REDOSPLAT DSL for timetabling requirements

Razija Turčinhodžić Mulahasanović ·
Samir Ribić

Abstract This paper presents a declarative language called REDOSPLAT intended to describe timetabling problems. It is designed for textual entry. Unlike most timetable data formats, REDOSPLAT uses easy to understand syntax based on sentences which describe requirements. DSL is complemented with a wizard which defines ordinary sentences.

Keywords Timetabling · DSL · REDOSPLAT

1 Introduction

Different formats and languages were developed to formulate requirements for timetabling problems. Some of them are line oriented languages [1], others mimic object-oriented programming languages [2], and some use language formats for expert systems [3]. XML is also base for wide number of languages [4]. In addition, there are languages similar to spoken languages. They are intuitive when it comes to data entry and include UniLang [5] and REDOSPLAT [6]. REDOSPLAT is a language that has been developed for about 10 years. Its latest version was introduced in 2018, and it is the topic of this paper. It is primarily made for school timetabling, but it can also be used for university and examination timetabling. It is a domain-specific declarative language whose sentences resemble those of spoken language. In our examples, the focus will be on school timetabling problems.

University of Sarajevo
Zmaja od Bosne bb, Sarajevo, Bosnia Herzegovina
Tel.: +387 33 250 700
Fax: +387 33 250 725
E-mail: samir.ribic@etf.unsa.ba, razija.turcinhodzic@etf.unsa.ba

2 Language scope

REDOSPLAT is designed to express different requirements for timetabling in schools across the world. It is used to describe time, resources, events, constraints, and solving goals. Time is expressible as days, working shifts (with a list of time slots they belong to) and time slots (with a day they belong to). Resources can have short and long names and they include sites (for schools in different cities and buildings), rooms (with optional capacity, price or site), classes (with optional default shift and the number of students), sub-class groups (with an optional number of students or percentage of class they belong to), joined multiple classes, teachers and courses (which can be grouped in categories). Event descriptions may be complex sentences, because describing events vary from simple ones (where the single teacher teaches a course to dedicated class), to events that allow selection between teachers, multiple teachers at the same time, joined classes, choice between rooms, lessons in multiple rooms, fixing time slots, spreading over the week, double and triple lessons, simultaneous lessons and lesson ordering. Constraints include unavailable time slots or days (for teachers, classes and rooms), wishes for time slots (soft constraint for teachers, classes and rooms with weight), default rooms (for teachers and classes), daily limits (minimal or maximal number of events per teacher/class/course category can be held in a day or shift), a prohibition that two teachers, rooms or classes can not be allocated at the same time, idle time policy (soft or hard) and travel time between sites. Preferred goals may be adjusted to define the main goal of solving (reducing idle time, reducing travel time, satisfy timeslot wishes, or make a balance between course category). Most of the requirements are optional, and do not need to be entered, but if the underlying solving engine supports it, they are used in the solution.

3 Language syntax examples

In schools, time is represented through time slots that generally have the same length. Time slots may belong to days and to shifts. Shifts are sets of time slots, usually morning or afternoon slots. These sentences describe one day, one timeslot and one shift.

```
Monday is day number 1.
MO1 is time slot 1 on Sunday.
SH1 is a shift consisting of MO1, ...
```

Here is an example of possible sentences describing classes, subgroups and teachers.

```
I is a class.
II is a class from shift SH1.
I-a is a subgroup of I.
JohnFein is a teacher.
```

Courses can be divided by categories so that some special restrictions, such as favouring difficult courses to occur in the earlier time slots, can be defined as well.

```
Mathematics is a course. Chem is course of category 2.
FEE is a course called "Fundamentals of electrical engineering".
```

Rooms may have their own capacity (number of seats), a fixed price if rented, and locations if lessons are taught in multiple buildings.

```
R1 is a room.
R2 is room with 30 seats costs 18.
FS is a room called "Sports Hall".
SwimmingPool is a building.
SP is located in SwimmingPool.
```

It is possible to assign a room to a particular teacher, class or subject. It is also possible to assign a teacher to the class that he or she teaches.

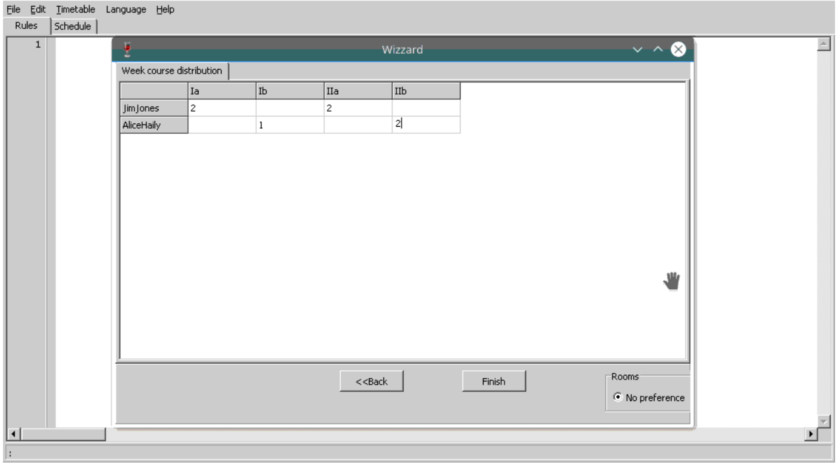
```
JohnFein is located in R1.
II is located in R2.
```

The main sentence creates an event (lesson). The event can have a teacher, a class, a room (or more of each of these resource), and the way lessons are spread over the week (doubles, triples). It is also possible to fix lessons of a course at predefined time slots if necessary.

```
KatinaFein teaches English to the group II1, II2
3 times a week, separately, with 1 double, in the room R1.
```

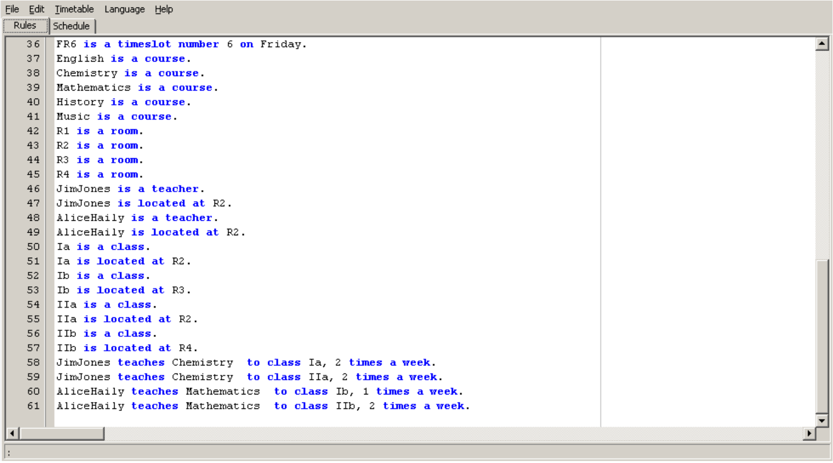
Following these basic requirements, special requirements may be added which include:

- unavailability of teachers, classes and rooms in designated time slots (this is a hard constraint);
  ```
  SportsHall is unavailable on MO10.
  JohnFein is unavailable on Friday.
  ```
- expressing the desire for the class, teacher or room to be used or not used within a certain time slot, which also sets the coefficient weight (soft constraint);
  ```
  JohnFein wishes time slots MO1 (200).
  ```
- requiring specific time slots by teachers, classes or rooms;
  ```
  ValeriBrowning requests time slot MO2.
  ```
- restrictions on the minimum and maximum hours for classes and teachers (and rooms) during the day or shift;
  ```
  Ia has between 5 and 7 lessons.
  NencyGreen has a maximum of 0 lessons on Friday.
  ```
- limiting the number of time breaks for teachers and classes;
  ```
  Pauses for I are forbidden.
  Pauses for ValerieBrowning are limited to 1.
  ```
- regulating special time slot overlaps;
  ```
  MO2 overlaps with MO1.
  ```
- regulation of special prohibition of simultaneous lessons for some teachers, classes and rooms;
  ```
  II1 conflicts with II2.
  ```
- class aggregation;
  ```
  CompetitionMath is composed of I-a, II1.
  ```
- travel time between locations;
  ```
  Travel time between SwimmingPool and MainBuilding is 2 timeslots.
  ```



	Ia	Ib	IIa	IIb
JimJones	2		2	
AliceHaily		1		2

● No preference



```
FR6 is a timeslot number 6 on Friday.
English is a course.
Chemistry is a course.
Mathematics is a course.
History is a course.
Music is a course.
R1 is a room.
R2 is a room.
R3 is a room.
R4 is a room.
JimJones is a teacher.
JimJones is located at R2.
AliceHaily is a teacher.
AliceHaily is located at R2.
Ia is a class.
Ia is located at R2.
Ib is a class.
Ib is located at R3.
IIa is a class.
IIa is located at R2.
IIb is a class.
IIb is located at R4.
JimJones teaches Chemistry  to class Ia, 2 times a week.
JimJones teaches Chemistry  to class IIa, 2 times a week.
AliceHaily teaches Mathematics  to class Ib, 1 times a week.
AliceHaily teaches Mathematics  to class IIb, 2 times a week.
```

4 IDE, wizzard, solving and usage

Although easy to read, typing in multiple similar sentences can be boring. To simmplify entry of sentences, a wizzard-based GUI was created allowing quick generation of basic constraints (time, teachers, classes, courses, rooms, capacities, workloads).

However, less used constrains are not covered by GUI, because they will tend to clutter and slow down the entry. Specific requirements and exceptions to the default values are entered subsequently by modifying the generated sentences and entering new ones. The editor includes syntax highlighting.

After selecting the algorithm and all other parameters needed to solve it, a timetable is generated in the case of syntactically correct instances. The solution can be presented in HTML, TABLE and XHSTT format. Conversion from XHSTT to REDOSPLAT is also possible. The solution includes lists of all requirements that are not met. REDOSPLAT is integrated with KHE engine [7] based solvers [8] or linear programming based solvers [9] and multiple algorithms for automatic timetable generation.

REDOSPLAT was tested at 16 local schools, of which 13 schools have shifts. Within the selected schools we have 7 primary schools, 3 grammar schools, 4 mixed secondary schools, 1 technical secondary school and 1 university department. They vary in effort required to express the demands. The easiest school has 20 classes with about 500 events, and works in 2 shifts. The most difficult school has 56 classes and about 2500 events and it works in 4 shifts (2 for students and 2 for teachers). The university department has about 150 courses and 1900 events. Schools were visited 2-3 times, the first and second time data were taken and the language was presented, and the second or third visit was used to present a solution. The data entry for smaller schools lasted 2-3 hours, including verification of entered data. The entry for larger schools took 2-3 days, but the duration of data entry is not longer than the time that schools normally need.

5 Concluding remark

REDOSPLAT is GPL licensed tool and it is available at
https://sourceforge.net/projects/redosplat/

We invite the interested reader to use and contribute to our software: by adding better algorithms, speeding up the engine, suggesting new syntax elements to describe requirements that exist in some countries etc.

The simplified syntax may attract researchers to contribute with more real-life examples for time table requirements.

References

1. L. Di Gaspero B. McCollum A. Schaerf., The second international timetabling competition (ITC-2007): Curriculum based course timetabling (track 3), Technical Report QUB/IEEE/Tech/,,2007
2. E.K.Burke J.H.Kingston P.A.Pepper, A standard data format for timetabling instances, International Conference on the Practice and Theory of Automated Timetabling,213-222, 1997
3. L. F. Lai LF C. C. Wu et. al. , An artificial intelligence approach to course timetabling, International Journal on Artificial Intelligence Tools, 17(01):223-240, 2008.
4. G. Post G S. Ahmadi et. al, An XML format for benchmarks in high school timetabling, Annals of Operations Research, 194(1), 385-397, 2012
5. L. P. Reis E. Oliveira , Language for specifying complete timetabling problems, International Conference on the Practice and Theory of Automated Timetabling, 322-341, 2000
6. S. Ribić R Turčinhodžić, A. Muratović-Ribić, T. Kosar,REDOSPLAT: A readable domain-specific language for timetabling requirements definition, Computer Languages Systems and Structures, 54, 252-272, 2018
7. H. Kingston, The khe high school timetabling engine,, 2010
8. G. H. Fonseca H. G. Santos E. G. Carrano, Integrating matheuristics and metaheuristics for timetabling, Computers and Operations Research, 74, 108-117,2016
9. S. Ribić R. Turčinhožić A. Muratović-Ribić, Modelling constraints in school timetabling using integer linear programming, 2015 XXV International Conference on Information Communication and Automation Technologies (ICAT), (pp. 1-6), 2015