High school timetabling at a federal educational institute in Brazil

Lucas H. A. Dantas, Romerito C. Andrade
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte (IFRN)
Natal, Brazil
{lucas.dantas,romerito.campos}@ifrn.edu.br

Leonardo C. T. Bezerra
Instituto Metrópole Digital (IMD)
Universidade Federal do Rio Grande do Norte (UFRN)
Natal, Brazil
leobezerra@imd.ufrn.br

***Abstract*—High school timetabling (HST) is a relevant problem traditionally addressed by (meta)heuristic approaches. This work addresses the HST in the context of the technical courses offered at the Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte (IFRN). Part of the largest Brazilian federal educational network, IFRN comprises 22 campi located in 18 different cities, which makes an HST approach for IFRN challenging, critical, and potentially seminal for other institutes of the network. Our contributions are two-fold. First, we model the HST problem at IFRN both as to mathematical formulation and real-world instances, which we create from data gathered at different campi. Second, we propose a greedy randomized adaptive search procedure (GRASP) algorithm specific for this scenario. To validate our contributions, we benchmark on the instances we create (i) state-of-the-art, (ii) commercial, and (iii) the proposed GRASP algorithms. Our approach produces feasible solutions for more instances than the remaining algorithms, with also competitive solution quality.**

***Index Terms*—high school timetabling, metaheuristic, GRASP**

I. Introduction

In education, *timetabling* problems involve allocating lesson schedules from a school, university courses, or even exams [1]. Generally, the goal of such problems is to schedule a series of meetings, often within a week, obeying a set of restrictions that involve resources, pedagogical aspects, and personal preferences. The high school timetabling (HST) problem is a relevant example [2], in which time slots, teachers, students, and classrooms must be assigned to a collection of lessons, and none of the participants may be assigned to more than one lesson simultaneously. Similarly to other timetabling problems, the HST is NP-hard, meaning exact algorithms are severely limited w.r.t. the instance sizes they can solve in reasonable computational times [1]. As an alternative, (meta)heuristic methods have shown relevant performance on the HST [3], [4]. Yet, solutions proposed for one country may not be straightforward to generalize. This is largely justified by the variations in the educational systems from different countries. As such, there is a striking difficulty in developing collaborative HST research [5].

The goal of this work is to address the HST problem in the context of the technical courses offered at the Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte (IFRN). IFRN is part of the major Brazilian federal education network targeting (technical) high-school and undergraduate studies. In total, the network comprises about 653 units and all 27 Brazilian federated units. IFRN, in particular, comprises 22 campi in 18 different cities of the state of Rio Grande do Norte. Every new school semester, each campus needs to compose its timetable in compliance with institutional and pedagogical requirements, which makes an HST approach both challenging and critical. Furthermore, given the expected similarities among IF campi across different Brazilian federated units, addressing the HST problem at IFRN is potentially seminal for a country-level solution to be deployed over all units.

Our contributions are two-fold. First, we model the HST problem requirements in the context of IFRN campi. Concretely, we (i) list and justify hard and soft constraints, (ii) formulate a mathematical model to account for the most relevant constraints identified, and; (iii) produce real-world instances from data collected from several campi. Though it exceeds the scope of this work, the proposed mathematical formulation can be used for research on mathematical programming approaches. Regarding real-world instances, we use an internationally adopted description format (XHSTT [6]), which allows us to benckmark state-of-the-art [7], [8] algorithms on these instances. Unfortunately, preliminary experiments demonstrated that existing algorithms are unable to solve some of the proposed instances due to their characteristics.

To address this limitation, our second contribution is the proposal and proof-of-concept validation of a metaheuristic optimizer specific for the HST problem at IFRN. Our proposed approach is a greedy randomized adaptive search procedure (GRASP) algorithm [9], a method that has repeatedly demonstrated its efficacy in several NP-hard problem applications [10]. GRASP requires both a constructive procedure and local search operators, which we propose and assess. We validate our GRASP approach benchmarking it against the state-of-the-art (meta)heuristics and also the commercial solutions previously adopted by the given campus. Results demonstrate that our approach is able to find feasible solutions for more instances than the remaining algorithms. Though state-of-the-art performance is not our goal, solutions devised are competitive in quality with the solutions from the remaining algorithms. Furthermore, though devised for the HST problem at IFRN, it is likely that the proposed algorithm

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Finance Code 001

be applicable to the HST problem at campi from different federated units, with appropriate adjustments.

The remainder of this work is structured as follows. Section II briefly reviews theoretical background. In Section III, we model the HST problem at IFRN. Our proposed GRASP algorithm is detailed in Section IV, and validated in Section V. We conclude and discuss future work in Section VI.

II. Background

Timetabling problems are classified by theoretical computer science as NP-hard [1]. This is also the case for the HST, and as a result most approaches proposed for this problem are (meta)heuristic. In this section, we first review metaheuristics and their application to combinatorial optimization. Later, we discuss educational timetabling problems and the HST, reviewing its literature.

A. Metaheuristics and combinatorial optimization

For combinatorial optimization problems (COP) like timetabling, solution enumeration or related methods thereof are usually impracticable. Scalable approaches are usually approximate or heuristic. Problem-general heuristics are known as metaheuristics [11], and can be broadly classified as constructive and/or refinement. Constructive methods iteratively build solutions, and are often the fastest among metaheuristic methods. By contrast, refinement algorithms focus on improving an incumbent solution, iteratively replacing its solution components for alternative ones. It is often the case that approaches combine constructive and refinement procedures, given their complementary nature.

A locally-optimal solution found by a metaheuristic algorithm may not be the global optimum for the given COP [9], being dependent on the incumbent solution and *neighborhood operator* adopted. A neighborhood operator is the change pattern that produces novel solutions from an incumbent solution. Figure 1 illustrates a swap between lessons in neighbor timetables. Given the existence of local optima, metaheuristic approaches must be designed to effectively and efficiently explore the search space. The balance between search intensification (exploitation) and diversification (exploration) is regulated by (numerical) hyperparameters that must be configured.

B. Educational timetabling problems

Due to their frequent need to generate timetables, educational institutions are an important focus of works that address scheduling problems. Next, we briefly describe timetabling problem categories in education [1], [12].

School timetabling. Scheduling weekly lessons for all classes at a school, avoiding conflicts between teachers and classes. In these problems, idle times between lessons for students are generally not acceptable.

Course timetabling. The weekly allocation of all lectures in a set of university courses, minimizing overlaps between classes with students in common. Idle times between lectures for students are generally acceptable.

Examination timetabling. Scheduling exams for a set of university courses, avoiding overlaps between exam times in courses with students in common.

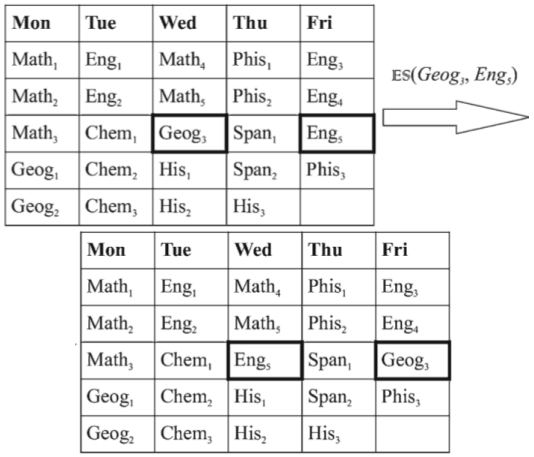


Fig. 1: Example of an event swap (ES) between weekdays [7].

Our work addresses the school scenario, known as the *high school timetabling* (HST) problem. For the HST, meetings are *lessons* on a *subject*, taught by a teacher to students grouped as a *class*. Lessons are scheduled in *time slots*, and idle slots between lessons are referred to as *windows*. In Figure 1 (top), for instance, the timetable presented considers a single class, and lessons scheduled for that class on Monday cover *Mathematics* on the first three time slots and *Geography* on the last two. The only idle time in the timetable given in Figure 1 (top) is the last time slot for Friday, but it is not considered a window for students as it is not situated between lessons of that class.

One of the challenges in timetabling problems is distinguishing feasibility constraints from optimality criteria. The former usually refers to institutional requirements that must be obeyed. However, when all organizational, pedagogical, and personal interests must be met in a mandatory manner, problem complexity and the time necessary for its solution increases. Alternatively, institutional and pedagogical requirements are modeled as hard constraints, i.e. solution feasibility. By contrast, preferences are modeled as soft constraints, and their minimization comprises solution optimality. Though HST problems vary substantially from one educational institution to another, hard constraints are commonly linked to physical conflicts such as overlapping lessons from a given teacher for different classes. Soft restrictions commonly found include: (i) meeting teaching staff personal preferences for specific time slots, and; (ii) minimizing windows for students and teachers.

C. Related work

Given the heterogeneous nature of the HST literature, we focus our review on efforts that help promote collaboration. Specifically, we discuss: (i) the most commonly used heuristic for HST problems, namely the Kingston high school timetabling engine (KHE [8]), and (ii) the International Timetabling Competition (ITC) edition that focused on the HST. The KHE heuristic has been open sourced as a programming library to provide a fast and robust alternative for the HST. KHE builds on hierarchical schedules, i.e., merging smaller timetables recursively until a complete solution is

produced. The algorithm is under continuous development, with its current version incorporating as polymorphic ejection chains [8]. ITC is an international competition that focuses on a different timetabling problem each year. Its third edition, held in 2011, addressed the HST problem. Next, we discuss its main contributions.

Algorithms. An algorithm combining simulated annealing (SA [9]) and iterated local search (ILS [9]) to refine initial solutions produced by KHE won the competition. In detail, the algorithm was proposed by the Group of Optimization and Algorithms (GOAL) [7], and produced the best solutions for almost all datasets included. More recently, the same group proposed an improved HST algorithm, replacing SA and ILS for variable neighborhood search (VNS [9]) and *matheuristics* ([9]) [13]. We refer to this hybrid solver as HS, and remark that it achieved best solutions for 15 out of 17 datasets considered in its evaluation.

Dataset description format. ITC 2011 helped popularize a description format for HST datasets. XHSTT [6] helps standardize both problem input and output, an provides flexibility for modeling the HST from different contexts. Currently, .

As discussed in this section, the HST is a problem that varies among different countries and educational systems. Though effective algorithms exist, their application to different contexts may be limited. In the next section, we model the HST problem for the technical courses offered at IFRN.

III. High school timetabling at IFRN

IFRN is part of a federal network of educational institutions that serves the 27 Brazilian federated units. In this section, we initially further detail its context and relevance. Later, we formulate the HST problem at IFRN from the perspective of constraints, mathematical model, and benchmark instances.

A. Context and relevance

The *Rede Federal de Educação Profissional, Científica e Tecnológica* network was established in 2008, as an expansion of the previous federal education networks in Brazil that originated in the early 20th century. As of 2019, the network comprised 38 federal institutes, in addition to technical universities and associated technical schools. Overall, the network comprises 653 educational units and serves over two million students, nearly half of which attend technical high-school courses. Importantly, the network plays a critical role in social equality, with over half of the students that declared income, gender, and ethnicity coming from low income families, being women, and self-declaring as non-white. Staff-wise, the network comprises over 50,000 teachers and professors, with nearly 20,000 holding a Master degree and circa 15,000 holding a doctoral degree.

IFRN is the federal institute located in the state of Rio Grande do Norte, in the Northeast region of Brazil. In total, IFRN has 21 campi in 18 different cities for on-site education, plus an online campus for remote education. Across its on-site campi, IFRN offers nearly 500 (technical) high-school and undergraduate level courses in all three shifts, serving over 80,000 students. Nearly half of IFRN students attend technical high-school courses. Given the state and region social and economical indicators, IFRN plays an even more important role in social equality, with low-income, female, and non-white student ratios above the national average. Staff-wise, IFRN has around 1,500 professors, with nearly 1,000 holding a Master degree and circa 500 holding a doctoral degree.

TABLE I: IFRN timetabling constraints.

Hard constraints		**Soft constraints**	
ID	***Description***	***ID***	***Description***
H1	Curriculum compliance	S1	Student window minimization
H2	Teacher exclusivity	S2	Teacher window minimization
H3	Class exclusivity	S3	Unwarranted weekly distribution
H4	Twin lessons	S4	Critical-time history
H5	Justified teacher availability		
H6	Lesson distribution		
H7	Maximum teacher daily lessons		
H8	Theoretical lessons limit per class		

B. Problem constraints

For each IFRN campus, high-school technical course, and academic semester, subjects are offered according to the curriculum of the given course. Despite the existence of common subjects for different courses, there is usually no sharing of classrooms between different classes. This way, teachers are pre-allocated to subjects for a specific course. The prior allocation of teachers increases the constraints of the problem, but eliminates the need to allocate classrooms.

Table I details the set of problem constraints we identify at IFRN, classifying them as hard or soft. Below, we briefly discuss the hard constraints.

H1: a class must be assigned the number of lessons proposed for each subject, as defined in its course curriculum.

H2 & H3: for a given time slot on a given school day, neither a teacher nor a class can be assigned to multiple lessons.

H4: a set of lessons from a given subject and class at a given school day must be arranged contiguously.

H5: a teacher can determine his teaching days provided legal justification.

H6: pedagogical preferences, e.g. to avoid a large number of consecutive lessons from a given teacher on a given subject for a given class on a given day.

H7: a teacher can be assigned maximum 10 lesson hours on a given school day.

H8: a teacher can be assigned a maximum four theoretical lesson hours for a given class on a given school day.

Soft constraints at IFRN comprise the minimization of student (S1) and teacher (S2) windows. For practical purposes, we also include constraints S3 and S4. The former refers to teacher preferences regarding concentrating lessons on a subset of school days, given other activities conducted in the institute such as research and outreach. The latter is an approach adopted by course coordinators to rotate teacher allocation on lessons assigned to the first time slots of Monday and to the last time slots on Friday. Finally, we remark that in this case study constraints H8 and S4 are not addressed due to the dependence of exogeneous data. Respectively, constraint H8 requires as input the split between theoretical and practical hours for a given subject of a given course, whereas constraint S4 requires teacher assignment history. In

addition, we merge constraints H5 and S3 for simplicity, i.e., all required distribution should be met.

C. Mathematical formulation

The input to the proposed model comprises sets of teachers and/or professors (P), classes (C), subjects (S), weekdays (D), and time slots (T). The model also requires (i) hour loads H, i.e. how many lessons h_{ps} teacher p should teach for subject s, and; (ii) teacher availability A, i.e. whether ($a_{pd} = 1$) or not ($a_{pd} = 0$) teacher p is available on day d. We assume that each lesson has a pre-allocated teacher and class, so subjects and teachers are not shared among classes and the *H6* constraint is assured. The goal function is given by Equation 1, a weighted sum of windows for teachers (W_p) and students (W_c) using weight importances I_p and I_c that are provided for decision-maker configuration. Auxiliary functions W_p and W_c are provided as supplementary material [14], for brevity.

$$min \sum_{d \in D} \left(\left(\sum_{p \in P} I_p \cdot W_p(p,d) \right) + \left(\sum_{c \in C} I_c \cdot W_c(c,d) \right) \right) \tag{1}$$

Equations 2–9 ensure hard constraints, where x_{psdt} indicates if teacher p is scheduled for subject s on day d and time slot t. Equation 2 ensures curriculum compliance (*H1*). Equations 3 and 4 respectively prevent conflicts among teacher (*H2*) and class schedules (*H3*). Note that Eq. 4 refers to a set of subjects from a given class as S_c. Equations 5–7 ensure that lessons for the same subject taught on the same weekday be twinned (*H4*). Equation 8 ensures teacher availability (*H5*), and Eq. 9 limits the number of daily lessons per teacher (*H7*).

$$\sum_{d \in D} \sum_{t \in T} x_{psdt} = h_{ps} \qquad \forall p \in P, s \in S \tag{2}$$

$$\sum_{s \in S} x_{psdt} \leq 1 \qquad \forall p \in P, d \in D, t \in T \tag{3}$$

$$\sum_{p \in P} \sum_{s \in S_c} x_{psdt} \leq 1 \qquad \forall c \in C, d \in D, t \in T \tag{4}$$

$$\left(\sum_{t \in T} x_{psdt} \cdot x_{psd(t+1)} \right) + 1 = \sum_{t \in T} x_{psdt} \quad \forall p \in P, s \in S, d \in D \tag{5}$$

$$\left(\sum_{t \in T} x_{psdt} \cdot x_{psd(t-1)} \right) + 1 = \sum_{t \in T} x_{psdt} \quad \forall p \in P, s \in S, d \in D \tag{6}$$

$$\left(\sum_{t \in T} x_{psdt} \cdot x_{psd(t+1)} \right) = \left(\sum_{t \in T} x_{psdt} \cdot x_{psd(t-1)} \right) \quad \forall p \in P, s \in S, d \in D \tag{7}$$

$$\sum_{s \in S} \sum_{t \in T} x_{psdt} = a_{pd} \cdot \sum_{s \in S} \sum_{t \in T} x_{psdt} \qquad \forall p \in P, d \in D \tag{8}$$

TABLE II: Instance characteristics. M: morning; A: afternoon.

Campus	Semester	Shift	ID	T	C	S	P	U
Caicó	2018.2	M	CA182M	30	11	109	41	71
		M&A	CA182MA	60	22	209	58	105
João Câmara	2018.1	M	JC181M	30	9	87	39	77
		M&A	JC181MA	60	15	147	52	98
	2018.2	M	JC182M	30	9	86	44	91
		M&A	JC182MA	60	14	144	52	107
Ipanguaçu	2018.1	M	IP181M	30	10	84	56	112
Pau dos Ferros	2018.1	M	PF181M	30	12	122	51	102
	2018.2	M	PF182M	30	12	124	55	110
		A	PF182A	30	13	98	55	109

$$\sum_{s \in S} \sum_{t \in T} x_{psdt} \leq 10 \qquad \forall p \in P, d \in D \tag{9}$$

D. Dataset modeling

To foster research on the HST at IFRN and other federal institutes, we devise XHSTT instances based on real timetables deployed in 2018 at four different IFRN campi on the first (2018.1) and second (2018.2) academic semesters. Details on how we model the HST at IFRN using the XHSTT format are provided as supplementary material [14], for brevity. The instances devised comprise both high school and undergraduate courses, given that teachers and professors at IFRN are often assigned to both. For this case study, we produced ten instances. Due to an absence of pattern among campi, we could not automate dataset production for an arbitrary semester/campus. Instead, the instances were reverse engineered from the timetables deployed, assuming that (i) lesson distribution adopted was requested by teachers, and; (ii) teacher unavailability was daily-wise rather than slot-wise. These assumptions increase problem constraints, and should be addressed when producing future timetables.

Table II depicts the general characteristics of each instance produced. Besides campus, semester, shift(s), and an unique identifier, we also give: (i) the number of time slots, classes, subjects, and teachers, and; (ii) the sum of unavailable days registered for the set of teachers. In average, each teacher presented between 1.5 and two unavailable days, whether justified or unwarranted. For this case study, we have not considered classes that attend the night shift. Furthermore, we have devised instances with (i) 30 time slots, comprising only classes that attend the morning or afternoon shifts, and; (ii) 60 time slots, comprising classes that attend both shifts.

As discussed in this section, we have modeled the HST problem at IFRN as to constraints, mathematical formulation, and real-world instances. Proposing a mathematical programming approach to benefit from our formulation exceeds the scope of this paper, and is planned as future work. In the next section, we propose an HST algorithm tailored for IFRN, which we later benchmark on the instances we provide.

IV. Proposed algorithm

As previously discussed, the HST is an NP-Hard problem for which (meta)heuristic algorithms are typically the method-of-choice. In addition, the specificities of the educational systems render customized algorithms important when addressing the HST at a given institution. In this section, we propose an algorithm based on the GRASP metaheuristic [9] for the HST at IFRN. GRASP is an effective yet simple metaheuristic

Algorithm 1 GRASP

Require: P, A, S, H
1: $LB \leftarrow$ splitSubjects(S,H)
2: **for** $i \leftarrow 1$ **to** *maxItr* **do**
3: $\tau \leftarrow$ construct(P, A, LB)
4: **if** τ **then**
5: **return** refine(A, τ)
Ensure: timetable τ

Algorithm 2 Refinement

Require: A, timetable τ
1: fill(τ)
2: **for** $i \leftarrow 1$ **to** δ **do**
3: b_1, $b_2 \leftarrow$ selection(τ)
4: **if** b_2 **then**
5: $\tau' \leftarrow$ swap(b_1, b_2, τ)
6: evaluate(τ, τ')
7: move(τ)
Ensure: timetable τ

that has been successfully employed in timetabling problems and combines constructive and refinement procedures. For simplicity, we initially describe the GRASP metaheuristic and the refinement procedure we employ, which follows the existing HST literature [7]. Later, we detail the constructive procedure tailored for the HST at IFRN.

A. The GRASP metaheuristic

GRASP [9] is a multi-start metaheuristic that combines a greedy randomized adaptive constructive procedure with a refinement procedure based on local search. GRASP has been successfully applied to several timetabling problems [10], and is especially suited for constrained problems like the HST. To regulate the exploitation/exploration trade-off, GRASP presents a numerical (hyper)parameter α that must be configured on a problem-basis. Alternatively, reactive GRASP [9] self-adjusts α, as we do in this work.

The GRASP algorithm we propose for the HST at IFRN is given in Algorithm 1. For consistency, we describe its input using the same notation adopted in the mathematical formulation. Before proceeding to its main loop, procedure splitSubjects processes the input data, returning a set of lesson blocks LB respecting hard constraint $H6$. Details on this procedure are provided as supplementary material [14]. The main cycle given in Lines 2–5 alternates between solution construction and refinement. For simplicity, we initially describe solution refinement, given in Algorithm 2 and detailed below.

B. Refinement

Given teacher availability and a feasible timetable τ, the refinement procedure exploits its neighborhood using the *event swap* (ES) operator [7], depicted in Figure 1. In that example, an event is a single lesson; more generally, we consider that lesson blocks can be swapped, as in the *event block swap* operator [7]. Furthermore, we fill the empty slots in τ with dummy lessons (Line 1) to render the ES operator also equivalent to the *event move* operator [7]. Neighborhood exploitation (Lines 2–6) is performed for a given number δ of local search iterations. Once exploration is concluded, we remove dummy slots and move lesson blocks up or down within the given shifts to remove windows (Line 7).

The loop given in Lines 2–6 comprises three procedures. First, procedure selection attempts to select two compatible lesson blocks for swap. In detail, the first block (b_1) is selected at random. The second block (b_2) must match the size of b_1, respect teacher availability, and target a different subject. If b_2 exists, a neighbor solution τ' is produced in Line 5 and evaluated in Line 6. Since we adopt a first-improvement pivoting rule, τ is replaced by τ' within procedure evaluate if the latter outperforms the former according to the following evaluation function:

$$f(s) = (I_p \cdot W_p) + (I_c \cdot W_c) + (\phi \cdot v) \tag{10}$$

where v corresponds to the number of hard constraint violations that swap operations may produce. Weights I_p, I_c, and, ϕ are exposed as (hyper)parameters.

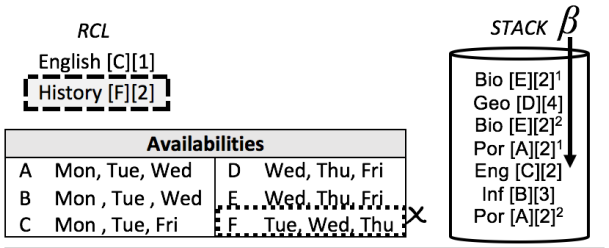
Algorithm 3 Construction

Require: P, LB
1: **while** LB **and** γ **do**
2: **if** *RCL* = $\emptyset$ **or** *deadend* **then**
3: *RCL* $\leftarrow$ buildRCL(LB, α)
4: increaseAlpha($\Delta\,\alpha$)
5: b, $t \leftarrow$ selectBlockTime(*RCL*)
6: *deadend* $\leftarrow t = \emptyset$
7: **if not** *deadend* **then**
8: allocate(b, t, τ, LB, *stack*)
9: **else**
10: backtrack(β, *stack*, τ, LB, γ)
Ensure: timetable τ **if** $LB = \emptyset$

C. Construction

The constructive procedure is given in Algorithm 3, which executes a maximum γ attempts to build a feasible timetable from the lesson blocks provided in LB (Lines 1–10). In particular, we say that the algorithm has reached a *dead end* if no further lesson block can be scheduled without violating hard constraints. For every unsuccessful attempt, the algorithm self-adjusts to increase its success odds. If γ is reached, no timetable is returned.

The success odds for building a feasible solution are regulated by the (hyper)parameter α, as follows. In GRASP algorithms, the constructive procedure is greedy, randomized, and adaptive. This is implemented through a restricted candidate list (RCL). In detail, the RCL is greedily built according to a given problem-specific heuristic. Yet, the next solution component to be assigned is randomly chosen from the RCL, helping to avoid local optima and/or violate problem constraints. In this approach [10], we build the RCL at the beginning of each attempt (Lines 2–3). The greedy maximization function $g(b)$ for each lesson block $b \in LB$ is defined as:



Mon	Tue	Wed	Thu	Fri
Por [A][2]^2		Por [A][2]^1	Geo [D][4]	Bio [E][2]^1
Por [A][2]^2	Inf [B][3]	Por [A][2]^1	Geo [D][4]	Bio [E][2]^1
Eng [C][2]	Inf [B][3]	Bio [E][2]^2	Geo [D][4]	
Eng [C][2]	Inf [B][3]	Bio [E][2]^2	Geo [D][4]	

Fig. 2: Example of timetable construction

$$g(s) = \lambda \cdot g_1(p) + \mu \cdot g_2(p) + \nu \cdot g_3(b) \tag{11}$$

where λ, μ, and ν are weights for the auxiliary functions g_1, g_2, and g_3, and $p \in P$ is the teacher for lesson block b. $g_1(p)$ and $g_2(p)$ respectively count the teacher's (i) number of unavailable days and (ii) weekly workload in the shift of b, whereas $g_3(b)$ returns the course load of the subject of b. As such, teachers with higher unavailability and workloads and subjects with higher loads are favored. The RCL comprises lesson blocks $b \in LB$ for which $g(b) \in [\overline{b}, \overline{b} + \alpha(\underline{b} - \overline{b})]$, where $\overline{b}$ and $\underline{b}$ are the lesson blocks in LB with maximum and minimum g values, respectively. Parameter $\alpha \in [0, 1]$ regulates RCL size, and hence the odds of building a feasible timetable. In the extreme scenarios, RCL comprises only lesson blocks with maximum value for g ($\alpha = 0$) or all lesson blocks available for allocation ($\alpha = 1$). In this work, we start from $\alpha = 0$, and increase it by $\Delta\,\alpha$ everytime the RCL is (re)built (Line 4).

At Line 5, procedure selectBlockTime randomly selects a lesson block b, and searches for a feasible time slot t for b. If t exists (Lines 6–8), procedure allocate assigns b to τ, removing it from LB. Else, a backtracking procedure is employed to reverse the last β assignments recorded in a history stack (Lines 9–10). This is illustrated in Fig. 2, where the block selected from the RCL comprises two History lessons from teacher F, who is not available on the day with two empty slots (Friday). Backtracking enables building an alternative, feasible timetable.

In this section, we have described a GRASP algorithm tailored for the HST at IFRN. Besides the inherent (hyper)parameters of GRASP, we have exposed further (hyper)parameters for configuration on a problem-basis. In the next section, we detail how we configure and benchmark our proposed approach, and discuss how it compares to commercial and state-of-the-art HST optimizers.

V. Validation

The experimental analysis we conduct in this section has two major goals: (i) to demonstrate the benefits of the real-world instances we produce, benchmarking existing state-of-the-art approaches on this dataset, and; (ii) to validate the proposed GRASP algorithm, assessing how it compares to the literature and the importance of its components. Initially, we describe how we configure our proposal and benchmark algorithms. Later, we discuss results both from benchmarking and from ablating our proposed approach.

A. Configuration and benchmarking setup

Our initial experiments considered a single GRASP iteration to understand the effectiveness of its components. However, as we will later discuss in this section the results observed for *maxItr* = 1 were already remarkably good. As such, we leave the analysis of multiple-iteration executions for future work. Considering the number of other (hyper)parameters we expose, we employ the irace heuristic configurator [15], as follows.

Problem sampling. We use holdout to separate between training and testing instances with a 60%/40% ratio.[1] Considering the limited number of training instances, we followed [16] and created meta-instances, each containing three training instances sampled without repetition. Performance on a meta-instance is the average performance from its comprising instances.

Configuration space. For self-adjusment, we consider $\Delta\alpha \in [0.001, 0.05]$, with an initial $\alpha = 0$. For the heuristic function $g(b)$, we consider $\lambda, \mu, \nu \in \{1, \ldots, 15\}$. Finally, the backtracking procedure removes $\beta \in \{25, 35, \ldots, 85\}$ lessons from the allocation stack for a maximum $\gamma \in \{1\,500, 2\,000, \ldots, 6\,500\}$ times. In total, this parameter space comprises 1.3×10^7 configurations.

Configuration setup. The proposed algorithm was implemented in Java and executed on a single core of an Intel Core i5 @ 2.3 GHz with 4MB of cache running the macOS 10.13 operating system. irace could execute our algorithm 1.000 times, each limited to four CPU minutes. Configurations were evaluated using the goal function given in Eq. 10 with weights $I_p = 1$, $I_c = 100$, and $\phi = 10000$. In a given irace iteration, configurations could be discarded after three meta-instances were executed. Under these experimental conditions, irace was able to evaluate 151 configurations and its total execution took five CPU days. The selected configuration is provided as supplementary material [14], for brevity, and discussed along this section when relevant.

The evaluation metric, stopping criterion, and execution environment used for benchmarking are the same adopted for (hyper)parameter configuration. Concerning algorithms, we used the KHE heuristic as baseline to evaluate our proposed constructive procedure. To compare final timetables, we benchmark the: (i) HS algorithm [13]; (ii) KHE heuristic coupled with our proposed refinement procedure, and; (iii) commercial software adopted at the given IFRN campus. Furthermore, considering the stochastic nature of heuristic optimization algorithms, multiple repetitions per instance are necessary for an analysis of performance variability. In this validation, each algorithm that has a stochastic component was executed 10 times per testing instance.

[1]Training: CA182M, JC181M, JC182MT, IP181M, PF181M, PF182M. Testing: CA182MT, JC181MT, JC182M, PF182T.

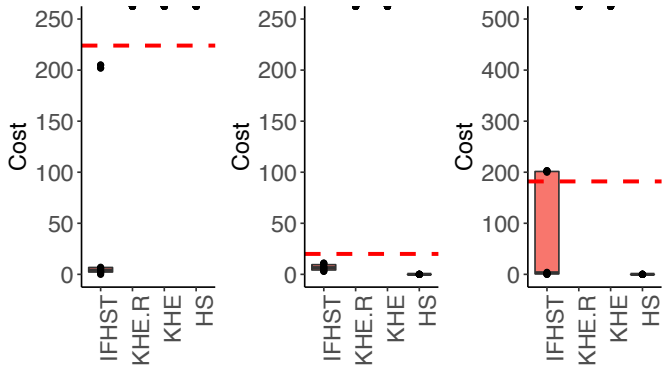


Fig. 3: Algorithm benchmark performance

B. Benchmarking results

Test instance results are given on the boxplots depicted in Figure 3, which compare the performance (y-axis) of the different algorithms considered in this work (x-axis) in each test instance. Only three instances are given, as no algorithm was able to produce feasible timetables for CA182MT. We remark the y-axis range varies per graph depending on the difficulty presented by the given instance. The dashed horizontal line indicates the quality of the solution produced by the commercial software that originated the timetable deployed at the given campus and semester, which we used to make the instance. Note that the solution quality reported for commercial software disregards penalties for violations we consider for other algorithms, since it is not possible to assess all hard constraints from the timetables deployed. The GRASP algorithm is labeled IFHST, whereas KHE coupled with our refinement procedure is labeled KHE.R. Points at the top edge of a given plot represent incomplete or unfeasible solutions. Next, we discuss the main insights observed.

IFHST produced feasible solutions for three out of the four testing instances considered. In general, solutions returned did not present student windows and had a very reduced number of teacher windows. Additional experiments comparing IFHST performance on training and testing instances provided in the supplementary material [14] showed that instances from the same campus and number of shifts presented similar results. An ablation analysis of runtime and solution quality component importance is given later in this section.

KHE was unable to produce a complete and feasible solution for any of the testing instances. For the two-shift instances, KHE could not handle window minimization using the XHSTT modeling we adopted. Regarding JC182M and PF182T, solutions produced were respectively unfeasible and incomplete. The issues presented by KHE could not be repaired by our refinement procedure on the testing instances. This was only observed for two training instances, which we report in the supplementary material [14].

HS delivered optimal solutions with a negligible runtime when it was able to produce a feasible and complete solution, namely for the single-shift instances. The inability of HS to produce feasible and complete solutions for the two-shift instances is in part justified by using KHE as a constructive step. Compared to IFHST, we observe that both approaches can be considered competitive. In detail, IFHST was the only algorithm able to produce a feasible solution for a two-shift instance (JC181MT). For JC182M, both algorithms presented solutions with high quality. Finally, HS performed better than IFHST on PF182T. Yet, despite the variance in IFHST results, the median cost remained close to zero. Finally, we remark that IFHST and HS were implemented in different programming languages, preventing a direct runtime comparison.

Commercial software results were in general outperformed by IFHST and HS. In detail, URÂNIA produced the timetables for the JC181MT and JC182M campus, whereas ascTimetables produced the timetables for the PF182T campus. Only for PF182T the commercial solution was better than a few IFHST solutions. It is important to note that commercial software are designed in a general way, without considering variations in hard restrictions among different institutions.

C. Ablation analysis

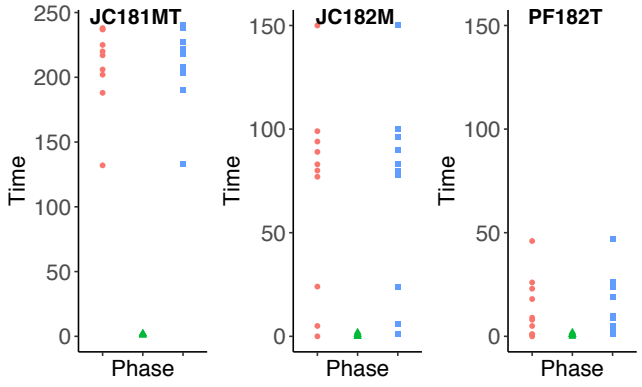
To further understand the contributions from IFHST procedures, we conduct an ablation analysis on the testing instances given in Figure 4. On the top, scatter plots show the runtimes from the constructive and refinement procedures individually and combined. On the bottom, boxplots give the solution cost produced by the constructive procedure and later by the swap and move neighborhood operators. We remark the y-axis range varies per graph depending on the difficulty presented by the different instances considered.

Regarding execution times, we observe that most of the computational effort is due to the constructive procedure. This is explained by the difficulty in producing feasible timetables, reflected in the configuration selected by irace with $\beta = 65$, $\gamma = 3\,500$, and $\Delta\,\alpha = 0.001$. Concerning solution cost, we note that the solutions returned by the constructive procedure are good regarding teacher windows and could be further improved regarding student windows. The quality of these solutions reflect the configuration selected for irace for the (hyper)parameters of the heuristic function $g(b)$, with $\lambda = 2, \mu = 7,$ and $\nu = 9$. Student windows are further improved for all instances by the refinement procedure. Furthermore, only for instance JC182M the move operator is unable to further improve results obtained with the swap operator.

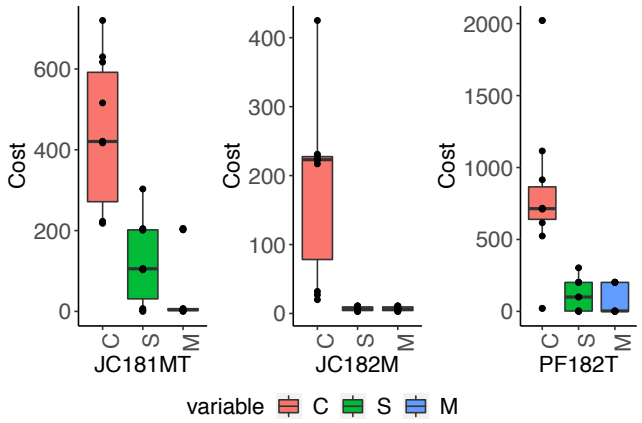
Overall, the analysis we have presented in this section confirms the contributions of our work. For most instances, we were able to benchmark state-of-the-art algorithms. More importantly, our proposed algorithm was able to solve more instances than the remaining algorithms benchmarked, displaying a competitive performance. Furthermore, we have observed how its individual components contribute to its performance, indicating possibilities for future improvements.

VI. Conclusion

Rede Federal de Educação Profissional, Científica e Tecnológica is one of the largest educational networks in Brazil and in the world. Its role in social equality has been established since the beginning of the 20th century, and over the past decades it has been significantly expanded. Network importance is even more pressing in low-income states such



(a) Runtime



(b) Solution cost

Fig. 4: Ablation analysis of GRASP component. C: constructive procedure; S: swap operator; M: move operator.

as Rio Grande do Norte, where *Instituto Federal de Educação, Ciência e Tecnologia* (IFRN) is located. Given the high qualification of its staff and the broad reach of its courses, optimizing resources in this network is a critical and demanding task.

In this work, we have modeled the high school timetabling (HST) problem at IFRN, identifying constraints, proposing a mathematical formulation, and devising ten real-world instances using the international XHSTT format [6]. Moreover, the algorithm we proposed was able to produce feasible timetables within the time limit adopted for most of the real-world instances considered, showing competitive performance with the state-of-the-art. This in line with results from (i) international competitions, where even the best algorithms cannot produce viable solutions for all datasets, and; (ii) the literature approaches considered, which could not solve instances with multiple shifts. Furthermore, our solution has been made configurable for decision makers regarding the importance of teacher and student windows. Nevertheless, the proposed algorithm was able to produce timetables with (close to) zero teacher and student windows.

Our work is seminal in many relevant ways. First and foremost, future work should focus on automating instance production to devise a dataset comprising most units from the network, including also scenarios with three shifts. In this context, novel constraints and exogenous data should be addressed, as well as automated validation to ensure the existence of feasible solutions. Algorithm-wise, future work should (i) investigate mathematical (hybrid) approaches for the proposed formulation, given the successful literature on the topic; (ii) render the proposed algorithm applicable to the general dataset, with appropriate adjustments as needed, and; (iii) investigate multiple-iteration scenarios, for instance allowing construction to accept violations to be repaired by refinement, potentially balancing the computational cost of the algorithmic components.

References

[1] R. J. Willemen, "School timetable construction: algorithms and complexity," Ph.D. dissertation, Technische Universiteit Eindhoven, 2002.

[2] T. B. Cooper and J. H. Kingston, "The solution of real instances of the timetabling problem," *The Comput. J.*, vol. 36, no. 7, pp. 645–653, 1993.

[3] G. H. Fonseca, H. G. Santos, and E. G. Carrano, "Late acceptance hill-climbing for high school timetabling," *J. of Sched.*, vol. 19, no. 4, pp. 453–465, 2016.

[4] L. Saviniec and A. A. Constantino, "Effective local search algorithms for high school timetabling problems," *Applied Soft Computing*, vol. 60, pp. 363–373, 2017.

[5] N. Pillay, "A survey of school timetabling research," *Ann. of Oper. Res.*, vol. 218, no. 1, pp. 261–293, 2014.

[6] G. Post, S. Ahmadi, S. Daskalaki, J. H. Kingston, J. Kyngas, C. Nurmi, and D. Ranson, "An XML format for benchmarks in high school timetabling," *Ann. of Oper. Res.*, vol. 194, no. 1, pp. 385–397, 2012.

[7] G. H. G. da Fonseca, H. G. Santos, T. Â. M. Toffolo, S. S. Brito, and M. J. F. Souza, "GOAL solver: a hybrid local search based solver for high school timetabling," *Ann. of Oper. Res.*, vol. 239, no. 1, pp. 77–97, 2016.

[8] J. H. Kingston, "Repairing high school timetables with polymorphic ejection chains," *Ann. of Oper. Res.*, vol. 239, no. 1, pp. 119–134, 2016.

[9] M. Gendreau, J.-Y. Potvin *et al.*, *Handbook of metaheuristics*. Springer, 2010, vol. 2.

[10] A. V. Moura and R. A. Scaraficci, "A GRASP strategy for a more constrained school timetabling problem," *Intern. J. of Oper. Res.*, vol. 7, no. 2, pp. 152–170, 2010.

[11] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 268–308, 2003.

[12] A. Schaerf, "A survey of automated timetabling," *Artif. Intell. Rev.*, vol. 13, no. 2, pp. 87–127, 1999.

[13] G. H. Fonseca, H. G. Santos, and E. G. Carrano, "Integrating matheuristics and metaheuristics for timetabling," *Comput. & Oper. Res.*, vol. 74, pp. 108–117, 2016.

[14] L. H. A. Dantas, L. C. T. Bezerra, and R. C. Andrade, "High school timetabling at a federal educational institute in brazil (supplementary material)," 2022. [Online]. Available: https://anonymous.4open.science/r/ifhst-supp/

[15] M. López-Ibánez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle, "The irace package: Iterated racing for automatic algorithm configuration," *Oper. Res. Persp.*, vol. 3, pp. 43–58, 2016.

[16] C. Vieira, A. d. Araújo, J. E. Andrade Júnior, and L. C. T. Bezerra, "isklearn: automated machine learning with irace," in *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2021.