High school timetabling at a federal educational institute in Brazil (supplementary material)

No Author Given

No Institute Given

1 Mathematical formulation auxiliary functions

The auxiliary functions operationalize the counting of class windows of professors and students for the goal function. The functions $L_i(t)$ and $L_f(t)$ determine, respectively, the initial and final limits for each of the three possible class shifts from a time $t \in T$:

$$L_i(t) = \begin{cases} 1, & \text{if } t \in \{1, \ldots, 6\} \\ 7, & \text{if } t \in \{7, \ldots, 12\} \\ 13, & \text{otherwise} \end{cases} \tag{1}$$

$$L_f(t) = \begin{cases} 6, & \text{if } t \in \{1, \ldots, 6\} \\ 12, & \text{if } t \in \{7, \ldots, 12\} \\ 16, & \text{otherwise} \end{cases} \tag{2}$$

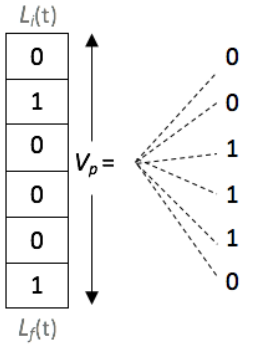
The $V_p(p, d, t)$ function checks if the class schedule for the time t on a school day d for which there are no classes allocated to the professor p ($\sum_{s \in S} x_{psdt} = 0$) is in fact a window for that professor. For a given time t on a school day d, the function returns the value 1 if previous classes are allocated ($\sum_{s \in S} \sum_{w=t-1}^{L_i(t)} x_{psdw} > 0$) and next ($\sum_{s \in S} \sum_{w=t+1}^{L_f(t)} x_{psdw} > 0$) for professor p, featuring t as a window for this professor. Otherwise, t is not considered a p window and the function return 0.

$$V_p(p, d, t) = \begin{cases} 1, & \text{if } \sum_{s \in S} \sum_{w=t-1}^{L_i(t)} x_{psdw} > 0 \text{ and } \sum_{s \in S} \sum_{w=t+1}^{L_f(t)} x_{psdw} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

The purpose of the auxiliary function $V_p(p, d, t)$ is illustrated (see Fig. 1), shows a professor's class allocation for a school day containing only two lessons, indicated by number 1, within the 6 possible times in the shift. The first time isn't considered a window, although there's no class allocated, since there are no lessons allocated before this time in the shift. The remaining times without allocated lessons are considered windows.

Similarly, the $V_c(c, d, t)$ function checks if the class schedule for the time t on the school day d for which there are no classes allocated to a class c ($\sum_{p \in P} \sum_{s \in S_c} x_{psdt} = 0$) is in fact a class window.

Fig. 1: Example of lesson window patterns.



$$V_c(c,d,t) = \begin{cases} 1, & \text{if } \sum_{p \in P} \sum_{s \in S_c} \sum_{w=t-1}^{L_i(t)} x_{psdw} > 0 \text{ and } \sum_{p \in P} \sum_{s \in S_c} \sum_{w=t+1}^{L_f(t)} x_{psdw} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The auxiliary functions V_p and V_c are used by professor and class window accounting functions, respectively W_p and W_c. The $W_p(p,d)$ function uses the $V_p(p,d,t)$ function to account for the number of windows present in a professor's p time allocation on a school day d in each shift of class. More precisely, the function W_p runs through all times $t \in T$ and uses the function $V_p(p,d,t)$ to check if it's a window. The sum of the results returned by the auxiliary function indicates the number of class windows found in the allocation of professor p hours for the school day d. Multiplication ensures that only times without lesson allocation contribute to this calculation.

$$W_p(p,d) = \sum_{t \in T} \left(V_p(p,d,t) \cdot (1 - \sum_{s \in S} x_{psdt}) \right) \tag{5}$$

Similarly, the $J_c(c,d)$ function uses the $V_c(c,d,t)$ function to account for the number of windows present in the lesson allocation of a class c on a school day d.

$$W_c(c,d) = \sum_{t \in T} \left(V_c(c,d,t) \cdot (1 - \sum_{p \in P} \sum_{s \in S_c} x_{psdt}) \right) \tag{6}$$

2 Dataset modeling as XHSTT

The dataset constraints modeling was done using the XHSTT notation. The modeling of constraint *H6* (lesson distribution) is ensured by pre-processing the subjects according to the distribution preferences informed by the professors. Two types of resources were used to model this constraint: *groups* and *splits*[1].

Fig. 2: Example of teacher availability constraint in XHSTT

```
<AvoidUnavailableTimesConstraint Id="AvoidUnavailableTimes_Th_Fr">
    <Name>ForbiddenTimes_Th_Fr</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <Resources>
            <Resource Reference="T2"/>
            <Resource Reference="T3"/>
            <Resource Reference="T10"/>
        </Resources>
    </AppliesTo>
    <TimeGroups>
        <TimeGroup Reference="gr_Th"/>
        <TimeGroup Reference="gr_Fr"/>
    </TimeGroups>
</AvoidUnavailableTimesConstraint>
```

Specifically, a *split* promotes the division of an event, thus subjects lessons splitted and modeled as a group. The other models are described below.

H1 - Curriculum compliance: to indicate to the optimizer the need to associate all events with the times available for allocation, a constraint of type *AssignTime* was created applied to the group *gr_AllEvents*, which encompasses all lessons.

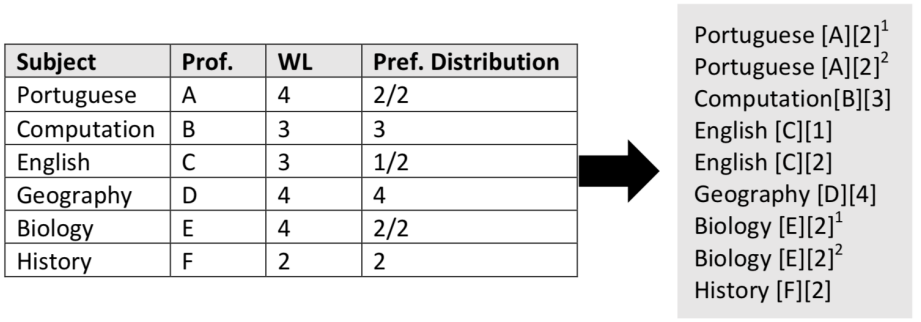
H2 & H3 - Teacher and class exclusivity: the constraint *AvoidClashes* was applied to two resources groups: (i) *gr_Teachers* and (ii) *gr_Classes*, which represent, respectively, all teachers and classes in the dataset.

H4 - Twin lessons: to avoid the lessons concentration of the same subject in a single school day, a constraint of type *SpreadEvents* was added, applied to all subjects that have more than one scheduled meeting. The restriction indicates that for the same subject, per day, there must be a maximum occurrence of one meeting, which does not exclude the possibility of having twinned classes.

H5 - Justified teacher availability: to reflect the teachers unavailability or preferences for specific days for teaching lessons, constraints of type *AvoidUnavailableTimes* were created. Each such constraint indicates the teacher-type resources associated with it, as well as the specific weekdays that should be avoided for allocating the referenced resources. Figure 2 shows the restriction of the teachers *T2*, *T3* and *T10* to three weekdays is exemplified. More specifically, lessons on Thursdays (*gr_Th*) and Fridays (*gr_Fr*) should be avoided. As it is identified with the value *true* in the *Required* attribute, this constraint must be met by the optimizer, as recommended by the *H5* problem constraint.

H7 - Maximum teacher daily lessons: the constraint of type *LimitBusyTimes* was used. The restriction was applied to the teachers group (*gr_Teachers*) and *TimeGroups* that represent the times of each weekday, limiting teacher-type resources to a maximum allocation of 10 lessons per school day.

Fig. 3: Subjects partitioning example: *Portuguese*, *English* and *Biology* subjects are subdivided to suit the lessons distribution requested by their respective professors.



Subject	Prof.	WL	Pref. Distribution
Portuguese	A	4	2/2
Computation	B	3	3
English	C	3	1/2
Geography	D	4	4
Biology	E	4	2/2
History	F	2	2

S1 & S2 - Student and teacher window minimization: for modeling window constraints two constraints of type *LimitIdleTimes* were added, which apply to the groups *gr_Classes* and *gr_Teachers*. In addition, they include the *TimeGroups* of each weekday.

3 Input data processing

The procedure splitSubjects processes the input data, returning a set of lesson blocks LB respecting hard constraint $H6$. For example, if a subject has six credits and is requested by its respective professor to separate them into twinned classes of two credits, the collection will now have three partitioned subjects, each containing two credits and under the ownership of the same professor. Additionally, the weekly workload of each professor is calculated based on the sum of credits for each subject assigned to them. Figure 3 illustrates the partitioning of subjects. From the distribution preferences of six disciplines, arranged in the fourth table column, a new collection is generated - to the right of image - now containing nine elements.

4 IFHST execution results for all instances

For an results overview obtained by the algorithm when submitted to the set of *datasets* produced in the modeling stage, the average values collected in different sensitive solution points are presented in the Table 1. Among all the instances, three of them were not solved by the proposed algorithm – such cases were present in the two highlighted scenarios. Complementarily, the column *Average T* represents the average time, calculated from the sum of execution times, divided by the total algorithm executions. The columns *Initial WC*, *Initial WP*, *Final WC* and *Final WP* represent, respectively, the average values obtained from the number of *C*lasses and *P*rofessors *W*indows after the construction and improvement phases.

Table 1: Run results for all instances

	Instance	Average T	Initial WC	Initial WP	Final WC	Final WP	Best T	Best WC	Best WP
TEST	CA182MT				–				
	JC181MT	2m57s	4,2	20,2	0,4	3,4	42s	0	0
	JC182M	1m11s	1,6	25,2	0	6,8	1s	0	3
	PF182T	15s	7,9	16,9	0,8	1,6	01s	0	0
TRAINING	CA182M				–				
	JC181M	2m17s	3	17,3	0	5,1	40s	0	0
	JC182MT	1m13s	4,2	22,2	0,1	3,8	7s	0	2
	IP181M	04s	7,3	19,7	0,8	8,2	01s	0	3
	PF181M	18s	2	28,1	0,1	9,2	02s	0	4
	PF182M				–				

5 Solution cost by heuristic method

The expanded Table 2 presents the solution costs of all methods for each of instances (test and training).

Table 2: Solution cost by heuristic method for window class (WC), window professor (WP) and violation (V)

	IFHST			**KHE + R**			**KHE**			**HS**		
	WC	WP	V	WC	WP	V	WC	WP	V	WC	WP	V
CA182M		-		0	2	0	0	2	0	0	0	0
CA182MT		-			-			-			-	
JC181M	0	5,1	0	0	0	0	0	0	**1**	0	0	**1**
JC181MT	0,4	3,4	0		-			-			-	
JC182M	0	6,8	0	0	0	**1**	0	0	**1**	0	0	0
JC182MT	0,1	3,8	0		-			-			-	
IP181M	0,8	8,2	0	0	0	**2**	0	0	**2**	0	0	0
PF181M	0,1	9,2	0	0	4	0	0	2	**1**	0	0	0
PF182M		-		0	0	**2**	0	0	**2**	0	0	0
PF182T	0,8	1,6	0		-			-		0	0	0

References

1. Kingston, J.H.: High school timetable data format specification (2018), http://www.it.usyd.edu.au/ jeff/cgi-bin/hseval.cgi?op=spec