Reasoning over student-related study regulations hard and soft constraints in ASP (Extended Abstract)*

Henry Otunuya[0000−0002−1012−7887]

University of Potsdam, Germany

Introduction

In Hahn et al. 2024, we presented a set-based mathematical structure for a study regulations problem. There we defined a *basic study regulation problem with examination tasks* as a pair of study plan and examinations plan.

With that study regulations structure, students are able to get study plans which satisfy the general constraints present in the study regulations legal document.

Usually study regulations leave room for additional constraints, on top of what is given in the legal document, which may be tailored to the varied needs of individual students. These additional constraints relate to both modules and examinations.

This abstract highlights some current work built on what was presented in Hahn et al. 2024. In particular, the following student-related constraints have been formalized and encoded in ASP[1]:

- Semester-to-ECTS credit bound (**C1**)
- Module-to-semester bound (**C2**)
- Semester-to-number of modules bound (**C3**)
- Number of semesters bound (**C4**)
- Semester-to-number of examinations bound (**C5**)
- Liked and disliked modules (**C6**)
- Liked and disliked module keywords (**C7**)
- Shortest study plan (**C8**)
- Module dependency with shortest gap in between (**C9**)

Figure 1 shows the current user interface implementation, made with *clinguin*[2]. Labels[3] were added to the screenshot to aid the following explanation.

With study plans shown in the middle of the user interface, labels **C1** to **C4** show HTML cards for the first four constraints. **C1** to **C3** have three input

* This is preliminary work.
[1] Only the first four are presented here.
[2] `https://clinguin.readthedocs.io/en/latest/` (last accessed: November 2025)
[3] That is, **C1**, **C2**, **C3**, **C4** and **Ac**.

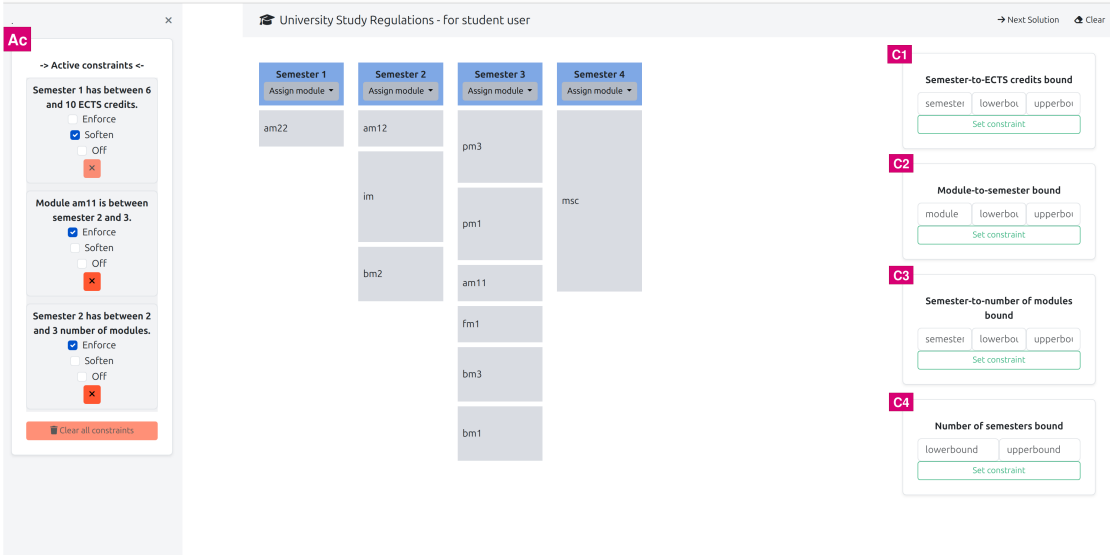

Fig. 1. User interface with student-related constraints

fields and **C4** two fields. With **C1**, a student is able to enter a semester with the lowerbound and upperbound ECTS credits values. Clicking on *Set constraint* introduces a constraint into the solution space.

With **C2**, a student is able to enter a module and set the lower and upper bound of its semester values. With **C3**, a student is able to enter a semester with the lower and upper bound number of modules for that semester. And lastly with **C4**, the lower and upper bound semester values can be set.

The **Ac** window shows the constraints which were set by clicking a *Set constraint* button. Each of the three cards in display share the same structure. First is the constraint in English sentence. It is followed by the *Enforce*, *Soften* and *Off* checkboxes. The first two checkboxes behave like HTML radio-buttons, that is, a constraint is either in *Enforce* or in *Soften* and not in both. As can be observed, the first card shows a **C1** constraint where for semester 1, 6 and 10 were set as the lowerbound and the upperbound respectively.

Of the three checkboxes, *Enforce* is the default, that is, it is checked when a *Set constraint* button is clicked. *Soften*, when checked, turns the associated constraint into an ASP weak constraint. The weak constraint minimizes an objective function shared among constraints **C1** to **C4**. This objective function essentially holds the deviation from a given lower and upper bound.

Checkbox *Off* when checked deactivates the associated constraint. With it a user is able to toggle on and off any constraint.

And next, the button with × when clicked, removes a constraint from the solution space and from the user interface. Similarly, the *Clear all constraints*

button removes all active constraints from the solution space and from the user interface.

In closing, to the best of my knowledge this is the first time these constraints are formalized, encoded in ASP and visualized with *clinguin*. Some relate works include Baldoni et al. 2011, Wagner et al. 2023 and Richards and Tsay 2020.

Acknowledgements This work is funded by the German Federal Ministry of Research, Technology and Space under contract number 16DHBKI024. I would like to thank Sebastian Schellhorn for assistance in the developement of the mathematical formalizations and Javier Romero for assistance with the weak constraints conceptualization of this work.

References

Baldoni, M. et al. (2011). "Constraint modeling for curriculum planning and validation". In: *Interactive Learning Environments* 19.1, pp. 81–123.

Hahn, S. et al. (2024). "Reasoning About Study Regulations in Answer Set Programming". In: *Theory and Practice of Logic Programming* 24.4, pp. 790–804. DOI: 10.1017/S1471068424000383.

Richards, A. L. and R. Tsay (2020). "An Optimal Slack-Based Course Scheduling Algorithm for Personalised Study Plans". In: *Proceedings of the 2020 9th International Conference on Educational and Information Technology*, pp. 1–7.

Wagner, M. et al. (2023). "A Combined Approach of Process Mining and Rule-based AI for Study Planning and Monitoring in Higher Education". In: *Proceedings of the International Conference on Process Mining (ICPM'22): Process Mining Workshops*. Springer-Verlag, pp. 513–525.