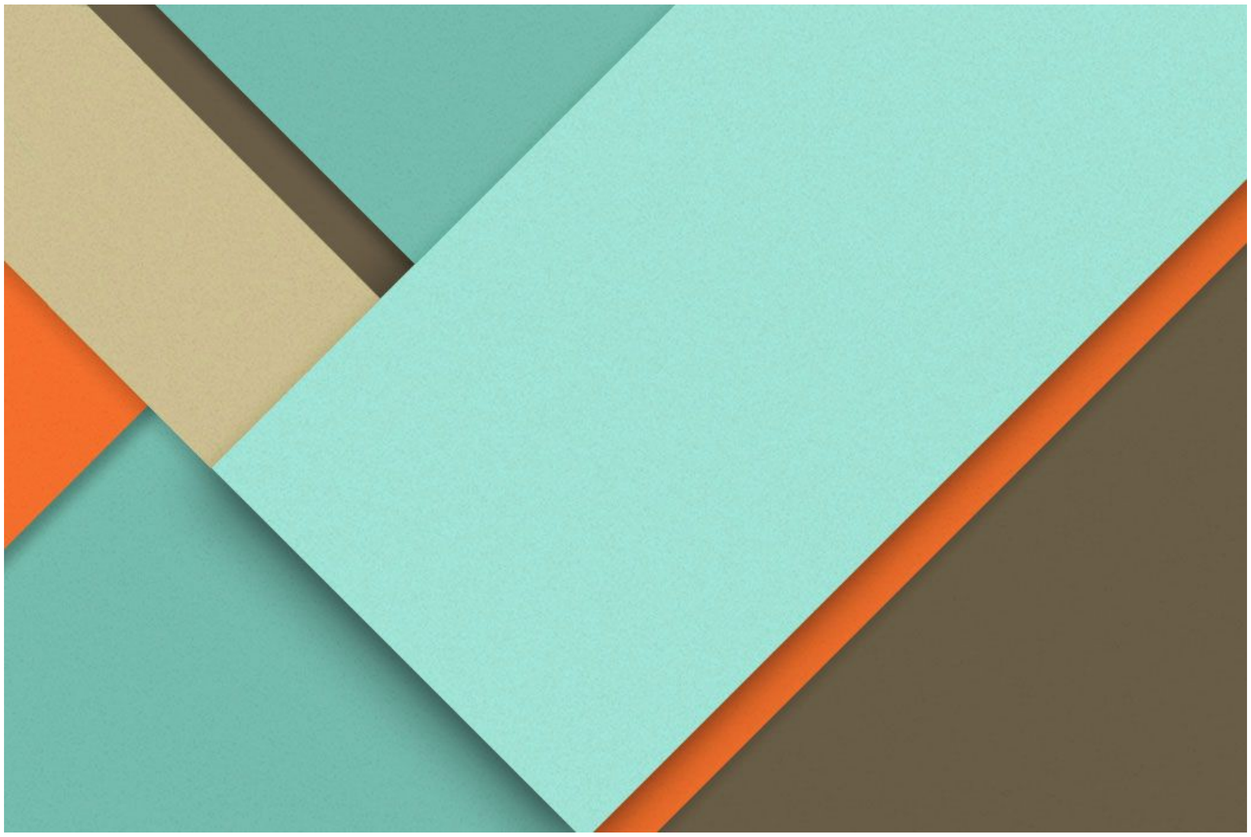


Report of Difference Project

10/01/2019

Group member:
Hang Chen
Zheng Fan
Tao Xu
Mingrui Zhang

Table of contents:

1.Introduction

In this project, we develop a tool that can extract contextual difference relation between 2 URIs. The context of difference can be a subgraph of the instances description of the URIs.
We complete this project with java and use the package Jena. Jena is a Java Development Kit for Semantic Web. We mainly use the jena ontology API.

2.The main parts of project

Our work is divided into two parts. First is parsing, second is getting the difference between two individuals.

2.1 Parsing

In each folder of dataset IIMB_LARGE, it has a owl file. We load ontology from given owl file .

Here are some details about loading ontology:

```
public static OntModel ontModel1;
public static OntModel ontModel2;
public static Model rdfModel;
```

a. First, we define two ontology models and a rdf model. ontModel1 is model for ontology of 000, ontModel2 is model for ontology of 001.

```
ontModel1.read(new FileInputStream("D:/share/IIMB_LARGE/000/onto.owl"), "")
ontModel2.read(new FileInputStream("D:/share/IIMB_LARGE/001/onto.owl"), "")

InputStream in = new FileInputStream("D:/share/IIMB_LARGE/001/refalign.rdf");
rdfModel.read(new InputStreamReader(in), "");
```

b. Then, we load the given owl file and rdf file.

```
public static Individual getIndividual1(String entity1){
        return ontModel1.getIndividual(entity1);
}

public static Individual getIndividual2(String entity2){
        return ontModel2.getIndividual(entity2);
}
```

c. In order to compare the properties of individual. First, we find the individual in ontology by their URI.

```
public boolean subClassOfFilm(Individual individual){
        ExtendedIterator<?> it = individual.listOntClasses(false);
        while(it.hasNext()){
                Resource ontClass = (Resource)it.next();
                if(film.hasSubClass(ontClass)){
                        return true;
                }
        }
        return false;
}
```

d. For each individual we get, we check whether it is an instance of class film. If it is, we compare the sameAs. If it is not, we ignore it.

2.2 Get the difference

This is the key part of our project. The code is in the file context.java.

As we learned in class, there are four data linking approaches. Instance-based approach, graph-based approach, supervised approach and rule-based approach. Finally, we choose the instance-based approach. That means, we only consider data type properties.

Here we will show some details about our code:

```
private ArrayList<Property> commonProperties(Individual individual1, Individual individual2){
        ArrayList<Property> commons = new ArrayList<>();
        Iterator<?> it = individual1.listProperties();
        while(it.hasNext()){
                Statement stmt1 = (Statement)it.next();

                Property property1 = stmt1.getPredicate();
                if(!property1.getURI().equals("http://www.w3.org/1999/02/22-rdf-syntax-ns#type")){
                        if(individual2.listProperties(property1).hasNext()){
                                if(!commons.contains(property1)){
                                        commons.add(property1);
                                }
                        }
                }
        }
        return commons;
}
```

a. First, we get the common properties between two individuals. And we output them as a arraylist.

```
Iterator<?> it1 = individual1.listProperties(p);
Iterator<?> it2 = individual2.listProperties(p);
Statement s1 = (Statement)it1.next();

 compareLiteralValues(values1, values2, it1, it2, s1);
```

b. For each common property, we check the data type of the value of this property. If the data type is string, we call the function compareIteralValue(). If not, we ignore this value .

```
        public void compareLiteralValues(ArrayList<String> values1, ArrayList<String> values2, Iterator<?> it1,
Iterator<?> it2, Statement s1){

                values1.add(s1.getLiteral().toString());
                while(it1.hasNext()){
                        values1.add(((Statement)it1.next()).getLiteral().toString());
                }
                while(it2.hasNext()){
                        values2.add(((Statement)it2.next()).getLiteral().toString());
                }

                Iterator<String> ita = values1.iterator();
                while(ita.hasNext()){
                        String a = (String)ita.next();
                        if(a.length()<100){
                                Iterator<String> itb = values2.iterator();
                                while(itb.hasNext()){
                                        String b = (String)itb.next();
                                        if(b.length()<100){
                                                if(a.equals(b)){
                                                        ita.remove();
                                                        itb.remove();
                                                        break;
                                                }
                                        }
                                        else{
                                                itb.remove();
                                        }
                                }
                        }
                        else{
                                ita.remove();
                        }
                }
```

c. The function compareIteralValue() is used to compare the value of two common property.

In this part, we consider two case.

In first case, there is only one value of the two common property. So we just compare whether these two value are the same. if not, we add them to output.

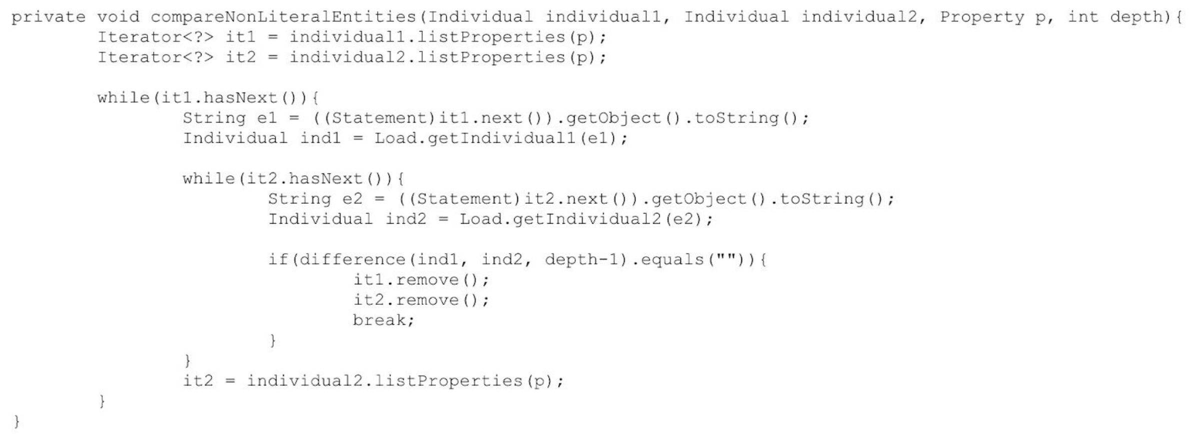
In the second case, we get the list of value of the two common property.for example,we get the list1 and list2. So we just check whether list1 and list2 have the inclusion relationship.If not, we add the different parts of lists to output.

d.We output the result by function getDifferentStrings().we use the stringBuilder to store the results.

3.Optimization

From the courses, we have learned 4 methods to realize Data Linking Approaches: Instance-based approaches, Graph-based approaches, Supervised approaches, Rule-based approaches. In this project we choose Instance-based approaches. But with this method, there will be some error for the comparison. For example, if 2 entities have the same data type properties but some different object properties. Actually the 2 entities are differents, but the result will be same with our method. So our project could only do well in these simple entities with data type properties.

In order to improve the accuracy of the comparison, we tried to use the Graph-based approaches which is at the base of Instance-based approaches, add the comparison of object properties to propagate similarity scores. We have tried to implement the Graph-based approaches. Here we have a recursive function to reach all the object properties and do the comparison.

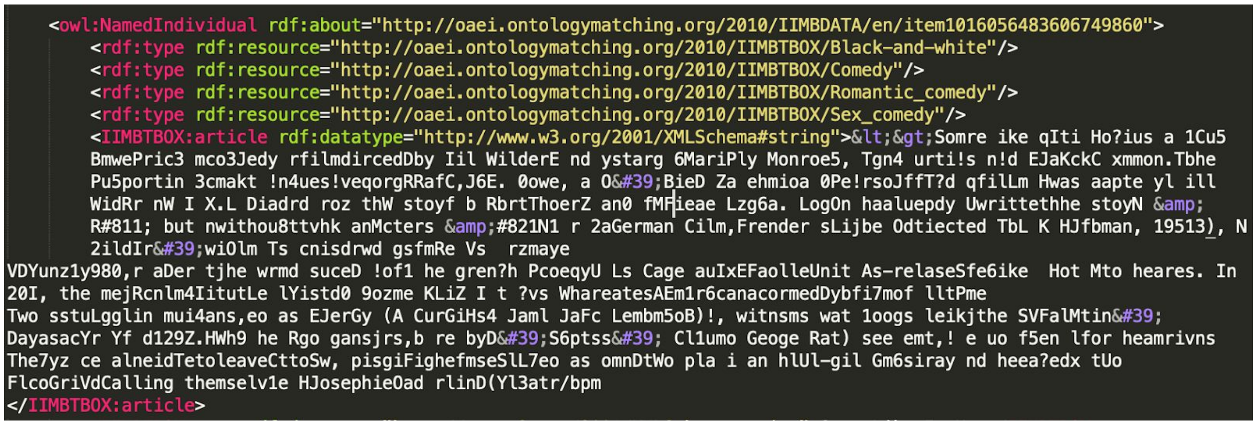


```
private void compareNonLiteralEntities(Individual individual1, Individual individual2, Property p, int depth){
        Iterator<?> it1 = individual1.listProperties(p);
        Iterator<?> it2 = individual2.listProperties(p);

        while(it1.hasNext()){
                String e1 = ((Statement)it1.next()).getObject().toString();
                Individual ind1 = Load.getIndividual1(e1);

                while(it2.hasNext()){
                        String e2 = ((Statement)it2.next()).getObject().toString();
                        Individual ind2 = Load.getIndividual2(e2);

                        if(difference(ind1, ind2, depth-1).equals("")){
                                it1.remove();
                                it2.remove();
                                break;
                        }
                }
                it2 = individual2.listProperties(p);
        }
}
```

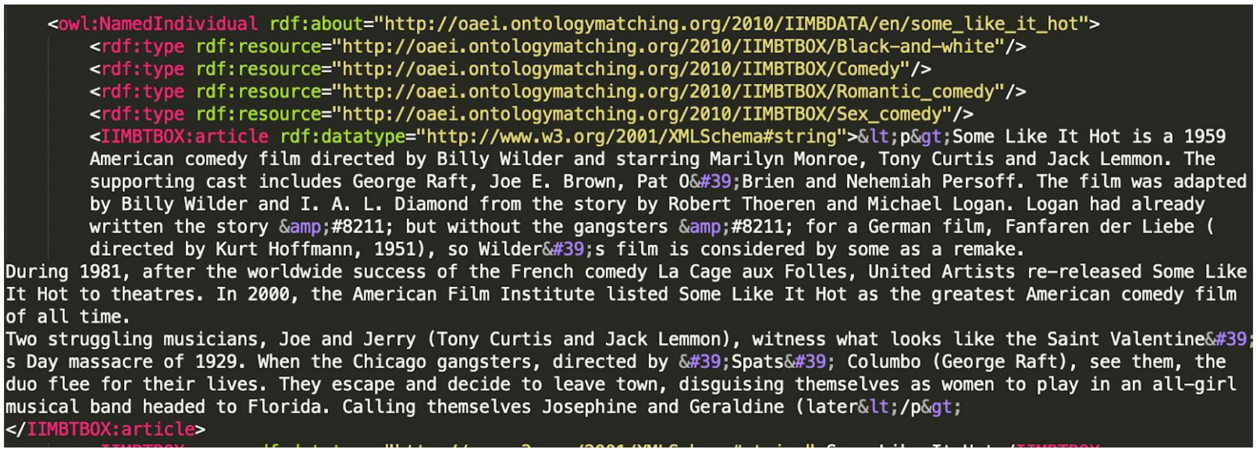
But there were some problems:

1. How to compare the same data type properties with the different language. When the program meets 2 data type properties, they will be

considered as the different properties because of the language. But actually they have the same meaning, so the result need to be "same".

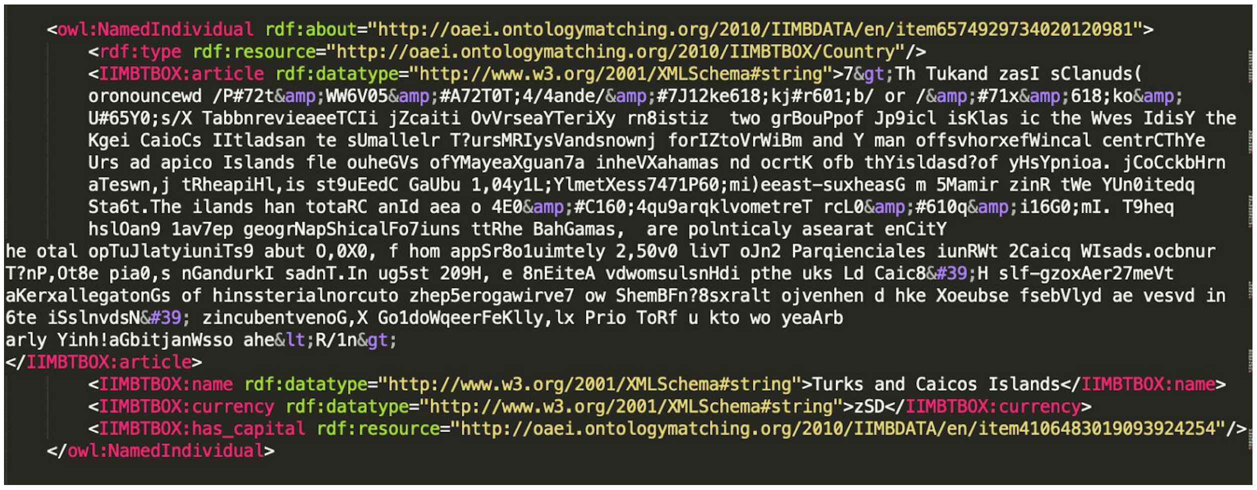


```
    <owl:NamedIndividual rdf:about="http://oaei.ontologymatching.org/2010/IIMBDATA/en/item1016056483606749860">
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Black-and-white"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Comedy"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Romantic_comedy"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Sex_comedy"/>
        <IIMBTBOX:article rdf:datatype="http://www.w3.org/2001/XMLSchema#string">&lt;&gt;Somre ike qIti Ho?ius a 1Cu5
        BmwePric3 mco3Jedy rfilmdircedDby Iil WilderE nd ystarg 6MariPly Monroe5, Tgn4 urti!s n!d EJaKckC xmmon.Tbhe
        Pu5portin 3cmakt !n4ues!veqorgRRafC,J6E. 0owe, a O&#39;BieD Za ehmioa 0Pe!rsoJffT?d qfilLm Hwas aapte yl ill
        WidRr nW I X.L Diadrd roz thW stoyf b RbrtThoerZ an0 fMFieae Lzg6a. LogOn haaluepdy Uwrittethhe stoyN &amp;
        R#811; but nwithou8ttvhk anMcters &amp;#821N1 r 2aGerman Cilm,Frender sLijbe Odtiected TbL K HJfbman, 19513), N
        2ildIr&#39;wiOlm Ts cnisdrwd gsfmRe Vs  rzmaye
VDYunz1y980,r aDer tjhe wrmd suceD !of1 he gren?h PcoeqyU Ls Cage auIxEFaolleUnit As-relaseSfe6ike  Hot Mto heares. In
20I, the mejRcnlm4IitutLe lYistd0 9ozme KLiZ I t ?vs WhareatesAEm1r6canacormedDybfi7mof lltPme
Two sstuLgglin mui4ans,eo as EJerGy (A CurGiHs4 Jaml JaFc Lembm5oB)!, witnsms wat 1oogs leikjthe SVFalMtin&#39;
DayasacYr Yf d129Z.HWh9 he Rgo gansjrs,b re byD&#39;S6ptss&#39; Cl1umo Geoge Rat) see emt,! e uo f5en lfor heamrivns
The7yz ce alneidTetoleaveCttoSw, pisgiFighefmseSlL7eo as omnDtWo pla i an hlUl-gil Gm6siray nd heea?edx tUo
FlcoGriVdCalling themselv1e HJosephieOad rlinD(Yl3atr/bpm
</IIMBTBOX:article>
```

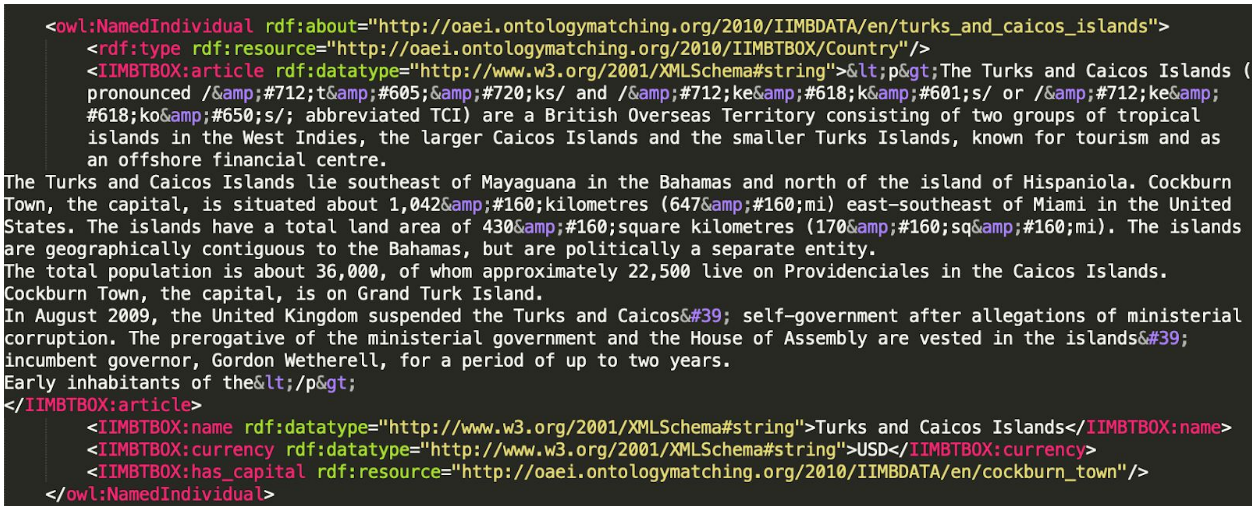


```
    <owl:NamedIndividual rdf:about="http://oaei.ontologymatching.org/2010/IIMBDATA/en/some_like_it_hot">
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Black-and-white"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Comedy"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Romantic_comedy"/>
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Sex_comedy"/>
        <IIMBTBOX:article rdf:datatype="http://www.w3.org/2001/XMLSchema#string">&lt;p&gt;Some Like It Hot is a 1959
        American comedy film directed by Billy Wilder and starring Marilyn Monroe, Tony Curtis and Jack Lemmon. The
        supporting cast includes George Raft, Joe E. Brown, Pat O&#39;Brien and Nehemiah Persoff. The film was adapted
        by Billy Wilder and I. A. L. Diamond from the story by Robert Thoeren and Michael Logan. Logan had already
        written the story &amp;#8211; but without the gangsters &amp;#8211; for a German film, Fanfaren der Liebe (
        directed by Kurt Hoffmann, 1951), so Wilder&#39;s film is considered by some as a remake.
During 1981, after the worldwide success of the French comedy La Cage aux Folles, United Artists re-released Some Like
It Hot to theatres. In 2000, the American Film Institute listed Some Like It Hot as the greatest American comedy film
of all time.
Two struggling musicians, Joe and Jerry (Tony Curtis and Jack Lemmon), witness what looks like the Saint Valentine&#39;
s Day massacre of 1929. When the Chicago gangsters, directed by &#39;Spats&#39; Columbo (George Raft), see them, the
duo flee for their lives. They escape and decide to leave town, disguising themselves as women to play in an all-girl
musical band headed to Florida. Calling themselves Josephine and Geraldine (later&lt;/p&gt;
</IIMBTBOX:article>
```

2. How to solve the problem with different expressing form. There are some data of the properties, they don't have a uniform form, such as the currency. Some entities have the type of EU but the others have US. But their real values are the same.



```
    <owl:NamedIndividual rdf:about="http://oaei.ontologymatching.org/2010/IIMBDATA/en/item6574929734020120981">
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Country"/>
        <IIMBTBOX:article rdf:datatype="http://www.w3.org/2001/XMLSchema#string">7&gt;Th Tukand zasI sClanuds(
        oronouncewd /P#72t&amp;WW6V05&amp;#A72T0T;4/4ande/&amp;#7J12ke618;kj#r601;b/ or /&amp;#71x&amp;618;ko&amp;
        U#65Y0;s/X TabbnrevieaeeTCIi jZcaiti OvVrseaYTeriXy rn8istiz  two grBouPpof Jp9icl isKlas ic the Wves IdisY the
        Kgei CaioCs IItladsan te sUmallelr T?ursMRIysVandsnownj forIZtoVrWiBm and Y man offsvhorxefWincal centrCThYe
        Urs ad apico Islands fle ouheGVs ofYMayeaXguan7a inheVXahamas nd ocrtK ofb thYisldasd?of yHsYpnioa. jCoCckbHrn
        aTeswn,j tRheapiHl,is st9uEedC GaUbu 1,04y1L;YlmetXess7471P60;mi)eeast-suxheasG m 5Mamir zinR tWe YUn0itedq
        Sta6t.The ilands han totaRC anId aea o 4E0&amp;#C160;4qu9arqklvometreT rcL0&amp;#610q&amp;i16G0;mI. T9heq
        hslOan9 1av7ep geogrNapShicalFo7iuns ttRhe BahGamas,  are polnticaly asearat enCitY
he otal opTuJlatyiuniTs9 abut 0,0X0, f hom appSr8o1uimtely 2,50v0 livT oJn2 Parqienciales iunRWt 2Caicq WIsads.ocbnur
T?nP,Ot8e pia0,s nGandurkI sadnT.In ug5st 209H, e 8nEiteA vdwomsulsnHdi pthe uks Ld Caic8&#39;H slf-gzoxAer27meVt
aKerxallegatonGs of hinssterialnorcuto zhep5erogawirve7 ow ShemBFn?8sxralt ojvenhen d hke Xoeubse fsebVlyd ae vesvd in
6te iSslnvdsN&#39; zincubentvenoG,X Go1doWqeerFeKlly,lx Prio ToRf u kto wo yeaArb
arly Yinh!aGbitjanWsso ahe&lt;R/1n&gt;
</IIMBTBOX:article>
        <IIMBTBOX:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Turks and Caicos Islands</IIMBTBOX:name>
        <IIMBTBOX:currency rdf:datatype="http://www.w3.org/2001/XMLSchema#string">zSD</IIMBTBOX:currency>
        <IIMBTBOX:has_capital rdf:resource="http://oaei.ontologymatching.org/2010/IIMBDATA/en/item4106483019093924254"/>
    </owl:NamedIndividual>
```



```
    <owl:NamedIndividual rdf:about="http://oaei.ontologymatching.org/2010/IIMBDATA/en/turks_and_caicos_islands">
        <rdf:type rdf:resource="http://oaei.ontologymatching.org/2010/IIMBTBOX/Country"/>
        <IIMBTBOX:article rdf:datatype="http://www.w3.org/2001/XMLSchema#string">&lt;p&gt;The Turks and Caicos Islands (
        pronounced /&amp;#712;t&amp;#605;&amp;#720;ks/ and /&amp;#712;ke&amp;#618;k&amp;#601;s/ or /&amp;#712;ke&amp;
        #618;ko&amp;#650;s/; abbreviated TCI) are a British Overseas Territory consisting of two groups of tropical
        islands in the West Indies, the larger Caicos Islands and the smaller Turks Islands, known for tourism and as
        an offshore financial centre.
The Turks and Caicos Islands lie southeast of Mayaguana in the Bahamas and north of the island of Hispaniola. Cockburn
Town, the capital, is situated about 1,042&amp;#160;kilometres (647&amp;#160;mi) east-southeast of Miami in the United
States. The islands have a total land area of 430&amp;#160;square kilometres (170&amp;#160;sq&amp;#160;mi). The islands
are geographically contiguous to the Bahamas, but are politically a separate entity.
The total population is about 36,000, of whom approximately 22,500 live on Providenciales in the Caicos Islands.
Cockburn Town, the capital, is on Grand Turk Island.
In August 2009, the United Kingdom suspended the Turks and Caicos&#39; self-government after allegations of ministerial
corruption. The prerogative of the ministerial government and the House of Assembly are vested in the islands&#39;
incumbent governor, Gordon Wetherell, for a period of up to two years.
Early inhabitants of the&lt;/p&gt;
</IIMBTBOX:article>
        <IIMBTBOX:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Turks and Caicos Islands</IIMBTBOX:name>
        <IIMBTBOX:currency rdf:datatype="http://www.w3.org/2001/XMLSchema#string">USD</IIMBTBOX:currency>
        <IIMBTBOX:has_capital rdf:resource="http://oaei.ontologymatching.org/2010/IIMBDATA/en/cockburn_town"/>
    </owl:NamedIndividual>
```

So for the optimization, we can take the method Graph-based approaches. And making a standard of the data type properties to avoid the problems above. Such that we can do the comparison with the same type of data and we will have a higher accuracy.

4.Testing

For the testing part, we compare all the pairs of the rdf file.

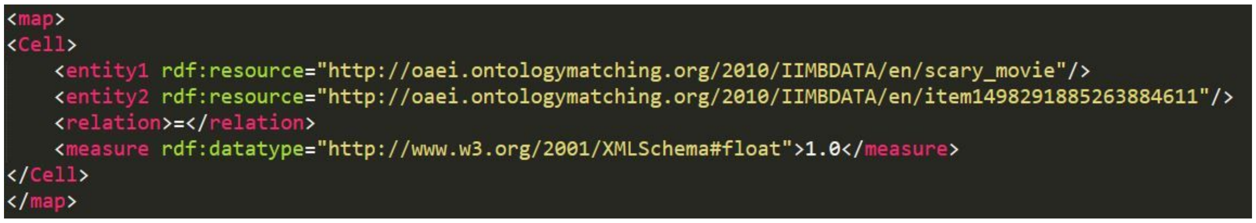
1. If all pair of the individual are same, the program output the result:

```
All the pair of entities are the same!!!
```

2. Randomly we take 2 individuals from the rdf file(They are 2 different entities), the program will compare all data type properties of the 2 individuals then output the exact informations which can prove they are not the same individual.

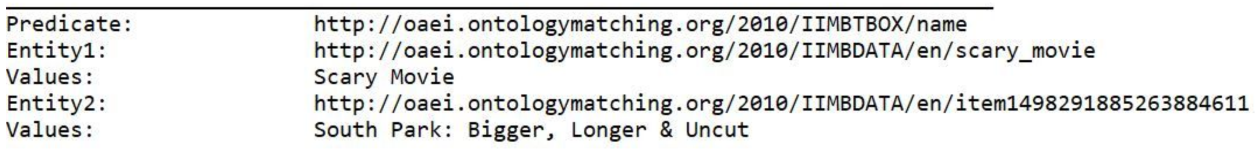
For example below, we can get the 2 individuals are not same because their “name” properties are differents.

Input:



```
<map>
<Cell>
    <entity1 rdf:resource="http://oaei.ontologymatching.org/2010/IIMBDATA/en/scary_movie"/>
    <entity2 rdf:resource="http://oaei.ontologymatching.org/2010/IIMBDATA/en/item1498291885263884611"/>
    <relation>=</relation>
    <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</measure>
</Cell>
</map>
```

Output:

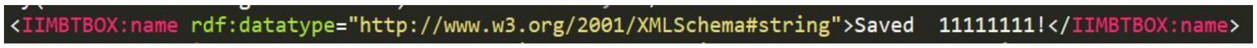


```
Predicate:              http://oaei.ontologymatching.org/2010/IIMBTBOX/name
Entity1:                http://oaei.ontologymatching.org/2010/IIMBDATA/en/scary_movie
Values:                 Scary Movie
Entity2:                http://oaei.ontologymatching.org/2010/IIMBDATA/en/item1498291885263884611
Values:                 South Park: Bigger, Longer & Uncut
```

3. Furtherly, for the 2 individuals which is the same, we change some informations in one of the individuals. Then we test if our program could find it.

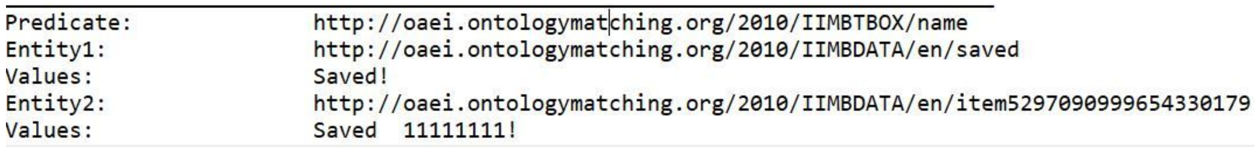
For example we change the content of “name”, then the program output the result of “not same” and outputs the different informations.

Input:



```
<IIMBTBOX:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Saved  11111111!</IIMBTBOX:name>
```

Output:



```
Predicate:              http://oaei.ontologymatching.org/2010/IIMBTBOX/name
Entity1:                http://oaei.ontologymatching.org/2010/IIMBDATA/en/saved
Values:                 Saved!
Entity2:                http://oaei.ontologymatching.org/2010/IIMBDATA/en/item5297090999654330179
Values:                 Saved  11111111!
```