Lab 6 – Ontologies

Today we are going to extend a small ontology about courses at IDMC to see how to add to a simple vocabulary some logical expressions, and how this affects inferences, i.e. the new information that can be derived from those logical expressions.

T1: In Blazegraph, create a new namespace with inference (you can, for example, call it "IDMC" or "SWLab5") and load into it (make sure the new namespace is selected in Blazegraph) the file at `https://mdaquin.github.io/t/2122/SW_Lab6/idmc.ttl`. Have a look at what the file contains as Blazegraph shows it to you, and using a SPARQL describe query, inspect the entities:

- `<https://idmc.univ-lorraine.fr/data/mathieu>`
- `<https://idmc.univ-lorraine.fr/data/SWLecture5>`

T2: At `https://mdaquin.github.io/t/2122/SW_Lab6/idmc_voc.ttl` you will find an RDF Schema vocabulary for the small graph previously loaded. Open it in a text editor and look at what it says. Draw a quick diagram of the classes, and of the way properties relate them.

T3: Load the vocabulary onto the previously created namespace in Blazegraph. Inspect again the two entities from T1. What changed and why?

T4: We are now going to use Protégé to edit the vocabulary and add some logical definitions in OWL. Go to `https://protege.stanford.edu/`, download and install Protégé. Start it and open the `idmc_voc.ttl` file. Check how things from the vocabulary appear in the interface of Protégé.

T5: Below is a list of things to change in the ontology using OWL operators (don't forget to save):

- Declare that the property `givenBy` is the inverse of the property `gives`: In the entities tab, subtab 'Object Properties', select `givenBy` and add `gives` as its inverse.
- Create a new property `attendedBy` as the inverse of `attends`.
- Define that the class `SCLectureInEnglish` is the class of things which are at the same time a `SCLecture` and `LectureInEnglish`: In the class subtab, select `SCLectureInEnglish` and add an equivalent class. Use the class expression editor to declare it using the `and` operator between `SCLecture` and `LectureInEnglish` (note that ctrl+space activates autocompletion in Protégé).
- Define `LectureInEnglish` as a `Lecture` which has for language (property `inLanguage`) English (`value english`)
- Define `NLPLecture` as a `Lecture` attended by someone (`some`) enrolled in some NLPSemester.
- Define `SCLLecture` as a `Lecture` attended by someone (`some`) enrolled in some SCSemester.
- Make `NLPLecture` a subclass of `LectureInEnglish` to indicate that an NLP lecture is necessarily in English.
- Say, by adding an expression of which `Lecture` is a subclass, that a lecture is given by exactly one person and that lectures are given only by people who have for role (property `hasRole`) lecturer (`value`).

T6: Open the file containing the ontology (`idmc_voc.ttl` if you didn't rename it) and check how the expressions you have entered in Manchester syntax where translated in turtle.

T7: Load the ontology into the namespace created at T1 and check the same entities as before. What changed and why?

T8: Some things should have changed, but because the inference capabilities of blazegraph are limited, they didn't. Do you know what they are? The file at `https://mdaquin.github.io/t/2122/SW_Lab6/idmc_ontoindi.ttl` contains a clean version of the ontology together with the elements in the original file (`idmc.ttl`). Load it into Protégé and activate one of its reasoners (e.g. Pellet) to check what it can infer that Blazegraph couldn't.