Declarative Programming project: Exam timetabling

Manual

Arno Moonens
Studentnr.: 500513
arno.moonens@vub.ac.be

1. Introduction

In this document, I will explain how to run each predicate of my solution. First I will say how to load the code, after which I will list all the predicates that can be run and with which arguments.

2. Loading the code and problem instance

First, the *Swipl* program should be started. This can be done in multiple ways depending on your installation. On Linux or Mac, one can use *Terminal* to go to the directory of my project and execute `swipl` there.
Next, he `load_code.pl` file should be consulted. If you started *Swipl* using *Terminal* in the project directory, it is sufficient to execute `consult(load_code)`.
The instance to be used also needs to be consulted. This can be done before or after consulting the `load_code.pl` file.
When this is all done, it is possible to run the predicates listed in Section 3.

3. Usable predicates

The following predicates can all be executed after loading the code and problem instance as defined in Section 2. In the arguments, one can see that `+`, `-` and `?` are used. `+` means that the variable already needs to be instantiated, while this isn't the case when a `-` precedes the variable. When a `?` is used, it doesn't matter if the variable following this character is already instantiated or not.

There are the executable predicates:

- `is_valid(?S)`
- `cost(+S,?Cost)`
- `violates_sc(+S,-SC)`
- `find_optimal(-S)`

- `is_optimal(?S)`
- `find_heuristically(-S)`
- `find_heuristically(-S,+T)`
- `pretty_print(+S)`
- `pretty_print(+SID,+S)`

4. Example run

Here an example run is given in the *Terminal* program on a *Mac* operating system. Note that the directory in which we start contains the files all the project files and problem instances files. We first start *SWI-Prolog* using the `swipl` command, after which we execute some commands as an example, namely `is_valid` to generate a valid schedule, `is_valid` together with `cost` and `find_optimal`.

```
MacBook-Air-van-Arno:Project arnomoonens$ swipl
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 7.2.3)
Copyright (c) 1990-2015 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

?- consult(small_instance).
true.

?- consult(load_code).
true.

?- is_valid(X).
X = schedule([event(e1, r2, 4, 10), event(e2, r2, 3, 12), event(e3, r1, 3, 10),
event(e4, r1, 2, 10), event(e5, r2, 1, 10)]) .

?- is_valid(X),cost(X,Cost).
X = schedule([event(e1, r2, 4, 10), event(e2, r2, 3, 12), event(e3, r1, 3, 10),
event(e4, r1, 2, 10), event(e5, r2, 1, 10)]),
Cost = 5.875 .

?- find_optimal(X).
X = schedule([event(e1, r2, 2, 10), event(e2, r2, 5, 10), event(e3, r1, 4, 10),
event(e4, r2, 4, 10), event(e5, r2, 3, 13)]).
```