Reasoning over Student Study Plans with Answer Set Programming

Henry Otunuya | Stefan Lindow | Mariia Merzliakova

University of Potsdam, Germany

19 September, 2025

HS2035@Informatik Festival 2025

Gefördert durch:

Bundesministerium
für Forschung, Technologie
und Raumfahrt

Outline

- Aim & Objectives
- University study regulations
- Study regulations modelling with SemaLogic
- Our *ASPSR* service
- Questions and Requests on the Study Regulations Model
- Summary
- Conclusion
- Acknowledgement

Aim & Objectives

- **Aim:**
  - To reason with student-based constraints over university study regulations.

- **Objectives:**
  - Model university study regulations in a computer interpretable manner.
  - Convert interview-retrieved questions and requests into interpretable constraints.
  - Introduce constraints into the study regulations model.

University study regulations

University study regulations

- University study regulations are legal documents which specify requirements needed to earn a degree.
- These requirements may include: modules, courses and examinations.
- The choice of what modules to take usually lie in the hands of the student.

University study regulations

- These choices, based on the number of modules involved, usually induce an explosion of possible sets of modules to choose from, resulting in several alternative study plans.
- Combination of modules which possess certain characteristics is desirable, however it can be an overwhelming task.

University study regulations

- For example:
  - a study plan within a given range of semesters,
  - a study plan which respects a certain ECTS credit bound, or
  - a student may be interested in a study plan which prioritizes one characteristic over another.

University study regulations

- Here we highlight the approach we have taken in addressing the aforementioned problems, that is,
  - we will talk about modelled study regulations in SemaLogic,
  - how we solve them using the combinatorial solving paradigm Answer Set Programming (ASP), using our ASP Study Regulations API service (ASPSR service), and
  - some interview-retrieved questions and requests a user (a student user to be precise) is able to ask with our service.

Study regulations modelling with SemaLogic

Study regulations modelling with SemaLogic

- *SemaLogic* is a formal specification language suitable for modelling requirements contained in university study regulations in a generic fashion.
- Study regulations modules are modelled as *symbols* while properties of modules, such as ECTS, are modelled as *attributes*.
- Let us see some modelling examples.

Study regulations modelling with SemaLogic

Examples:

```
Mandatory Modules [ BM1, BM2, BM3 ]

BM1.ECTS := 9;
BM2.ECTS := 9;
BM3.ECTS := 9;
```

Study regulations modelling with SemaLogic

Examples:

```
Optional Modules 2|2 { AM11, AM12, AM21, AM22 }

Optional Modules.Child.ECTS := 6;
```

Study regulations modelling with SemaLogic

```
MSc. Cognitive Systems Study Regulations in SemaLogic notation:

§2
Master of Science [necessary credit points, Cognitive Systems]

§4
¿[Decision Examining Board, Bachelor of Science] ⇀ Cognitive Systems;?
Master of Science.regular duration := four semesters;
necessary credit points := (sum(Cognitive Systems.Under, ECTS) == 120);

§5
¿ sum(Mandatory Modules.Child, ECTS) == 27?
Mandatory Modules [BM1, BM2, BM3]
Mandatory Modules.Child.ECTS := 9;

¿ sum(Optional Modules.Child, ECTS) == 24?
Optional Modules 4|4 {AM11, AM12, AM21, AM22, AM31, AM32, Bridge Modules}
Optional Modules.Child.ECTS := 6;
Bridge Modules ~FM1, FM2, FM3~
Bridge Modules.Child.ECTS := 6;
Decision Examining Board 0|2 {Bridge Modules}

¿ sum(Project Seminars.Child, ECTS) == 24?
Project Seminars 2|2 { PM1, PM2, PM3 }
Project Seminars.Child.ECTS := 12;

¿ sum (Scholarly Work Methods, ECTS) == 15?
Scholarly Work Methods [IM1]
IM1.ECTS := 15;

Thesis.This.ECTS := 30;
Thesis [Masters Thesis, Oral Exam, Get Thesis Topic]

Cognitive Systems [Thesis, Courseload]
Courseload [Scholarly Work Methods, Mandatory Modules, Optional Modules, Project Seminars]

Get Thesis Topic ⇀ Masters Thesis;
Masters Thesis ⇀ Oral Exam;

Courseload ⇀ Thesis;
```

```
§6
Get Thesis Topic { Enough credits, [Some Credits, Registered for Examination] }
Enough credits := (sum (Courseload.Under, ECTS) >= 90);
Registered for Examination := (30 <= sum (Courseload.Under, ECTS, Registered Exams));
Some Credits := (sum (Courseload.Under, ECTS) >= 60);

// Semester data
YearSemRange := 20241,20242,20251,20252,20261,20262;

BM1.yearsem := YearSemRange;
BM2.yearsem := YearSemRange;
BM3.yearsem := YearSemRange;

AM11.yearsem := YearSemRange;
AM12.yearsem := YearSemRange;
AM21.yearsem := YearSemRange;
AM22.yearsem := YearSemRange;
AM31.yearsem := YearSemRange;
AM32.yearsem := YearSemRange;

FM1.yearsem := YearSemRange;
FM2.yearsem := YearSemRange;
FM3.yearsem := YearSemRange;

PM1.yearsem := YearSemRange;
PM2.yearsem := YearSemRange;
PM3.yearsem := YearSemRange;

IM1.yearsem := YearSemRange;

Thesis.yearsem := YearSemRange;

// Weight data
BM2.weight := 10;
AM12.weight := 10;
AM21.weight := 10;
AM22.weight := 5;
PM1.weight := 5;
PM2.weight := 5;
PM3.weight := 3;
```

Fig. 1: Adapted SemaLogic model of the MSc. Cognitive Systems Study Regulations

Our *ASPSR* service

Our *ASPSR* service

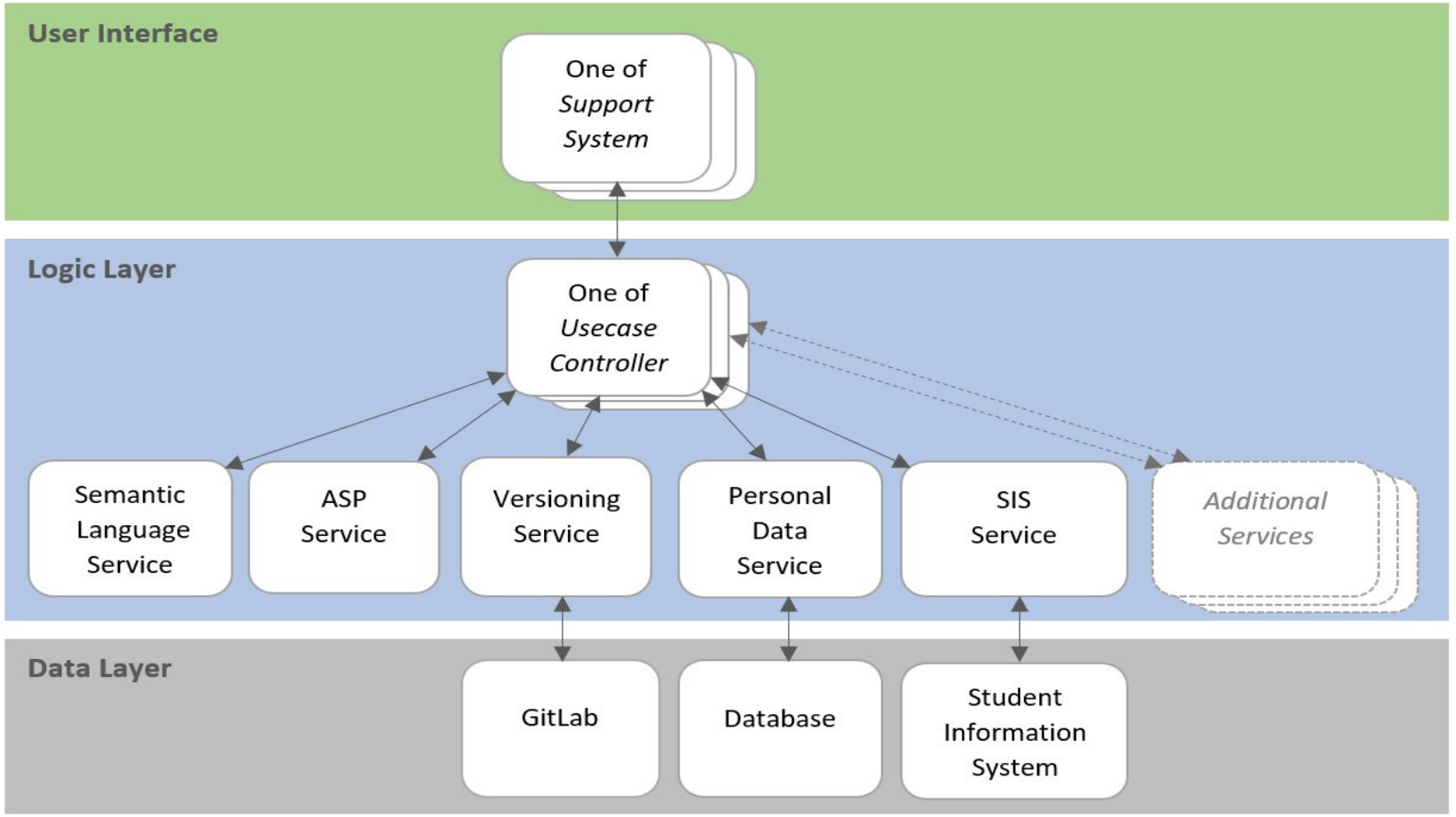


Fig. 2: Developed support system infrastructure (*von der Heyde et al., 2023*)

Our *ASPSR* service

- Our *ASPSR service* relies on Answer Set Programming (ASP) technology.
- ASP is a declarative programming paradigm suitable for solving combinatorial search problems.
- Examples of such problems include train scheduling, course timetabling, games like Sudoku, and university study regulations.
- In ASP, what constitutes the solution to a particular problem is encoded. This represents what is called the *encoding*.

Our *ASPSR* service

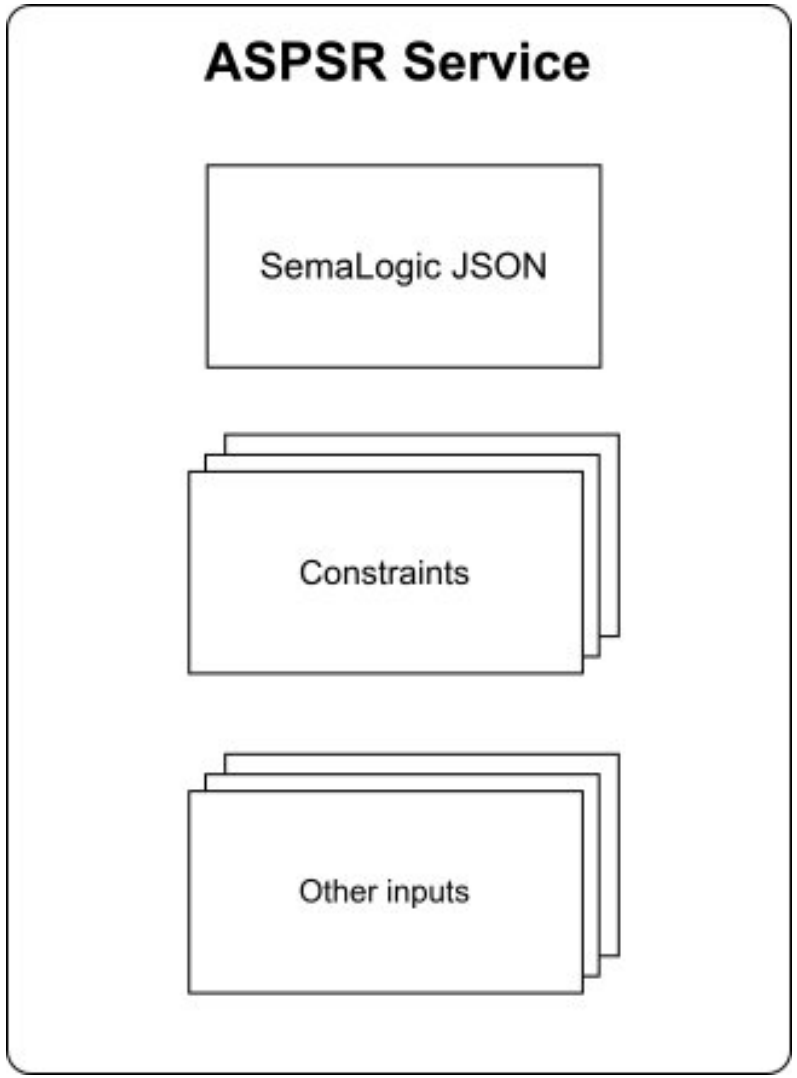
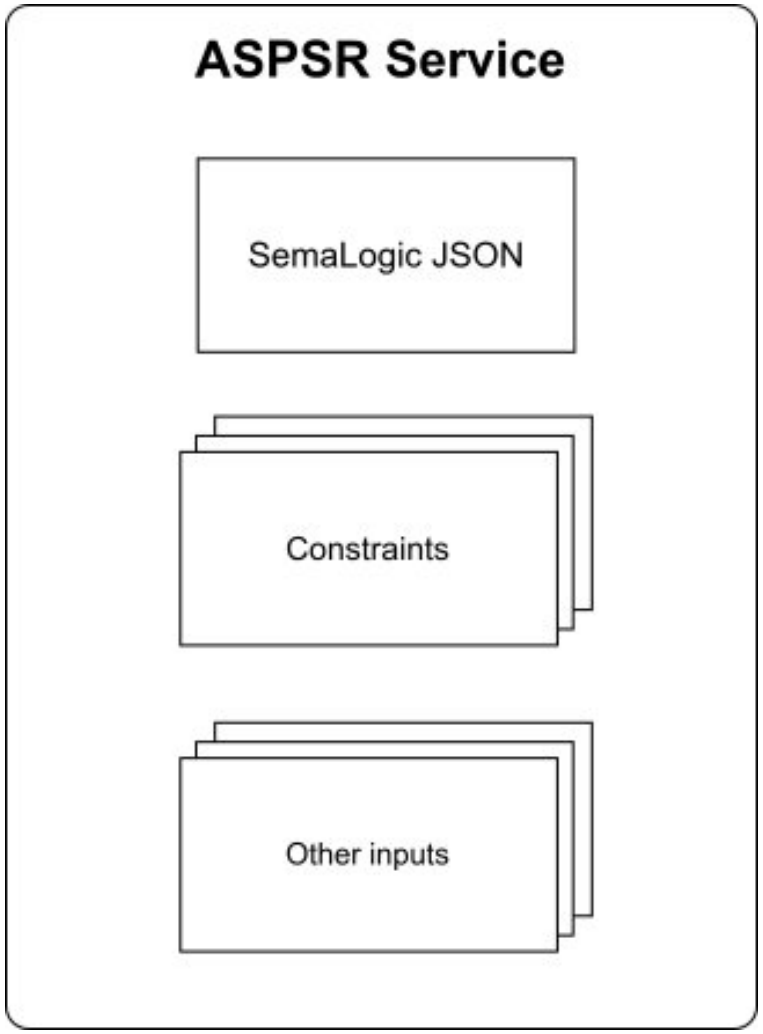


Fig. 2: ASPSR service overview

Our *ASPSR* service



- Our ASPSR service is one micro-service in the entire software architecture.
- The ASPSR service consumes a SemaLogic JSON in addition to some *constraints*.
- Then it gives solutions which satisfy the constraints.

Our *ASPSR* service

Constraints

- A constraint is simply a condition which has to be satisfied within the *solution space*.
- These constraints may be hard or soft:
  - a *hard constraint* is one which must be respected, and
  - a *soft constraint* is one which optimizes some objective function.
- Constraints, if satisfied, generally give solutions which is a subset of the entire solution space.

Our *ASPSR* service

Constraints > Hard constraints

- We have three hard constraints - *sum*, *count*, and *range* :
  1. sum : this enables us to restrict the solution space by setting a summation bound on an attribute.
  2. count : this enables us to set a cardinality bound on an attribute.
  3. range : enables us to restrict the range of an attribute allowable within the solution space to a given set of values.

Our *ASPSR* service

Constraints > Soft constraints

- The three hard constraints can be converted into soft constraints.
- Each soft constraint has a *priority field* whose value is an ordering (in descending order) among the soft constraints.
- Additionally, our `sort` soft constraint orders (in descending order) the solution space by a numeric bound on a given attribute.

Our *ASPSR* service

Constraints

- Those constraints are given as string delimited by parenthesis. Example:
  - *(sum,ects,yearsem,15,20)*
- And two constraints are appended with a semi-colon, “ ; ”. Example:
  - *(sum,ects,yearsem,15,20);(count,ects,yearsem,2,4)*

Questions and Requests on the Study Regulations Model

Questions and Requests on the Study Regulations Model

- The SemaLogic model mentioned previously will now be used here as the base on which we will ask and answer the following questions and requests.

Questions and Requests on the Study Regulations Model

- We put our interview-retrieved questions into two categories:
  1. Questions concerning results
  2. Requests and questions for workload management

Questions and Requests on the Study Regulations Model

Questions concerning results

Q/n	Question
1	What is the fastest way to get finished, respecting a certain amount of ECTS (e.g. 30)?
2	How can I get finished within the standard period of study?

Questions and Requests on the Study Regulations Model

Questions concerning results > ASPSR service constraints

Q/n	Question	ASPSR service constraint
1	What is the fastest way to get finished, respecting a certain amount of ECTS (e.g. 30)?	(maximize,2,shortest,yearsem); (minimize,1,count, yearsem); (sum,ects,yearsem,30,30)
2	How can I get finished within the standard period of study?	(count,yearsem,ects,4,4)

Questions and Requests on the Study Regulations Model

Requests and questions for workload management

Q/n	Question
3	I want to stretch my studies to a certain number of semesters.
4	I need to finish my studies until a certain date.
5	How can I get a plan with approx. equal ECTS numbers in every semester (e.g. 20-25 ECTS)?

Questions and Requests on the Study Regulations Model

Requests and questions for workload management > ASPSR service constraints

Q/n	Question	ASPSR service constraint
3	I want to stretch my studies to a certain number of semesters.	(count,yearsem,ects,5,6)
4	I need to finish my studies until a certain date.	(range,yearsem,20241,20261)
5	How can I get a plan with approx. equal ECTS numbers in every semester (e.g. 20-25 ECTS)?	(sum,ects,yearsem,20,25)

Questions and Requests on the Study Regulations Model

- *Questions 1 - 4* each induce above 100,000 study plans.
- This is mainly due to the available number of semesters a module can be assigned to.
- *Question 5* gives no solution since “Thesis” has 30 ECTS.
- However, without applying any constraints, the total number of study plans is 360.

Questions and Requests on the Study Regulations Model

Some preference-based constraints

C/n	ASPSR service constraint
1	(sum,weight,18,18)
2	(minimize,sum,weight,18,18)
3	(sort,weight,30,40)
4	(maximize,2,sum,weight,20,20); (minimize,1,range,weight,5,9)

Questions and Requests on the Study Regulations Model

Some preference-based constraints

C/n	ASPSR service constraint	Result
1	(sum,weight,18,18)	24
2	(minimize,sum,weight,18,18)	336
3	(sort,weight,30,40)	173
4	(maximize,2,sum,weight,20,20); (minimize,1,range,weight,5,9)	12

Summary

Summary

- University study regulations as a combinatorial problem.
- University study regulations modelling in SemaLogic.
- Our ASPSR service for reasoning with the SemaLogic JSON.
- Interview-retrieved student questions and requests were posed to our service as hard and soft constraints with *sum*, *count* and *range*.

Conclusion

Conclusion

- We worked with the assumption that the students are freshers, hence we did not take into account academic achievements.
- Though we have some implementation for that.
- Current possibilities:
  - modelling examinations
  - *synchronization* of modules semester with associated examinations semester.
  - *level*-based constraint for modules and examinations.
  - semester specific *sum* and *count* constraints.

Acknowledgement

- *CAVAS*+ colleagues for their collective work within the project.
- In particular Magdalena Vock and Jonas Arndt (who conducted the interviews).

+ This work is funded by the Federal Ministry of Research, Technology and Space (BMFTR) under contract number 16DHBKI024.

Thank you

otunuya@uni-potsdam.de

References

- von der Heyde, M., Goebel, M., Zoerner, D., & Lucke, U. (2023). Integrating AI tools with campus infrastructure to support the life cycle of study regulations. *Proceedings of European University*, *95*, 332-344.