## Cadinality test – example

## Ontology

## Let the ontology will be defined as below:

```
@prefix : <http://www.example.org/tkubik/ct#> .
                                                                             owl:onProperty :hasBrother ;
                                                                             owl:minCardinality "1"^^xsd:nonNegativeInteger . ]
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
                                                                          [ a owl:Restriction ;
                                                                             owl:onProperty :hasBrother ;
<http://www.example.org/tkubik/ct#> a owl:Ontology .
                                                                             owl:maxCardinality "2"^^xsd:nonNegativeInteger . ]
                                                                         );
:hasBrother a owl:SymmetricProperty .
                                                                      1.
:Any a owl:Class ;
                                                                  :Many a owl:Class ;
        owl:equivalentClass [
                                                                           owl:equivalentClass [
        a owl:Restriction ;
                                                                          a owl:Class ;
        owl:onProperty :hasBrother ;
                                                                          owl:intersectionOf (
        owl:someValuesFrom owl:Thing .
                                                                          [ a owl:Restriction ;
                                                                            owl:onProperty :hasBrother ;
        1.
                                                                             owl:minCardinality "5"^^xsd:nonNegativeInteger . ]
:Average a owl:Class ;
                                                                         );
        owl:equivalentClass [
                                                                       ].
       a owl:Class ;
       owl:intersectionOf (
                                                                  :Person a owl:Class .
        [ a owl:Restriction ;
          owl:onProperty :hasBrother ;
                                                                  :i1 a owl:NamedIndividual , :Person ;
                                                                  :hasBrother :i2 , :i3 .
:i2 a owl:NamedIndividual , :Person .
          owl:minCardinality "3"^^xsd:nonNegativeInteger . ]
        [ a owl:Restriction ;
          owl:onProperty :hasBrother ;
                                                                  :i3 a owl:NamedIndividual , :Person .
          owl:maxCardinality "4"^^xsd:nonNegativeInteger . ]
                                                                  :i4 a owl:NamedIndividual , :Person ;
                                                                           :hasBrother :i5 , :i6 , :i7, :i8, :i9 .
      );
    1.
                                                                  :i5 a owl:NamedIndividual , :Person .
                                                                  :i6 a owl:NamedIndividual , :Person
:Few a owl:Class ;
                                                                  :i7 a owl:NamedIndividual , :Person
        owl:equivalentClass [
                                                                  :i8 a owl:NamedIndividual , :Person ;
:hasBrother :i5 , :i6 , :i7
       a owl:Class ;
       owl:intersectionOf (
                                                                  :i9 a owl:NamedIndividual , :Person .
        [ a owl:Restriction ;
```

Consider definitions of Average, Few and Many classes. They are defined with the use of owl:equivalentClass that allows put some restrictions on the class membership.

Such ontology, when processed with semantical reasoning turned, will be extended by inferred facts as follows.

- 1. i1 is an instance of Any and has two brothers declared explicitly (these are i2 and i3).
- 2. i2 is an instance of Any class and hasBrother i1, but not i3. It comes from the declaration of hasBrother as a Symmetric property but not Transitive. In fact this property cannot be declared as Transitive due to restrictions declared on classes Average, Few and Many. This is covered by the OWL 2 Web Ontology Language, Structural Specification and Functional-Style Syntax (Second Edition) (<u>https://www.w3.org/TR/owl2-syntax/#Global Restrictions%20 on Axioms in OWL 2 DL</u>) which requires that only simple object properties may be asserted to be reflexive or symmetric. Asserting that a property is transitive automatically makes it composite. Then the reasoner would rise an exception:

Non-simple property '<http://www.example.org/tkubik/ct#hasBrother>' or its inverse appears in the cardinality restriction 'ObjectMaxCardinality(4 <http://www.example.org/tkubik/ct#hasBrother> owl:Thing)'.

			ibik\Desktop\Sei	mOnt\cardinalityTest.ttl]	- • ×
e Edit View Reasoner Tools Re t thttp://www.example.org/tka ive Ontology × Entities × Individuals I ass hierarchy Class hierarchy (inferred) ss hierarchy: Person UIERCE t t t t t t t t t t t t t t t t t t t		× SPARQL Query ×			Search
<ul> <li>owl:Thing</li> <li>Any</li> <li>Average</li> <li>Few</li> <li>Hany</li> <li>Person</li> </ul>	<ul> <li>↓ 1.</li> <li>↓ 13</li> <li>↓ 14</li> <li>↓ 15</li> <li>↓ 16</li> <li>↓ 17</li> <li>↓ 18</li> <li>↓ 19</li> </ul>	Descriptions 2 Types Person Any Same Individual As Different Individuals		Property assertions 12 Object property assertions hasBrother 11 Data property assertions Negative object property assertions Negative data property assertions	0000

3. i4 is an instance of Any class and has five brothers declared explicitly.

< ct (http://www.example.org/tkubik/ct#): [C:\Users\tkubik\Desktop\SemOnt\cardinalityTest.ttl] – 🗆 🗙							
File Edit View Reasoner Tools Ret	actor Window H	elp					
< > Image: the state of the sta							
Active Ontology × Entities × Individuals by class × DL Query × SPARQL Query ×							
Class hierarchy Class hierarchy (inferred)	Instances: i400008	Annotations Usage					
Class hierarchy (inferred): Person	◆* 💥	Annotations: i4					
▼ ● owl:Thing ▼ ⊖ Any	For: 😑 Person	Annotations 🕀			-		
😑 Average	♦ i1 ♦ i2	Description: i4	0800	Property assertions: i4	0802		
	• i3	Types 🛨		Object property assertions 🕀			
	<ul> <li>● i4</li> <li>● i5</li> </ul>	Person	0000	hasBrother i5	0000		
	• i6	Any	<b>?</b> @	hasBrother i7			
	♦ i7 ♦ i8	Same Individual As 🕀		hasBrother i8	0000		
	• i9			hasBrother i9	0000		
		Different Individuals 🕀		Data property assertions 🕂			
				Negative object property assertions 🕂			
				Negative data property assertions 🕀			
Reasoner active 📝 Show Interences							

4. Because i1 has two brothers and i4 has five brothers one may think that both should belong to, respectively, Few and Many classes. But this is not the case. With the use of owl:minCardinality restriction one can set sufficient conditions for an instance to be a member of particular class (e.g. if the knowledge base includes information that someone has 2 brothers, this person can be assigned to Few class). But owl:maxCardinality restrictions are used to declare necessary conditions. Because of open world assumption it is difficult to say that someone has no more brothers that already recorded in the knowledge base (e.g. if the knowledge base includes information that someone has no more brothers that already recorded in the knowledge base (e.g. if the knowledge base includes information that someone has 3 brothers there is no guarantee that this person has no more brothers, but for sure this person cannot be assign to Few class because the max cardinality restriction would be broken). Combining necessary and sufficient conditions (by intersection operator) makes the resulting formula a necessary condition.

Fortunatelly there is a way of closing the world. The instances can be assign to the proper class by running SPARQL querry as in the example below (which result with a list list entities bolonging to Few class).

🭕 ct (http://www.example.org/tkubik/ct#) : [C:\Users\tkubik\Desktop\SemOnt\cardinalityTest.ttl] – 🗖 🗙						
File Edit View Reasoner Tools Refactor V	Vindow Help					
< > 🔄 ct (http://www.example.org/tkubik/ct#)	< > 🔄 ct (http://www.example.org/tkubik/ct#)					
Active Ontology × Entities × Individuals by class × DL Query × SPARQL Query ×						
SPARQL query:	SPARQL query:					
PREFIX cnt < http://www.example.org/tkubik/ct#> CONSTRUCT [ts a cntFew] WHERE WHERE WHERE GBOUP BY ts { GBOUP BY ts HAVING (?numBrothers>0 && ?numBrothers<=2) }						
Subject	Predicate	Object				
i1	rdf:type	Few				
	Execute					
		Reasoner active V Show Inferences				