KRR: Activity 3 – Introduction to OWL and SWRL

Alexandru Sorici

26.10.2020

Introduction

Structured knowledge is used in many production-level systems, to date. Schema.org is an initiative supported by companies such as Google, Microsoft, Pinterest or Yandex. The vocabularies defined on schema.org are used by over 10 million websites. Google's Knowledge Graph makes use of the vocabulary to present the *medallion* widgets when querying for factual things, such as movie titles, historical figures, etc.

Derivations of schema.org find applications in more industrial settings as well: FIBO (Finance), GoodRelations (e-commerce), VVO (Volkswagen Vehicle Ontology), POPE (Purdue Ontology for Pharmaceutical Engineering).

The representation format used on websites is typically one of Microdata[1], RDFa[2] or JSON-LD[3].

In the academic domain, nonetheless, the standard, well studied language for defining ontologies and implementing description-logic based reasoning is OWL (Web Ontology Language). In this Activity we explore the concepts that make up OWL.

OWL Concepts

We will overview OWL Concepts on hand of two example ontologies: one for defining Activities of Daily Living (adlont.owl) and one for defining genealogy (generations.owl).

Entities

- Overview of the Activity hierarchy for `subclassOf` relationship (e.g. `PreparingMeal`)
- Overview of the Artifact hierarchy: e.g. `FoodPreparationFurniture`
- Overview of axiomatic definitions of concepts

OWL Object Properties

- Overview of object properties that are used also in chain composition: `producesEvent`, `sensesUsageOf`, `usedFor`, `requiresUsageOf`
- Overview of the resulting properties defined using chain composition: `necessaryEventFor` and `predictiveSensorEventFor`

[1]https://www.w3.org/TR/microdata/
[2]https://rdfa.info/
[3]https://json-ld.org/

OWL Datatype Properties

- Overview of datatype properties that deal with linking an individual to a literal value.
- Exemplification on `hasPhoneStatus` and `hasDoorStatus` to show how a *range* is created from xsd:string individuals

Working with the reasoner

- Installing and configuring possible inferences of an ontology reasoner.
- Viewing the *inferred hierarchy*: exemplification on the case for *chain property* reasoning.

Working with the DL query tab Exemplification of working with the DL query tab via the following examples:

- SensorEvent and predictiveSensorEventFor some ChooseOutfit = $SensorEvent \sqcap \exists predictiveSensorEventFor.ChooseOutfit$
- Furniture or Fixture or Device that is required for the activity of choosing an outfit = $(Furniture \sqcup Fixture \sqcup Device) \sqcap (\exists sensesUsageOf^{-}.(\exists neceseary SensorFor.ChooseOutfit))$

Tasks

Task 1 Use the DL Query Tab on the ADL ontology to:

- retrieve all the sensors that are used to predict either the *WatchDVD* activity or the *ChooseOutfit* one
- retrieve the type of furniture that is used for either *cleaning* or *waterplants* activities

Task 2 Load and inspect the ontology file describing the concepts and relations existing between family members. It defines classes such as `Man`, `Woman`, `Father`, `Mother`, `Son`, `Daughter` and relations such as `hasChild`, `hasSibling`.

The tasks are the following:

- Open the family tree of the *Baggins of Hobbiton*: http://tolkiengateway.net/wiki/Baggins_Family#Family_Tree_of_the_Baggins_of_Hobbiton
- Populate your generations ontology with *individuals* from Frodo's genealogy. Make sure you *only* define them as instances of `Person` and that you relate them via the corresponding *object properties*: `hasSex`, `hasChild`, `hasSibling`. Select your instances such that you covereu pot face the following concepts (some are not yet defined in the ontology).
  - Father, Mother, Grandfather, Grandmother, Uncle, Aunt, Cousin, Grandchild, Nephew, Niece
- Initialize and run the inference engine. Make sure that you can see your defined instances under the *Instances* description for the *Class hierarchy (inferred)* view.
- Define the following missing concepts: `Uncle`, `Aunt`, `Cousin`, `Nephew`, `Niece`. Define the missing property `marriedTo` (used for specifying that an Aunt can also be the spouse of an uncle). Use the DL Query Tab to construct the expressions that define these concepts using the existing classes and relations. Use the inference engine to show that you have at least one individual from each of the introduced concepts.

SWRL Tasks

SWRL (Semantic Web Rule Language) is a complementary means to produce inferences in an ontology based setting. It proposes a rule-based inference mechanism that mitigates some of the expressiveness limitations present in OWL 2 (e.g. the tree model property for property composition).

We will be using the same toy ontology (generations), as a means to exemplify usage of SWRL rules.

Pre-requisits

- Follow the explanations of SWRL capabilities on hand of the slide notes, included on the course web site, and your TA :-)
- Differentiate your individuals :-)
  - OWL reasoners operate on the *open world assumption*, which means that two individuals *could be the same*, **unless** explicitly stated / inferred to be different.
  - To do this for the individuals of your generations ontology, select one individual and **use the *Different Individuals* property** from the *description tab* to specify that it is different from all others. The reasoner will infer the rest.
- Install the SWRL Tab and SQWRL Tab plugins in Protege.
- Be sure to have the latest version of the HermiT reasoner installed.

Task 1

- Remove all `hasSibling` relations from your ontology.
- Infer `hasSibling` from `hasChild` and its inverse `hasParent`.
- Infer all `hasChild` instances, using `hasChild` and `marriedTo`.

- Add the following properties to your ontology: **hasUncle, hasAunt, hasNiece, hasNephew, hasGrandchild**. Infer the instances of these properties by means of the appropriate SWRL rules.

Task 2

- Define a new Datatype Property, called `hasAge`.
- Define the classes `Adult` and `Minor` and infer their members using the condition that individuals over 18 years of age are adults and the rest are minors.