

Thursday' talks

Who cares about DBpedia properties?
Use infoboxes' properties to create
better SPARQL queries

General motivation

- You learn something
- I get feedback
- Implicit agreement
 - I do my best trying to teach you something **valuable**
 - You give me **valuable** feedback
 - Please, be as polite as possible 😊

My motivation

- Long tail work on DBpedia properties
- Submitted as short paper at iSEMANTICS

Who cares about DBpedia properties? Use infoboxes' properties to create better SPARQL queries

Mariano Rico^{*}
Ontology Engineering Group
UPM
Spain
mariano.rico@fi.upm.es

Nandana
Mihindukulasooriya[†]
Ontology Engineering Group
UPM
Spain
nmihindu@fi.upm.es

Asunción Gomez-Perez
Ontology Engineering Group
UPM
Spain
asun@fi.upm.es

ABSTRACT

DBpedia extracts most of its data from Wikipedia's infoboxes. Manually-created "mappings" link infobox attributes to DBpedia ontology properties (dbo properties) producing most used DBpedia triples. However, infobox attributes without a mapping produce triples with properties in a differ-

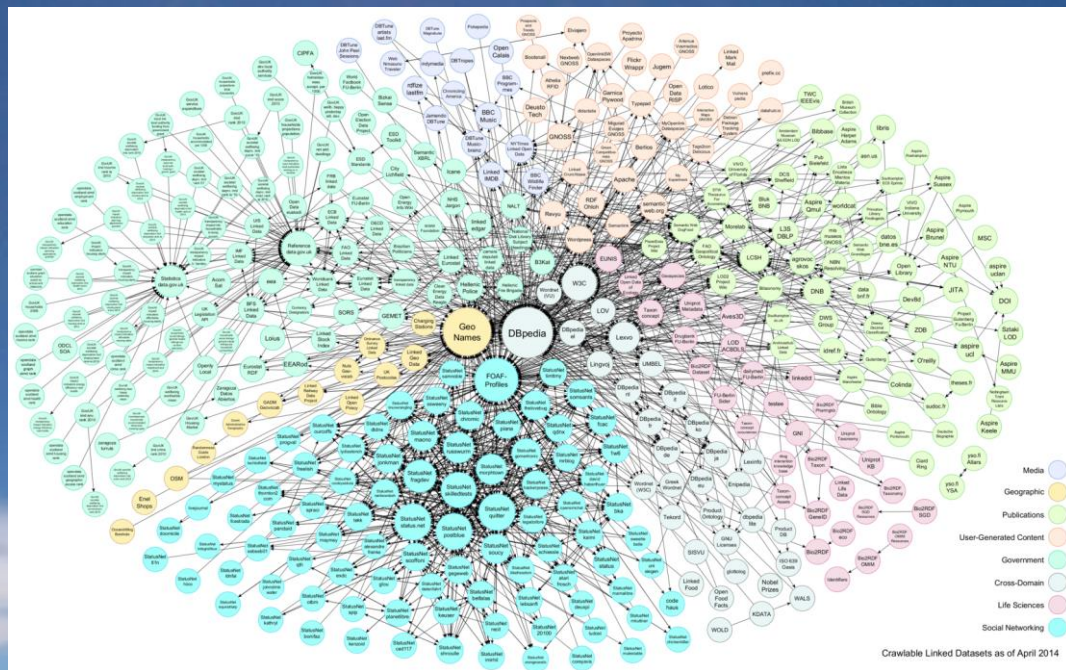
created mappings¹. The analysis of the Spanish DBpedia (esDBpedia) found [4] that, despite the high number of mappings (100+ classes), for each 4 triples containing a dbo property there is 1 triple containing a dbp property. In this work, we extend this analysis to English and German DBpedias, with similar results. For instance, in the English

In the LOD galaxy...



DBpedia

- Kernel of the LOD
- Several DBpedias
 - English
 - Spanish
 - ... up to 16



DBpedia

- How to explore these huge datasets?
 - By means of SPARQL queries ☹
 - Explore them using LOUPE ☺
- loupe.linkeddata.es

Explore 2, 209, 980, 256 triples from 32 datasets

What is Loupe?
Loupe is a tool that helps you to inspect a dataset to understand which vocabularies (classes, and properties) are used including statistics and frequent triple patterns. Starting to from the high-level statistics, Loupe allows you to zoom into details down to the corresponding triples.

Select one of our indexed datasets:

Dataset: DBpedia (English) [Explore](#)

DBpedia (EN), Wikidata Stmts, Wikidata Terms, Linked Brainz, DBpedia (ES), DBpedia (IT), DBpedia (DE), DBpedia (FR), DBpedia (NL), DBpedia (PL), DBpedia (PT), DBpedia (RU), OpenAIRE

Developed by Ontology Engineering Group, Universidad Politécnica de Madrid
Supported by the 4V Spanish national project (TIN2013-46238-C4-2-R) and UNPM13-4E-1814
Powered by Elastic Search, Docker, OpenLink Virtuoso Open Source
Built with Bootstrap, DataTables, D3.js
Icons from Fontawesome

Ontology Engineering Group

Claims

- C1: DBpedia has more dbp properties than expected
 - you know dbo properties
- C2: SPARQL queries barely use dbp properties
 - We need logs
- C3: The method proposed can enhance DBpedia SPARQL queries
 - Enhance? (more results)

```

N C [img alt="Spanish flag icon" data-bbox="180 15 215 45"] [img alt="English flag icon" data-bbox="220 15 255 45"] [img alt="Document icon" data-bbox="360 15 395 45"] ➤ Avanzado ➤ Caracteres especi
{{referencias adicionales:}}
{{Ficha de actor
|nombre = Woody Allen |[[Archivo:Prince of
Emblem.svg|15px|Premio Principe de Asturias]]
|foto = Woody Allen at the Tribeca Film
|nombre de nacimiento = Allan Stewart Konigsberg
|fecha de nacimiento = {{fecha|112|1935|edad}}
|lugar de nacimiento = [[bandera|USA]] [[Brooklyn]], [[
|nacionalidad = [[Estados Unidos|Estadounidense]]
|ocupación = [[Director de cine|Director]], [[
|[[dramaturgo]], [[humorista]], [[escritor]]
|año debut = 1955
|año retiro = presente
|cónyuge = [[Soon-Yi Previnn]] <small>[[1997-
|hijos = 2
|premios óscar = ''[[Anexo:Óscar a la mejor direc
[[[:Categoria:Películas de 1977|1977]] ''[[
|original|''Mejor guion original'']] ''<br>[[
Hall]]''<br>[[[:Categoria:Películas de 11
|[:Categoria:Películas de 2011|2011]] ''[[
|premios globo de oro = ''[[Anexo:Globo d
|[:Categoria:Películas de 1985|1985]] ''[[
|[:Categoria:Películas de 2012|2012]] ''[[
|]] ''<br>[[[:Categoria:Películas de 2014
|ta = ''[[Anexo:BAFTA

```

Extractors

Label	Category	Image
-------	----------	-------

Redirect	Disambiguation
----------	----------------

Abstract	Geo	Pagelink
----------	-----	----------

Generic Infobox

Mapping-based Infobox

Parsers

DateTime

units | Geo

String-List

Numbers



Property	dbpedia-owl:abstract
----------	----------------------

Value

- Allan Stewart Königsberg (Brooklyn, 1 de diciembre de 1915 Allen, es un director, guionista, actor, músico, dramaturgo ganador del premio Óscar en cuatro ocasiones. Es uno de la era moderna, rodando una película al año desde 1919 película considerada por muchos como una de las mejores premio Óscar al Mejor director en 1977. Mantiene una gran Keaton. Sus grandes influencias cinematográficas oscilan Federico Fellini, hasta comediantes como Groucho Marx;
- [dpedia: Annie_Hall](#)
- [dpedia: Hannah_y_sus_hermanas](#)
- [dpedia: Midnight_in_Paris](#)
- [dpedia: Medianoche_en_Paris](#)
- [dpedia: Hannah_and_Her_Sisters](#)
- [dpedia: Anexo_César_a_la_mejor_pelicula_extranjera](#)
- [dpedia: Anexo:Premio_Donostia_del_Festival_de_San_S](#)
- [dpedia: Manhattan_\(película\)](#)
- [dpedia: Premio_Príncipe_de_Asturias_de_las_Artes](#)
- [dpedia: The_Purple_Rose_of_Cairo](#)
- [dpedia: César_a_la_mejor_pelicula_extranjera](#)
- [dpedia: Premio_Donostia](#)
- [Allan Stewart Königsberg](#)
- [dpedia: Brooklyn](#)

Rendering

Triple store

RDF

Claim C1

- DBpedia has more dbp properties than expected
 - At least for the 3 DBpedias Analyzed (Eng, Spa, Ger)
- We get this from LOUPE (Eng)
 - Number of properties for the namespaces
 - But, what about triples using that properties?
 - LOUPE does not know (yet)

List of distinct namespaces and the number of properties per each (483 namespaces)

Show entries

Namespace	Property Count
http://dbpedia.org/property/	58239
http://dbpedia.org/ontology/	1338
http://www.openlinksw.com/schemas/virtrdf#	86
http://dbpedia.org/ontology/Planet/	13
http://xmlns.com/foaf/0.1/	12

Claim C1

- We need a “strong” SPARQL query
 - Any EP would produce timeout
 - We have superpowers ☺

Table 1: Top 10 DBpedia properties for the English, Spanish and German DBpedias (2015-04 version).

English DBpedia		Spanish DBpedia		German DBpedia	
dbpedia.org		dbpedia.org		dbpedia.org	
URI	Triples	URI	Triples	URI	Triples
dbp:hasPhotoCollection	4,041,585	dbp:wikiPageUsesTemplate	3,402,499	dbp:name	494,852
dbp:name	4,021,368	dbp:nombre	558,837	dbp:geburtsort	305,063
dbp:title	1,452,504	dbp:título	327,498	dbp:kurzbeschreibung	283,695
dbp:subdivisionType	1,257,766	dbp:name	230,763	dbp:geburtsdatum	283,405
dbp:shortDescription	1,194,274	dbp:tipoSuperior	225,868	dbp:typ	232,702
dbp:dateOfBirth	1,023,951	dbp:horario	203,890	dbp:viaf	169,145
dbp:subdivisionName	1,004,294	dbp:imagen	183,887	dbp:gnd	165,362
dbp:goals	969,216	dbp:familia	152,430	dbp:jahre	156,498
dbp:placeOfBirth	908,819	dbp:title	144,724	dbp:sterbedatum	144,209
dbp:birthPlace	903,529	dbp:ordo	142,196	dbp:alternativnamen	143,893
#props dbp	58,239	#props dbp	17,111	#props dbp	12,167
#props dbo	1,338	#props dbo	559	#props dbo	534
#triples dbp	78,125,087	#triples dbp	28,234,292	#triples dbp	10,483,987
#triples dbo	82,369,408	#triples dbo	90,389,560	#triples dbo	50,750,486

Claim C2

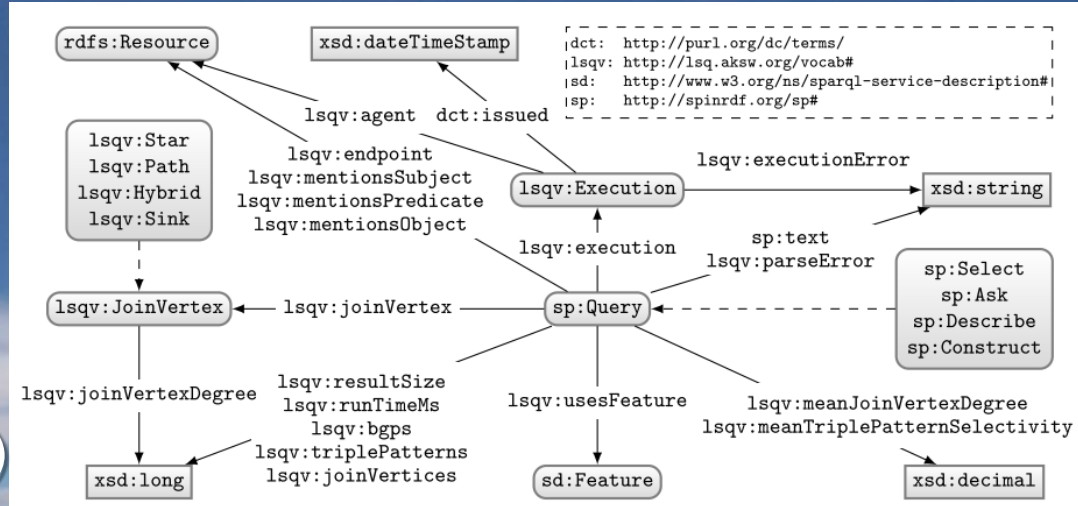
- SPARQL queries barely use dbp properties
 - We need to analyze SPARQL repositories (logs)
 - Fortunately
 - We have logs for the Spanish DBpedia
 - There is a public repository at aksw.github.io/LSQ
 - DBpedia (logs from 30/04/2010–20/07/2010). 1.7M queries
 - Linked Geo Data (24/11/2010–06/07/2011). 1.6 M queries
 - Semantic Web Dog Food (16/05/2014–12/11/2014). 1.4M queries
 - British Museum (08/11/2014–01/12/2014) 0.8M queries

Claim C2

- SPARQL queries barely use dbp properties
 - In the English DBpedia
 - We looked for SPARQL queries containing **both** dbp and dbo properties

– Results

- Out of 1,208,762 distinct queries
 - Only 2,328 queries
- Out of 3,041 distinct agents (IPs)
 - Only 473 IPs



Claim C3

- The method proposed can enhance DBpedia SPARQL queries
 - More results
- How? See an example

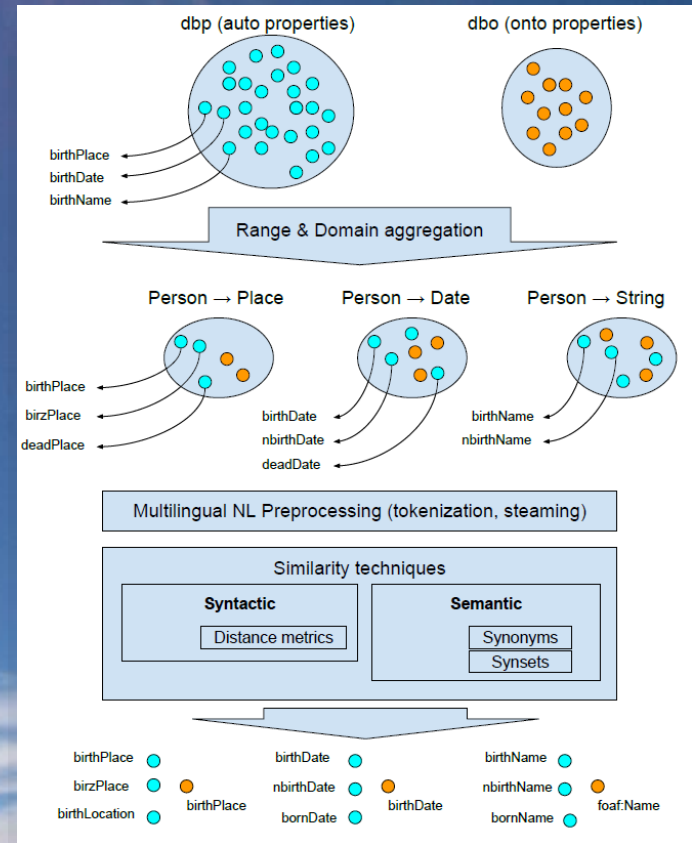
```
1 PREFIX dbo: <http://dbpedia.org/ontology/>
2 select ?s ?bp {
3   ?s dbo:birthPlace ?bp .
4 }
```

Listing 1: Original SPARQL query

```
1 PREFIX dbo: <http://dbpedia.org/ontology/>
2 PREFIX dbp: <http://dbpedia.org/property/>
3
4 select ?s ?bp where {
5   ?s ?p ?bp .
6   VALUES ?p {
7     dbo:birthPlace #typical dbo property
8     #Alternative dbp properties
9     dbp:birthPlcace dbp:birthplace
10    dbp:birhPlace   dbp:bithPlace
11    dbp:birtPlace   dbp:biRthPlace
12  }
13 }
```

Claim C3

- The method
- Step 1: aggregate properties into groups according to their domain and range
- Step 2: Multilingual NL preprocessing
- Step 3: aggregate properties by similarity (syntactic and semantic)



Claim C3

- Results. dbp→dbo mapping. Example for dbo:birthPlace for Eng, Spa, Ger.

DBpedia	dbo prop	dbp prop					Δ_1
		Syntactic			Semantic		
English	birthPlace	birthPlace	birthplace	placeofbirth	cityofbirth	cityofbirthPlace	350%
		birthPlac	birthdplace	birthPalce	cityOfBirth	birthLocation	
		birthPlace	PlaceOfBirth	laceOfBirth			
		oplaceOfBirth	birthPlace.	birthPlacE			
		birthPalce	birthPlae	birthPace	birthPlaxe		
		birtPlace	birthPlace	bithPlace	brithPlace		
		nbirthPlace	birthplace	birghPlace			
		birthdplace	biRthPlace	birth	placebirth		
		placeOfBirth	placOfBirth	birthPlaceOf			
		birthPlae					
Spanish	birthPlace	lugarDeNacimiento		lugarNacimiento	ciudadnacimiento	221%	
		lugarNacimiento		lugarNacimiento	ciudadDenacimiento		
		lugarDenacimiento		lugarNacimiento	paidsnacimiento		paissNacimiento
		lugarNaciento			birthPlace		birthplace
German	birthPlace	geburtsort	birthplace	birthPlace	geburtsland	countryofbirth	134%
		placeOfBirth	placeofbirth				

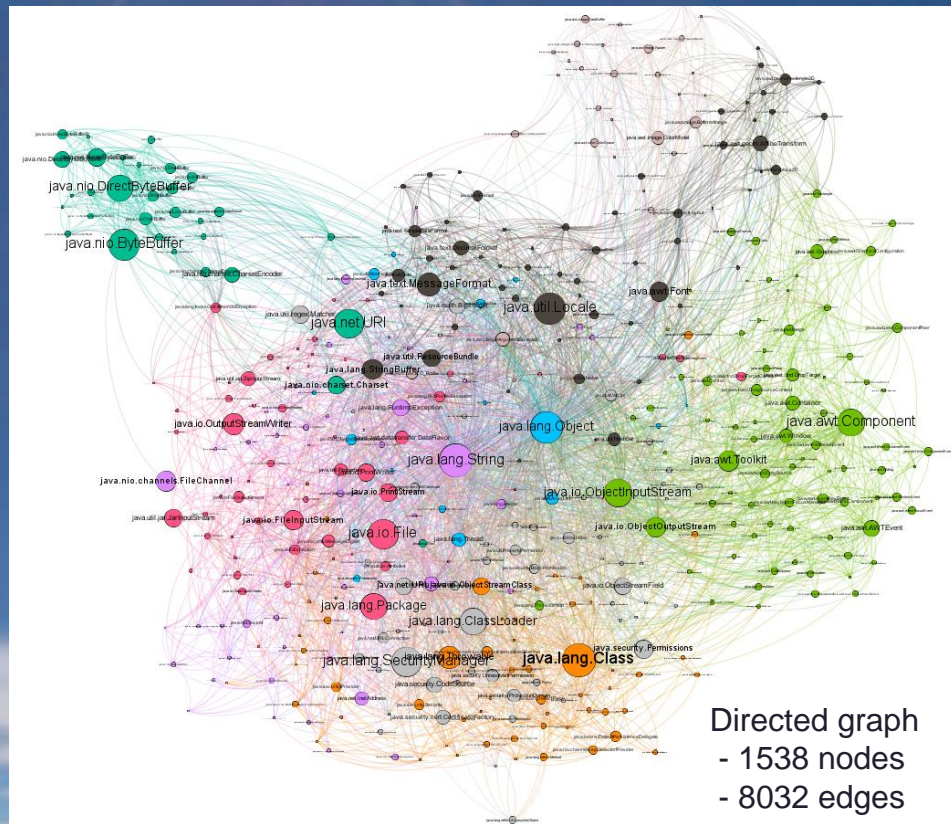
Future work

- Ideal objective: a stable mapping for a given language
 - Drawbacks
 - Fine tuning of algorithms and parameters
 - This requires users validation
 - Detailed study of precision and recall
 - For many common classes
 - For many common SPARQL queries



And another thing...

- Visualizing large datasets
 - E.g. The Java Compile-Time Dependency graph
 - Nodes are Java classes and directed edges are compile-time dependencies between two classes.
 - The data provided contains the dependencies for all classes under the `java.*` packages for JDK 1.4.2



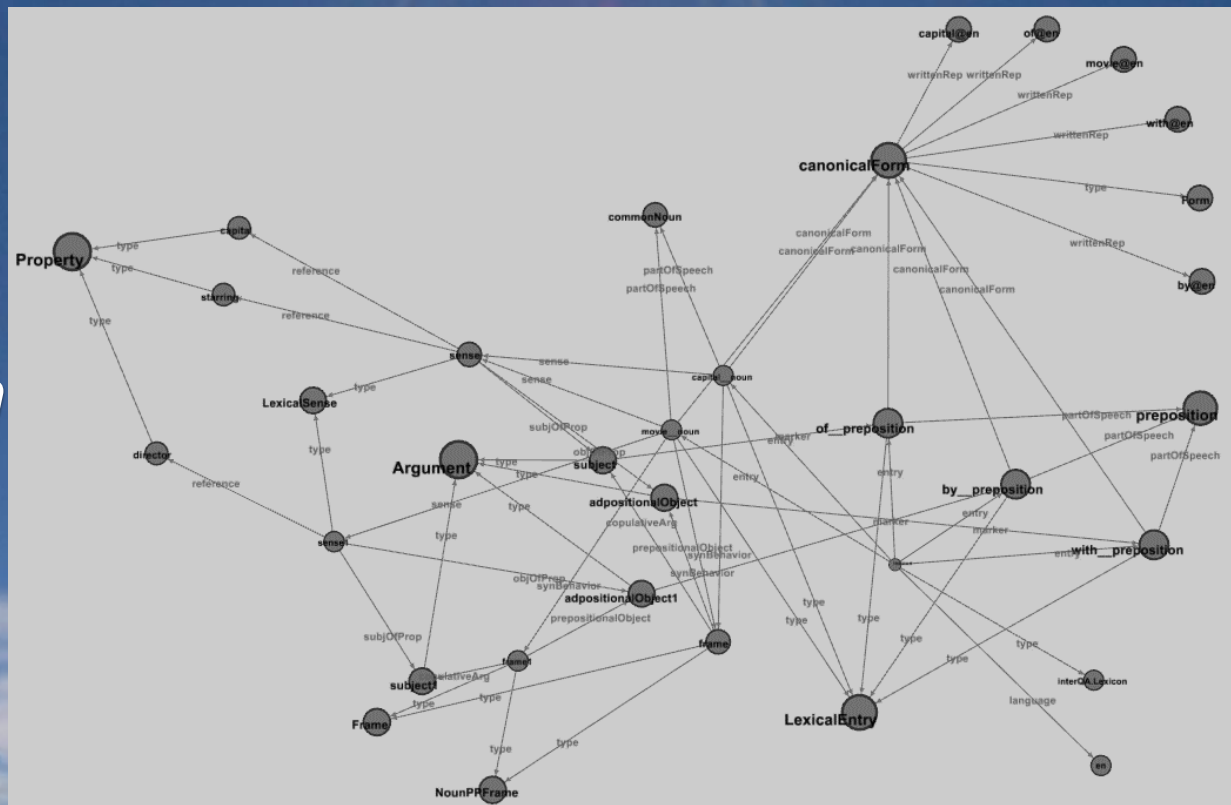
And another thing...

- Visualizing (very) large datasets
 - E.g. a *Lemon* dataset



The image displays a dense network graph representing the Lemon dataset. Nodes are represented by dark grey circles, and edges are thin grey lines. The graph is highly interconnected, with many nodes having multiple incoming and outgoing edges. Key nodes include 'Property', 'LexicalEntry', 'Argument', 'Frame', 'NounPPFrame', 'LexicalSense', 'sense', 'subject', 'adpositionalObject', 'adpositionalObject1', 'frame', 'canonicalForm', 'capitalization', 'writtenRep', 'movie', 'with', 'by', 'of', 'preposition', 'with_preposition', 'by_preposition', 'of_preposition', 'entry', 'marker', 'interCLexicon', 'language', 'on', 'type', 'reference', 'sense', 'subjProp', 'relativeArg', 'type', 'partOfSpeech', 'commonNoun', 'capitalNoun', 'entity', 'entity1', 'entity2', 'entity3', 'entity4', 'entity5', 'entity6', 'entity7', 'entity8', 'entity9', 'entity10', 'entity11', 'entity12', 'entity13', 'entity14', 'entity15', 'entity16', 'entity17', 'entity18', 'entity19', 'entity20', 'entity21', 'entity22', 'entity23', 'entity24', 'entity25', 'entity26', 'entity27', 'entity28', 'entity29', 'entity30', 'entity31', 'entity32', 'entity33', 'entity34', 'entity35', 'entity36', 'entity37', 'entity38', 'entity39', 'entity40', 'entity41', 'entity42', 'entity43', 'entity44', 'entity45', 'entity46', 'entity47', 'entity48', 'entity49', 'entity50', 'entity51', 'entity52', 'entity53', 'entity54', 'entity55', 'entity56', 'entity57', 'entity58', 'entity59', 'entity60', 'entity61', 'entity62', 'entity63', 'entity64', 'entity65', 'entity66', 'entity67', 'entity68', 'entity69', 'entity70', 'entity71', 'entity72', 'entity73', 'entity74', 'entity75', 'entity76', 'entity77', 'entity78', 'entity79', 'entity80', 'entity81', 'entity82', 'entity83', 'entity84', 'entity85', 'entity86', 'entity87', 'entity88', 'entity89', 'entity90', 'entity91', 'entity92', 'entity93', 'entity94', 'entity95', 'entity96', 'entity97', 'entity98', 'entity99', 'entity100'. The graph is set against a light grey background with a subtle blue and white gradient at the top.

- Visualizing (very) large datasets
 - E.g. a *Lemon* dataset





Thanks for your attention

Mariano.Rico@upm.es

