Knowledge Engineering and Semantic Web

Exercise Sheet: 4

Will be discussed on: May 23, 2023

Leibniz Universität Hannover

TUTORS:
Yaser Jaradeh, Hassan Hussien, and some other ORKG members

QUESTIONS: Please don't hesitate to ask any questions. Questions help you and your peers.
PRINT: Please consider the environment before printing the exercise.

1 Review questions

1. Semantics and Syntax : which statement is correct?
   (a) **Syntax:** meaning of the character strings
   ✗ **Syntax: character strings without meaning**
   (b) **Semantics:** meaning of the character strings ✓
2. Implicit Knowledge: which statement is correct?

```
ex:ExampleBook rdf:type ex:TextBook .
ex:TextBook rdfs:subClassOf ex:Book .
ex:PrintMedia rdfs:subClassOf ex:Media .
```

   (a) `ex:TextBook rdf:type ex:ExampleBook .` ✗ **not in knowledge**
   (b) `ex:TextBook rdf:type ex:PrintMedia .` ✗ **not in knowledge**
   (c) `ex:ExampleBook rdf:type ex:Book.` ✓ **true**
3. rdfs:subClassOf "characteristics"
   (a) symmetric ✗ **A subClass of B >> does not mean B subClassOf A**
   (b) transitive ✓ **true**
   (c) reflexive ✓ **true A subclass of A**

2 Semantic Entailment Rules

Write down the semantic entailment rules.
Note:
if $a\ \eta\ b \in K$ and $b\ \eta\ c \in K$ then $K \leftarrow K \cup \{a\ \eta\ c\}$
also written as :$\frac{a\ \eta\ b \quad b\ \eta\ b}{a\ \eta\ c}$

Notation Triples : < subject predicate object >

a and b $\widehat{=}$ any URI (predicates)

$\_:n$ $\widehat{=}$ blank node ID

u and v $\widehat{=}$ any URI or blank node ID for (subject)

x and y $\widehat{=}$ any URI, or blank node ID or literals (object)

l $\widehat{=}$ any literal

Simple Entailment Rules

Rule	Formula
Simple Entailment 1	**Solution:** $\frac{u\ a\ x\ .}{u\ a\ \_:n\ .}$
Simple Entailment 2	**Solution:** $\frac{u\ a\ x\ .}{\_:n\ a\ x\ .}$

Explain in own words the two rules.
Rule 1:

Solution: We can replace x through a blank node (the object position in the triple)
Rule 2:

Solution: We can replace u through a blank node (the subject position in the triple)

RDF Entailment Rules

Rule	Formula
RDF axioms	**Solution:** $\frac{}{u\ a\ x\ .}$
Literal Grounding	**Solution:** $\frac{u\ a\ l\ .}{u\ a\ \_{:}n\ .}$ where :_n does not yet occur in the graph
RDF Rule 1	**Solution:** $\frac{u\ a\ y\ .}{a\ rdf{:}type\ rdf{:}Property\ .}$
RDF Rule 2	**Solution:** $\frac{u\ a\ l\ .}{\_{:}n\ rdf{:}type\ rdf{:}XMLLiteral\ .}$ where _:n has been introduced through LG rule

RDFS Entailment Rules

Rule	Formula
RDFS axioms	**Solution:** $\frac{}{u\ a\ x\ .}$ for all RDFS axiomatic triples u a x
RDFS Rule 1	**Solution:** $\frac{u\ a\ l\ .}{\_{:}n\ rdf{:}type\ rdfs{:}Literal\ .}$ whit :_n as usual
RDFS Rule 2 (domain)	**Solution:** $\frac{a\ rdfs{:}domain\ x\ .\ u\ a\ y\ .}{u\ rdf{:}type\ x\ .}$
RDFS Rule 2 (range)	**Solution:** $\frac{a\ rdfs{:}range\ x\ .\ u\ a\ v\ .}{v\ rdf{:}type\ x\ .}$
RDFS Rule 4a	**Solution:** $\frac{u\ a\ x\ .}{u\ rdf{:}type\ rdfs{:}Resource\ .}$
RDFS Rule 4b	**Solution:** $\frac{u\ a\ v\ .}{v\ rdf{:}type\ rdfs{:}Resource\ .}$
RDFS Rule 5	**Solution:** $\frac{u\ rdfs{:}subPropertyOf\ v\ .\ v\ rdfs{:}subPropertyOf\ x\ .}{u\ rdfs{:}subPropertyOf\ x\ .}$
RDFS Rule 6	**Solution:** $\frac{u\ rdf{:}type\ rdf{:}Property\ .}{u\ rdfs{:}subPropertyOf\ u\ .}$
RDFS Rule 7	**Solution:** $\frac{a\ rdfs{:}subPropertyOf\ b\ .\ \ u\ a\ y\ .}{u\ b\ y\ .}$
RDFS Rule 8	**Solution:** $\frac{u\ rdf{:}type\ rdfs{:}Class\ .}{u\ rdfs{:}subClassOf\ rdfs{:}Resource\ .}$
RDFS Rule 9	**Solution:** $\frac{u\ rdfs{:}subClassOf\ x\ .\ \ v\ rdf{:}type\ u\ .}{u\ rdf{:}type\ x\ .}$
RDFS Rule 10	**Solution:** $\frac{u\ rdf{:}type\ rdfs{:}Class\ .}{u\ rdfs{:}subClassOf\ u\ .}$
RDFS Rule 11	**Solution:** $\frac{u\ rdfs{:}subClassOf\ v\ .\ \ v\ rdfs{:}subClassOf\ x\ .}{u\ rdfs{:}subClassOf\ x\ .}$
RDFS Rule 12	**Solution:** $\frac{u\ rdf{:}type\ rdfs{:}ContainerMemberShipProperty\ .}{u\ rdfs{:}subPropertyOf\ rdfs{:}member\ .}$
RDFS Rule 13	**Solution:** $\frac{u\ rdf{:}type\ rdfs{:}Datatype\ .}{u\ rdfs{:}subClassOf\ rdfs{:}Literal\ .}$

3 Inference

1. Hierarchy of properties: Select the correct inferences among the following ones.

a)
```
:a :p1 :b .
:a :p2 :c .
->
:b rdfs:subPropertyOf :c .
```

b)
```
:a :p1 :b .
:b rdf:type :C .
->
:p1 rdfs:range :C .
```

c)
```
:a :p1 :b .
:p2 rdfs:subPropertyOf :p1 .
->
:a :p2 :b .
```

d)
```
:p1 rdfs:subPropertyOf :p2 .
:a :p1 :b .
->
:a :p2 :b .
```

Solution:

a) ✗ **`:a`, `:b`, and `:c` are entities not properties, so `:b rdfs:subPropertyOf :c` is completely wrong, as domain and range of `rdfs:subPropertyOf` are `rdf:Property`.**

b) ✗ **`:b` can have too many types which are irrelevant to range of `:p1`.**

c) ✗ **With `:a :p2 :b` and `:p2 :subPropertyOf :p1`, we can infer `:a :p1 :b`, not vice versa.**

d) ✓ **Look at explanation of choice c.**

2. Hierarchy of Classes: Select the correct inferences among the following ones.

a)
```
:A rdfs:subClassOf :B .
:c rdf:type :A .
->
:c rdf:type :B .
```

b)
```
:a :p1 :b .
:a :p2 :c .
:b rdf:type :B .
:c rdf:type :C .
:B rdfs:subClassOf :C .
->
:p1 rdfs:subPropertyOf :p2 .
```

c)
```
:p1 rdfs:domain :A .
:p1 rdfs:range :C .
:p2 rdfs:domain :B .
:p2 rdfs:range :D .
:p1 rdfs:subPropertyOf :p2 .
->
:A rdfs:subClassOf :B .
:C rdfs:subClassOf :D .
```

d)
```
:a :p1 :b .
:p2 rdfs:domain :C .
:p1 rdfs:subPropertyOf :p2.
->
:a rdf:type :C .
```

Solution:

a) ✓ **Obvious by definition of `rdfs:subClassOf`**

b) ✗ **It's totally irrelevant inference.**

c) ✗ **Not all members of domain set necessarily contribute in a property relations. So there could be some members of `:A` which are not contained in :B. Similar explanation holds for `:C` and `:D`**

d) ✓ **`:a :p1 :c` and `:p1 rdfs:subPropertyOf :p2` entail: `:a :p2 :b` and so `:a rdf:type :C` as `:C` is the domain of `:p2`.**

3. Equivalence of Classes: Select the correct inferences among the following ones.

a)
```
:A rdfs:subClassOf :B .
:B rdfs:subClassOf :C .
:C rdfs:subClassOf :D .
:D rdfs:subClassOf :A .
->
:A , :B , :C , :D
are equivalent classes.
```

b)
```
:A rdfs:subClassOf :B .
:B rdfs:subClassOf :C .
:c rdf:type :A .
->
:c rdf:type :C .
```

c)
```
:A rdfs:subClassOf :B .
:B rdfs:subClassOf :A .
:c rdf:type :A .
:d rdf:type :A .
->
:c and :d are equivalent.
```

d)
```
:p1 rdfs:subPropertyOf :p2 .
:p2 rdfs:subPropertyOf :p1 .
:p1 rdfs:range  :B ;
    rdfs:domain :A .
:p2 rdfs:range  :D .
:p2 rdfs:domain :C .
->
:A is equal to :C and :B is equal to :D.
```

Solution:

a) ✓ `:A rdfs:subClassOf :B` **and** `:B rdfs:subClassOf :C` **entails** `:A rdfs:subClassOf :C`**. Similarly we can infer:** `:A rdfs:subClassOf :D`**. And then by** `:A rdfs:subClassOf :D` **and** `:D rdfs:subClassOf :A` **we can entail that** `:A` **and** `:D` **are equivalence. Similar reasoning can be applied for the other equivalencies.**

b) ✓ `:c rdf:type :A` **and** `:A rdfs:subClassOf :B` **so it would be entailed that** `:a rdf:type :B`**. Similarly we can infer :a rdf:type :C.**

c) ✗ **Equivalence of two class doesn't mean that all members of each class are the same.**

d) ✗ **Not all members of domain and range sets of property P are related to each other with property P. So there are some members of** `:A` **and** `:C` **which are not contributed in relation defined by P, so they be not contained in** `:A` **and** `:A` **respectively.**

4 Consider the following statements:

a) Represent them in RDF Turtle serialization.

b) Select the correct ones.

1. $< rdfs:subClassOf^I, rdfs:Resource^I > \in I_{EXT}(rdfs:domain^I)$.
   ✗ `rdfs:subClassOf rdfs:domain rdfs:Resource.`, **The domain of** `rdfs:subClassOf` **is** `rdfs:Class`

2. $< rdf:List^I, rdf:rest^I > \in I_{EXT}(rdfs:domain^I)$.
   ✗ `rdf:List rdfs:domain rdf:rest.`, **Subject should be replaced with object.**

3. $I_{CEXT}(rdfs:Class^I) \subseteq I_{CEXT}(rdfs:Resource^I)$.
   ✓ `rdfs:Class rdfs:subClassOf rdfs:Resource.`

4. $< rdfs:domain^I, rdf:Property^I > \in I_{EXT}(rdf:type^I)$.
   ✓ `rdfs:domain rdf:type rdf:Property.`

5. If $< x, y > \in I_{EXT}(rdfs:domain^I)$ and $< u, v > \in I_{EXT}(x) \rightarrow u \in I_{CEXT}(x)$.
   ✗ `:x rdfs:domain :y.` **and** `:u :x :v.` → `:u rdf:type :x.`, **False, as** `:u` **is in type of** `:y` **not** `:x`**.**

5 For the following knowledge base, indicate which statement can be entailed. Prove the true answers with proof-theoretic semantics.

```
@prefix ex:   <http://example.org> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .



ex:dog      rdfs:subClassOf   ex:animal .
ex:horse    rdfs:subClassOf   ex:creature .
```

```
ex:person   rdfs:subClassOf   ex:creature .


ex:isEnemyOf     rdfs:subPropertyOf   ex:knows .
ex:isEnemyOf     rdfs:domain          ex:person;
                 rdfs:range           ex:person .
ex:isFriendOf    rdfs:subPropertyOf   ex:knows .




ex:LuckLuke       a   ex:person .
ex:JollyJumper    a   ex:horse .
ex:Rantanplan     a   ex:dog .


ex:LuckyLuke      ex:isFriendOf   ex:JollyJumper .
ex:JollyJumper    ex:isFriendOf   ex:Rantanplan .
ex:LuckyLuke      ex:isEnemyOf   exJoeDalton .
```

Statements:

1. ex:Rantanplan a ex:creature.
   ✗ **Type of** ex:Rantanplan **is** ex:dog **and there is no triple to show that dog is subclass of** ex:creature **or any subclass of this class.**

2. ex:Rantanplan ex:isFriendOf ex:JollyJumper.
   ✗ **The triple** ex:JollyJumper ex:isFriendOf ex:Rantanplan. **doesn't infer that** ex:RantanPlan **is also friend of** ex:JollyJumper, **as there is no triple which entails** ex:isFriendOf **is symmetric property (this property is not covered by RDFS semantic, and is part of OWL ontology). Based on RDFS semantic we can say** ex:RantanPlan ex:isFriendOf ex:JollyJumper. **only if have this triple in our knowledge base. In RDFS semantic we can point on statements and talk about them, but, we cannot make rules about them.**

3. ex:LuckyLuke ex:isFriendOf ex:RantanPlan.
   ✗ **Similar reasoning with last part.**

4. ex:LuckyLuke ex:knows ex:JoeDalton.
   ✓ **According to our knowledge base we know** ex:LuckyLuke ex:isEnemyOf exJoeDalton. **And also we know** ex:isEnemyOf rdfs:subPropertyOf ex:knows. **So** ex:LuckLuke ex:knows ex:JoeDalton. **is inferred.**

5. ex:JoeDalton ex:isEnemyOf ex:LuckyLuke.
   ✗ **Similar reasoning with second and third statements.**

6. ex:JoeDalton a ex:creature.
   ✓ ex:JoeDalton **is a** ex:person **as the range of** ex:isEnemyOf **is** ex:Person. **And as** ex:person **is subclass of** ex:creature, **so** ex:JoeDalton **is ex:creature .**