European Journal of Operational Research 312 (2024) 92–116

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A multi-objective mixed integer linear programming model for thesis defence scheduling

João Almeida[a,*], Daniel Santos[a], José Rui Figueira[a], Alexandre P. Francisco[b]

[a] CEGIST, Instituto Superior Técnico, Universidade de Lisboa, Portugal
[b] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal

ARTICLE INFO

Article history:
Received 17 April 2023
Accepted 16 June 2023
Available online 22 June 2023

Keywords:
Timetabling
Education
Integer programming
Multiple objective programming
Scheduling

ABSTRACT

In this paper, we address the thesis defence scheduling problem, a critical academic scheduling management process, which has been overshadowed in the literature by its counterparts, course timetabling and exam scheduling. Specifically, we address the single defence assignment type of thesis defence scheduling problems, where each committee is assigned to a single defence, scheduled for a specific day, hour and room. We formulate a multi-objective mixed-integer linear programming model, which aims to be applicable to a broader set of cases than other single defence assignment models present in the literature, which have a focus on the characteristics of their universities. For such a purpose, we introduce a different decision variable, propose constraint formulations that are not regulation and policy specific, and cover and offer new takes on the more common objectives seen in the literature. We also include new objective functions based on our experience with the problem at our university and by applying knowledge from other academic scheduling problems. We also propose a two-stage solution approach. The first stage is employed to find the number of schedulable defences, enabling the optimisation of instances with unschedulable defences. The second stage is an implementation of the augmented ϵ-constraint method, which allows for the search of a set of different and non-dominated solutions while skipping redundant iterations. The methodology is tested for case-studies from our university, significantly outperforming the solutions found by human schedulers. A novel instance generator for thesis scheduling problems is presented. Its main benefit is the generation of the availability of committee members and rooms in availability and unavailability blocks, resembling their real-world counterparts. A set of 96 randomly generated instances of varying sizes is solved and analysed regarding their relative computational performance, the number of schedulable defences and the distribution of the considered types of iterations. The proposed method can find the optimal number of schedulable defences and present non-dominated solutions within the set time limits for every tested instance.

1. Introduction

The scheduling of thesis defences is a fundamental problem in the academic world. Millions of students worldwide need to prepare and defend their theses in what is often their greatest academic challenge up to that point. Thus, considering the many resource allocation challenges it presents for colleges and universities every year, where it is often assigned to a single person who must solve it using e-mails, excel sheets, and other less-than-ideal tools, thesis defence scheduling is one of the most important academic management problems. The literature has established the problem, focusing on each country and university's regulations and culture. Even so, it can be defined through the 6W's question framework: we want to know Who (the committee member) is to be assigned to Which examination committee, performing What role, to Whom (students/thesis defences), When (day and hour) and Where (the specific room). Ultimately, the association of these assignments is called a schedule.

A feasible schedule fulfils a particular set of constraints. For thesis defence scheduling those can be classified under three categories: scheduling complete committees, fulfilling committee composition rules and ensuring committee member and room availability. Moreover, a feasible schedule is not necessarily a "good" one. Likewise, two points of view assess the schedules: committee assignment quality and schedule quality. These points of view

* Corresponding author.
E-mail address: joao.carvalho.almeida@tecnico.ulisboa.pt (J. Almeida).

https://doi.org/10.1016/j.ejor.2023.06.031

are rendered operational by constructing several criteria or objectives to be maximised or minimised. Moreover, regarding committee assignment quality, the criteria can go from fairly distributing assignments between committee members to assessing their suitability to evaluate the thesis defences. Additionally, the schedule quality objectives might include, for example, promoting compact schedules, satisfying preferred time slot requests from committee members or preventing room changes.

Conversely, many problems may arise when such measures are not considered or when a human scheduler cannot adequately meet them. To begin with, there is the perceived (or actual) bias of the scheduler in favour of some committee members, which might lead to disagreements between the affected parts regarding fairness and transparency considerations. Moreover, if the assigned committee is not knowledgeable enough in the necessary research subjects to assess a thesis, this might lead to inaccurate evaluations of the student's work. Furthermore, as most committee members often have other tasks and are not available at all times, finding a schedule that is as small of an inconvenience for them as possible is not a simple task, which leads to committee members having less available time to dedicate to their research, teaching, or any other activity. Lastly, there is the problem of the scheduling process itself, which just burns too much time for the scheduler, who could put it to more productive use. Likewise, several approaches for dealing with such problems have been considered in the literature.

Regarding a fair assignment distribution, Kochaniková & Rudová (2013) proposes a constructive heuristic that iteratively assigns defences to committees in such a manner that each committee is appointed the same or at most one more defence than the other committee. Moreover, their improvement phase keeps this distribution intact. Pham et al. (2015) takes a different approach, minimising the workload differences between the most and least burdened committee members. Conversely, Battistutta et al. (2019) is the first to impose an exponential penalty for the number of times a committee member is assigned to a different committee. Finally, Christopher & Wicaksana (2021) includes a maximum quota of assignments for each committee member.

In terms of the methodologies used to assess committee suitability, there is minimal variance. Each consideration of this point of view aims to optimise the matching between expertise or research areas of the committee members and the subject of the defences they are assigned to (Battistutta et al., 2019; Christopher & Wicaksana, 2021; Huynh et al., 2012; Pham et al., 2015). Nonetheless, other aspects are also regarded in Christopher & Wicaksana (2021), specifically committee member academic level and previous experience moderating defences.

As far as reducing the inconvenience caused for committee members, the literature has considered different measures that better suit their situation. Huynh et al. (2012) considers an objective that minimises the total number of room changes. Moreover, the same work introduces a compactness measure where an increasing number of time slots between two consecutive assignments is penalised. In Battistutta et al. (2019), it is stated that this is a valuable objective. However, due to their structuring of thesis defences into sessions, it is automatically guaranteed. Concerns regarding time slot preferences also arise in Huynh et al. (2012) and Tawakkal & Suyanto (2020), with the latter including an objective specifically regarding students.

There are several types of defence scheduling problems, distinguished by the number of defences that a committee is assigned to: (1) *Single defence assignment* - Each committee is assigned to one defence, which will take place in one slot (day, hour and room). This type of scheduling is addressed in Christopher & Wicaksana (2021); Huynh et al. (2012); Limanto et al. (2019); Pham et al. (2015); Tawakkal & Suyanto (2020) and our work; (2) *Session of defences assignment* - Each committee is assigned to a group of defences (session), which will take place in one slot (day, period and room). Evidently, the time slot cannot be just a particular hour in such instances. Instead, it represents an extended period, such as a morning, afternoon, or day. This type of scheduling is addressed in Battistutta et al. (2019); (3) *Hybrid assignment* - A committee is assigned to an extended period, like in the session of defences assignment type. However, single defences and other required additional committee members are then assigned to an hour within such a period. This type of scheduling is addressed in Kochaniková & Rudová (2013).

Several different solution approaches have been applied to the thesis defence scheduling problem, specifically: Mixed-integer linear programming (Battistutta et al., 2019); Constraint programming (Battistutta et al., 2019); Greedy backtracking hybrid algorithm (Su et al., 2020); Local search (Battistutta et al., 2019; Kochaniková & Rudová, 2013; Pham et al., 2015; Tawakkal & Suyanto, 2020); Genetic algorithm (Huynh et al., 2012; Limanto et al., 2019); and Particle swarm optimisation (Christopher & Wicaksana, 2021).

In contrast with the most studied academic scheduling problems, exam scheduling and course timetabling (Chaudhuri & De, 2010); for a state-of-the-art review of those, we refer the reader to Babaei et al. (2015); Ceschia et al. (2023); Chen et al. (2021); and for how inefficient and time-consuming assigning and scheduling committees is for the unfortunate people to whom such tasks are delegated, thesis defence scheduling is remarkably underrepresented in the literature. We propose a multi-objective mixed-integer linear programming model, which, to the best of our knowledge, is the first formulation of its type for the single defence assignment type. Moreover, our approach aims to be applicable to a broader range of instances and regulations than the ones found in the single defence assignment literature. To achieve such a goal, our model includes the following novel characteristics, organised under three levels:

1. Main decision variable - our fundamental decision variable, built from the 6W's question framework, is the first to include the committee member's role in each assignment. This is beneficial as it allows for a greater committee composition flexibility than the previous formulations;
2. Constraints - instead of modelling our university's regulations, we take advantage of the novel decision variable and input the eligible committee members for each role in each thesis defence as a parameter. Therefore, it can then be adapted to fit each instance's needs. Moreover, we are the first to model the problem in a way that considers the possibility of not scheduling every defence, as not all of them might be schedulable, and presenting an @incomplete" schedule can still be valuable for the decision-maker;
3. Objectives - to the best of our knowledge, we are the first to introduce an objective that minimises the number of days a committee member is scheduled to attend a defence. Additionally, we also propose different formulations for several previously defined objectives. Specifically, we introduce a linearisation of an exponential penalty for the number of assignments, compactness, and room change objectives better suited for cases where a committee member is not available for every time slot and a differentiation mechanism for preferred time slots with multiple preference levels.

As for the solution approach, we must be able to find how many thesis defences can be scheduled. Consequently, we propose a two-stage approach, with the first stage being an adaption of our formulation, but with the objective of finding the number of schedulable thesis defences. That number will then be used as a parameter in our model. While new to thesis defence scheduling, similar two-stage approaches have been studied in academic

timetabling problems with mixed-integer based solution methods (Burke et al., 2010; Sørensen & Dahms, 2014; Vermuyten et al., 2016). Their primary motivation was to simplify the search space of an otherwise computationally difficult problem to solve with mixed-integer linear programming. This is not our primary goal, as, instead of using it to solve different objectives separately, we want to find the value of a necessary parameter. Moreover, course timetabling works with heuristic solution methods have also applied two-stage approaches. Similarly to our approach, in Bellio et al. (2021); Goh et al. (2017), the first stage of the heuristic is employed to guarantee the feasibility of the solution. Conversely, other works decompose their heuristics based on different objectives (Al-Yakoob & Sherali, 2015; Santiago-Mozos et al., 2005). Lastly, while most works regard multiple objectives, none propose approaches that allow the search of the solution space for several solutions, commonly employed in other scheduling problems (Amiri & Farvaresh, 2023; Guo et al., 2023; Gülcü & Akkan, 2020; Koziel & Pietrenko-Dabrowska, 2022; Urbani et al., 2023). Contrarily, we adapt the augmented ϵ-constraint method, introduced in Mavrotas (2009) and Mavrotas & Florios (2013). The main benefits of such an implementation are the assurance that the found solutions are non-dominated (or Pareto optimal) and the existence of mechanisms to foresee and skip some iterations that will not yield new solutions. Moreover, having better knowledge about the trade-offs between solutions can be valuable to the decision-maker (Mesquita-Cunha et al., 2022).

The methodology is tested in case studies from different departments and outperforms the human schedulers. We also analyse how our model should be parameterised to solve problems from the literature.

We propose an instance generator for thesis defence scheduling. It generates the availability of committee members and rooms in availability blocks, resembling their real-world counterparts. Computational experiments are conducted on a set of 96 instances. The main conclusions are that the number of schedulable defences was always relatively easy to optimally determine during the first stage and that, even for the largest considered instances, some non-dominated solutions were found within the set time limits.

The remainder of this paper is organised as follows. Section 2 introduces the multi-objective mixed-integer linear programming model and briefly describes its parameters, variables, constraints, and objective functions. Section 3 is devoted to the approach for identifying non-dominated solutions of the model presented in the previous section. Section 4 presents case studies from two different departments and explains how the model can be parameterised to fit different problems from the literature. Section 5 addresses the instance generation method. Section 6 presents the computational experiments conducted to test the scalability of the approach and an analysis of the findings. Lastly, Section 7 includes concluding remarks and future research path suggestions.

2. The multi-objective mixed integer linear programming model

This section presents the multi-objective mixed-integer linear programming model to schedule and assign committees to thesis defences. It introduces the indices of variables and parameters. Then it presents the necessary parameters and variables, followed by a general overview of the objectives and the definition of the constraints. Lastly, the objectives will be revisited and appropriately defined. For a more detailed description of the mathematical formulation, the reader is directed to Appendix A.

2.1. Indices

The necessary indices for the parameters and variables are presented in this subsection. Let us note that, while they all start with 0, this value is never used to represent an object. Nonetheless, it is necessary to portray the absence of such objects in specific constraints and objective functions. For example, while there is no committee member 0, we still need the ability to quantify 0 committee members.

1. $i = 0, \ldots, n_i$, are the indices related to the master's thesis defence committee members;
2. $j = 0, \ldots, n_j$, are the indices related to the master's thesis defences;
3. $t = 0, \ldots, n_t$, are the indices related to the role of the committee members;
4. $k = 0, \ldots, n_k$, are the indices related to the days;
5. $\ell = 0, \ldots, n_\ell$, are the indices related to the available hour slots in each day;
6. $p = 0, \ldots, n_p$, are the indices related to the available rooms;
7. $q = 0, \ldots, n_q$, are the indices related to research subjects.

2.2. Parameters

The parameters for the model are presented in this subsection. They have been divided into three groups, the first about individual committee members, the second regarding both committee members and rooms, and the last one related to thesis defences.

1. *Related to committee members.*
   (a) $e_{ijt} \in \{0, 1\}$, is 1 if a committee member i is eligible to be assigned to a certain role t in a designated defence j; and 0 otherwise, for all $i = 1, \ldots, n_i$, $j = 1, \ldots, n_j$, $t = 1, \ldots, n_t$;
   (b) $c_i \in \mathbb{N}$, is the maximum number of committees a committee member i can be assigned to, for all $i = 1, \ldots, n_i$;
   (c) $u_i \in \mathbb{N}$, is the weight assigned to each committee member i, for all $i = 1, \ldots, n_i$;
   (d) $l_{ik\ell} \in \mathbb{N}_0$, is the preference level that each committee member i holds for every time slot (day k and hour ℓ), 0 represents unavailability, for all $i = 1, \ldots, n_i$, $k = 1, \ldots, n_k$, $\ell = 1, \ldots, n_\ell$;
   (e) $r_{iq} \in \{0, 1\}$, is 1 if a committee member i has knowledge regarding the research subject q; and 0 otherwise, for all $i = 1, \ldots, n_i$, $q = 1, \ldots, n_q$;
2. *Related to committee members and rooms.*
   (a) $b_i \in \{0, \ldots, d-1\}$, is the number of hour slots following the end of a defence in which a committee member i would consider the scheduling of a different one as compact, in our formulation we considered it as always being smaller than the duration d of a defence, for all $i = 1, \ldots, n_i$;
   (b) $v_{i\ell} \in \{0, \ldots, n_{v_i}\}$, is the weight given by a committee member i to each hour slot ℓ considered compact. A higher weight for an hour-slot represents a preferable assignment, for all $i = 1, \ldots, n_i$, $\ell = 0, \ldots b_i$;
   (c) $a_i \in \{0, \ldots, d-1\}$, is the number of hour slots following the end of a defence in which a committee member i would consider changing rooms problematic, for all $i = 1, \ldots, n_i$;
   (d) $h_{i\ell} \in \{0, \ldots, n_{h_i}\}$, is the penalty given by a committee member i after a room change to each hour slot ℓ considered problematic. A higher weight for an hour-slot represents a less desirable assignment, for all $i = 1, \ldots, n_i$, $\ell = 0, \ldots, a_i$;
   (e) $m_{k\ell p} \in \{0, 1\}$, is 1 if a room p is available at a certain time slot (k, ℓ); and 0 otherwise, for all $k = 1, \ldots, n_k$, $\ell = 1, \ldots, n_\ell$, $p = 1, \ldots, n_p$;
3. *Related to thesis defences.*
   (a) $d \in \mathbb{N}$, is the length (duration) of a defence in hour slots;

(b) $\bar{t}_{jq} \in \{0,1\}$, is 1 if a defence j studies a certain research subject q; and 0 otherwise, for all $j = 1,\ldots,n_j$, $q = 1,\ldots,n_q$;
(c) $g \in \{0,\ldots,n_j\}$, is the number of complete committees (thesis defences) to be scheduled. If one knows that all defences are schedulable, then $g = n_j$. Otherwise, finding g becomes part of the problem itself.

2.3. Variables

The definition of the decision and auxiliary variables is presented in this subsection. The first is built from the 6W's question framework and represents a committee member's assignment to a role within a thesis defence, happening at a specific time and room. The others are used to define concepts such as the number of scheduled days, workloads, defence research subject coverage by its committee, compactness and room change penalties, and assigned roles within a committee.

1. *Decision variables.*
   (a) $x_{ijtk\ell p} \in \{0,1\}$, which is equal to 1 if a committee member i is assigned to thesis defence j, performing a role t, in day k, at hour slot ℓ and in room p; and 0 otherwise, for all $i = 1,\ldots,n_i$, $j = 1,\ldots,n_j$, $t = 1,\ldots,n_t$, $k = 1,\ldots,n_k$, $\ell = 1,\ldots,n_\ell$, and $p = 1,\ldots,n_p$;
2. *Auxiliary variables.*
   (a) $y_{jk\ell p} \in \{0,1\}$, is 1 if the thesis defence j is scheduled for day k, at hour ℓ in room p; and 0 otherwise, for all $j = 1,\ldots,n_j$, $k = 1,\ldots,n_k$, $\ell = 1,\ldots,n_\ell$, and $p = 1,\ldots,n_p$;
   (b) $\bar{y}_{ik\ell p} \in \{0,1\}$, is 1 if committee member i is assigned to any defence on day k, hour ℓ and room p; and 0 otherwise, for all $i = 1,\ldots,n_i$, $k = 1,\ldots,n_k$, $\ell = 1,\ldots,n_\ell$, and $p = 1,\ldots,n_p$;
   (c) $\hat{y}_{ijk} \in \{0,1\}$, is 1 if committee member i is assigned to j defences on day k; and 0 otherwise, for all $i = 1,\ldots,n_i$, $j = 0,\ldots,n_j$ and $k = 1,\ldots,n_k$;
   (d) $w_{ij} \in \{0,1\}$, is 1 if committee member i is assigned to j thesis defences; and 0 otherwise, for all $i = 1,\ldots,n_i$, and $j = 0,\ldots,n_j$;
   (e) $\overline{w}_{ik} \in \{0,1\}$, is 1 if committee member i is assigned to thesis defences in k days; and 0 otherwise, for all $i = 1,\ldots,n_i$, and $k = 0,\ldots,n_k$;
   (f) $s_{ijq} \in \{0,1\}$, is 1 if i committee members of the committee of defence j have q research subject in common with said defence; and 0 otherwise, for all $i = 0,\ldots,n_i$, $j = 1,\ldots,n_j$, and $q = 1,\ldots,n_q$;
   (g) $\bar{s}_{ik\ell p} \in \mathbb{N}_0$, is the compactness value for committee member i, scheduled to attend any defence at day k, hour ℓ and room p, for all $i = 1,\ldots,n_i$, $k = 1,\ldots,n_k$, $\ell = 1,\ldots,n_\ell$, and $p = 1,\ldots,n_p$;
   (h) $\hat{s}_{ik\ell p} \in \mathbb{N}_0$, is a room change penalty for committee member i, scheduled to attend any defence at day k, hour ℓ and room p, for all $i = 1,\ldots,n_i$, $k = 1,\ldots,n_k$, $\ell = 1,\ldots,n_\ell$, and $p = 1,\ldots,n_p$;

2.4. Objective functions

The objective functions are presented in this subsection. Two points of view are used to assess the assignment quality. The first one, committee assignment quality, is related to both fair workload distribution and the matching between the expertise of the committee members and the defences they are assigned to, accordingly, it impacts the quality of the defence assessment process by not having overburdened committees, and a good matching of members to defences. The second one, schedule quality, is related to the assessment of the quality of the schedules of individual committee members, and aims to reduce the inconvenience caused by scheduling them to certain time-slots and rooms. The objective functions used to render them operational are introduced here. Nonetheless, further constraints and variables must be defined before each objective's mathematical expressions can be specified. In Section 2.6 we will return to the objective functions and properly define them. Furthermore, the additional constraints are listed in Section 2.5.

1. *Point of view of committee assignment quality.* This point of view is operationalised by the following objectives.
   (a) *Minimise workloads*. The workload is the number of committee assignments a committee member has. To ensure fairness, this number is squared. With this criterion, we want to achieve a balanced workload distribution between committee members.
   (b) *Maximise research subject coverage*. Research subject coverage is the percentage of research subjects in a defence its committee covers. With this criterion, we want to maximise such coverage and aim to ensure that every subject in a defence is covered by at least one of the committee members.
   (c) *Maximise committee member suitability*. Committee member suitability is defined as the number of research subjects each committee member has in common with their assigned defences. With this criterion, we want to maximise such suitability and aim to ensure that each member is a specialist in as many of the defence's subjects as possible.
2. *Point of view of schedule quality.* This point of view is operationalised by the following objectives.
   (a) *Minimise non-consecutive assignments*. Each assignment is given a compactness value. This value is based on the committee member's weight and the weight for the interval in which a defence is scheduled. The committee member defines the latter regarding their preferences over time intervals between defences. With this criterion, we want to produce more compact schedules according to the committee members' preferences.
   (b) *Minimise the non-satisfaction of time slot preferences*. A penalty is given whenever a committee member is assigned one of their stated undesirable time slots. With this criterion, we want to minimise the occurrence of such assignments.
   (c) *Minimise committee days*. Committee days are the number of days a committee member is scheduled to attend a defence. To ensure fairness, this number is squared. With this criterion, we want to minimise such a number.
   (d) *Minimise room changes*. Each committee member defines a time frame considered problematic for a room change between defences, penalising such events. With this criterion, we want to promote room stability.

2.5. Constraints

The necessary constraints to model the feasible region of the problem are presented in this subsection. They fall under four categories. The first concerns the scheduling of complete committees or thesis defences. The second regards the respect for committee member assignment rules. The third guarantees that committee members and rooms are available for their corresponding assignments. Finally, the fourth defines the values for several auxiliary variables present in the objective functions.

1. *Scheduling complete committees.* These constraints define a complete committee and ensure that every schedulable defence is assigned one.
   (a) *Complete committee definition*. A complete committee is a set of n_t assignments for a defence, j, all with a different appointed role, t, in the same slot, (day k, hour ℓ, and room p). Moreover, for a defence to occur, it must have such a committee assigned to it. This constraint ensures that either

a defence is assigned to a complete committee or no assignments for such a defence can occur.

$$\sum_{i=1}^{n_i} x_{ijtk\ell p} = y_{jk\ell p}, \quad j = 1, \ldots, n_j,\ t = 1, \ldots, n_t$$
$$k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell,\ p = 1, \ldots, n_p \tag{1}$$

(b) *Single committee assignment*. If a defence, j, can be scheduled, it should only be assigned one committee and appointed one slot, (day k, hour ℓ, and room p). Thus, in this constraint, we state that for a defence, j, the number of complete committees assigned to it is less or equal to 1.

$$\sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} y_{jk\ell p} \leqslant 1, \quad j = 1, \ldots, n_j \tag{2}$$

(c) *Complete committees (thesis defences) to be scheduled*. In each instance of the thesis defence scheduling problem, a defined number of committees (thesis defences) can be assigned and scheduled, denoted by g. If one knows that all the defences can be scheduled, then this number is the number of defences, i.e., $g = n_j$. However, some defences may not be schedulable due to conflicting committee member availabilities, lack of enough eligible committee members, lack of rooms, or others. In such cases, finding the value for g becomes an indispensable part of the problem. In this constraint, assuming the value of g is already known, we enforce the number of assigned complete committees (thesis defences) as the number of schedulable complete committees, g.

$$\sum_{j=1}^{n_j} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} y_{jk\ell p} = g \tag{3}$$

2. *Committee Composition.* These constraints ensure the eligibility of the committee members to perform their assignments.

(a) *Committee member eligibility*. Different universities and their departments have distinct regulations for the eligibility of committee members, i, to perform specific roles, t, within each committee for a defence, j. Thus, we do not attempt to include such rules within our model. Conversely, we aggregate them in a parameter, e_{ijt}, which takes the value 1 if a committee member, i, is eligible to perform a role, t, in a defence, j, and 0 otherwise. In this constraint, we state that only eligible members can be assigned to a committee.

$$\sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant e_{ijt}, i = 1, \ldots, n_i,$$
$$j = 1, \ldots, n_j,\ t = 1, \ldots, n_t \tag{4}$$

(b) *Maximum number of committees assigned to a committee member*. In this constraint, we ensure that the sum of the assignments, for a committee member, i, does not exceed the maximum allowed number of committees, c_i, for that committee member.

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant c_i, \quad i = 1, \ldots, n_i \tag{5}$$

3. *Committee member and room availability*. These constraints guarantee that committee members and rooms are available for each assignment.

(a) *Committee member time slot availability*. Committee members have different obligations other than attending thesis defences. Consequently, they are not available to be assigned to every time slot. This constraint ensures that members are not assigned for defences occurring at their unavailable times.

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant l_{ik\ell}, \quad i = 1, \ldots, n_i,$$
$$k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell \tag{6}$$

(b) *Committee member assignment juxtaposition*. A committee member, cannot be assigned to more than one defence at the same time. Additionally, this constraint also ensures that a committee member is not assigned more than one role in a defence, as that would mean that said committee member would have two different assignments in the same time slot.

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{\ell=\bar{\ell}}^{\bar{\ell}+d-1} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant 1, \quad i = 1, \ldots, n_i,$$
$$k = 1, \ldots, n_k,\ \bar{\ell} = 1, \ldots, n_\ell - d + 1 \tag{7}$$

(c) *Room time slot availability*. A room's purpose might not just be hosting thesis defences. Thus, it is natural that it happens to be booked for any other event at some point. Accordingly, whenever a room is unavailable, it cannot host any defence.

$$\sum_{j=1}^{n_j} y_{jk\ell p} \leqslant m_{k\ell p}, \quad k = 1, \ldots, n_k,\ \ell$$
$$= 1, \ldots, n_\ell,\ p = 1, \ldots, n_p \tag{8}$$

(d) *Room capacity*. In our formulation, we considered that a room could not hold more than one defence at a time.

$$\sum_{j=1}^{n_j} \sum_{\ell=\bar{\ell}}^{\bar{\ell}+d-1} y_{jk\ell p} \leqslant 1, \quad k = 1,$$
$$\ldots, n_k,\ \bar{\ell} = 1, \ldots, n_\ell - d + 1,\ p = 1, \ldots, n_p \tag{9}$$

4. *Objective functions measures.* These constraints define the values for the auxiliary variables necessary for some of the objective functions.

(a) *Research subject coverage definition.* For a defence, j, research subject coverage is the percentage of its studied research subject covered by the areas of expertise of its committee members. To implement such an objective, we first need to define an auxiliary binary variable, s_{ijq}, which takes the value 1 if a defence, j, which studies a research subject, q, that is, $\bar{t}_{jq} = 1$, has a number, i, of committee members assigned to its committee, who have said subject as one of their areas of expertise.

$$\sum_{i=0}^{n_i} i s_{ijq} = \sum_{i=1}^{n_i} \sum_{t=1}^{n_t} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} r_{iq} \bar{t}_{jq} x_{ijtk\ell p},$$
$$j = 1, \ldots, n_j,\ q = 1, \ldots, n_q \tag{10}$$

$$\sum_{i=0}^{n_i} s_{ijq} = 1, \quad j = 1, \ldots, n_j,\ q = 1, \ldots, n_q \tag{11}$$

(b) *Compactness value definition.* We defined a compact assignment of a committee member, i, to a day, k, at an hour, ℓ, as one that occurs within a specific time frame, b_i, after the end of a different assignment for such a committee member. That is, if a committee member, i, is assigned to a defence, j, at a day, k, and an hour, ℓ, this assignment is considered compact on the condition that the same committee member is assigned to a different defence, $\bar{j}$, in the same day, k, between hours $\ell - d$ and $\ell - d - b_i$. Moreover, the parameter

$v_{i\ell}$, distinguishes the hour slots within such a time frame, as they might have different perceived values for a committee member, i. Thus, the compactness value for an assignment, $\bar{s}_{ik\ell}$, is 0 if a committee member, i, does not have a different assignment ending between hours ℓ and $\ell - b_i$, or $v_{i\bar{\ell}}$ if he does have such an assignment ending at $\bar{\ell}$ hour slots before the start of the new assignment a different hour slot, ℓ.

$$\bar{y}_{ik\ell p} = \sum_{j=1}^{n_j}\sum_{t=1}^{n_t} x_{ijtk\ell p}, \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = 1,\dots,n_\ell,\ p = 1,\dots,n_p \tag{12}$$

$$\bar{s}_{ik\ell} \geqslant 0, \quad i{=}1,\dots,n_i,\ k{=}1,\dots,n_k,\ \ell{=}1,\dots,n_\ell \tag{13}$$

$$\bar{s}_{ik\ell} \leqslant n_{v_i}\sum_{p=1}^{n_p}\bar{y}_{ik\ell p} \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = 1,\dots,n_\ell \tag{14}$$

$$\bar{s}_{ik\ell} \leqslant \sum_{\bar{\ell}=0}^{b_i}\sum_{p=1}^{n_p} v_{i\bar{\ell}}\bar{y}_{ik\hat{\ell}p}, \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = d,\dots,n_\ell,\ \hat{\ell} = \ell - d - \bar{\ell} \tag{15}$$

$$\bar{s}_{ik\ell} \geqslant \sum_{\bar{\ell}=0}^{b_i}\sum_{p=1}^{n_p} v_{i\bar{\ell}}\bar{y}_{ik\hat{\ell}p} - n_{v_i}\left(1 - \sum_{p=1}^{n_p}\bar{y}_{ik\ell p}\right), \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = d,\dots,n_\ell,\ \hat{\ell} = \ell - d - \bar{\ell} \tag{16}$$

(c) *Workload definition.* The workload for a committee member, i, is defined as the number, j, of committees they are assigned to. It would be possible to represent it as an integer variable. Still, in such a case, it would not be possible to consider its square in the objective function while keeping its linearity. However, the exponential penalty in the objective function can be linearised by representing it through a variable, w_{ij}, which takes the value 1 if a committee member, i, is assigned to a number, j, of committees, and 0 otherwise.

$$\sum_{j=0}^{c_i} j w_{ij} = \sum_{j=1}^{n_j}\sum_{t=1}^{n_t}\sum_{k=1}^{n_k}\sum_{\ell=1}^{n_\ell}\sum_{p=1}^{n_p} x_{ijtk\ell p}, \quad i{=}1,\dots,n_i \tag{17}$$

$$\sum_{j=0}^{c_i} w_{ij} = 1, \quad i = 1,\dots,n_i \tag{18}$$

(d) *Committee days definition.* A committee day is defined as a day when a committee member has a defence scheduled. To represent this concept, we introduce a variable, $\overline{w}_{ik}$, which takes the value 1 if a committee member, i, has defences scheduled on a number, k, of days, and 0 otherwise.

$$\sum_{j=0}^{c_i} j\hat{y}_{ijk} = \sum_{j=1}^{n_j}\sum_{t=1}^{n_t}\sum_{\ell=1}^{n_\ell}\sum_{p=1}^{n_p} x_{ijtk\ell p}, \quad i{=}1,\dots,n_i,\ k = 1,\dots,n_k \tag{19}$$

$$\sum_{j=0}^{n_j}\hat{y}_{ijk} = 1, \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k \tag{20}$$

$$\sum_{k=0}^{n_k} k\overline{w}_{ik} = \sum_{j=1}^{c_i}\sum_{k=1}^{n_k}\hat{y}_{ijk}, \quad i = 1,\dots,n_i \tag{21}$$

$$\sum_{k=0}^{n_k}\overline{w}_{ik} = 1, \quad i = 1,\dots,n_i \tag{22}$$

(e) *Room change penalty definition.* A room change is considered problematic if a committee member, i, is not given a certain amount of time, a_i, between the end of an assignment, $\bar{j}$, and the beginning of another, j, which is scheduled for a different room, $\overline{p}$, than the first one, p. Moreover, parameter $h_{i\ell}$ distinguishes the hour slots within such a timeframe, as they might have different perceived penalties for a committee member, i. To assess this objective we define the room change variable, $\hat{s}_{ik\ell p}$, which represents the room change penalty that the assignment of a committee member, i, to a day, k, an hour, ℓ and a room, p, would incur.

$$\hat{s}_{ik\ell p} \geqslant 0, \quad i{=}1,\dots,n_i,\ k{=}1,\dots,n_k,\ \ell{=}1,\dots,n_\ell,\ p{=}1,\dots,n_p \tag{23}$$

$$\hat{s}_{ik\ell p} \leqslant n_{h_i}\bar{y}_{ik\ell p} \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = 1,\dots,n_\ell,\ p = 1,\dots,n_p \tag{24}$$

$$\hat{s}_{ik\ell p} \leqslant \sum_{\bar{\ell}=0}^{a_i}\sum_{\overline{p}=1}^{n_p} h_{i\bar{\ell}}\bar{y}_{ik\hat{\ell}\overline{p}}, \ i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = d,\dots,n_\ell,\ \hat{\ell} = \ell - d - \bar{\ell},\ p = 1,\dots,n_p,\ p \neq \overline{p} \tag{25}$$

$$\hat{s}_{ik\ell p} \leqslant \sum_{\bar{\ell}=0}^{a_i}\sum_{\overline{p}=1}^{n_p} h_{i\bar{\ell}}\bar{y}_{ik\hat{\ell}\overline{p}} - n_{h_i}(1 - \bar{y}_{ik\ell p}), \quad i = 1,\dots,n_i,\ k = 1,\dots,n_k,\ \ell = d,\dots,n_\ell,\ \hat{\ell} = \ell - d - \bar{\ell},\ p = 1,\dots,n_p,\ p \neq \overline{p} \tag{26}$$

2.6. Back to the objective functions

In this subsection, now that we have defined all the necessary constraints, the objective functions within the two points of view are revisited and adequately determined.

1. *Point of view of committee assignment quality*. This point of view is operationalised by the following objectives.
   (a) *Minimise workloads*. This criterion considers a product, $u_i j^2 w_{ij}$. The variable, w_{ij}, takes the value 1 if a committee member, i, is assigned to a number, j, of defences, and 0 otherwise. By multiplying it by j^2, we can keep the linearity of the model while applying an exponential penalty representing the square of the number of defences a committee member, i, is assigned to, promoting fairness in workload distribution. Moreover, the committee member's weight, u_i, is also considered. We want to promote a fair distribution of workloads between committee members by minimising this objective.

$$\min z_1(w) = \sum_{i=1}^{n_i}\sum_{j=1}^{n_j} u_i j^2 w_{ij} \tag{27}$$

(b) *Maximise research subject coverage*. The research subject coverage for a defence, j, is computed through a quotient between the number of its research subjects, q, covered by its committee as the numerator and the sum of all of its research subjects as the denominator. Let us note that, similarly to Constraint (21), while variable s_{ijq} is defined for $i = 0, \ldots, n_i$, the sum must start on $i = 1$, as we do not want to include the research subjects that are covered 0 times by the defence's assigned committee. We want to maximise the sum of the coverages of all defences with this objective.

$$\max z_2(s) = \left(\sum_{j=1}^{n_j} \sum_{q=1}^{n_q} \bar{t}_{jq} \right)^{-1} \sum_{i=1}^{n_i} \sum_{j=1}^{n_j} \sum_{q=1}^{n_q} s_{ijq} \tag{28}$$

(c) *Maximise committee member suitability*. This criterion considers a product, $r_{iq}\bar{t}_{jq}x_{ijtk\ell p}$, which will be 0 unless a committee member, i, is assigned to a defence, j, that is $x_{ijtk\ell p} = 1$, and a research subject, q, is within the areas of expertise of a committee member, i, that is, $r_{iq} = 1$, and the subjects addressed in the defence, j, that is, $\bar{t}_{jq} = 1$, in which case the product will be 1. We want to maximise the sum of these products with this objective.

$$\max z_3(x) = \sum_{i=1}^{n_i} \sum_{q=1}^{n_q} \sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} r_{iq}\bar{t}_{jq}x_{ijtk\ell p} \tag{29}$$

2. *Point of view of schedule quality*. This point of view is operationalised by the following objectives.

(a) *Minimise non-consecutive assignments*. The highest potential compactness value for an assignment for a committee member, i, is parameter n_{v_i}. Thus, for a committee member, i, the maximum potential sum of compactness values would be the product of said maximum value by the committee member's number, j, of assignments minus one assignment, as, logically, the committee member's first assignment cannot be scheduled within a certain time-frame after another has ended. Thus, and by weighting each assignment by the weight conferred to its committee member, the maximum sum of compactness values for the problem is represented by $\sum_{i=1}^{n_i} \sum_{j=1}^{n_j} u_i n_{v_i} (j-1) w_{ij}$. On the contrary, the effective sum of the compactness values, weighted by their correspondent committee member's weight, is represented by $\sum_{i=1}^{n_i} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} u_i \bar{s}_{ik\ell}$. This criterion considers the difference between the compactness value for an ideal schedule and the effective value for the proposed schedule. We want to minimise such differences with this objective.

$$\min z_4(w, \bar{s}) = \sum_{i=1}^{n_i} \sum_{j=1}^{n_j} u_i n_{v_i} (j-1) w_{ij} - \sum_{i=1}^{n_i} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} u_i \bar{s}_{ik\ell} \tag{30}$$

(b) *Minimise the non-satisfaction of time slot preferences*. This criterion considers a product, $u_i(l_{ik\ell} - 1)x_{ijtk\ell p}$, which represents the penalty of assigning a committee member, i, to a day, k, at an hour, ℓ, that is $x_{ijtk\ell p} = 1$. In this case, it takes the value of the product of the committee member's weight, u_i, with the penalty level assigned by the committee member to the combination of day, k, at an hour, ℓ, that is, $l_{ik\ell} - 1$. The parameter, $l_{ik\ell}$, can take the value 0 if a committee member, i, is unavailable at a day, k, and an hour, ℓ, but such an assignment would be infeasible, or a natural number, with larger values representing lower preference levels. We want to minimise the sum of such penalties with this objective.

$$\min z_5(x) = \sum_{i=1}^{n_i} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{p=1}^{n_p} u_i (l_{ik\ell} - 1) x_{ijtk\ell p} \tag{31}$$

(c) *Minimise committee days*. This criterion takes into account a product, $u_i k^2 \overline{w}_{ik}$. The variable, $\overline{w}_{ik}$, takes the value 1 if a committee member, i, is assigned to committees in a number, k, of days, and 0 otherwise. By multiplying it by k^2, we keep the model linear while applying an exponential penalty representing the square of the number of days a committee member, i, is assigned to, promoting fairness in the committee days distribution. Moreover, the committee member's weight, u_i, is also considered. We want a fair distribution of committee days between committee members and to minimise the sum of such a product with this objective.

$$\min z_6(\overline{w}) = \sum_{i=1}^{n_i} \sum_{k=1}^{n_k} u_i k^2 \overline{w}_{ik} \tag{32}$$

(d) *Minimise room changes*. The variable $\hat{s}_{ik\ell p}$ is the assigned room change penalty for a committee member, i, who will change rooms within a problematic time frame due to being assigned to any defence in a day, k, an hour, ℓ, and a room, p. The penalty is incurred on the condition that the committee member, i, is also assigned to another defence, in a different room, $\overline{p}$, in the same day, k, which ends within a specific time-frame, between ℓ and $\ell - a_i$, with the parameter, a_i, representing the number of hour slots before an assignment where the committee member considers scheduling the end of another assignment as problematic for a room change. We want to minimise the sum of the product of such penalties by their incurring committee member's weight with this objective.

$$\min z_7(\hat{s}) = \sum_{i=1}^{n_i} \sum_{j=1}^{n_j} u_i \hat{s}_{ij} \tag{33}$$

2.7. Summary

In thesis defence scheduling, we want to know Who (the committee member) is to be assigned to Which committee, performing What role, to Whom (thesis defences), When (day and hour) and Where (room).

In every MOMILP problem, the feasible region of the decision space is defined by a certain number of constraints. In our thesis defence scheduling model, we divided these constraints into three groups:

1. *Scheduling complete committees*. These constraints define a complete committee and ensure that every schedulable defence is assigned one.
2. *Committee composition*. These constraints ensure the eligibility of the committee members to perform their assignments.
3. *Committee member and room availability*. These constraints guarantee that committee members and rooms are available for each assignment.

Nonetheless, not all feasible solutions are equivalent. To assess their relative standing, we identified two points of view, rendered operational by specific criteria:

1. *Committee assignment quality*. This point of view includes criteria related to the committees assigned to each defence. Accordingly, it impacts the quality of the defences and the workload distribution between committee members.

2. *Schedule quality*. This point of view includes criteria for minimising the inconvenience the assignments generate for the committee members.

3. An approach to the MOMILP problem

This section addresses the approach chosen to solve the MOMILP problem introduced in the previous section. It starts by presenting some fundamental concepts and definitions, followed by the employed algorithm and, finally, the practical aspects regarding the actual use of the algorithm.

3.1. Some fundamental concepts, their definitions, and notation

Consider the following MOMILP problem,

$$\begin{aligned} &\max\ z_1(x),\\ &\quad\vdots\\ &\max\ z_i(x),\\ &\quad\vdots\\ &\max\ z_{n_z}(x),\\ &\text{subject to:}\\ &x \in X \end{aligned} \tag{34}$$

where $x = (x_1, \ldots, x_j, \ldots, x_{n_x})$ is the vector of the decision variables, X is the feasible region in the decision space, and z_i is the $i-th$ linear objective function, for $i = 1, \ldots, n_z$. The image of X according to all the objective functions defines the feasible region, Z, in the criterion space.

A fundamental concept in multi-objective optimisation is the notion of *dominance*. A solution or outcome vector z' in the objective space dominates another solution z'' if and only if $z'_i \geqslant z''_i$, for all $i = 1, \ldots, n_z$, with at least one of these being a strict inequality, i.e., $z'_i > z''_i$ for some i.

A feasible solution, $\bar{z} \in Z$, is said to be *non-dominated* if and only if there is no other feasible solution, $z \in Z$, such that z dominates $\bar{z}$. The set of all non-dominated solutions is known in the literature as the *Pareto front*. The inverse image of a non-dominated solution, $\bar{x} = F^{-1}(\bar{z})$ is called an *efficient solution* (in the decision space).

Our objective is to identify a subset of the Pareto front, denoted by N. For this purpose, we will use a well-known scalarisation technique based on the resolution of a sequence of ϵ-constraint problems of the form:

$$\begin{aligned} &\max\ z_1(x),\\ &\text{subject to:}\\ &x \in X,\\ &z_i(x) \geqslant \epsilon_i,\ i = 2, \ldots, n_z \end{aligned} \tag{35}$$

where only one of the objective functions, arbitrarily chosen here as $z_1(x)$, is being maximised. Whereas the others are instead included in constraints, which set lower bounds, ϵ_i, for each of the remaining objective functions, $z_i(x)$, for $i = 2, \ldots, n_z$. Moreover, different non-dominated solutions are found by setting different values for the lower bounds, ϵ_i.

3.2. Algorithmic framework

This subsection addresses the multi-objective algorithmic framework. It is divided into three sequential steps. The first is finding the number of schedulable defences, g, which is then set as a parameter for the subsequent steps. The second is computing the ideal, z^{id}, and approximated nadir, z^{nad}, points. These points are necessary as in the next step a number of equally spaced bounds between z^{id} and z^{nad} is defined for each objective and used as the values for ϵ. This methodology allows us to control the maximum number of solutions we wish to obtain. Finally, the augmented ϵ-constraint method itself.

In fact, the proposed method adapts the augmented ϵ-constraint method, introduced in Mavrotas (2009) and Mavrotas & Florios (2013). Our adaption obtains a subset of the Pareto front by iteratively increasing the lower bounds, ϵ_i, for each objective, $i = 2, \ldots, n_z$. However, in contrast with some ϵ-constraint methods, this method guarantees that all solutions found are non-dominated. To achieve this, instead of just using the objective function $z_1(x)$, a component related to the remaining objective functions is also included in the objective function with the help of surplus variables.

3.2.1. Computing the ideal and the approximate nadir points

For each $i = 1, \ldots, n_z$, the problem of Eq. (36) is solved. Each objective function is divided into two components. The first is an objective function, $z_i(x)$, for $i = 1, \ldots, n_z$. The second is the sum of the remaining objective functions $z_j(x)$, for $j = 1, \ldots, n_z,\ j \neq i$, multiplied by a suitable number 10^{-E} that ensures that, regardless of the value the second component takes, the value of the first component is the same as if we would instead maximise $z_i(x)$ separately, i.e., $\max_{x \in X} z_i(x),\ i = 1, \ldots, n_z$.

$$z_i^{\rho_i *} = \max_{x \in X} \left\{ z_i(x) + \left(10^{-E}\right) \sum_{j=1,\ i \neq j}^{n_z} z_j(x) \right\},\quad i = 1, \ldots, n_z. \tag{36}$$

Let us denote the perturbation of the objective function, $z_i(x)$, in the previous equation by,

$$\rho_i = \left(10^{-E}\right) \sum_{j=1,\ i \neq j}^{n_z} z_j(x),\quad i = 1, \ldots, n_z, \tag{37}$$

where $0 \leqslant \rho_i < 1$, i.e., since $z_i(x) \in \mathbb{Z}$, then ρ_i will not influence the value of $z_i(x)$. We can now define, $z_i^* = z_i^{\rho_i *} - \rho_i$, for all $i = 1, \ldots, n_z$, and the ideal point, z^{id}, can be defined as follows:

$$z^{id} = \left(z_1^*, \ldots, z_i^*, \ldots, z_{n_z}^*\right). \tag{38}$$

Let z_j^{i*}, denote the value obtained for an objective function, $z_j(x),\ j = 1, \ldots, n_z$, when maximising $z_i(x),\ i = 1, \ldots, n_z$, i.e., when solving the problem of Eq. (36), for $z_i(x)$. Each scalar component, z_i^{nad}, of the approximate nadir vector, z^{nad}, can be defined as follows:

$$z_i^{nad} = \min_{j=1,\ldots,n_z} \left\{ z_i^{j*} \right\},\quad i = 1, \ldots, n_z. \tag{39}$$

Accordingly, the approximate nadir vector can be stated as follows:

$$z^{nad} = \left(z_1^{nad}, \ldots, z_i^{nad}, \ldots, z_{n_z}^{nad}\right). \tag{40}$$

This process is represented in Algorithm 1, denoting the problem of Eq. (36), for each $z_i(x)$, as P^{z_i}.

Algorithm 1. Compute z^{id} and z^{nad}.

```
input: P^{z_i};
output: z^{id}, z^{nad};
for (j = 1, ..., n_z) do
    z^{nad} ← z^{nad} ∪ ∞;
end for
for (i = 1, ..., n_z) do z_i^{ρ_i*} ← optimise(P^{z_i});
    z^{id} ← z^{id} ∪ (z_i^{ρ_i*} − ρ_i);
    for (j = 1, ..., n_z, i ≠ j) do
        if (z_j < z_j^{nad}) then
            z_j^{nad} ← z_j;
return(z^{id}, z^{nad});
```

3.2.2. Augmented ϵ-constraint

Finally, we can define the surplus variables, $s_i(x)$, for $x \in X$, and for each objective, $z_i(x)$, for $i = 2, \ldots, n_z$, as in Eq. (41). These are computed as the ratio of the difference between the objective, $z_i(x)$, and its corresponding value in the nadir point, z_i^{nad}, and the difference between the optimum value for the objective, z_i^*, and its corresponding value in the nadir point, z_i^{nad}.

$$s_i(x) = \frac{z_i(x) - z_i^{nad}}{z_i^* - z_i^{nad}}, \quad i = 2, \ldots, n_z, \; x \in X. \tag{41}$$

Now, to define the objective function used in the augmented ϵ-constraint method, z^ϵ, as in Equation Eq. (42). This objective function is divided into two components. The objective function $z_1(x)$ and a perturbation component, computed through the sum of the surplus variables, $s_i(x)$, employed to guarantee non-dominated solutions, while ensuring that the value of the objective function $z_1(x)$ is the same as if we would instead maximise $z_1(x)$ separately, i.e., $\max_{x \in X} z_1(x)$, while considering that the remaining objectives $z_j(x)$ are subject to the lower bound vector ϵ, i.e., $z_j(x) \geqslant \epsilon_j$, for $j = 2, \ldots, n_z$.

$$z^\epsilon = \max_{x \in X} \left\{ z_1(x) + (n_z - 0.9)^{-1} \sum_{i=2}^{n_z} s_i(x) \right\} \tag{42}$$

Let us denote the perturbation of the objective function $z_1(x)$ by

$$\phi = (n_z - 0.9)^{-1} \sum_{i=2}^{n_z} s_i(x), \tag{43}$$

where, given that $0 \leqslant s_i(x) \leqslant 1$, it follows that $0 \leqslant \phi < 1$, i.e., since $z_1(x) \in \mathbb{Z}$, then ϕ will not influence the optimality of $z_1(x)$, considering that the remaining objective functions, $z_j(x)$, are subject to the vector of lower bounds, ϵ.

To iterate between the vector of lower bounds, ϵ, we first specify a parameter, p_i, subject to $\frac{1}{p_i} \in \mathbb{N}$, which is the percentage of the gap between the correspondent ideal, z_i^*, and nadir, z_i^{nad}, scalar components, which is incremented between each iteration for an objective, $z_i(x)$. Moreover, we also define a vector, v, of dimension $n_z - 1$, such that each scalar component, v_i, represents the number of times each lower bound, ϵ_i, is to be incremented by its corresponding percentage, p_i, in the current iteration. Thus, Eq. (44) defines the lower bound, ϵ_i, for a given objective, $z_i(x)$, as the increment of the scalar component, z_i^{nad}, by a percentage, $v_i p_i$, of the difference between the scalar components, z_i^* and z_i^{nad}. Note that, while all other parameters are constants, v_i, is updated between each iteration. Moreover, the minimum value for the lower bound, ϵ_i, must be z_i^{nad}, and its maximum value z_i^*, therefore $v_i \in \{0, \ldots, \frac{1}{p_i}\}$.

$$\epsilon_i = z_i^{nad} + v_i p_i \left(z_i^* - z_i^{nad} \right), \; i = 2, \ldots, n_z \tag{44}$$

To update v between iterations, we use Algorithm 2. As input, the algorithm receives the vector v, the vector of percentages, p, and the variable *stop*, returning the updated values for v and *stop* as output. The final iteration is reached and *stop* is set to *true* when for all bounded objectives $z_i(x)$, $v_i = \frac{1}{p_i}$, after which the main algorithm stops and the set of the obtained non-dominated solutions, N, is returned. Otherwise, following an ascending order of indexes, $i = 2, \ldots, n_z$, the first $v_i < \frac{1}{p_i}$, is incremented by 1. Furthermore, all $v_{\hat{i}}$, $\hat{i} < i$, which necessarily are equal to $\frac{1}{p_i}$, are reset to 0. This ensures that every possible vector v is considered.

Algorithm 2. Update v.

input: $v, p, stop \in \{true, false\}$;
output: $v, stop$;
if $(v = (1/p_1, \ldots, 1/p_{n_z}))$ **then** $stop \leftarrow true$;
else
    i^* is the first i, such that $v_i < 1/p_i$;
    $v_{i^*} \leftarrow v_{i^*} + 1$;
    $v_{\hat{i}} \leftarrow 0, \; \hat{i} < i^*$;
return$(v, stop)$;

However, not all vectors of lower bounds, ϵ, generated by vectors v, have the potential to obtain new solutions. Accordingly, we use Algorithm 3, which receives as input the vector of lower bounds, ϵ, and the set of already obtained solutions, N, and assesses whether there is no already obtained solution, $z \in N$, that dominates the current lower bounds vector, ϵ. If there is such a solution, z, then the current lower bounds vector, ϵ, does not have the potential to generate a new solution, and the variable *skip* is set to *true*. If $skip = true$, the main algorithm skips the optimisation of the problem of Eq. (42) for the current ϵ (which we denote as P^ϵ), improving its overall efficiency.

Algorithm 3. Skip obtained solutions: *SkipSolutions*().

input: N, ϵ;
output: $skip \in \{true, false\}$;
$skip \leftarrow false$;
for $(z \in N)$ **and** $(skip = false)$ **do**
    if (z dominates ϵ) **then** $skip \leftarrow true$;
return$(skip)$;

Conversely, some combinations can be *a priori* proven to be infeasible. Similarly to the previous algorithm, Algorithm 4 tests if an iteration has the potential to enrich the pool of solutions. Accordingly, it receives as input the vector of lower bounds, ϵ, and the set of lower bounds already identified as infeasible, I. If ϵ dominates any lower bound combination in I it follows that the corresponding optimisation cannot find a feasible solution. Therefore, it can be skipped. Similarly to Algorithm 3, the variable *skip* is set and returned as the output.

Algorithm 4. Skip infeasible models: *SkipInfModels*().

input: I, ϵ;
output: $skip \in \{true, false\}$;
$skip \leftarrow false$;
for $(\bar{\epsilon} \in I)$ **and** $(skip = false)$ **do**
    if (ϵ dominates $\bar{\epsilon}$) **then** $skip \leftarrow true$;
return$(skip)$;

Finally, we can design Algorithm 5, which presents the overall augmented ϵ-constraint method. As input, it receives the problem, P^ϵ, the percentages vector, p, to be considered, and the nadir, z^{nad}, and ideal, z_i^*, points., As output, it returns the set of obtained non-dominated solutions, N.

Algorithm 5. Augmented ϵ-constraint.

input: P^{ϵ}, p, z^{nad}, z^{id};
output: N;
$v \leftarrow (0, \ldots, 0)$;
$I \leftarrow \{\}$;
$N \leftarrow \{\}$;
$stop \leftarrow false$;
while $(stop = false)$ **do**
  for $(i = 2, \ldots, n_z)$ **do**
    $\epsilon_i \leftarrow z_i^{nad} + v_i p_i (z_i^* - z_i^{nad})$;
  end for
  $skip \leftarrow SkipSolutions(N, \epsilon)$;
  if $(skip = false)$ **then**
    $skip \leftarrow SkipInfModels(I, \epsilon)$;
    if $(skip = false)$ **then** $z \leftarrow optimise(P^{\epsilon})$;
      if $(z = -\infty)$ **then** $I \leftarrow I \cup \{\epsilon\}$;
      else $N \leftarrow N \cup \{z\}$;
      end if
    end if
  end if
  $(v, stop) \leftarrow Update(v, p, stop)$;
end while
return(N);

The first step is to set up the initial maximisation, which considers $\epsilon_i = z_i^{nad}$, $i = 2, \ldots, n_z$. Additionally, the obtained solutions set, N, and the known infeasible lower bound combinations set, I, must also be defined. Necessarily, the first iteration always produces a feasible solution to be saved in the set of already obtained solutions, N. Nonetheless, thereupon, before every iteration, Algorithms 3 and 4 assess if the vector of lower bounds, ϵ, generated by vector v, has the potential to find a new solution. If it does, Problem P^{ϵ} is optimised, considering the current vector of lower bounds, ϵ, in which case, if a new solution, z, is found, it is kept in the solutions set, N, otherwise, the lower bounds vector, ϵ, is saved as infeasible in set I. Regardless of the present optimisation being skipped or not, Algorithm 2 determines if the last iteration has already been reached, in which case Algorithm 5 stops and returns the obtained solutions set, N. Otherwise, Algorithm 2 updates v for the next iteration.

3.2.3. Finding the number of schedulable defences

Before computing z^{id}, z^{nad}, and initialising Algorithm 5, we need to set the value for the number of defences that can be scheduled in any given instance, g. For such a purpose, we solve the alternative problem of Eq. (45), which is similar to Problem P^{ϵ}, but without considering Constraint (3), which sets the number of defences that are to be scheduled, Constraints (10)–(26), which define the values for the objectives, and lastly, the objectives themselves (27)–(33).

Conversely, we instead include the objective function of Eq. (45), which maximises the number of scheduled complete committees (thesis defences), computed as the sum of a variable, $y_{jk\ell p}$, which takes the value 1 if a defence, j, is scheduled at a day, k, an hour, ℓ, and a room, p.

$$g = \max z_g(y) = \sum_{j=1}^{n_j} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} y_{jk\ell p} \tag{45}$$

To sum up, the first stage of the procedure is to find the maximum number of thesis defences that can be scheduled for a given instance and set that value as a parameter in the following steps. For the second stage, we compute the ideal point, z^{id}, and the approximate nadir point, z^{nad} through Algorithm 1. Finally, we have all the necessary parameters to initialise the augmented ϵ-constraint method, Algorithm 5. A schematic representation of the whole two-stage procedure is presented in Fig. 1.

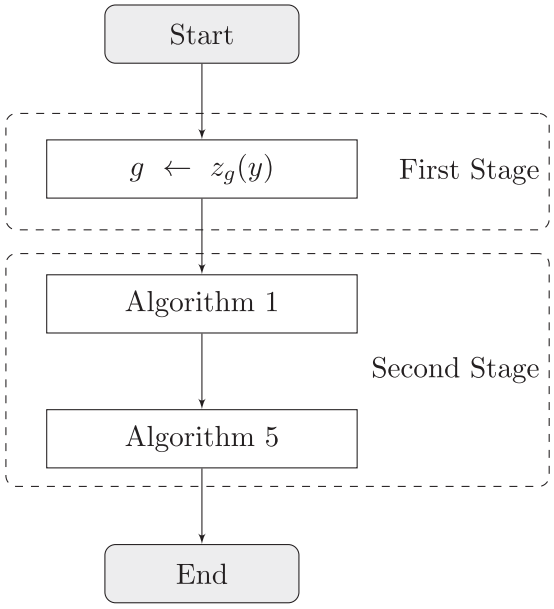

Fig. 1. Full procedure diagram.

4. Case studies

This section addresses adaptions of our model to different thesis scheduling case studies, classified as single defence assignment problems. The first two case studies are adapted from different departments within our university. The third and fourth case studies are taken from the literature and demonstrate how flexible our model is while being applied to problems of different characteristics. They are taken from Huynh et al. (2012) and Tawakkal & Suyanto (2020), respectively. For each, a summary of the problem and an explanation of the parameterisation is presented. For the case studies from our university, the computational results are presented and discussed.

Regarding both hardware and software characteristics for the computational experiments: (1) CPU: Intel(R) Core(TM) i7-8565U CPU @ 1.80 gigahertz 1.99 gigahertz; (2) RAM: 8 gigabyte; (3) Implementation of the algorithms: Python 3.11; (4) Solver: Gurobi 10.0.0.

4.1. Engineering and management department

In this subsection, the case study of the Engineering and Management Department (DEG) of the Instituto Superior Técnico (IST) of the University of Lisbon is addressed.

4.1.1. Summary of the problem

Following IST's policy, the committees are composed of three members, the supervisor, a president, and another member. Moreover, the president must be part of DEG's scientific committee.

Two main concerns have been voiced by the professors of the department regarding the quality of their schedules. Quite often, some of them end up being assigned to a large number of committees. This added workload disturbs their other responsibilities, such as teaching, supervising, and doing their own research. Moreover, the thesis defences of this department are conducted in a *campus* outside of Lisbon. Therefore, most professors have to make additional trips to be present for a defence. Thus, they would prefer

if their defences would be scheduled in as few days as possible. Accordingly, the objectives considered for this case study are promoting a fair workload distribution and minimising the scheduled days.

4.1.2. Model parameterisation

For this case study, there are 47 committee members, n_i, 37 thesis defences, n_j, 3 roles, n_t, 16 days, n_k, 31 hours slots in each day, n_ℓ, and 2 available rooms, n_p. Each defence lasts 1 hour, d, equivalent to 4 hours slots.

The rooms, p, can be considered to be available for every timeslot, (k, ℓ), i.e., $m_{k\ell p} = 1$. The member availability, $l_{ik\ell}$, was gathered by the department's secretary, who was responsible for coordinating the scheduling process with the different committees and ensuring the feasibility of the schedules.

As for the committee eligibility, e_{ijt}, four scenarios are considered. The first considers that the appointed committees are fixed and equal to the ones which were effectively assigned by the decision-makers. The second considers that the president of each defence can be chosen from a set of 11 members of the scientific committee. The third considers that the other member can be chosen from all the available members. The final scenario considers that both the president and the other member can be chosen, respecting the aforementioned rules.

No differentiation was considered between different members, *i.e.*, the individual weight, u_i, was set to 1. The omitted parameters can be disregarded, as they are only necessary for objectives which are not analysed in this case study.

4.1.3. Results discussion

No time limit was set for each iteration. The objective functions are fair workload distribution, z_1, and minimisation of committee days, z_6. The number of equally spaced bounds between, and including, z_6^{nad} and z_6^*, was 20. Thus, at most, 20 solutions can be found for each scenario.

The results for the four scenarios tested with the model are presented in Table 1. This table includes information about the computational time and the number of solutions.

Naturally, in the scenario with a fixed committee, only one non-dominated solution can be found, as the workload objective is only affected by the composition of the committees, which is fixed. The scenarios which allow for more committee combinations invariably had more non-dominated solutions but took more time to solve.

Moreover, the number of solutions and computational times are not only affected by the number of roles to be assigned by the model but also by the number of members available to fill these roles. This is evident when comparing the results for the scenarios where only the presidents and the other members are assigned. In both, the committees have 2 fixed positions, but only 11 members can be chosen as presidents, and 47 can be chosen as other members, leading to a larger variety of possible combinations.

The performance of each solution of the different scenarios regarding the workload objective, z_1, and the committee days objective, z_6, is presented in Fig. 2.

Table 1
Department of engineering and management computational results.

Scenario	Solutions	CPU(s)
Fixed committees	1	1
President	5	20
Other member	14	193
President & other member	15	3530

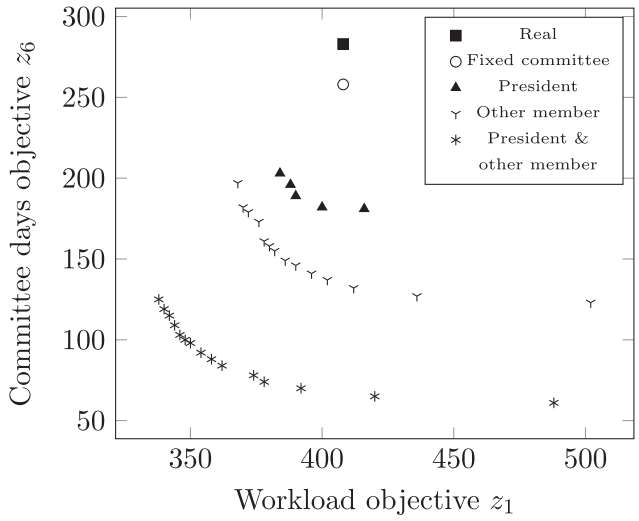

Fig. 2. Performance of each solution found for the four scenarios and the solution found by the decision-makers.

Regarding the workload objective, the solution found by the decision-makers is competitive with some solutions found by the model. Nonetheless, if we consider the performance of the committee days objective, the usefulness of using a model becomes evident.

While a better-performing solution is found for the scenario with fixed committees, the difference in performance is somewhat small. Nonetheless, we hypothesize that if the member availability had been gathered differently, there would have been a greater discrepancy. The data for each committee was obtained through doodles where the supervisor and other member would vote on a couple of different dates suggested by the president. This is a method which facilitates the work of a human scheduler as only a couple of options are considered, but the model is able to handle larger amounts of information and find better solutions.

Nonetheless, the major benefit comes from allowing different committee combinations. It was expected that the workload objective could be greatly improved in this case. However, what is shown is that it is possible to find much better solutions regarding the committee days objective by assigning members with better-matching availability to the same committees. For example, in the scenario where the other member is being assigned, the best-performing solution regarding the committee days objective has a value of $z_6 = 123$. Comparatively, for the scenario where both the president and other member are assigned, the worst-performing solution for this objective is only slightly worse, at $z_6 = 125$.

For a clearer understanding of the trade-offs and meaning of each objective, we now look with more detail into the solution found by the human schedulers ($z_1 = 408,\ z_6 = 283$) and three solutions of the scenario where the president and other member are assigned. Specifically, the solution with the best workload performance ($z_1 = 338,\ z_6 = 125$), an intermediate solution ($z_1 = 354,\ z_2 = 92$), and the solution with the best committee days performance ($z_1 = 488,\ z_6 = 61$). Fig. 3 presents the percentage of members who participate in a certain number of committees, and Fig. 4 presents the percentage of members with a certain number of scheduled days.

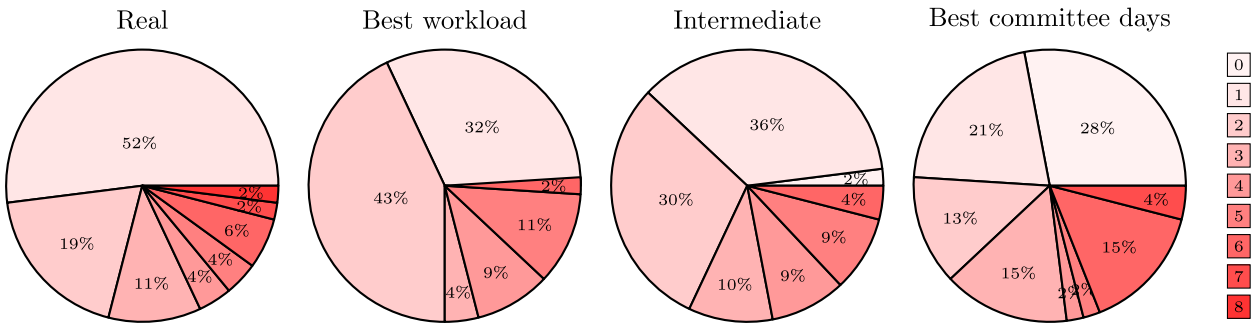

Fig. 3. Percentage of committee members with a number of committees.

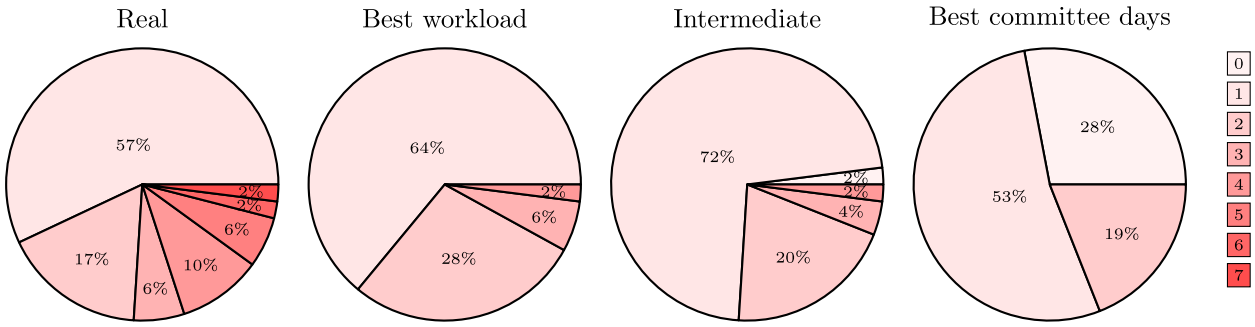

Fig. 4. Percentage of committee members with a number of committee days.

The solution with the fairest workload distribution favours the attribution of 2 committees to most members. Moreover, only the member who was supervising 6 students participated in 6 or more committees. Comparatively, in the solution obtained by the human schedulers, 10% of professors participated in 6 or more committees and in the solution with the best committee days, 19% of professors participated in 6 or more committees.

Regarding the scheduled days, the poor performance of the human schedulers is evident. The best solution regarding this objective considered that only 19% of members had to be present on 2 different days in Taguspark *campus* for a defence. Comparatively, in the schedule found by the decision-makers, 43% of professors had to be present for more than 2 days. Moreover, even in the solution which had the best workload distribution, only 36% of members had defences scheduled for more than 2 days.

There is a clear trade-off between these objectives. On the one hand, solutions which distribute the workload more fairly have to rely on scheduling members for more days to accommodate the more varied committees. On the other hand, the solutions which schedule fewer committee days, distribute more defences to members with matching availability, who can then evaluate them all on a smaller number of days.

4.2. Informatics engineering department

In this subsection, the case study of the Informatics Engineering Department (DEI) of the Instituto Superior Técnico of the University of Lisbon is addressed.

4.2.1. Summary of the problem

DEI also follows IST's standard committee composition, including a president, the supervisor and another member. However, this department's thesis scheduling process has some fundamental differences when compared to DEG's. DEI operates in two different *campi*. Accordingly, some students defend their theses in the Alameda *campus*, in Lisbon, and others in the Taguspark *campus*, where DEG's defences also take place.

Participating in a committee in the Alameda *campus* is usually not as disruptive to the members. Nonetheless, DEI has a considerably larger number of defences occurring in this *campus* when compared to DEG in Taguspark. This results in an increased workload for the presidents. To improve the schedules for members in both *campi*, four objectives are regarded. Specifically, promoting a fair workload distribution, compact schedules, minimising the committee days, and avoiding room changes.

4.2.2. Model parameterisation

In the Alameda *campus* case-study there are 161 members, n_i, 110 defences, 13 days, n_k, and 4 rooms, n_p. In the Taguspark *campus* case-study there are 89 members, n_i, 49 defences, 10 days, n_k, and 2 rooms, n_p. Both consider 3 different roles, n_t, 6 hours slots where a defence can be scheduled, and 2 of mandatory breaks, i.e., $n_\ell = 8$. The break hour slots could have been disregarded if not for the compactness objective. Each defence lasts for a single hour slot, d.

The rooms, p, can be considered to be available for every time-slot, (k, ℓ), i.e., $m_{k\ell p} = 1$. We had access to the president's availability, but not for the remaining members. Thus, for the availability parameter, $l_{ik\ell}$, three scenarios were randomly generated considering the time-slot availability percentage of supervisors and other members: 20% availability, 30% availability, 40% availability.

As for the committee eligibility, e_{ijt}, the supervisors and other members are already pre-assigned, and the model can only choose the president of each defence. In the Alameda *campus* case-study there are 14 members who can be presidents, and in Taguspark there are 9.

Since the presidents are the most affected, the remaining members were assigned an individual weight, $u_i = 0$, in the Alameda *campus*. This means that they are not regarded in the computation of the objectives' values, but their availability is still taken into ac-

count. Contrarily, in the Taguspark *campus*, only members with a single defence pre-assigned were disregarded.

Considering that the hour slots represent 1.5 hours periods, it was considered that any free time between defences should be penalised, i.e., for all members, $b_i = 0$, and $v_{i0} = 1$. For the same reason, the number of hour slots following the end of a defence in which a committee member,*i*, would consider changing rooms problematic, a_i, is also 0 for all members, and $h_{i0} = 1$.

4.2.3. Results discussion

No time limit was set for each iteration. The objective functions are fair workload distribution, z_1, minimisation of non-consecutive assignments, z_4, minimisation of committee days, z_6, and minimisation of room changes z_7. Regarding objective z_7, for every solution found, its value was 0, which means that no member had to change rooms within consecutive time slots. The number of equally spaced bounds between, and including, z_4^{nad} and z_4^*, and z_6^{nad} and z_6^*, was 5. Thus, at most, 25 solutions can be found for each scenario.

The results for the case-studies from both *campi* and considering the three availability scenarios are presented in Table 2. This table includes information about the number of solutions, the percentage of defences that can be scheduled in each of them, and the computational time.

Table 2
Department of informatics engineering results .

Campus	Availability	Solutions	Scheduled defences	CPU (seconds)
Alameda	20%	7	90%	20
Alameda	30%	4	100%	210
Alameda	40%	5	100%	405
Taguspark	20%	3	84%	6
Taguspark	30%	9	100%	20
Taguspark	40%	5	100%	25

Let us note that for both *campi* it stopped being possible to schedule all defences after the availability percentage was dropped to 20%. Accordingly, we can assume that the real percentage would have been higher than that and that the other two scenarios can provide a fair benchmark for analysis of the results which would be achievable through our methodology.

The performance of the schedules obtained by the decision-makers and those obtained by our model for the scenarios were all the defences are scheduled are presented in Fig. 5.

In DEG's case study, we found that while the human schedulers could find workload distributions, z_1, that were somewhat competitive with some non-dominated solutions, the same was not true for the committee days objective, z_6. In the Taguspark case study, the same conclusion can be taken. The values for z_1 in the non-dominated solutions vary from 366 to 386, and the solution found by the schedulers had a performance of 380. Moreover, the value for the committee days objective, z_6, was 130, whereas the solution which performs the worst in this objective had $z_6 = 106$.

However, the same cannot be said for the Alameda case study. Perhaps due to this being a larger problem than the other two, with more than double the number of defences to be scheduled, the human schedulers performed worse in all objectives than the non-dominated solutions found by the
model.

As for the consecutiveness objective, z_4, in both *campi*, the solution found by the human schedulers performed worse than any solution found by the model. Let us note that since all the weights that can influence this objective, u_i and $v_{i\ell}$, are set to 1 or 0, the differences in the performance of two solutions translate the difference in the number of times a member had a defence start without having had another defence scheduled in the previous time-slot. This objective does not vary too much between the non-dominated solutions of the same scenario, with the largest difference being only 3. Nonetheless, for both *campi*, the human scheduler found solutions which performed worse than any non-dominated solution. Specifically, in the Alameda case study, the solution found by the human schedulers had a difference of 15 to the worst-performing non-dominated solution. For the smaller *campus*, the difference was only 3.

4.3. Huynh et al. (2012)'s case study

This work addresses the thesis scheduling problem at the School of Information and Communication Technology, Hanoi University of Science and Technology.

4.3.1. Summary of the problem

Following the policy of most Vietnamese universities (Huynh et al., 2012), the committees are composed of five members. Moreover, while the thesis supervisors cannot be included, they appoint two reviewers who must enter the committee. Additionally, two committee members must be external professors and all committee members must have expertise in the research subject of the thesis. This problem regards all time slots as available for

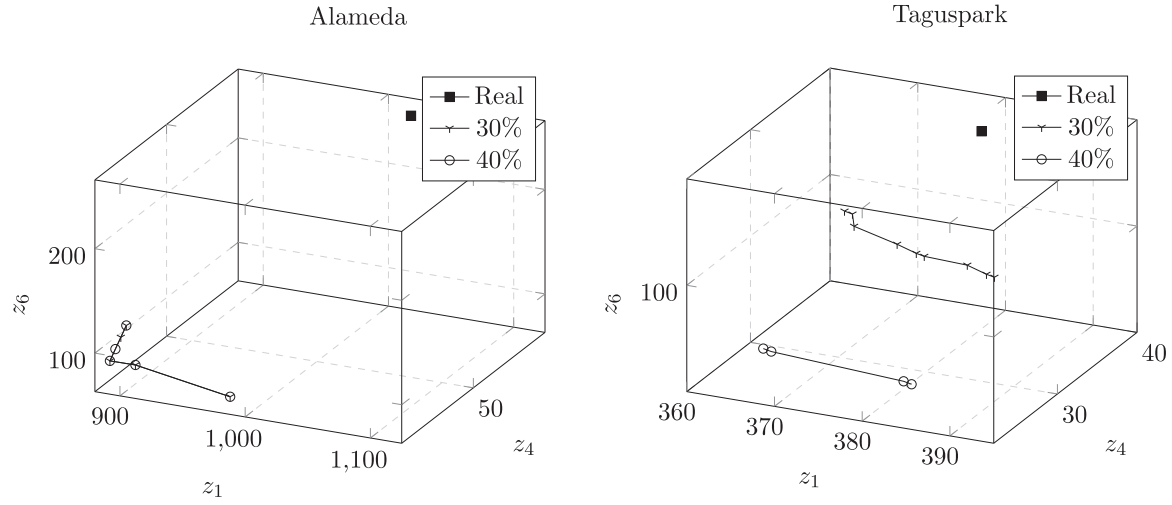

Fig. 5. Performance of each solution found for the department of informatics engineering problem.

both committee members and rooms. However, some should be avoided.

Four objectives are considered. Specifically, minimising committee member assignments in time-slots which should be avoided, maximising committee member suitability regarding the research subject of the thesis, maximising consecutive assignments for committee members, and minimising room changes.

4.3.2. Model parameterisation

The definition of parameters such as the number of defences, n_j, number of members, n_i, duration of the defences, d, research subjects of a committee member, r_{iq}, etc., is simple and we will not explain it here. Moreover, we will not analyse the definition of parameters that reflect the decision-maker's preferences, such as the weight assigned to different members, u_i, or the hour slots after the end of a defence that would be considered inconvenient for a room change, a_i. In fact, only three parameters require some level of attention in this specific case study.

There is no reference to a limit to the number of committees a committee member can be a part of. Hence, the maximum number of defences to be assigned to a committee member, c_i, should be equal to the number of thesis defences, n_j.

Since all time-slots are considered as available, the committee member, $l_{ik\ell}$, and room, $m_{k\ell p}$, availability parameters cannot take the value 0, as this represents unavailability.

The last parameter, e_{ijt}, represents committee member eligibility. It takes the value 1 if a committee member, i, is eligible to perform a role, t, in a certain defence, j, and 0 otherwise. For the two committee members that are appointed by the supervisor there must be two fixed roles, i.e., two roles with only one eligible member. For example, if committee member $i = 1$ is appointed by the supervisor for thesis defence $j = 2$ to perform role $t = 3$, then it follows that $e_{123} = 1$ and $\sum_{i=1}^{n_i} e_{i23} = 1$. Regardless of which roles are fixed, there are two roles that are reserved for external members, meaning that all internal members would be ineligible, i.e., for these roles if a committee member, i, is part of the university staff and a role, t, is reserved for external members, then $e_{ijt} = 0$. And *vice-versa* for the three internal roles.

As for the objective functions, while we do not follow the same formulation, a comparable version of each of them is present in our model. Thus, to adapt our approach to this problem, one could just disregard the objectives that are not considered in this specific case study.

4.4. Tawakkal & Suyanto (2020)'s case study

This work addresses the thesis scheduling problem at the Faculty of Informatics, Telkom University.

4.4.1. Summary of the problem

The committees are composed of three members. Moreover, they include the supervisor and two examiners, who are lecturers at the Faculty of Informatics. To avoid conflicts with their lectures, the committee members must be regarded as unavailable for certain time-slots. However, the rooms are available at all times.

A single objective is considered. Specifically, students who will defend their theses can state their own preferences regarding the time-slots in which they would prefer to do so.

4.4.2. Model parameterisation

Similarly to the previous case study, we will not analyse parameters whose definition is straightforward. Moreover, there is also no limit to the number of committees a committee member can be a part of. Therefore, the definition of the parameter, c_i, follows the same rule as the previous case study.

While the committees are composed of three members, the students must also be considered as part of the committee, so that their preferences can be included as an objective. Thus, the number of committee roles, n_t, is 4. Accordingly, the number of committee members, n_i, must also include the students.

The definition of the committee member time-slot availability and preference parameter, $l_{ik\ell}$, is different between lecturers and students. For a lecturer, i, the parameter can take the value 0, if they are not available, or 1 if they are. For students, i, who are available for all time-slots, the parameter must never take the value 0. Moreover, their preference requests are to be considered. Hence, the parameter should take the value 1, for their preferred time-slots, and a larger value for their non-preferred ones. This differentiation impacts the penalty of choosing a certain time-slot for a defence.

Out of the four roles, t, in these committees, the supervisor and student roles are fixed. Suitably, the committee eligibility parameter, e_{ijt}, follows the rules explained for fixed roles in the previous case study. For the examiner roles, any lecturer that is not the supervisor can be selected. Thus, the parameter takes the value 1 for these lecturers, and 0 for the supervisor and the students.

For the objective function, the time-slot preference objective is the only one which should be regarded.

5. Designing an instance generator for thesis defence scheduling

This section addresses the design of the instance generator for thesis defence scheduling problems that we propose. It starts by presenting the different types of instances that were considered and references the different additional necessary inputs. Moreover, the specific procedures for the random instance generation are also presented, with a focus on the availability parameters, defined based on conditional probabilities.

5.1. Types of instances and other inputs

To test the proposed method (see Section 6), a set of 96 instances, denoted by $p(n_i.n_j.n_t.n_k.n_\ell.n_p.n_q)$, was generated. Let us point out that the following parameters are identical for all of these instances:

1. $n_t = 3$, which is the defined number of roles;
2. $n_k = 15$, which is the defined number of days;
3. $n_\ell = 16$, which is the defined number of hour slots in a day. In these instances each hour slot represents 30 minutes;
4. $n_q = 15$, which is the defined number of research subjects.

Moreover, the instances were divided into six different types by varying the number of committee members, n_i, the number of defences, n_j, and the number of rooms, n_p:

1. Instances of type, $p(25.20.3.15.16.3.15)$: These instances consider 25 committee members ($n_i = 25$), 20 defences ($n_j = 20$) and 3 rooms ($n_p = 3$), instances (1)–(16).
2. Instances of type, $p(25.20.3.15.16.4.15)$: These instances consider 25 committee members ($n_i = 25$), 20 defences ($n_j = 20$) and 4 rooms ($n_p = 4$), instances (17)–(32).
3. Instances of type, $p(38.30.3.15.16.3.15)$: These instances consider 38 committee members ($n_i = 38$), 30 defences ($n_j = 30$) and 3 rooms ($n_p = 3$), instances (33)–(48).
4. Instances of type, $p(38.30.3.15.16.4.15)$: These instances consider 38 committee members ($n_i = 38$), 30 defences ($n_j = 30$) and 4 rooms ($n_p = 4$), instances (49)–(64).
5. Instances of type, $p(50.40.3.15.16.3.15)$: These instances consider 50 committee members ($n_i = 50$), 40 defences ($n_j = 40$) and 3 rooms ($n_p = 3$), instances (65)–(80).

6. Instances of type, $p(50.40.3.15.16.4.15)$: These instances consider 50 committee members ($n_i = 50$), 40 defences ($n_j = 40$) and 4 rooms ($n_p = 4$), instances (81)–(96).

As for the remaining parameters, some were considered the same for every instance, specifically:

1. $d = 2$, which is the duration of a thesis defence, in the number of time slots;
2. $p(u_i = 1) = 0.7,\ p(u_i = 2) = 0.3$, which are the probabilities of a committee member i to be assigned to a certain individual weight u_i;
3. $c_i = 0.5n_i$, which is the maximum number of allowed defences *per* committee member i;
4. $\sum_{q=1}^{n_q} r_{iq} = 3$, which is the number of research subjects q for committee members i;
5. $\sum_{q=1}^{n_q} t_{jq} = 3$, which is the number of research subjects q for defences j.

Additionally, for each type of instance, we generated sixteen different randomised instances while varying some data inputs corresponding to the remaining parameters of our model:

1. $t = 1$ and $t = 2$, or $t = 2$: These are the fixed roles in e_{ijt}. If a role, t, is fixed, it means that, for each defence, j, there is only one eligible committee member, i, it can be assigned to. Regardless of a role being fixed, there is an overall set of committee members that can be assigned to that role.
2. $p(l_{ik\ell} = 0) = 0.78,\ p(l_{ik\ell} = 0) = 0.82,$ or $p(l_{ik\ell} = 0) = 0.86$: These are the unavailability percentages, $p(l_{ik\ell} = 0)$, for the committee member time slot preference and availability parameter, $l_{ik\ell}$, which are defined as the percentage of $l_{ik\ell} = 0$. The percentages for the instances with 2 fixed roles were 0.78 or 0.82. Conversely, for the instances with a single fixed role, they were 0.82 or 0.86. This differentiation between instances with 1 or 2 fixed roles was necessary due to increased computational complexity.
3. $p(m_{k\ell p} = 0) = 0.8$ or $p(m_{k\ell p} = 0) = 0.86$, for $m_{k\ell p}$: These are the unavailability percentages, $p(m_{k\ell p} = 0)$, for the room availability parameter, $m_{k\ell p}$, which are defined as the percentage of $m_{k\ell p} = 0$.
4. $p(v_{i\ell} = [1]) = 0.7,\ p(v_{i\ell} = [2, 1]) = 0.3$ or $p(v_{i\ell} = [1]) = 0.8,\ p(v_{i\ell} = [2, 1]) = 0.2$: These are the probabilities of a committee member, i, being assigned values [1] or [2,1] for the compactness preference parameter, $v_{i\ell}$, which will affect the big-M upper bounds, n_{v_i}.
5. $p(h_{i\ell} = [1]) = 0.7,\ p(h_{i\ell} = [2, 1]) = 0.3$ or $p(h_{i\ell} = [1]) = 0.8,\ p(h_{i\ell} = [2, 1]) = 0.2$: These are the probabilities of a committee member, i, being assigned values [1] or [2,1] for the room change penalty parameter, $h_{i\ell}$, which will affect the big-M upper bounds, n_{h_i}.

A diagram summarising this instance generation procedure is presented in Fig. 6.

The generation of most parameters is done through simple probability-based random choices. Thus, we do not explain them in detail. However, the availability parameters follow some additional rules, which are explained in the next subsection.

5.2. Availability parameters generation for committee members and rooms

In real-world thesis defence scheduling applications, the availability periods for committee members and rooms usually occur in blocks, between lectures, before the first lecture of a given day, or after all the daily assignments. Algorithm 6 was designed to replicate such behaviour. The algorithm can be described as a Markov chain, which will help estimate the probability distributions of the different possible values for the availability parameters, $l_{ik\ell}$ and $m_{k\ell p}$. For simplification, when we refer to an individual τ, it can represent a committee member or a room.

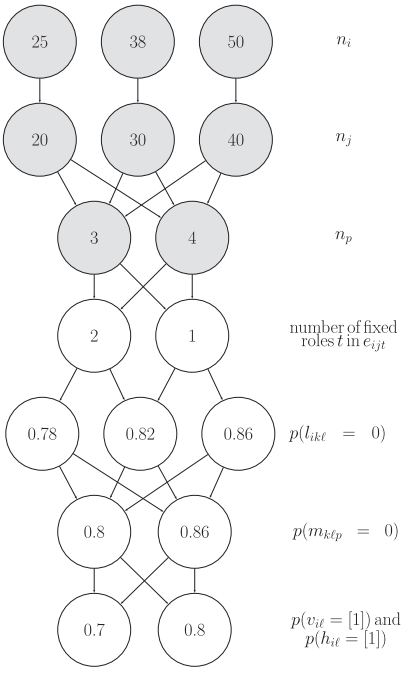

Fig. 6. Generated instances diagram.

Algorithm 6. Generate availability parameters.

input: n_i,n_k, n_ℓ, $p(\alpha|\alpha)$,Δ,d;
output: $l_{ik\ell},\ i = 1, \ldots, n_n,\ k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell$;
for $(i = 1, \ldots, n_i)$ **do**
  for $(k = 1, \ldots, n_k)$ **do**
    $l_{ik0} \leftarrow 0$;
    for $(\ell = 1, \ldots, n_\ell + \Delta)$ **do**
      if $(l_{ik\overline{\ell}} = 0_e)$; **then**
        $l_{ik\ell} \leftarrow 0$;
      else
        $l_{ik\ell} \leftarrow random(\alpha),\ p(\alpha) = p(\alpha|\overline{\alpha}),\ l_{ik\overline{\ell}} = \overline{\alpha}$;
      end if
    end for
    for $(\ell = 0, \ldots, \Delta)$ **do**
      $delete(l_{ik\ell})$;
return$(l_{ik\ell},\ i = 1, \ldots, n_n,\ k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell)$;

To generate the availability parameters in a manner which represents their real-world behaviour, we defined the probability of an individual, τ, to have a certain availability status, α, at any day, k, and hour, ℓ, (time slot (k, ℓ)), as conditional on its status, $\overline{\alpha}$, in a previous time slot, $(k, \overline{\ell})$, for $\overline{\ell} = \ell - 1$. Let us note that, when we mention a conditional probability, $p(\alpha|\overline{\alpha})$, what we are referring to is the probability of having $l_{ik\ell} = \alpha$ or $m_{k\ell p} = \alpha$, given that $l_{ik\overline{\ell}} = \overline{\alpha}$ or $m_{k\overline{\ell}p} = \overline{\alpha}$.

Algorithm 6 receives the following notable inputs:

1. Δ: This is the duration of the initial warm-up period for each day, k, within which the generated parameters will be disregarded. This is important as it will allow the Markov chain to reach a steady state.
2. $p(\alpha|\alpha)$: These are the probabilities of an $l_{ik\ell} = \alpha$ or $m_{k\ell p} = \alpha$ to remain unchanged between ℓ and $\overline{\ell},\ \ell = \overline{\ell} + 1$.
3. $p(\alpha|\overline{\alpha})$: These are the probabilities of an $l_{ik\ell} = \alpha$ or $m_{k\ell p} = \alpha$ changing from a state, $\overline{\alpha}$, to another, α, between ℓ and

$\overline{\ell}$, $\ell = \overline{\ell} + 1$. This is not an input *per se*, but computed through Eq. (46), based on the input values, $p(\alpha|\alpha)$.

Computing the remaining probabilities, $p(\alpha|\overline{\alpha})$, through Eq. (46) promotes the proportionality between $p(\alpha|\overline{\alpha})$ and all other $p(\hat{\alpha}|\overline{\alpha})$, based on their conditional probabilities, $p(\alpha|\alpha)$. Moreover, this equation also guarantees that the sum of these probabilities is always equal to 1.

$$p(\alpha|\overline{\alpha}) = p(\alpha|\alpha)\left(\sum_{\hat{\alpha}=0}^{n_\alpha} p(\hat{\alpha}|\hat{\alpha})\right)^{-1}(1 - p(\overline{\alpha}|\overline{\alpha})),\ \alpha, \hat{\alpha} \neq \overline{\alpha} \quad (46)$$

For the committee member availability parameter, $l_{ik\ell}$, three values were considered for assignment, $l_{ik\ell} = 0$, representing unavailability, $l_{ik\ell} = 1$, representing preferred time slots, and, $l_{ik\ell} = 2$, representing less preferred time slots. The room availability parameter, $m_{k\ell p}$, was defined as binary, hence, the possible values are, $m_{k\ell p} = 0$, representing unavailability, and, $m_{k\ell p} = 1$, representing availability. The inputs, $p(\alpha|\alpha)$, per each defined unavailability percentage, are presented in Table 3 for the committee member availability parameter, $p(l_{ik\ell} = 0)$, and in Table 4 for the room availability parameter, $p(m_{k\ell p} = 0)$.

The first step of the algorithm is to define how the values for the first hour slot, $\ell = 1$, for an individual, τ, and day, k, l_{ik1} and m_{k1p}, are generated. We determined that, for each parameter regarding $\ell = 1$, the conditional probabilities of it being assigned a status α, $p(\alpha|\overline{\alpha})$, are to consider $l_{ik0} = 0$ or $m_{k0p} = 0$, i.e., $\overline{\alpha} = 0$. Nonetheless, this creates a biased availability distribution for the first hour slots for each day. However, we want to ensure that the probability, $p(\alpha)$, of an hour slot, ℓ, being assigned availability status, α, is independent of the hour slot, ℓ. Thus, for every committee member or room, and day, we generate an excess of parameter values to uniformise their distribution. Then we disregard the initial excess values, i.e., we consider a warm-up period, Δ. In our experiments, we used a $\Delta = 40$ *per* committee member or room, and day, which proved sufficient to eliminate the initial value's effect.

Then, we generate each availability parameter value in sequential order, based on the presented conditional probabilities.

Nonetheless, there is an exception where these input probabilities do not apply. It occurs after an individual, τ, which was available to start an assignment in a time slot, $(k, \overline{\ell})$, stops being available to do so at the next time slot, (k, ℓ), for $\ell = \overline{\ell} + 1$. Let us note that for an individual τ to be able to start an assignment at time slot $(k, \overline{\ell})$, its effective available time must extend up until $(k, \hat{\ell})$, for $\hat{\ell} = \ell + d$, as, otherwise, the individual, τ, could not be present for the whole duration, d, of a defence. Thus, we defined that when a change from availability to unavailability occurs, the individual, τ, must be unavailable for at least a long enough time, such that if they were scheduled for their last available time slot before the unavailability block, and for the first available one after its occurrence, the individual, τ, would have at least one time slot of effectively unassigned time. Therefore, whenever such a change occurs, $d - 1$ unavailable time slots are automatically added, ensuring the rule above is respected.

An illustrative example of the time slot preference and availability generated by this method is presented in Fig. 7. Let us note that, since we considered $d = 2$, the exceptional addition was of a single time slot (i.e., $d - 1$). For example, if we had used $d = 4$ instead, 3 time slots would have been added.

For additional theoretical background on the concept of Markov chains and an explanation of how they can be used to predict the availability percentages the reader is directed to Appendix B.

Table 3
Conditional probabilities $p(\alpha|\alpha)$*per* generated unavailability percentage $p(l_{ik\ell} = 0)$ for parameter $l_{ik\ell}$.

	$p(l_{ik\ell} = 0)$		
$p(\alpha\|\alpha)$	0.78	0.82	0.86
$p(0\|0)$	0.95	0.95	0.95
$p(1\|1)$	0.7	0.63	0.55
$p(2\|2)$	0.7	0.63	0.55

Table 4
Conditional probabilities $p(\alpha|\alpha)$ *per* generated unavailability percentage $p(m_{k\ell p} = 0)$ for parameter $m_{k\ell p}$.

	$p(m_{k\ell p} = 0)$	
$p(\alpha\|\alpha)$	0.8	0.86
$p(0\|0)$	0.95	0.95
$p(1\|1)$	0.7	0.8

6. Computational experiments, results, and some comments

This section addresses the computational experiments made on the generated instances. It starts by specifying some essential practical aspects, and then the analysis of the results of the computational experiments is presented.

6.1. Practical aspects

There are several practical aspects we need to consider regarding hardware and software, iteration time limits, parameters of the augmented ϵ-constraint method, and the display of the computational experiments.

6.1.1. Hardware and software

Regarding both hardware and software characteristics: (1) CPU: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40 gigahertz 2.42 gigahertz; (2) RAM: 15.8 gigabyte;(3) Implementation of the algorithms: Python 3.10; (4) Solver: Gurobi 9.5.0.

6.1.2. Time limits

Regarding the time limits set for each step of the procedure: (1) Finding parameter g: 30 minutes; (2) Algorithm 1: 2 hours, equally divided between the seven objectives; (3) Algorithm 5: For each iteration, 12 hours minus the time used for the previous steps and iterations in the procedure, divided by the remaining number of iterations.

6.1.3. Parameters of the algorithms

Regarding the parameters of the augmented ϵ-constraint method and other algorithms: (1) Objective to be fully considered in z^{ϵ}: z_1; (2) Bounded objectives: z_3, z_4; (3) Number of equally spaced bounds between, and including, z_i^{nad} and z_i^*: 10; (4) Continuous objective z_2: While z_2 is defined as continuous, it was rounded up to the nearest integer, as, otherwise, Algorithm 1 could not be used to assess its optimum accurately.

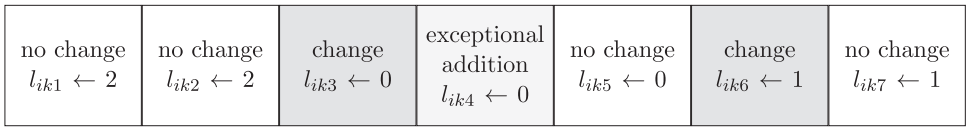

Fig. 7. Committee member time slot preference and availability parameter $l_{ik\ell}$ generation illustrative diagram.

6.1.4. Computational experiments display

Regarding the computational experiments, displayed in the Appendix, in Tables C.1, C.2 and C.3, for small ($n_j = 20$), medium ($n_j = 30$) and large ($n_j = 40$) instances, respectively:

1. Table row: Presents the number of the instance, the identification of the type of instance, the corresponding input data and the outputs.
2. Types of output: Presents the number of non-dominated solutions found, $|N|$, the number of infeasible iterations, $|I|$, the number of skipped non-dominated solutions, $skip^N$, the number of skipped infeasible solutions, $skip^I$, the number of solutions found that were not proven as non-dominated due to a time limit being reached, $time^N$, the number of iterations that were stopped before a solution was found due to a time limit being reached, $time^I$, the number of defences that can be scheduled, g, and the CPU time required.

6.2. Computational experiments

This subsection addresses the analysis of the computational experiments. For such a purpose, several aspects must be taken into account. Specifically, the computational performance, number of schedulable defences, and the type of iteration distribution. Moreover, some concluding comments are also made.

6.2.1. Computational performance

Regarding computational performance, which can be assessed by the time an instance takes to solve and the number of iterations that were stopped due to time limit conditions being reached, the following remarks can be made:

1. Increases in the size of the instance, for the considered instance types $p(n_i.n_j.n_t.n_k.n_\ell.n_p.n_q)$, invariably lead to an increase in computational complexity;
2. For the considered ranges, a decrease in the number of fixed roles in e_{ijt} or in the unavailability percentages for the availability parameters, $l_{ik\ell}$ and $m_{k\ell p}$, also lead to notable increases in the CPU times;
3. Conversely, the variation on the percentages for each big-M upper bound, $v_{i\ell}$ and $h_{i\ell}$, did not produce such a unidimensional variation in computational complexity, with some instance types being solved more efficiently when the smaller upper bound is more frequent, and other instances seemingly showing the opposite trend.

For the first two points, the mentioned parameter variations increase the number of possible feasible variable combinations. Thus, making the instances more challenging to solve. For the irregular behaviour presented in the last remark, the explanation might be that, occasionally, the increase in the number of committee members that are assigned the parameter values $v_{i\ell} = [2, 1]$ and $h_{i\ell} = [2, 1]$, instead of $v_{i\ell} = [1]$ and $h_{i\ell} = [1]$, is potentially reducing solution symmetries. This effect surpasses the impact of the weaker linear relaxations induced by larger big-M upper bounds.

6.2.2. Number of schedulable defences

Regarding the number of schedulable defences, logically, increasing the number of committee members and defences leads to more defences being scheduled. Additionally, the following remarks on the percentage of schedulable defences can be made:

1. Increasing the number of available rooms improves the number of schedulable defences;
2. Reducing the number of fixed roles in e_{ijt} or the unavailability percentages for the availability parameters, $l_{ik\ell}$ and $m_{k\ell p}$, promoted higher schedulability percentages.

These points show that these parameters, which are involved in the model, affect the probability that there is a suitable time slot for the scheduling of a given defence. Moreover, an increase in this probability improves the number of schedulable defences. Let us note that the same parameters that positively affect the number of schedulable defences also negatively impact the time it takes to solve the respective instance. This is explained by the increase in the number of possible assignment combinations promoted by the variation of these parameters. Conversely, the big-M upper bound distribution variation does not impact the number of schedulable defences, which might indicate why its effect on the computational efficiency is not as streamlined as it is for the other considered parameters.

6.2.3. Type of iteration distribution

Regarding the type of iteration distribution, while it is harder to take conclusions considering the more computationally complex instances, which had iterations stop due to the set CPU time limit being reached, the following remarks can be made for the instances where these time-related stop conditions were not met:

1. The distribution of feasible and infeasible iterations, $|N| + skip^N$ and $|I| + skip^I$, respectively, shows a slight variation, with most instances having between 75 and 85 feasible iterations out of 100;
2. Instances that had more effective iterations took longer to solve when compared to similarly-sized instances, which had more skipped iterations. Nonetheless, their time *per* effective iteration is not considerably different;
3. The number of different non-dominated solutions shows a slight positive correlation with the percentage of schedulable defences.

When looking at the results of the computational experiments, some outlier instances occasionally pop out, which take longer to solve when compared to the instances that are most similar to them. Nonetheless, these usually occur due to, for some reason, the outlier instances having a relatively high number of different non-dominated solutions. This leads to fewer skipped iterations and, thus, longer CPU times. Nonetheless, this does not mean that each iteration is harder to solve, just that there are more effective iterations. Moreover, besides these occasional outlier instances, the apparent rule is that there are more different solutions in instances with a higher schedulability percentage. Accordingly, when there are more assignments, more committee members are involved. This seems to promote more possible trade-offs between the different considered objectives.

6.2.4. Concluding comments

The proposed method showed a remarkable capacity for finding the number of schedulable defences, g, always reaching optimality in the first stage. This is a helpful step for real-world problems where the decision-makers are not *a priori* certain that all defences are schedulable. Furthermore, finding this parameter means that at least one feasible solution is always found.

Moreover, for almost every instance with two fixed roles, the method could map the desired subset of the Pareto front without reaching any of the defined time limits. Conversely, the same cannot be said for instances with a single fixed role, specifically for the medium and large instances, which had several iterations being stopped due to time limits. Still, we must note that, even for these instances, the method still returns several feasible solutions and some non-dominated different ones. Thus, if applied to larger real-world instances, several different options would still be presented to the decision-maker, even if their optimality could not be proven.

The number of fixed roles was the parameter whose variation had the highest impact on the computational performance and the number of schedulable defences. While there is no certainty that this remark will hold for different parameter ranges, it can be seen as an indicator of the effect this scheduling decision can have on the best thesis defence scheduling method to be employed by universities with different policies. For organisations where the scheduling process occurs synchronously with the assignment of the committees, hence having less fixed roles, it might be easier to schedule all the defences, but using deterministic methods might not be suitable due to the increased computational complexity. Conversely, for organisations where the scheduling process occurs separately from the assignment of the committees, hence having more fixed roles, finding an available time slot for each defence might be more challenging. Still, it is easier for a deterministic solution to their instance to be found within a reasonable time frame.

7. Conclusions

In this paper, we propose a MOMILP model for the single defence assignment class of the thesis defence scheduling problems. This problem consists of assigning a committee, time slot, and room to each thesis defence. Moreover, it is subject to a set of constraints that define the feasible region. Each of these constraints can be placed in one of three groups of constraints, specifically, scheduling complete committees, committee composition, or committee member and room availability. Moreover, the quality of the schedules can be assessed under two points of view, rendered operational through several criteria, specifically, committee assignment and schedule quality. To tackle the multi-objective nature of the problem, we implement an adaptation of the augmented ϵ-constraint method, which allows for the mapping of a subset of the Pareto front, presenting the decision-makers with a variable number of different non-dominated solutions, while employing iteration skipping mechanisms to improve the overall computational efficiency.

The thesis defence scheduling literature has primarily been focused on solving the problem at the authors' universities. Conversely, one of the main contributions of our work is that we formalise the problem in a manner that is easier to adapt to institutions with different regulations. Furthermore, besides offering a new take on the formulation of the most common objectives in thesis defence scheduling, we also regard some additional ones not previously considered. We also account for the possibility that, in instances where not all defences are schedulable, it can be valuable for the decision-makers to be presented with an "incomplete" schedule so that they have access to more information and may better assess how to proceed in solving the problem.

With this work, we aim to promote the study of a fundamental academic scheduling problem, which is remarkably underrepresented in the literature for how impactful and time-consuming it can be. Thus, we also present a novel random instance generator that can help to provide instances for future research.

Two case studies based on instances from different departments within our university are presented. The model finds solutions which dominate the solutions obtained by the human schedulers in every objective. Two case studies illustrating how our model can be parameterised for solving instances from literature are also explored.

The computational experiments proved that the first stage introduces a critical step for solving thesis defence scheduling instances where it is not known that every defence is schedulable. Moreover, even for larger instances, the method was always returned several solutions. Furthermore, for smaller instances or those with two fixed roles, the optimality of each returned solution is practically always proven.

While we attempt to include most concerns and policies addressed in the literature and inclusively consider new ones, it is entirely possible and expected that some additional regulations and preferences not yet discussed but present in other universities might not have been covered by our work. Moreover, unlike other academic scheduling problems, the development and improvement of novel solution methods and algorithms are lacking for the thesis scheduling problem. Thus, we believe this to be a promising new field for future research, and that new findings can help not only the optimisation of thesis scheduling in universities but also apply to other scheduling problems, such as course timetabling or exam scheduling.

Declaration of Competing Interest

Authors declare that they have no conflict of interest.

Acknowledgments

João Almeida acknowledges the help of Inês Marques, who, along with Daniel Rebelo dos Santos, introduced him to the topic of master's thesis defence scheduling problems. José Rui Figueira acknowledges the support by national funds through FCT (Fundação para a Ciência e a Tecnologia), under DOME Project (reference: PTDC/CCI-COM/31198/2017). All the authors acknowledge the Portuguese national funds through the FCT - Foundation for Science and Technology, I.P., the first three authors under project UIDB/00097/2020 and the last author under projects UIDB/50021/2020 and LA/P/0078/2020.

Appendix A. Mathematical model detailed description

A1. Constraints

The necessary constraints to model the feasible region of the problem are presented in this subsection. They fall under four categories. The first concerns the scheduling of complete committees or thesis defences. The second regards the respect for committee member assignment rules. The third guarantees that committee members and rooms are available for their corresponding assignments. Finally, the fourth defines the values for several auxiliary variables present in the objective functions.

1. *Scheduling complete committees.* These constraints define a complete committee and ensure that every schedulable defence is assigned one.
   (a) *Complete committee definition.* A complete committee is a set of n_t assignments for a defence, j, all with a different appointed role, t, in the same slot, (day k, hour ℓ, and room p). Moreover, for a defence to occur, it must have such a committee assigned to it. Conversely, no assignment that is not incorporated into one can exist, as it would occupy a slot that is not being used. Likewise, this constraint defines the auxiliary variable $y_{jk\ell p}$, which takes the value 1 if a defence, j, has a complete committee assigned to it and 0 otherwise. Thus, the left-hand-side of the equation can also only take binary values. The sum on the mentioned side represents the number of committee members, i, assigned to a defence, j, to perform a role, t, on a given slot, (k, ℓ, p). Consequently, since this sum can be at most 1, each role, t, can only be filled once in a complete committee. Moreover, evidently, $y_{jk\ell p}$ can only take one value for a given defence, j, and slot, (k, ℓ, p), ergo, the left-hand-side of the equality can also only take one value for the same defence, j, slot, (k, ℓ, p) and for

every role, $t = 1, \ldots, n_t$, meaning that for $t \neq \bar{t}$, the following equality must be verified: $\sum_{i=1}^{n_i} x_{ijtk\ell p} = \sum_{i=1}^{n_i} x_{ij\bar{t}k\ell p}$. Accordingly, if a role, t, is assigned to a defence, j, any other role, $\bar{t}$, must also be assigned. Interchangeably, if any role, t, is not assigned, any other role, $\bar{t}$, must also not be assigned. Therefore, either a defence is assigned to a complete committee or no assignments for such a defence can occur.

$$\sum_{i=1}^{n_i} x_{ijtk\ell p} = y_{jk\ell p}, \quad j = 1, \ldots, n_j,\ t = 1, \ldots, n_t$$
$$k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell,\ p = 1, \ldots, n_p \tag{A.1}$$

(b) *Single committee assignment*. If a defence, j, can be scheduled, it should only be assigned one committee and appointed one slot, (day k, hour ℓ, and room p). Thus, in this constraint, we state that for a defence, j, the number of complete committees assigned to it is less or equal to 1.

$$\sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} y_{jk\ell p} \leqslant 1, \quad j = 1, \ldots, n_j \tag{A.2}$$

(c) *Complete committees (thesis defences) to be scheduled*. In each instance of the thesis defence scheduling problem, a defined number of committees (thesis defences) can be assigned and scheduled, denoted by g. If one knows that all the defences can be scheduled, then this number is the number of defences, i.e., $g = n_j$. However, some defences may not be schedulable due to conflicting committee member availabilities, lack of enough eligible committee members, lack of rooms, or others. In such cases, finding the value for g becomes an indispensable part of the problem. In this constraint, assuming the value of g is already known, we enforce the number of assigned complete committees (thesis defences) as the number of schedulable complete committees, g.

$$\sum_{j=1}^{n_j} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} y_{jk\ell p} = g \tag{A.3}$$

2. *Committee Composition.* These constraints ensure the eligibility of the committee members to perform their assignments.

(a) *Committee member eligibility*. Different universities and their departments have distinct regulations for the eligibility of committee members, i, to perform specific roles, t, within each committee for a defence, j. Thus, we do not attempt to include such rules within our model. Conversely, we aggregate them in a parameter, e_{ijt}, which takes the value 1 if a committee member, i, is eligible to perform a role, t, in a defence, j, and 0 otherwise. In this constraint, we state that if a given committee member, i, is non-eligible to perform a role, t, in the committee of a defence, j, that is, if $e_{ijt} = 0$, then no assignment that involves such a combination can occur, that is, the left-hand-side of the equation must also be 0. Contrarily, if such a combination is possible, that is, if $e_{ijt} = 1$, then the equality still holds, as, logically, a committee member, i, can be assigned at most once to a given defence, j.

$$\sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant e_{ijt},$$
$$i = 1, \ldots, n_i,\ j = 1, \ldots, n_j,\ t = 1, \ldots, n_t \tag{A.4}$$

(b) *Maximum number of committees assigned to a committee member*. This committee member eligibility requirement cannot be represented by the eligibility parameter, e_{ijt}. Thus, we included it as another constraint. In cases where there is no such regulation, the value for the maximum number of committees assigned to a committee member, i, represented by c_i, is equal to the number of thesis defences, that is, $c_i = n_j$. In this constraint, we ensure that the sum of the assignments, which occur when $x_{ijtk\ell p} = 1$, for a committee member, i, does not exceed the maximum allowed number of committees, c_i, for that committee member.

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant c_i, \quad i = 1, \ldots, n_i \tag{A.5}$$

3. *Committee member and room availability*. These constraints guarantee that committee members and rooms are available for each assignment.

(a) *Committee member time slot availability*. Committee members have different obligations other than attending thesis defences. Consequently, they are not available to be assigned to every time slot. The committee member availability parameter, $l_{ik\ell}$, takes a value greater than or equal to 1 if a committee member, i, is available to be assigned at a day, k, and an hour, ℓ, and 0 if they are not. In this, we state that if a given committee member, i, is not available at a day, k, and an hour, ℓ, that is, if $l_{ik\ell} = 0$, then no assignment that involves such a combination can occur, that is, the left-hand-side of the equation must also be 0. Contrarily, if such a combination is possible, that is, if $l_{ik\ell} \geqslant 1$, then the equality still holds, as, logically, a committee member, i, can only be given at most one assignment at a particular time slot, (k, ℓ).

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant l_{ik\ell},$$
$$i = 1, \ldots, n_i,\ k = 1, \ldots, n_k,\ \ell = 1, \ldots, n_\ell \tag{A.6}$$

(b) *Committee member assignment juxtaposition*. A committee member, i, cannot be assigned to more than one defence, j, starting at a day, k, and an hour, ℓ. Moreover, that committee member is also unavailable to attend any other defence that begins at any given point before the end of such a defence, j. In other words, until the hour $\ell + d$ is reached on the same day, k. Thus, in this constraint, we ensure that if there is an assignment, $x_{ijtk\bar{\ell}p} = 1$, for a committee member, i, in a day, k, at an hour, $\bar{\ell}$, there cannot be any other assignment for the same committee member, in an hour that occurs before the end of the previous defence, that is, in any hour between and including $\bar{\ell}$ and $\bar{\ell} + d - 1$. Additionally, this constraint also ensures that a committee member is not assigned more than one role, t, in a defence, j, as that would mean that said committee member would have two different assignments in the same time slot, (k, ℓ).

$$\sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{\ell=\bar{\ell}}^{\bar{\ell}+d-1} \sum_{p=1}^{n_p} x_{ijtk\ell p} \leqslant 1, \quad i = 1, \ldots, n_i,$$
$$k = 1, \ldots, n_k,\ \bar{\ell} = 1, \ldots, n_\ell - d + 1 \tag{A.7}$$

(c) *Room time slot availability*. A room's purpose might not just be hosting thesis defences. Thus, it is natural that it happens to be booked for any other event at some point. The room availability parameter, $m_{k\ell p}$, takes the value 1 if a room, p, is available to host a defence at a day, k, and an hour, ℓ, and 0 otherwise. In this constraint, we state that, for a given slot, (k, ℓ, p), the sum of its assigned complete committees, $y_{jk\ell p} = 1$, always takes a value lower or equal to that of $m_{k\ell p}$. Accordingly, whenever a room is unavailable, that is, $m_{k\ell p} = 0$, it cannot host any defence, and this sum is correctly set to 0. Moreover, if the room is available, that

is, $m_{k\ell p} = 1$, then, at most, one defence, j, can be assigned such a slot, (k, ℓ, p).

$$\sum_{j=1}^{n_j} y_{jk\ell p} \leqslant m_{k\ell p}, \quad k = 1, \ldots, n_k, \\ \ell = 1, \ldots, n_\ell, \ p = 1, \ldots, n_p \tag{A.8}$$

(d) *Room capacity*. In our formulation, we considered that a room could not hold more than one defence at a time. Variable $y_{jk\ell p}$ takes the value 1 if a defence, j, is assigned to a day, k, an hour, ℓ, and a room, p. Consequently, for the same day, room and between the start of defence j, at hour ℓ, and its end, at hour $\ell + d$, the point at which the room can be scheduled again, there cannot be more than one $y_{jk\ell p} = 1$. In this constraint, this is achieved by stating that, for any day, k, hours between, and including, ℓ to $\ell + d - 1$, and a room, p, the sum of the values of the variable $y_{jk\ell p}$ must be less or equal to 1.

$$\sum_{j=1}^{n_j} \sum_{\ell=\bar{\ell}}^{\bar{\ell}+d-1} y_{jk\ell p} \leqslant 1, \quad k = 1, \ldots, n_k, \ \bar{\ell} = 1, \ldots, n_\ell \\ - d + 1, \ p = 1, \ldots, n_p \tag{A.9}$$

4. *Objective functions measures.* These constraints define the values for the auxiliary variables necessary for some of the objective functions.

(a) *Research subject coverage definition.* For a defence, j, research subject coverage is the percentage of its studied research subject covered by the areas of expertise of its committee members. To implement such an objective, we first need to define an auxiliary binary variable, s_{ijq}, which takes the value 1 if a defence, j, which studies a research subject, q, that is, $\bar{t}_{jq} = 1$, has a number, i, of committee members assigned to its committee, who have said subject as one of their areas of expertise. Let us note that, whenever a committee member, i, studies a research subject, q, then $r_{iq} = 1$. The value for such a variable is defined in Constraint (10) by stating that the product of s_{ijq} by the number, i, of committee members assigned to that defence is equal to the sum of the assignments, $x_{ijtk\ell p} = 1$, where the research subject, q, is in both the studied subjects of the defence and the areas of expertise of the committee member, that is $r_{iq}\bar{t}_{jq} = 1$. Furthermore, a defence, j, cannot be assigned to more than one number of committee members with a research subject, q, in common with it. Thus, with Constraint (11), we ensure that this value is unique for each combination of defence, j, and research subject, q.

$$\sum_{i=0}^{n_i} i s_{ijq} = \sum_{i=1}^{n_i} \sum_{t=1}^{n_t} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} r_{iq}\bar{t}_{jq} x_{ijtk\ell p}, \\ j = 1, \ldots, n_j, \ q = 1, \ldots, n_q \tag{A.10}$$

$$\sum_{i=0}^{n_i} s_{ijq} = 1, \quad j = 1, \ldots, n_j, \ q = 1, \ldots, n_q \tag{A.11}$$

(b) *Compactness value definition.* We defined a compact assignment of a committee member, i, to a day, k, at an hour, ℓ, as one that occurs within a specific time frame, b_i, after the end of a different assignment for such a committee member. That is, if a committee member, i, is assigned to a defence, j, at a day, k, and an hour, ℓ, this assignment is considered compact on the condition that the same committee member is assigned to a different defence, $\bar{j}$, in the same day, k, between hours $\ell - d$ and $\ell - d - b_i$. Moreover, the parameter $v_{i\ell}$, distinguishes the hour slots within such a time frame, as they might have different perceived values for a committee member, i. Thus, the compactness value for an assignment, $\bar{s}_{ik\ell}$, is 0 if a committee member, i, does not have a different assignment ending between hours ℓ and $\ell - b_i$, or $v_{i\bar{\ell}}$ if he does have such an assignment ending at $\bar{\ell}$ hour slots before the start of the new assignment a different hour slot, ℓ.

Before we define the compactness variable, $\bar{s}_{ik\ell}$, we define a different variable, $\bar{y}_{ik\ell p}$, which takes the value 1 if a committee member, i, is assigned to any defence at a day, k, an hour, ℓ and a room, p, and 0 otherwise. This is done in Constraint (12). Moreover, we denote that the right-hand-side of the equality in the constraint above never takes a value greater than 1, as a committee member, i, cannot be assigned more than one defence or role in the same slot, (k, ℓ, p).

With this new variable, we can define the compactness variable, $\bar{s}_{ik\ell}$, as the product between the sums, $\sum_{p=1}^{n_p} \bar{y}_{ik\ell p}$, and, $\sum_{\bar{\ell}=0}^{b_i} \sum_{p=1}^{n_p} v_{i\bar{\ell}} \bar{y}_{ik\hat{\ell}p}$, with $\hat{\ell} = \ell - d - \bar{\ell}$. Nonetheless, this would impair the linearity of the model. Thus, to linearise the aforementioned product, we opted for a big-M formulation, with $M = n_{vi}$, which is the highest value the compactness variable, $\bar{s}_{ik\ell}$, can take for a committee member, i. This formulation is represented in the following constraints.

$$\bar{y}_{ik\ell p} = \sum_{j=1}^{n_j} \sum_{t=1}^{n_t} x_{ijtk\ell p}, \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \\ \ell = 1, \ldots, n_\ell, \ p = 1, \ldots, n_p \tag{A.12}$$

$$\bar{s}_{ik\ell} \geqslant 0, \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \ \ell = 1, \ldots, n_\ell \tag{A.13}$$

$$\bar{s}_{ik\ell} \leqslant n_{v_i} \sum_{p=1}^{n_p} \bar{y}_{ik\ell p} \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \ \ell = 1, \ldots, n_\ell \tag{A.14}$$

$$\bar{s}_{ik\ell} \leqslant \sum_{\bar{\ell}=0}^{b_i} \sum_{p=1}^{n_p} v_{i\bar{\ell}} \bar{y}_{ik\hat{\ell}p}, \quad i = 1, \ldots, n_i, \\ k = 1, \ldots, n_k, \ \ell = d, \ldots, n_\ell, \ \hat{\ell} = \ell - d - \bar{\ell} \tag{A.15}$$

$$\bar{s}_{ik\ell} \geqslant \sum_{\bar{\ell}=0}^{b_i} \sum_{p=1}^{n_p} v_{i\bar{\ell}} \bar{y}_{ik\hat{\ell}p} - n_{v_i} \left(1 - \sum_{p=1}^{n_p} \bar{y}_{ik\ell p}\right), \\ i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \\ \ell = d, \ldots, n_\ell, \ \hat{\ell} = \ell - d - \bar{\ell} \tag{A.16}$$

(c) *Workload definition.* The workload for a committee member, i, is defined as the number, j, of committees they are assigned to. It would be possible to represent it as an integer variable. Still, in such a case, it would not be possible to consider its square in the objective function while keeping its linearity. However, the exponential penalty in the objective function can be linearised by representing it through a variable, w_{ij}, which takes the value 1 if a committee member, i, is assigned to a number, j, of committees, and 0 otherwise. The value for such a variable is defined in Constraint (17) by stating that the product of w_{ij} by the number, j, of defences assigned to a committee member, i, is equal to the sum of the assignments, $x_{ijtk\ell p} = 1$, for that same committee member. Furthermore, a committee member, i, cannot be assigned to more than one number of defences, j. Thus, with Constraint (18), we ensure that this value is unique for each one.

$$\sum_{j=0}^{c_i} j w_{ij} = \sum_{j=1}^{n_j} \sum_{t=1}^{n_t} \sum_{k=1}^{n_k} \sum_{\ell=1}^{n_\ell} \sum_{p=1}^{n_p} x_{ijtk\ell p}, \quad i = 1, \ldots, n_i \tag{A.17}$$

$$\sum_{j=0}^{c_i} w_{ij} = 1, \quad i = 1, \ldots, n_i \tag{A.18}$$

(d) *Committee days definition.* A committee day is defined as a day when a committee member has a defence scheduled. To represent this concept, we introduce a variable, $\hat{y}_{ijk}$, which takes value 1 if a committee member, i, is assigned to a number, j, of committees in a day, k, and 0 otherwise. To define such a variable, in Constraint (19), we define its value and in (20) we ensure its uniqueness for each combination of committee member, i, and day, k, similarly to the definitions of variables s_{ijq}, in Constraints (10) and (11), and w_{ij}, in Constraints (17) and (18). To institute fairness in the distribution of committee days, an exponential penalty on the number for each committee member is included in their respective objective function. However, to keep the model linear, we still need another variable, $\overline{w}_{ik}$, which takes the value 1 if a committee member, i, has defences scheduled on a number, k, of days, and 0 otherwise. The value for such a variable is defined in Constraint (21), and, in Constraint (22), we ensure that this value is unique for each committee member. Moreover, let us note that, while similar, the latter two constraints have a noteworthy difference when compared to the other constraints referenced in this point. Specifically, the right-hand-side of Constraint (21), does not include a variation of the sum of the assignments, $x_{ijtk\ell p} = 1$, involving instead another variable, $\hat{y}_{ijk}$, moreover, while this variable, $\hat{y}_{ijk}$, is defined for a $j = 0, \ldots, n_j$, the sum must start in $j = 1$, as we do not want to count the days, k, where a committee member, i, has 0 defences assigned.

$$\sum_{j=0}^{c_i} j\hat{y}_{ijk} = \sum_{j=1}^{n_j}\sum_{t=1}^{n_t}\sum_{\ell=1}^{n_\ell}\sum_{p=1}^{n_p} x_{ijtk\ell p}, \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k \tag{A.19}$$

$$\sum_{j=0}^{n_j} \hat{y}_{ijk} = 1, \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k \tag{A.20}$$

$$\sum_{k=0}^{n_k} k\overline{w}_{ik} = \sum_{j=1}^{c_i}\sum_{k=1}^{n_k} \hat{y}_{ijk}, \quad i = 1, \ldots, n_i \tag{A.21}$$

$$\sum_{k=0}^{n_k} \overline{w}_{ik} = 1, \quad i = 1, \ldots, n_i \tag{A.22}$$

(e) *Room change penalty definition.* A room change is considered problematic if a committee member, i, is not given a certain amount of time, a_i, between the end of an assignment, $\overline{j}$, and the beginning of another, j, which is scheduled for a different room, $\overline{p}$, than the first one, p. Moreover, parameter $h_{i\ell}$ distinguishes the hour slots within such a timeframe, as they might have different perceived penalties for a committee member, i. The room change variable, $\hat{s}_{ik\ell p}$, is the variable that represents the room change penalty that the assignment of a committee member, i, to a day, k, an hour, ℓ and a room, p, would incur. This variable can be defined as the product between the variable $\overline{y}_{ik\ell p}$, which takes the value 1 if a committee member, i, is assigned to any defence at a day, k, an hour, ℓ, and a room, p, and the sum $\sum_{\overline{\ell}=0}^{a_i}\sum_{\overline{p}=1}^{n_p} h_{i\overline{\ell}}\overline{y}_{ik\hat{\ell}\overline{p}}$, with $\hat{\ell} = \ell - d - \overline{\ell}$, which will take the value of parameter $h_{i\overline{\ell}}$ if a committee member, i, is assigned to a different defence, $\overline{j}$, in the same day, k, in hour $\ell - d - \overline{\ell}$, which is allocated a different room, $\overline{p}$. However, this product would not be linear. Thus, we opted for a big-M formulation, with the big-M being bounded by the highest value parameter $h_{i\ell}$ can take, that is, $M = n_{h_i}$ to linearise said product. This formulation is represented in the following constraints.

$$\hat{s}_{ik\ell p} \geqslant 0, \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \\ \ell = 1, \ldots, n_\ell, \ p = 1, \ldots, n_p \tag{A.23}$$

$$\hat{s}_{ik\ell p} \leqslant n_{h_i}\overline{y}_{ik\ell p} \quad i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \\ \ell = 1, \ldots, n_\ell, \ p = 1, \ldots, n_p \tag{A.24}$$

$$\hat{s}_{ik\ell p} \leqslant \sum_{\overline{\ell}=0}^{a_i}\sum_{\overline{p}=1}^{n_p} h_{i\overline{\ell}}\overline{y}_{ik\hat{\ell}\overline{p}}, \\ i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \ \ell = d, \ldots, n_\ell, \\ \hat{\ell} = \ell - d - \overline{\ell}, \ p = 1, \ldots, n_p, \ p \neq \overline{p} \tag{A.25}$$

$$\hat{s}_{ik\ell p} \leqslant \sum_{\overline{\ell}=0}^{a_i}\sum_{\overline{p}=1}^{n_p} h_{i\overline{\ell}}\overline{y}_{ik\hat{\ell}\overline{p}} - n_{h_i}(1 - \overline{y}_{ik\ell p}), \\ i = 1, \ldots, n_i, \ k = 1, \ldots, n_k, \\ \ell = d, \ldots, n_\ell, \ \hat{\ell} = \ell - d - \overline{\ell}, \\ p = 1, \ldots, n_p, \ p \neq \overline{p} \tag{A.26}$$

Appendix B. Markov chains and availability estimation

B1. Some fundamental concepts regarding Markov chains

Our availability generation algorithm can be defined as a Markov chain. Nonetheless, some fundamental concepts must be clarified before this representation can be addressed.

A Markov chain is a type of stochastic process, with the distinguishing characteristic that each of its states, α, is part of a set of discrete events and that the probability of each state, α, to occur at time ℓ, depends only on the previous state, $\overline{\alpha}$, occurring at time $\ell - 1$.

A square transition matrix T $i \times j$ can also represent such a system. Each entry of transition matrix T represents the probability $p(\alpha_j|\alpha_i)$ of the next state being α_j given that the previous state was α_i. Moreover, each entry must be within 0 and 1, as they represent probabilities, and the sum of each row must be 1, to correctly represent the total probability of a given set. An example of such a matrix T is displayed in Fig. B.1.

Accordingly, the power T^ℓ computes the probability of each state, α_j, to occur after ℓ repetitions, given that the initial state was α_i. Moreover, a transition matrix, T, is said to be regular if, after a certain number of repetitions, each of its columns stabilises at a certain value. Thus, if a transition matrix, T, is regular, there is a vector, V, such that, after a sufficiently large number of experiments, ℓ, and for any probability vector, $\hat{p}$, the following condition is verified,

$$\hat{p} \cdot T^\ell \approx V.$$

This suggests that after a certain number of experiments, regardless of the initial conditions, a regular Markov chain converges

$p(\alpha_1\|\alpha_1)$	$\ldots$	$p(\alpha_j\|\alpha_1)$	$\ldots$	$p(\alpha_{n_j}\|\alpha_1)$
		$\ldots$		
$p(\alpha_1\|\alpha_i)$	$\ldots$	$p(\alpha_j\|\alpha_i)$	$\ldots$	$p(\alpha_{n_j}\|\alpha_i)$
		$\ldots$		
$p(\alpha_1\|\alpha_{n_i})$	$\ldots$	$p(\alpha_j\|\alpha_{n_i})$	$\ldots$	$p(\alpha_{n_j}\|\alpha_{n_i})$

Fig. B.1. An example of a transition matrix T.

to a steady-state, with each possible state, α, occurring at a certain probability V_α. Furthermore, a transition matrix, T, is known to be regular, if, after any number of repetitions, ℓ, there is a T^ℓ such that $p(\alpha_j|\alpha_i) > 0$, for all $j = 1, \ldots, n_j,\ i = 1 \ldots, n_i$.

B2. Representation of algorithm 6 as a Markov chain and probability distribution estimation

Our availability generation algorithm can be described as a Markov chain with $n_\alpha + d$ possible states, where n_α is the number of different availability states and d is the duration of the defences.

For an availability state, $\alpha = 1, \ldots, n_\alpha$, occurring at time $\ell - 1$, the following state, i.e., the one occurring at time ℓ, can remain unchanged, with a probability $p(\alpha|\alpha)$, it can change to another availability state, $\overline{\alpha} = 1, \ldots, n_\alpha,\ \overline{\alpha} \neq \alpha$, with probability $p(\overline{\alpha}|\alpha)$ or it can start an unavailability block, with probability $p(0|\alpha)$.

When an unavailability block starts, we know that at least d zeros are to be added. Nonetheless, in a Markov chain, the probabilities must only depend on the previous state. Thus, before we have the unavailability state, $\alpha = 0$, there must be $d - 1$ different states, $0_{e_{\hat{i}}},\ \hat{i} = 1, \ldots, d-1$, that lead to the generation of a 0, with probability 1, i.e., for $\hat{i} = 1, \ldots, d-2$, the only probability is $p(0_{e_{\hat{i}+1}}|0_{e_{\hat{i}}}) = 1$ and for $\hat{i} = d-1$ it is $p(0|0_{e_{d-1}}) = 1$.

Finally, when the last exceptional 0 is added, the state $\alpha = 0$ functions similarly to the availability states. Specifically, it can repeat itself, with probability $p(0|0)$, or it can generate an availability state, $\alpha = 1, \ldots, n_\alpha$, with probability $p(\alpha|0)$. Let us note that, unlike the availability states, the unavailability state, $\alpha = 0$, cannot be followed by a state 0_{e_1}.

The Markov chain that represents the availability generation algorithm is displayed in Fig. B.2.

Accordingly, we can use the properties of each corresponding transition matrix, T, to estimate the probability distribution of the parameters generated through this method, $l_{ik\ell}$ and $m_{k\ell p}$.

Considering the warm-up period, $\Delta = 40$, and, as an example, the generation of $l_{ik\ell}$ with an unavailability percentage $p(l_{ik\ell} = 0) = 0.78$:

The transition matrix, T, is:

$p(\alpha_j\|\alpha_i)$	0_{e_1}	0	1	2
0_{e_1}	0	1	0	0
0	0	0.95	0.025	0.025
1	0.1728	0	0.7	0.1272
2	0.1728	0	0.1272	0.7

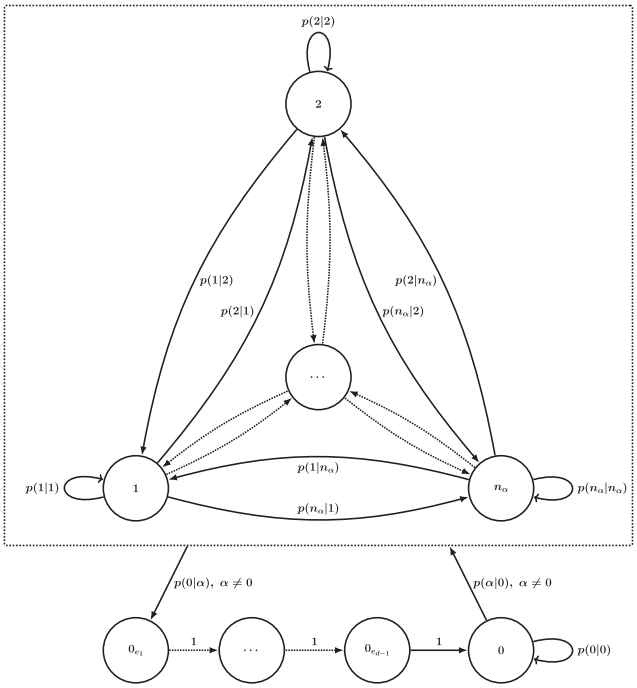

Fig. B.2. Markov chain representation of the availability generation algorithm.

The transition matrix, following 39 repetitions, T^{39}, is:

$p(\alpha_j\|\alpha_i)$	0_{e_1}	0	1	2
0_{e_1}	0.0373	0.7466	0.1080	0.1080
0	0.0373	0.7466	0.1080	0.1080
1	0.0373	0.7466	0.1080	0.1080
2	0.0373	0.7466	0.1080	0.1080

And the transition matrix, following 40 repetitions, T^{40}, is:

$p(\alpha_j\|\alpha_i)$	0_{e_1}	0	1	2
0_{e_1}	0.0373	0.7466	0.1080	0.1080
0	0.0373	0.7466	0.1080	0.1080
1	0.0373	0.7466	0.1080	0.1080
2	0.0373	0.7466	0.1080	0.1080

Consequently, $p(l_{ik\ell} = 0) = 0.0373 + 0.7466 \approx 0.78$ and $p(l_{ik\ell} = 1) = p(l_{ik\ell} = 2) \approx 0.11$. Let us note that, at a $\ell = \Delta = 40$, the matrixes were not yet fully stationary, with differences between T^{39} and T^{40} of order 10^{-5}. Nonetheless, we considered these to be small enough to conduct our computational experiments.

Appendix C. Computational results

Table C.1
Computational experiments - small instances ($n_i = 25,\ n_j = 20$).

Instance		Data										Output							
N	$p(n_i.n_j.n_t.n_k.n_\ell.n_p.n_q)$	d	u_i	e_{ijt}	c_i	$l_{ik\ell}$	$m_{k\ell p}$	$v_{i\ell}$	$h_{i\ell}$	r_{iq}	t_{iq}	$\|N\|$	$\|I\|$	$skip^N$	$skip^I$	$time^N$	$time^I$	g	CPU(seconds)
1	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	21	3	65	11	0	0	10	230
2	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	26	5	53	16	0	0	18	289
3	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	25	4	64	7	0	0	17	550
4	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	11	1	86	2	0	0	14	267
5	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	48	5	30	17	0	0	20	542
6	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	51	5	27	17	0	0	18	1067
7	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	59	5	22	14	0	0	17	590
8	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	38	5	44	13	0	0	19	1135
9	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	47	5	32	16	0	0	18	513
10	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	25	6	49	20	0	0	14	298
11	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	44	5	33	18	0	0	20	1418
12	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	38	5	36	21	0	0	20	417
13	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	42	6	34	18	0	0	20	485
14	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	50	5	29	16	0	0	20	777
15	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	57	7	12	24	0	0	19	1140
16	p(25.20.3.15.16.3.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	47	6	27	20	0	0	20	10246
17	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	33	5	41	21	0	0	16	488
18	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	25	3	56	16	0	0	15	384
19	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	52	4	32	12	0	0	17	818
20	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	60	6	13	21	0	0	20	1053
21	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	51	6	21	22	0	0	20	750
22	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	24	5	55	16	0	0	13	369
23	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	61	5	21	13	0	0	20	991
24	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	2	13	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	60	4	24	12	0	0	20	2631
25	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	38	6	35	21	0	0	19	1217
26	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	45	5	32	18	0	0	20	1004
27	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	47	5	31	17	0	0	20	4364
28	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	53	6	19	22	0	0	20	1522
29	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	42	5	37	16	0	0	20	1221
30	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	38	6	37	19	0	0	20	709
31	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	40	6	32	22	0	0	20	12951
32	p(25.20.3.15.16.4.15)	2	[0.7, 0.3]	1	13	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	41	5	37	15	0	2	20	21175

Table C.2
Computational experiments - medium instances ($n_i = 38,\ n_j = 30$).

Instance		Data										Output							
N	$p(n_i.n_j.n_t.n_k.n_\ell.n_p.n_q)$	d	u_i	e_{ijt}	c_i	$l_{ik\ell}$	$m_{k\ell p}$	$v_{i\ell}$	$h_{i\ell}$	r_{iq}	t_{iq}	$\|N\|$	$\|I\|$	$skip^N$	$skip^I$	$time^N$	$time^I$	g	CPU(seconds)
33	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	37	6	37	20	0	0	26	973
34	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	29	2	68	1	0	0	20	698
35	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	39	5	41	15	0	0	24	1003
36	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	60	4	24	12	0	0	26	1484
37	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	30	5	48	17	0	0	22	788
38	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	51	5	28	16	0	0	22	1593
39	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	43	4	41	12	0	0	26	1169
40	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	41	5	38	16	0	0	28	10129
41	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	59	6	16	19	0	0	30	1951
42	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	39	5	38	18	0	0	29	1215
43	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	26	5	31	18	16	4	30	35358
44	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	39	5	38	18	0	0	30	20317
45	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	44	6	28	22	0	0	30	6579
46	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	42	4	25	12	13	4	30	34954
47	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	14	4	37	11	20	14	30	38044
48	p(38.30.3.15.16.3.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	46	5	27	17	2	3	30	21212
49	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	40	5	40	15	0	0	23	1589
50	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	40	3	47	10	0	0	28	1351
51	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	38	6	37	19	0	0	30	2984
52	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	32	3	58	7	0	0	23	1248
53	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	53	6	25	16	0	0	25	1862
54	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	51	5	32	12	0	0	24	1883
55	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	40	5	41	14	0	0	26	2218
56	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	2	19	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	32	5	49	14	0	0	24	2382
57	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	54	5	26	15	0	0	26	2028
58	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	57	5	26	12	0	0	30	2493
59	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	53	5	29	13	0	0	30	18086
60	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	15	4	31	14	22	14	30	37324
61	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	22	5	36	18	11	8	30	31803
62	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	15	4	30	10	24	17	30	39081
63	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	8	4	23	11	31	23	30	39821
64	p(38.30.3.15.16.4.15)	2	[0.7, 0.3]	1	19	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	11	4	19	14	36	16	30	38628

Table C.3
Computational experiments - large instances ($n_i = 50,\ n_j = 40$).

Instance		Data										Output							
N	$p(n_i.n_j.n_t.n_k.n_\ell.n_p.n_q)$	d	u_i	e_{ijt}	c_i	$l_{ik\ell}$	$m_{k\ell p}$	$v_{i\ell}$	$h_{i\ell}$	r_{iq}	t_{iq}	$\|N\|$	$\|I\|$	$skip^N$	$skip^I$	$time^N$	$time^I$	g	CPU (seconds)
65	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	28	5	52	15	0	0	28	1366
66	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	51	4	36	9	0	0	32	2633
67	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	40	4	43	13	0	0	34	2744
68	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	36	5	42	17	0	0	35	6862
69	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	39	5	43	11	1	1	35	22050
70	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	49	5	30	16	0	0	32	2719
71	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	40	3	45	9	0	3	33	29643
72	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	40	5	39	15	1	0	37	21446
73	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	48	5	24	19	4	0	40	18000
74	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	39	5	39	17	0	0	36	13145
75	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	17	5	42	14	14	8	40	34408
76	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	42	5	35	16	0	2	40	20341
77	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	21	4	40	13	16	6	40	35309
78	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	34	5	32	14	9	6	40	31078
79	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	15	5	20	15	21	24	39	37261
80	p(50.40.3.15.16.3.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	3	3	0	11	39	44	40	41412
81	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	44	5	37	14	0	0	31	2840
82	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	39	6	34	21	0	0	26	2726
83	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	38	4	48	10	0	0	34	9626
84	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	44	5	35	16	0	0	32	3543
85	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	43	4	42	11	0	0	35	8356
86	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	32	5	50	13	0	0	32	2459
87	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	49	5	31	15	0	0	34	20443
88	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	2	25	[0.78, 0.11, 0.11]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	40	4	41	11	2	2	39	31311
89	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	12	4	12	14	33	25	40	39083
90	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	34	5	29	16	10	6	36	30877
91	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	10	4	18	10	17	41	40	40127
92	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.86, 0.07, 0.07]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	16	4	37	11	10	22	39	38514
93	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.8, 0.2]	[0.8, 0.2]	3	3	11	5	22	14	16	32	40	38139
94	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.86, 0.14]	[0.7, 0.3]	[0.7, 0.3]	3	3	19	5	40	13	13	10	40	35060
95	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.8, 0.2]	[0.8, 0.2]	3	3	7	3	17	9	31	33	40	41043
96	p(50.40.3.15.16.4.15)	2	[0.7, 0.3]	1	25	[0.82, 0.09, 0.09]	[0.8, 0.2]	[0.7, 0.3]	[0.7, 0.3]	3	3	6	3	16	10	23	42	40	40999

References

Al-Yakoob, S. M., & Sherali, H. D. (2015). Mathematical models and algorithms for a high school timetabling problem. *Computers and Operations Research, 61*, 56–68. https://doi.org/10.1016/j.cor.2015.02.011.

Amiri, M., & Farvaresh, H. (2023). Carrier collaboration with the simultaneous presence of transferable and non-transferable utilities. *European Journal of Operational Research, 304*(2), 596–617. https://doi.org/10.1016/j.ejor.2022.04.033.

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

Battistutta, M., Ceschia, S., De Cesco, F., Di Gaspero, L., & Schaerf, A. (2019). Modelling and solving the thesis defense timetabling problem. *Journal of the Operational Research Society, 70*(7), 1039–1050. https://doi.org/10.1080/01605682.2018.1495870.

Bellio, R., Ceschia, S., Di Gaspero, L., & Schaerf, A. (2021). Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. *Computers and Operations Research, 132*. https://doi.org/10.1016/j.cor.2021.105300. Article No. 105300

Burke, E. K., Mareček, J., Parkes, A. J., & Rudová, H. (2010). Decomposition, reformulation, and diving in university course timetabling. *Computers and Operations Research, 37*(3), 582–597. https://doi.org/10.1016/j.cor.2009.02.023. Hybrid Metaheuristics

Ceschia, S., Di Gaspero, L., & Schaerf, A. (2023). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research, 308*(1), 1–18. https://doi.org/10.1016/j.ejor.2022.07.011.

Chaudhuri, A., & De, K. (2010). Fuzzy genetic heuristic for university course timetable problem. *International Journal of Advances in Soft Computing and its Applications, 2*(1), 100–123.

Chen, M. C., Sze, S. N., Goh, S. L., Sabar, N. R., & Kendall, G. (2021). A survey of university course timetabling problem: Perspectives, trends and opportunities. *IEEE Access, 9*, 106515–106529. https://doi.org/10.1109/ACCESS.2021.3100613.

Christopher, G., & Wicaksana, A. (2021). Particle swarm optimization for solving thesis defense timetabling problem. *TELKOMNIKA Telecommunication, Computing, Electronics and Control, 19*(3), 762–769. https://doi.org/10.12928/TELKOMNIKA.v19i3.18792.

Goh, S. L., Kendall, G., & Sabar, N. R. (2017). Improved local search approaches to solve the post enrolment course timetabling problem. *European Journal of Operational Research, 261*(1), 17–29. https://doi.org/10.1016/j.ejor.2017.01.040.

Guo, Y., Zhang, Y., Boulaksil, Y., Qian, Y., & Allaoui, H. (2023). Modelling and analysis of online ride-sharing platforms—A sustainability perspective. *European Journal of Operational Research, 304*(2), 577–595. https://doi.org/10.1016/j.ejor.2022.04.035.

Gülcü, A., & Akkan, C. (2020). Robust university course timetabling problem subject to single and multiple disruptions. *European Journal of Operational Research, 283*(2), 630–646. https://doi.org/10.1016/j.ejor.2019.11.024.

Huynh, T. T. B., Pham, Q. D., & Pham, D. D. (2012). Genetic algorithm for solving the master thesis timetabling problem with multiple objectives. In *TAAI '12: Proceedings of the 2012 conference on technologies and applications of artificial intelligence* (pp. 74–79). Washington, D.C., USA: IEEE Computer Society. https://doi.org/10.1109/TAAI.2012.50. Article No. 6395009

Kochaniková, B., & Rudová, H. (2013). Student scheduling for bachelor state examinations. In G. Kendall, G. V. Berghe, & B. McCollum (Eds.), *6th multidisciplinary international conference on scheduling : Theory and applications*. New York, NY, USA: Springer.

Koziel, S., & Pietrenko-Dabrowska, A. (2022). Constrained multi-objective optimization of compact microwave circuits by design triangulation and Pareto front interpolation. *European Journal of Operational Research, 299*(1), 302–312. https://doi.org/10.1016/j.ejor.2021.08.021.

Limanto, S., Benarkah, N., & Adelia, T. (2019). Thesis examination timetabling using genetic algorithm. In T. H. Muliawati, M. F. Ardiansyah, D. M. Sari, D. I. Permatasari, & Mu'arifin (Eds.), *International electronics symposium on knowledge creation and intelligent computing, IES-KCIC 2018 - proceedings* (pp. 6–10). Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers Inc.. https://doi.org/10.1109/KCIC.2018.8628572. Article No. 8628572

Mavrotas, G. (2009). Effective implementation of the e-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation, 213*(2), 455–465. https://doi.org/10.1016/j.amc.2009.03.037.

Mavrotas, G., & Florios, K. (2013). An improved version of the augmented e-constraint method (AUGMECON2) for finding the exact Pareto set in multi-objective integer programming problems. *Applied Mathematics and Computation, 219*(18), 9652–9669. https://doi.org/10.1016/j.amc.2013.03.002.

Mesquita-Cunha, M., Figueira, J. R., & Barbosa-Póvoa, A. P. (2022). New ϵ-constraint methods for multi-objective integer linear programming: A Pareto front representation approach. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2022.07.044.

Pham, Q. D., Hoang, T. A. D., Huynh, T. T., & Nguyen, T. H. (2015). A Java library for constraint-based local search: Application to the master thesis defense timetabling problem. In H. Quyet Tang, P. Le Anh, L. De Raedt, Y. Deville, M. Bui, L. Trong Thi Dieu, … N. Nguyen Ba (Eds.), *Proceedings of the sixth international symposium on information and communication technology: vol. 03-04-December-2015* (pp. 67–74). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2833258.2833300.

Santiago-Mozos, R., Salcedo-Sanz, S., Deprado-Cumplido, M., & Bousoño-Calzón, C. (2005). A two-phase heuristic evolutionary algorithm for personal-

izing course timetables: A case study in a spanish university. *Computers and Operations Research, 32*(7), 1761–1776. https://doi.org/10.1016/j.cor.2003.11.030.

Su, P., Luo, B. X., Deng, F. Y., Xia, A. X., & Guo, Y. (2020). Group strategy of dissertation defense based on greedy retrospective hybrid algorithm. In *Journal of physics: Conference series: vol. 1634*. Bristol, UK: IOP Publishing Ltd. https://doi.org/10.1088/1742-6596/1634/1/012077. Article No. 012077

Sørensen, M., & Dahms, F. (2014). A two-stage decomposition of high school timetabling applied to cases in denmark. *Computers and Operations Research, 43*, 36–49. https://doi.org/10.1016/j.cor.2013.08.025.

Tawakkal, M. I., & Suyanto (2020). Exploration-exploitation balanced krill herd algorithm for thesis examination timetabling. In *2020 international conference on data science and its applications*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers Inc.. https://doi.org/10.1109/ICoDSA50139.2020.9212837. Article No. 9212837

Urbani, M., Brunelli, M., & Punkka, A. (2023). An approach for bi-objective maintenance scheduling on a networked system with limited resources. *European Journal of Operational Research, 305*(1), 101–113. https://doi.org/10.1016/j.ejor.2022.05.024.

Vermuyten, H., Lemmens, S., Marques, I., & Beliën, J. (2016). Developing compact course timetables with optimized student flows. *European Journal of Operational Research, 251*(2), 651–661. https://doi.org/10.1016/j.ejor.2015.11.028.