Lab 1

Víctor MONTESDEOCA FENOY

Efrén BOYARIZO GARGALLO

1. Start Protégé and imagine a good namespace prefix and a namespace URI for modeling the Cycling domain. Provide this couple prefix/uri in your report and explain why you choose them.

As we are using a GitHub repository to host our reports and the .rtf files, the couple prefix/uri can be extracted from the raw URL that points to the .owl file:

Prefix: https://raw.githubusercontent.com/efrenbg1/WebSem/main/Lab%201

URI: /cycling.owl

So the resulting IRI will be:

https://raw.githubusercontent.com/efrenbg1/WebSem/main/Lab%201/cycling.owl

2. In the cycling domain, a Race is either a OneDayRace or a SeveralStagesRace. There are no other types of race. Model these three classes in the ontology and write in the report the logical formula equivalent to this definition.

$OneDayRace \sqsubseteq Race$

$SeveralStagesRace \sqsubseteq Race$

$Race \equiv OneDayRace \sqcup SeveralStagesRace$

5. Define with necessary and sufficient conditions (i.e. strict equality) the Prologue class such as the Prologue is exactly always the first Stage of a SeveralStagesRace. You might have to introduce more properties or use the inverse of an existing property to do this modeling. Write in the report the logical formula equivalent to this definition.

We first need to define that we will always have on Prologue stage in a SeveralStagesRace (defined in the property composedOfPrologue):

$SeveralStagesRace \sqsubseteq \leq 1 Prologue$

We know also that a Prologue will always be the first stage, and so we can define a new DataProperty called stageOrder of type nonNegativeInteger (order goes from 1 to n) and set it to always be equal to 1:

$Prologue \sqsubseteq stageOrder(nonNegativeInteger)$

$\forall Prologue.stageOrder, stageOrder = 1$

6. Explain in the report what are the differences between object and datatype properties.

Both objects and datatypes are used to define relationships between entities. But they differ in several ways:

Object: links classes to other classes. One example is the property "composedOfPrologue" which links SeveralStagesRace with Prologue, meaning that a several stages race will always have to have a prologue stage

Datatype: it links classes to data values. One example is the property "age" which links Person with a value corresponding to its age

7. A Team is exactly composed of 1 Doctor, 1 Director and 10 RaceCyclist. Define this class using the belongsTo property. Explain in the report what can you say about this new property with respect to previously defined properties?

Within the "Person"/"TeamPerson" we have three different new classes: "Doctor", "Director" and "RaceCyclist". As subclasses of "TeamPerson" we can link them all to a "Team" using a general "belongsTo" property. But to be more specific we can define three new object properties under "belongsTo" that link each of the new classes to "Team" as well.

The difference with previously defined properties is that in this case we have exactly 1 "Doctor", 1 "Director" and 10 "RaceCyclist" in a "Team", and as such we have to define three DataProperties with the count values of each of the classes and as such we have to make sure they equal 1, 1 and 10 respectively.

Additionally we need to define the inverse property. To do so, we can add the following into "Equivalent to" of class "Team": "inverse (belongsTo) some TeamPerson". Where "TeamPerson" is the parent class of "Director", "Doctor" and "RaceCyclist".

9. Search for properties in the FOAF ontology to align with your own properties and add the corresponding mapping axioms. Write in the report the mappings you have discovered.

From the FOAF ontology we've mapped the following:

Classes:

FOAF	Cycling	Why?
Person	Person	They represent the same concept
Organization	Team	A team is really an organization of people

Data properties:

FOAF	Cycling	Why?
age	age	The data property age for Person can represents the same concept and so it can be directly mapped
firstName	name	The same for the name of Person

Object properties:

FOAF	Cycling	Why?
member	belongsTo	Member maps an Agent with a Group. belongsTo does the same thing but with a TeamPerson and a Team

11. Use the LOV search function at https://lov.linkeddata.es/dataset/lov/ and identify relevant classes and properties that could be re-used in your ontology. Write them down in your report.

This question can not be done as the website is down at the moment of making this report

12. Validate your ontology by clicking on Reasoner and Start reasoner where HermiT is selected as the reasoning service. What is the result? Did you get the message "your ontology is coherent and consistent"?

When running the reasoner no message is displayed to the user. The Reasoner will remain running until told to stop by the user, so any modifications will be analyzed immediately. If the reasoner finds

something wrong, the different classes or properties will be marked red.

If all goes well, then the tab “Individuals by class” will display both the Individuals defined by the users and the individuals inferred by the Reasoner. If everything is correct both tabs should match.

In our case, no classes or properties were marked red, and the inferred tab matched the individuals we’ve defined. Moreover, no errors where displayed in the logs having only the message “Ontologies processed in 40 ms by HermiT”.