CS5560 Knowledge Discovery and Management

Problem Set 7 & 8

Submission Deadline: July 28, 2017

ttps://goo.gl/forms/aTXnl4oRHMdS8j1L2

Name: Sreelakshmi Nandanamudi
Class ID: 17

References

I. Logical knowledge representation

First Order Logic Reference: http://pages.cs.wisc.edu/~dyer/cs540/notes/fopc.html

1) Let us define the statements as follows:
   - G(x): "x is a giraffe"
   - F(x): "x is 15 feet or higher,"
   - Z(x): "x is animal in this zoo"
   - M(x): "x belongs to me"

   Express each of the following statements in First-Order Logic using G(x), F(x), Z(x), and M(x).

   a) Nothing, except giraffes, can be 15 feet or higher;
   b) There is no animal in this zoo that does not belong to me;
   c) I have no animals less than 15 feet high.
   d) All animals in this zoo are giraffes.

2) Which of the following are semantically and syntactically correct translations of "No dog bites a child of its owner"? Justify your answer
   a) $\forall$ x Dog(x) $\Rightarrow$ ¬Bites(x, Child(Owner(x)))
   b) $\neg\exists$ x, y Dog(x) $\wedge$ Child(y, Owner(x)) $\wedge$ Bites(x, y)
   c) $\forall$ x Dog(x) $\Rightarrow$ ($\forall$ y Child(y, Owner(x)) $\Rightarrow$ ¬Bites(x, y))
   d) $\neg\exists$ x Dog(x) $\Rightarrow$ ($\exists$ y Child(y, Owner(x)) $\wedge$ Bites(x, y))

3) For each of the following queries, describe each using Description Logic
Reference: http://www.inf.ed.ac.uk/teaching/courses/kmm/PDF/L3-L4-DL.pdf

   a) Define a person is Vegan
   b) Define a person is Vegetarian
   c) Define a person is Omnivore

II. SPARQL

Reference: https://www.w3.org/2009/Talks/0615-qbe/

Design a SPARQL query for following queries and show an expected output.

Query #1: Multiple triple patterns: property retrieval
Find me all the people in Tim Berners-Lee's FOAF file that have names and email addresses. Return each person's URI, name, and email address.

Query #2: Multiple triple patterns: traversing a graph
Find me the homepage of anyone known by Tim Berners-Lee.

Query #3: Basic SPARQL filters
Find me all landlocked countries with a population greater than 15 million.

Query #4: Finding artists' info
Find all Jamendo artists along with their image, home page, and the location they're near, if any.

Query #5. Design your own query

III. SWRL

References:
https://www.w3.org/Submission/SWRL/
https://dior.ics.muni.cz/~makub/owl/

Design SWRL rules for the following cases

Rule #1: design hasUncle property using hasParent and hasBrother properties
Rule #2: an individual X from the Person class, which has parents Y and Z such that Y has spouse Z, belongs to a new class ChildOfMarriedParents.
Rue #3: persons who have age higher than 18 are adults.
Rue #4: Compute the person's born in year
Rule #5: Compute the person's age in years
Rule #6: Design your own rule

I

Logical knowledge representation:

Knowledge representation and reasoning (KR) is the field of artificial intelligence (AI) dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks such as diagnosing a medical condition or having a dialog in natural language.

First order Logic (FOL or FOPC) syntax:

User defines these primitives:

1) Constant symbols (i.e; "individuals in the world")
2) Function symbols (mapping individuals to individuals)
3) predicate symbols (mapping from individuals to truth values)

FOL supplies these primitives:

1) variable symbols. Ex: x, y
2) Connectives. not (~), and ($\wedge$), or ($\vee$), implies ($\Rightarrow$), if and only if ($\Leftrightarrow$)
3) Quantifiers: Universal ($\forall$) and Existential ($\exists$)

1) possible translations for the given statements are

$\forall x(\neg G(x) \rightarrow \neg F(x))$ OR $\forall x(F(x) \rightarrow G(x))$

$\neg \exists x(Z(x) \wedge \neg M(x))$ OR $\forall x(Z(x) \rightarrow M(x))$

$\forall x(M(x) \rightarrow F(x))$

$\forall x(Z(x) \rightarrow G(x))$

2. Syntactic Analysis:
The goal of syntactic analysis is to determine whether the text string on input is a sentence in the given natural language

Semantic Analysis:
Semantic and pragmatic analysis make up the most complex phase of language processing as they build up on results of all the above mentioned disciplines.

a) $\forall x \; Dog(x) \Rightarrow \neg Bites(x, child(owner(x)))$
No dog bites dogs and owner of children

b) $\neg E \; x, y \; Dog(x) \wedge child(y, owner(x)) \wedge Bites(x, y)$
No dog bites owners children

c) $\forall x \; Dog(x) \Rightarrow (\forall y child(y, Owner(x)) \Rightarrow \neg Bites(x, y)$
All dog donot bite their children of owner

d) $\neg E \; x \; Dog(x) \Rightarrow (\exists y \; child(y, owner(x)) \wedge Bites(x, y)$
Dog bite the children of owners.

Therefore, the correct translations are (b) and (c)

3. Description Logic: Description Logic allows formal concept definitions that can be reasoned about to be expressed. It is an important element of the semantic web.

a) Define a person is Vegan
people who does not eat or use animal products

$\forall$ eats $\neg$ Animal products

b) Define a person is Vegetarian.

people who doesnot eat animal products.

$\forall$ eats $\neg$ Animal

c) Define a person is Omnivore.

Animal | person eats food of both plant and Animal.

$\exists$ eats Animal

II SPARQL:

SPARQL is the query language of the Semantic web. It lets us:

1) pull values from structured and semi-structured data.
2) Explore data by quering unknown relationships.
3) perform Complex joins of disparate databases in a single, simple query.
4) Transform RDF data from one vocabulary to another.

Query #1 : Multiple triple patterns : property retrieval

Prefix PREFIX foaf: <http://xmlns.com/foaf/0.1>

```
SELECT *
WHERE {
        ?person foaf:name ?name
        ?person foaf:mbox ?email
```

Expected output :

person	name	email
<http://www.w3.org/People/Berners-Lee/card#amy>	"Amy Vander Hiel"	<mailto:amy@w3.org>

<http://www.w3.org/people/Berners-Lee/card#dj> "Dean Jackson" <mailto:dean@w3.org>
<http://www.w3.org/people/Berners-Lee/card#edd> "Edd Dumbill" <mailto:edd@usefulinc.com>
⋮

Query 2 : Multiple triple patterns : traversing a graph.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1>
PREFIX card: <http://www.w3.org/People/Berners-Lee/card#>

SELECT ?homepage
FROM <http://www.w3.org/people/Berners-Lee/card>
WHERE {
        card:i foaf:knows ?known
        ?known foaf:homepage ?homepage.
}
```

Expected output :

homepage
<http://purl.org/net/eric/>
<http://www.mellon.org/about_foundation/staff/program-area-staff/irafuchs>
<http://www.johnseelybrown.com/>
<http://heddley.com/edd>

Query 3 : Basic SPARQL filters.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX type: <http://dbpedia.org/class/yago/>
PREFIX prop: <http://dbpedia.org/property/>
SELECT ?country_name ?population
WHERE {
```

```
    ? country a type: LandlockedCountries;
              rdfs: label ? country_name;
              prop: populationEstimate ? population
    FILTER (?population > 15000000)
}
```

Expected output:

Country_name	population
Afghanistan	31889923
Afghanistan	31889923
⋮	
Etopia	75067000
Etopia	75067000
⋮	

Query 4: Finding artists' info:

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?img ?hp ?loc
WHERE {
    ?a a mo:MusicArtist;
     foaf:name ?name;
     foaf:img ? img;
     foaf:homepage ?hp;
     foaf:based_near ?loc;
}
```

Wrong way

```
OPTIONAL {?a foaf:img ?img}
OPTIONAL {?a foaf:homepage ?hp}
OPTIONAL {?a foaf:based_near ?loc}
```

Right way

Expected output

name	img	hp	loc
"Cicada"^^xsd:string	http://img.jamendo.com/artists/h/hattrickman.jpg	http://www.cicada-fr.st	http://sws.geonames.org/3031339

"Hace Soul"^^xsd:string http://img.jamendo.com/artists/h/hace.soul.jpg http://www.hacesoul.com http://sws.geonames.org/2510769

"Vincent j"^^xsd:string http://img.jamendo.com/artists/v/vincentj.jpg http://v.joudrier.free.fr/siteV http://sws.geonames.org/3020781

⋮

Query 5: Design your own query:

Asking a question → Is the Amazon river longer than the Nile River?

```
PREFIX prop: <http://dbpedia.org/property/>
Ask
{
<http://dbpedia.org/resource/Amazon_River> prop:length ?amazon.
<http://dbpedia.org/resource/Nile> prop:length ?nile.
FILTER (?amazon > ?nile).
}
```

Expected output:

```
<?xml version="1.0" ?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.w3.org/2001/sw/DataAccess/rf1/result.xsd">
<head></head>
<boolean>true</boolean>
</sparql>
```

III SWRL: A Semantic Web Rule Language

Combining OWL and RuleML

Rule#1: design hasUncle property using hasParent and hasBrother properties

$$hasParent(?x_1, ?x_2) \wedge hasBrother(?x_2, ?x_3) \Rightarrow hasUncle(?x_1, ?x_3)$$

Rule #2 : an individual X from the Person class, which has parents Y and Z such that Y has spouse Z, belongs to a new class childOfMarriedParents

person(?X), hasParent(?X, ?Y), hasParent(?X, ?Z), hasSpouse(?Y, ?Z) → childOfMarriedParents(?X)

Rule #3 : persons who have age higher than 18 are adults.

person(?P), hasAge(?P, ?age), swrlb: greaterThan(?age, 18) → Adult(?P)

Rule #4 : Compute the person's born in year

person(?P), bornOnDate(?P, ?date), swrlb: date(?date), swrlb: date(?date, ?year, ?month, ?date, ?timezone) → bornInyear(?P, ?year)

Rule #5 : Compute the person's age in years

person(?P), bornInYear(?P, ?year), my: thisYear(?newYear), swrlb: subtract(?age, ?newYear, ?Year) → hasAge(?P, ?age)

Rule #6 : design your own rule.

→ design hasChild ~~DaughterSon~~ property using hasChild and Man properties.

hasChild(?X, ?Y) ∧ Man(?Y) ⟹ hasSon(?X, ?Y)