Automatic and Interactive Validation of Study Regulations in Accreditation Processes of Higher Education Institutions

Markus von der Heyde¹, Chukwunwike Otunuya², Matthias Goebel¹, Dietmar Zoerner², Ulrike Lucke²

¹ SemaLogic UG, Germany
² University of Potsdam, Germany

EUNIS Congress 2023, Vigo, Spain
What if…

- we would get real time feedback
- on validity and misconceptions
- while writing or changing a study regulation.
We would need to...

- edit a text,
- do real time parsing,
- validate contained semantics,
- transform into other representations,
- generate feedback,
- and display feedback along with the text (or elsewhere).
How to edit

- technical language
- dynamically generated SVG
- MarkDown notation
- export PDFs (for archival)
- Obsidian plug-in to connect to REST API
Simple rule violations

- **Loop**: recognises recursive definitions of terms as well as prerequisites or temporal order.
- **Completeness**: detects if symbols are left undefined when they are needed.
- **Compartmentalisation**: detects if the rule set is split into multiple partitions, i.e., the defined symbols have no logical reference between them.
- **Ranges**: detects if instances of variables do not match the defined range of values.
- **Conflicting AND and OR conditions**: detects whether the requirements of the AND statements conflict with the constraints of the concurrent OR statements, leaving the solution set empty.
- **Empty dynamic groups**: detects whether a dynamic group defined by an interval of symbolic names has no members.
Generic JSON representation

Three generic sections of information:

- Attributes: as symbols and their properties as attributed values
- Groups: to flexibly name the segmentation of all symbols
- Terms: to cover the logical relation between symbols as constraints
ASP workflow (1/2)

- Beyond simple validation → use of highly specialized AI tool for Answer Set Programming (ASP)
ASP workflow (2/2)

- ASP was used to encode and solve the combinatorial problem.
- The generated SemaLogic JSON object serves as input.
- Our encoding contains the study regulations constraints which feeds on the data.
- The solving process returns an output of study paths which satisfy the constraints in the regulation. It returns no solution if the constraints are not satisfied.
Live demonstration

- Simple editing…
- What was understood by the system?
- SemaLogic / JSON / SVG output
- Display of validation results
Summary

- Use of Formal Specification Language
- Automatic parsing and further processing
- Simple (SemaLogic-based) and complex (ASP-based) validation scenarios
- Real time capabilities for OpenAPI based REST interface
- Integration in Obsidian editor by plug-in mechanism

→ Writing and reading study regulations, while receiving real time validation feedback from SemaLogic and ASP.