The Semantic Web

Serge Abiteboul
INRIA Saclay & ENS Cachan

Introduction

The goals

Allows querying data from different organizations/systems using a unique entry point

Allow machines to exchange data and process data from other machines

For this, allow machines to reason on the data

Difficulty

- Interaction between structured and unstructured data
- Different formats for structured data
- Different schema
- Imprecision, incompleteness, possibly inconsistencies

The semantic Web

The **Semantic Web** is an evolving extension of the World Wide Web, in which data is made available in one standardized semantic format

Standards of the W3C:

- Naming entities: URI
- Facts/relations: RDF
- Constraints on them: RDF/S or OWL
- Link data
- Queries: SPARQL

The Semantic Web

A Web in which the resources are **semantically** described

- Annotations give information about a page, explain an expression in a page, etc.

More precisely, a resource is anything on the Internet that can be referred to by a **Uniform Resource Identifier** (URI), i.e., a string of characters

- A web page, identified by a URL
- A fragment of an XML document, identified by an element node of the document,
- A web service,
- A thing, an object, a concept, a property, etc.

Semantic annotations: logical assertions that relate resources to some terms in pre-defined **ontologies**

Ontologies

Ontologies

Formal descriptions providing **human** users a shared understanding of a given domain

A controlled vocabulary

Formally defined so that it can also be processed by **machines**

Logical semantics that enables reasoning.

Reasoning is the key for different important tasks of Web data management, in particular

- to answer queries (over possibly distributed data)
- to relate objects in different data sources enabling their integration
- to detect inconsistencies or redundancies
- to refine queries with too many answers, or to relax queries with no answer

Classes and class hierarchy

Backbone of the ontology

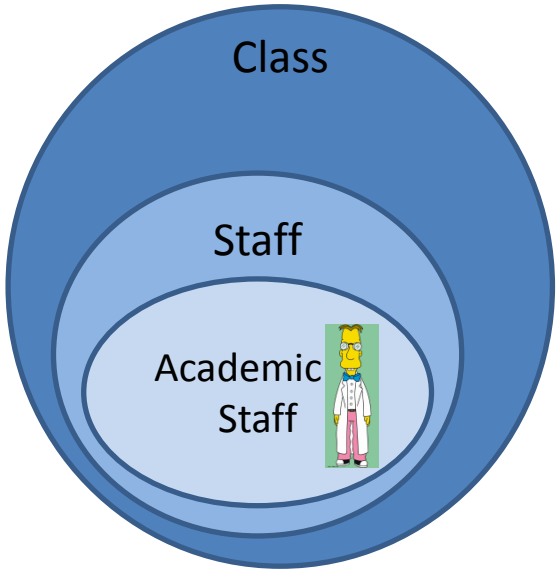
A class will be interpreted as a set of objects

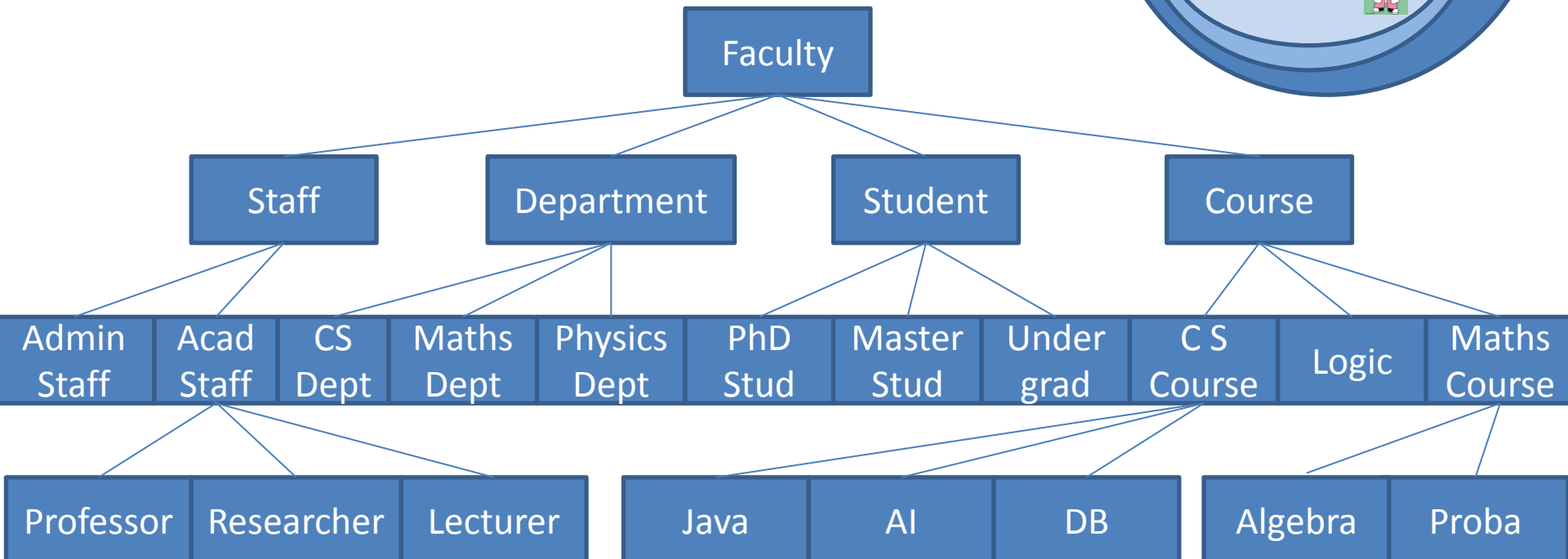
- Staff **isa Class**

A relation isa is interpreted as set inclusion

- AcademicStaff **isa** Staff

Taxonomy: hierarchy of classes





Relations with their domains

Declaration of **relations** with their signature

Relations are interpreted as binary relations between objects

- TeachesIn(AcademicStaff, Course)
- if one states that ``X TeachesIn Y'',

  then X belongs to AcademicStaff and Y to Course,

More examples

- TeachesTo(AcademicStaff, Student),
- Leads(Staff, Department)

Instances

Classes have instances

- Dupond is an instance of the class Professor
- It corresponds to the fact: Professor(Dupond)

Relations also have instances

- (Dupond,CS101) is an instance of the relation TeachesIn
- It corresponds to the fact: TeachesIn(Dupond,CS101)

The instance statements can be seen as (and stored in) a database

Ontology = schema + instance

Ontology = schema + instance (aka Knowledge base)

Schema

- The set of class and relation names
- The signatures of relations
- Other constraints that are used for
  - checking data consistency (like dependencies in databases)
  - inferring new facts

Instance

- The set of facts
- The set of base facts together with the inferred facts should satisfy the constraints

3 ontology languages for the Web

RDF: a very simple ontology language

RDFS: Schema for RDF

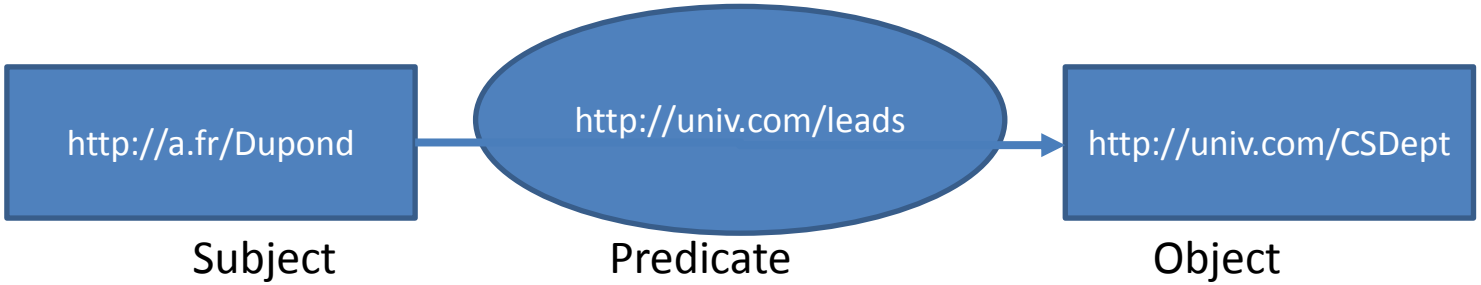
- Can be used to define richer ontologies

OWL: a much richer ontology language

We next present them rapidly

We will also mention a family of ontology languages: Description logics

RDF triples



Dupond leads the CS department

More triples

- Simplified, ignoring prefix

< :Dupond :TeachesIn :UE111 >

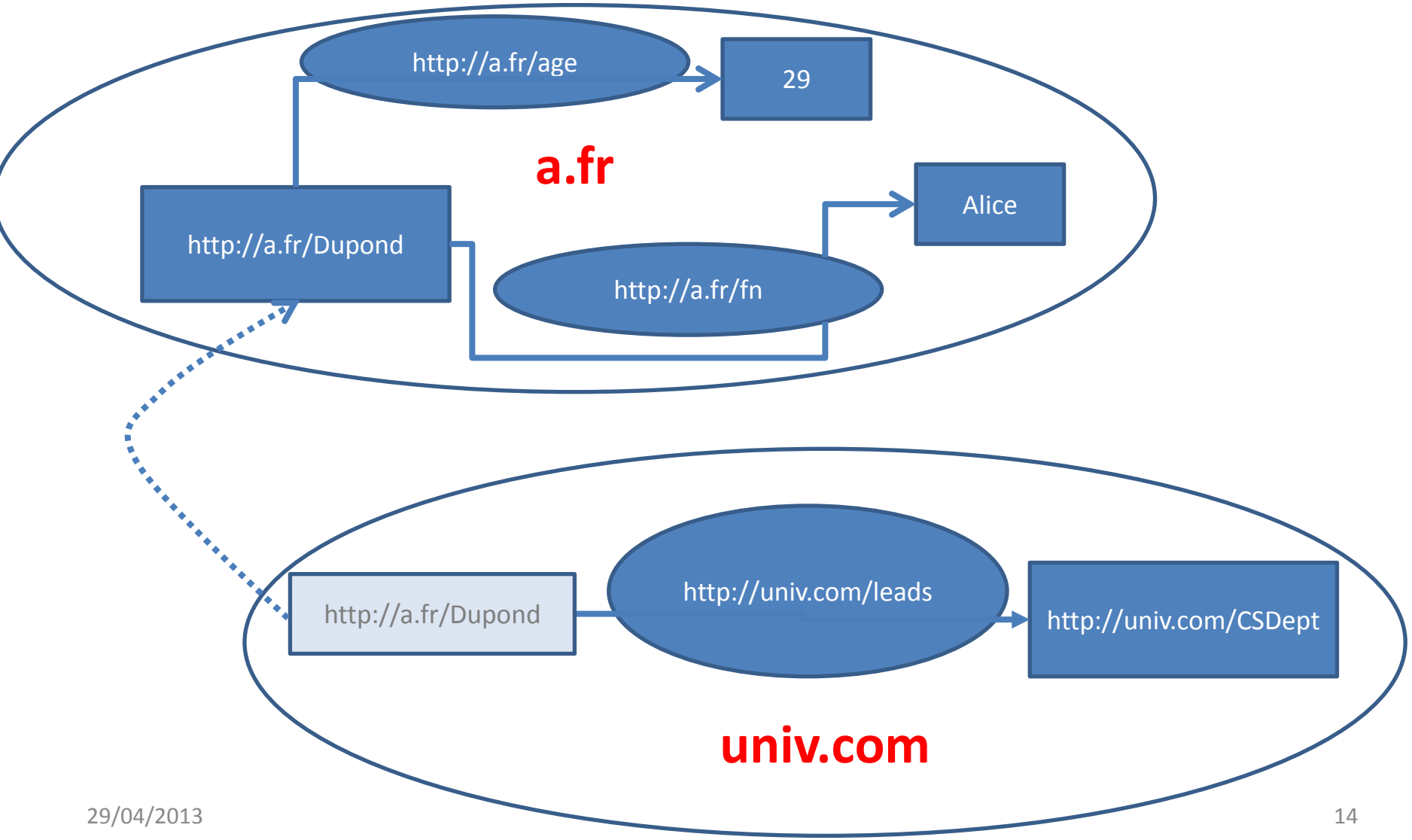
< :Dupond :TeachesTo :Pierre >

< :Pierre :EnrolledIn :CSDept >

< :Pierre :RegisteredTo :UE111 >

< :UE111 :OfferedBy :CSDept >

They live in a global graph



Some standard vocabularies

rdf: The basic RDF vocabulary

rdfs: RDF Schema vocabulary

dc: Dublin Core (predicates for describing documents)

s: Schema.org (predicates for Web content)

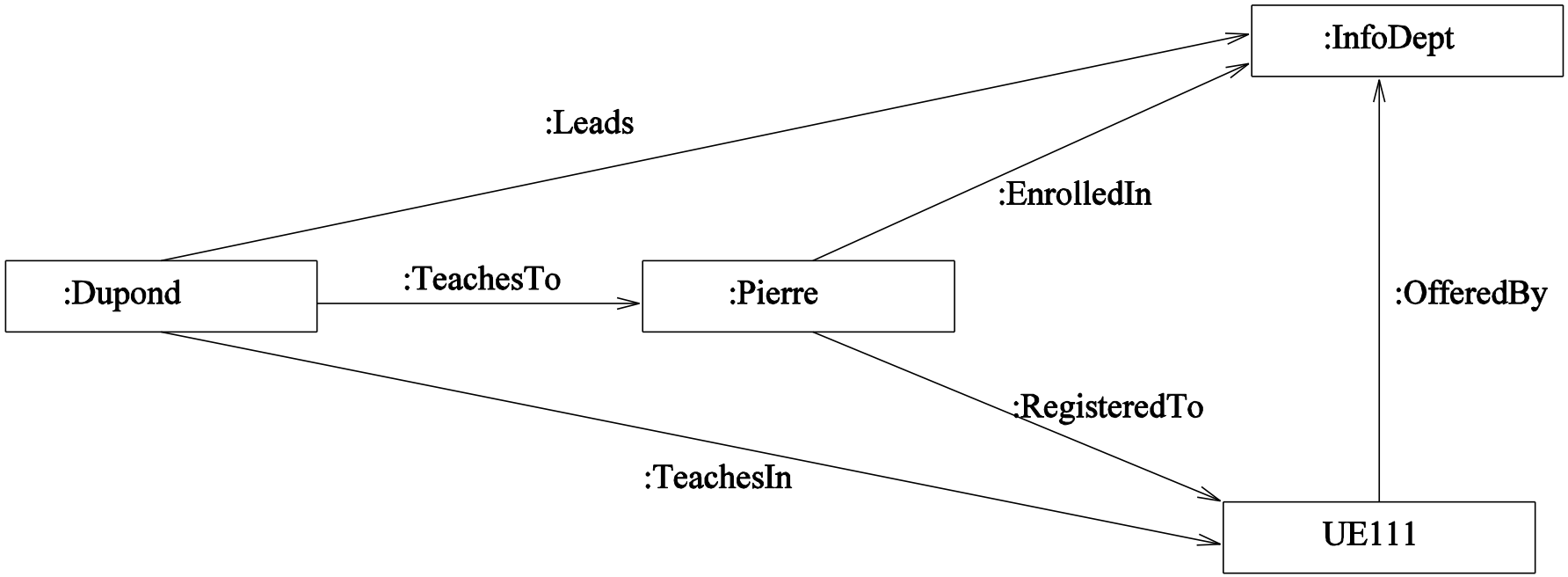
- Schema.org defines vocabulary for people, movies, events, restaurants, etc

cc: Creative Commons (types of licences)

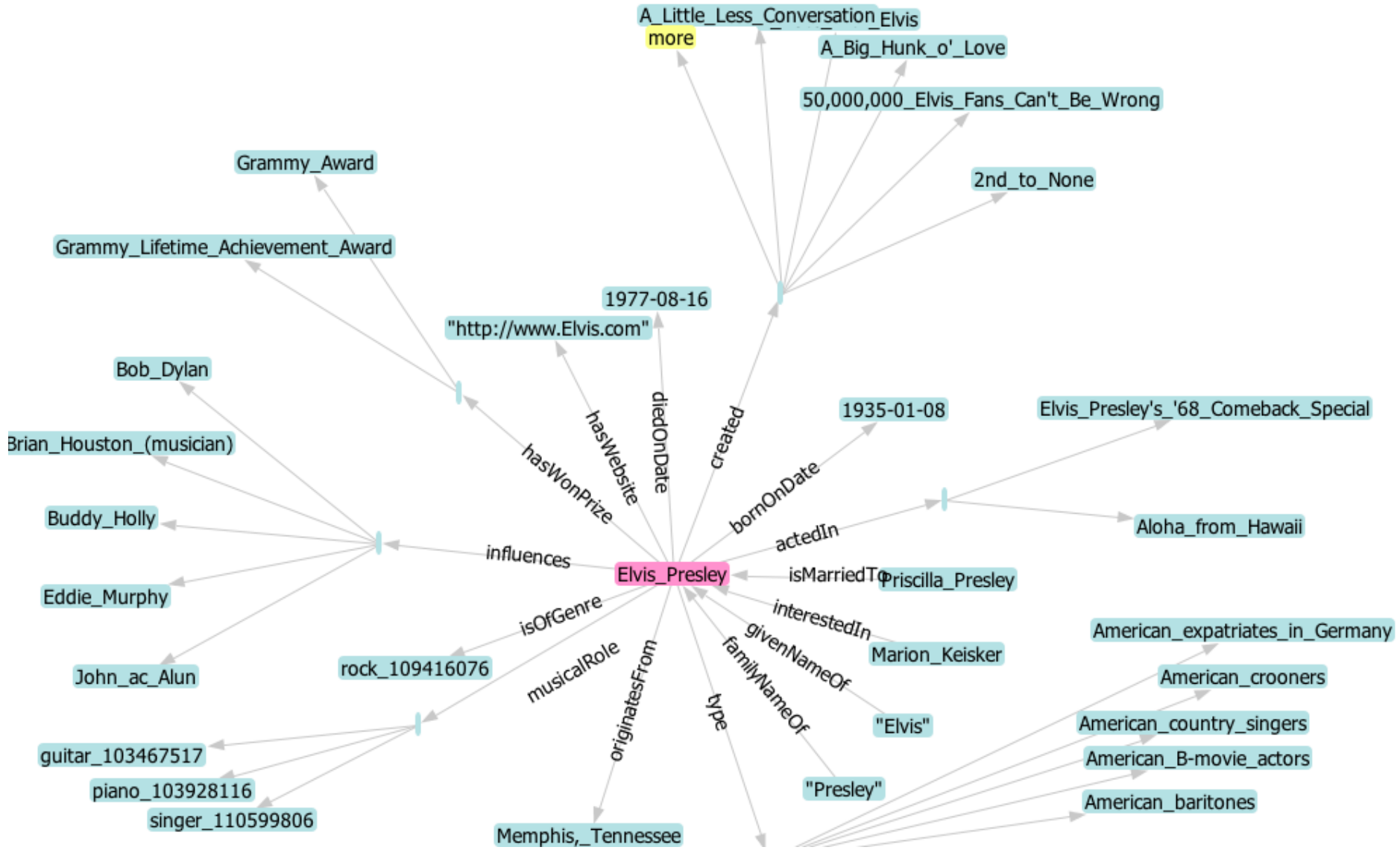
RDF graph

A set of RDF facts defines

- A set of relations between objects
- An **RDF graph** where the nodes are objects:



Sample RDF Graph: Elvis in Yago



RDF semantics

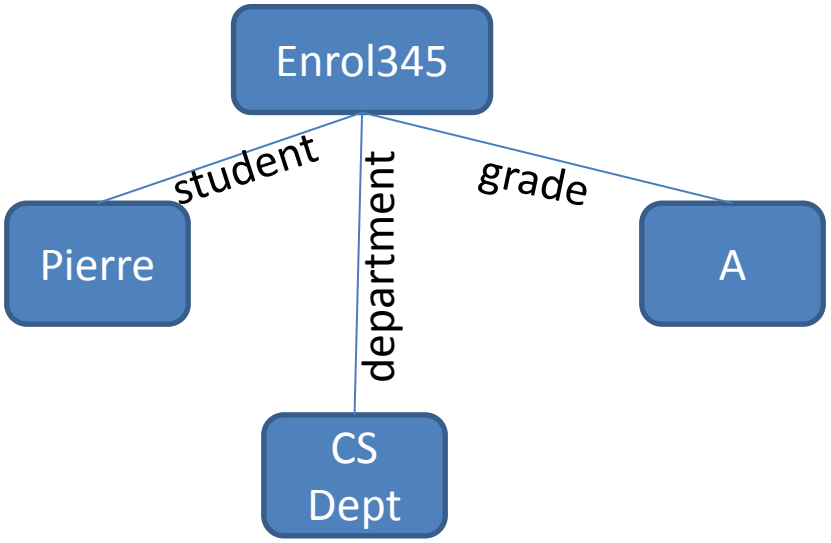
A triple < s P o > is interpreted in first-order logic (FOL) as a fact P(s,o)

Example:

- Leads(Dupond, CSDept)
- TeachesIn(Dupond, UE111)
- TeachesTo(Dupond,Pierre)
- EnrolledIn(Pierre, CSDept)
- RegisteredTo(Pierre, UE111)
- OfferedBy(UE111, CSDept)

More complex facts

Student	Department	Grade
Pierre	CS Dept	A



Enrol345	student	Pierre
Enrol345	Department	CS Dept
Enrol345	grade	A

Rather inelegant?

RDFS: RDF Schema

The schema in RDF is super simplistic

An RDF Schema defines the schema of a richer ontology

Do net get confused: RDFS can use RDF as syntax, i.e., RDFS statements can be themselves expressed as RDF triples using some specific **properties** and **objects** used as RDFS keywords with a particular meaning.

Declaration of classes and subclass relationships

- < Staff **rdf:type rdfs:Class** >
- < Java **rdfs:subClassOf** CSCourse >

Declaration of instances (beyond the pure schema)

- < Dupond **rdf:type** AcademicStaff >

RDF Schema – continued

Declaration of relations

- properties in RDFS terminology
- < RegisteredTo **rdf:type** rdf:Property >

Declaration of subproperty relationships

- < LateRegisteredTo **rdfs:subPropertyOf** RegisteredTo >

Declaration of domain/range restrictions for predicates

- < TeachesIn **rdfs:domain** AcademicStaff >
- < TeachesIn **rdfs:range** Course >
- TeachesIn(AcademicStaff , Course)

RDFS semantics in logic

RDF and RDFS statements FOL translation

< i rdf:type C >	$C(i)$
< i P j >	$P(i , j)$
< C rdfs:subClassOf D >	$\forall X\,(C(X) \Rightarrow D(X))$
< P rdfs:subPropertyOf R >	$\forall X\ \forall Y\,(P(X,Y) \Rightarrow R(X,Y))$
< P rdfs:domain C >	$\forall X\ \forall Y\,(P(X,Y) \Rightarrow C(X))$
< P rdfs:range D >	$\forall X\ \forall Y\,(P(X,Y) \Rightarrow D(Y))$

Owl

OWL extends RDFS with the possibility to express additional constraints

- Disjointness between classes
- Constraints of functionality and symmetry on predicates
- Intentional class definitions
- Class union and intersection

Examples

- departments can be lead only by professors
- only professors or lecturers may teach to undergraduate students.

Description Logics

Philosophy: isolate **decidable** fragments of first-order logic allowing reasoning on complex logical axioms over unary and binary predicates

These fragments are called Description Logics

The DL jargon:

- the classes are called **concepts**
- the properties are called **roles**
- the schema is called the **Tbox**
- the instance is called the **Abox**
- the ontology = Tbox + Abox

Semantics of main concepts

$I(C1 \sqcap C2) = I(C1) \cap I(C2)$

$I(\forall R.C) = \{o1 \mid \forall o2\,[(o1,o2) \in I(R) \Rightarrow o2 \in I(C)]\}$

$I(\exists R.C) = \{o1 \mid \exists o2.[(o1,o2) \in I(R) \wedge o2 \in I(C)]\}$

$I(\neg C) = dom(I) - I(C)$

$I(R^{-}) = \{(o2,o1) \mid (o1,o2) \in I(R)\}$

What kind of reasoning

Satisfiability checking: Given a knowledge base K = <T ,A>, is K satisfiable? (i.e., is the knowledge base consistent)

Subsumption checking: Given a Tbox T and two concept expressions *C* and *D, does* $T \models C \sqsubseteq D$*?*

- E.g. is C a subclass of C' or is w an instance of C

Instance checking: Given a knowledge base K = <T ,A>, an individual *e* and a concept expression *C, does* $K \models C(e)$*?*

Query answering: Given a knowledge base K = <T ,A>, and a concept expression *C,* finds the set of individuals *e* such that $K \models C(e)$*?*

- These problems are undecidable for full OWL and typically for DL with negation

Querying ontologies

Querying using RDFS

RDFS statements can be used to infer new triples

Example

- Base fact *ResponsibleOf (durand,ue111)*
- Use the statement <*ResponsibleOf rdfs:domain Professor*>
  i.e., the logical rule: *ResponsibleOf (X,Y)⇒Professor (X)*
- With substitution {X/durand, Y/ue111}
- Infer fact *Professor (durand)*
- Use the statement <*Professor rdfs:subClassOf AcademicStaff* >
  i.e., the rule *Professor (X)⇒AcademicStaff (X)*
- With substitution {X/durand}
- Infer fact *AcademicStaff (durand)*
- etc.

If we ask “who is in the Academic Staff?”, we want Durand in the answer

The saturation algorithm

RDFS is simple and very used but limited

Keep inferring new facts until a fixpoint is reached

Note: Only polynomially many facts can be added

PTIME

More complex: DL-Lite

Develop as a good compromise between expressive power and reasonable complexity of query answering

More complex DL: query answering is unfeasible

- Add some key constraints: still feasible
- Add some IncDs: still feasible
- Add both: unfeasible

Query answering: example

Abox:

Professor(Jim), HasTutor(John,Mary), TeachesTo(John,Bill)

Tbox:

$Professor \sqsubseteq \exists TeachesTo$

$Student \sqsubseteq \exists HasTutor$

$\exists TeachesTo^{-} \sqsubseteq Student$

$\exists HasTutor^{-} \sqsubseteq Professor$

$Professor \sqsubseteq \neg Student$

Queries: conjunctive queries on concepts and atomic roles

$q0(x) \leftarrow TeachesTo(x,y) \wedge HasTutor(y,z)$

Answering queries by reformulation

$q0(x) \leftarrow TeachesTo(x,y) \wedge HasTutor(y,z)$

$Student \sqsubseteq \exists HasTutor$

$HasTutor(y,z) \leftarrow Student(y)$

$q1(x) \leftarrow TeachesTo(x,y) \wedge Student(y)$

Query *q1* computes answers to *q0*

Answering queries by reformulation

Reformulations of *q0* given the Tbox *T*

*A*ll answers are obtained with:

q1(x)←TeachesTo(x,y) ∧ Student(y)

q2(x)←TeachesTo(x,y) ∧ TeachesTo(z',y)

q3(x)←TeachesTo(x,y')

q4(x)←Professor(x)

q5(x)←HasTutor(u,x)

One can use a standard query processor to evaluate the query

For complex ontology languages, not always possible

Consistency checking

Tbox: T ’

Professor ⊑ ∃TeachesTo

Student ⊑ ∃HasTutor

$\exists TeachesTo^{-} \sqsubseteq Student$

$\exists HasTutor^{-} \sqsubseteq Professor$

Professor ⊑ ¬Student

∃TeachesTo ⊑ ¬Student

∃HasTutor ⊑ Student

This Tbox has no model if the following query has an answer

qunsat ←TeachesTo(x,y) ∧ HasTutor(x,y’)

Why?

- TeachesTo(a,b) ∧ HasTutor(a,b’)
- TeachesTo(a,b) implies ¬Student(a)
- HasTutor(a,b’) implies Student(a)
- A contradiction

Illustration of what is desirable: DL-Lite

Query answering and data consistency checking can be performed in two separate steps:

A reasoning step with the Tbox alone (i.e., the ontology without the data)

- Ptime in the size of the Tbox
- Produces a polynomial number of reformulations and of *unsat queries*

The evaluation of conjunctive queries over the data in the Abox (without the Tbox)

- makes it possible to use an SQL engine and thus advantage of query optimization in relational DBMS
- data complexity in *AC0 (so contained in logspace)*

SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) is the query language of the Semantic Web

```
SELECT ?dep
WHERE {
<http://a.fr/Dupond>   <http://univ.com/leads>   ?dep }
```

Find me all the values for ?dep such that the triple is true

Pattern matching over the RDF graph

Many gadgets

Many ontologies provide “SPARQL endpoints”, i.e. a service than can receive SPARQL queries sent by a machine or typed by a human

Integration of data sources

Goal

Obtain data from different data sources with a single query/interface

Example:

- Sciences: query different databases recording information about genes
- Business: query catalogs of different vendors
- Administration: integrate financial data from different branches
- Web: find data on a person from many Web sources

Complex task: to describe possibly complex connection between data sources, use **semantics**

Buzz word: semantic Web

The data sources:

- have been developed independently
- are autonomous
- very heterogeneous

Semantics is needed to relate their concepts and their structures

Logic is used to describe the semantics

Example

Where can I see a film of Woody Allen today in Paris?

- Woody Allen *plays_in a film X*
- X *is_shown_at_theater Y*
- Y *is_located_in Paris*

Ignore irrelevant sources: Air France, etc.

Find the relevant sources and understand how to use them

Combine their results

Ask queries to a global schema & answer using data of the local schemas

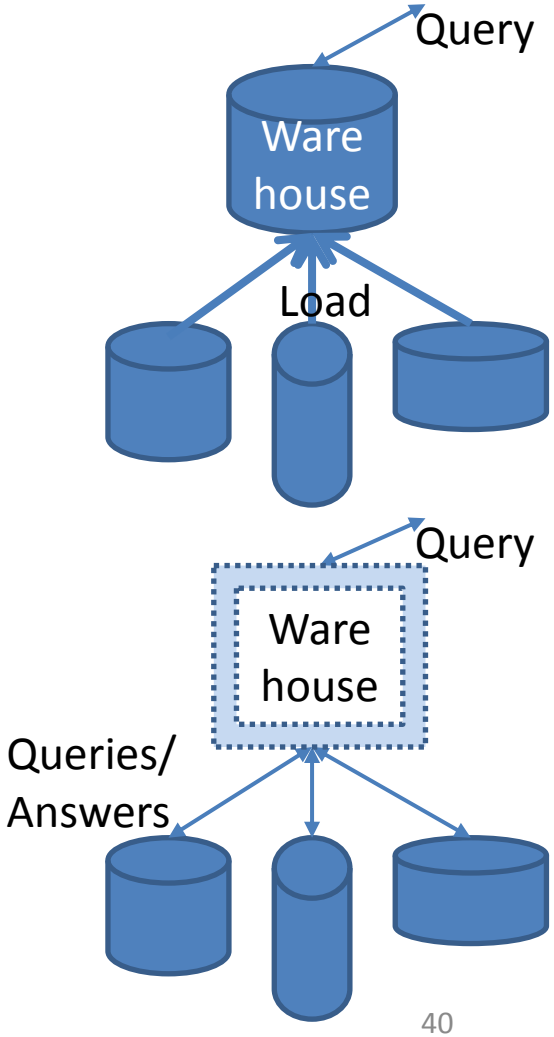
Two main approaches

Warehousing approach

- Global instance is materialized
- Creation: cost of computation and storage
- Query evaluation is very efficient
- Updates are costly: need to propagate local instance updates to the warehouse
- Otherwise data are possibly stale

Mediating approach

- Global instance is virtual
- Query: cost of reformulation
- Creation and updates: no cost



Database and FO

The mediator approach

Database schema
FO constraints

Database instances
Links: FO constraints

Expressed in FO
Evaluation uses a DB engine

Global schema: Define a mediated schem

- Structured vocabulary serving as a query interface for users queries
- Typically, one per domain

Local schemas: Declare a data source

- Model the content of the source to integra ated schema
- Relate the concepts/relations of the source to those of the mediated schema

Query processing

- Reformulate and decompose a user's quer a into queries over the local schema that are run at the data sources
- Combine the answers of local queries to construct the answer to the global query

The mediator approach – continued

Define a mediated schema (also called a global schema) that serves for the query interface for users

Declare the data sources: mappings between the global schema and the schemas of the local data sources

Two approaches:

- **Global-As-Views** (GAV) approach: the global relations are defined as views over the local relations
- **Local-As-Views** (LAV) approach: the local relations are defined as views over the global relations

Query processing

- Rewriting the users queries (expressed using global relations) in terms of local relations⇒ logical query plans
- Combine the answers of logical query plans to obtain the result

Global as view

Obtained by unfolding all the atoms in the query

Gives a conjunction of disjunctive queries

Rewrite into a disjunction of conjunctive queries

Remove useless ones (using the homomorphism theorem for conjunctive queries)

Limitations

- Adding or removing data sources requires to revise all the GAV mappings defining the global schema
- When a new data source arrives, we must consider how it may be combined with all the existing data sources to produce tuples of any global relation

Local as view

Several algorithms have been proposed

- Bucket
- Minicon: an optimization of Bucket
- Inverse-rules: in the spirit of algorithm for GAV

Inverse rules

Principle: The LAV mappings are transformed into GAV mappings (called *inverse rules*) independently of the query

To do that, we need to introduce existential variables

- For the existential variables, we use Skolem terms
- This keeps track of their provenance

At query time, the rewritings are obtained by unfolding like in GAV

The unfolding is a little trickier because of the Skolem terms

Introducing Skolems

V4(X):- cite(X,Y), cite(Y,X)

V5(X,Y):- sameTopic(X,Y)

V6(X,Y):- cite(X,Z) , cite(Z,Y) , sameTopic(X,Z)

Result of the inversion of the rule:

cite(X,f1(X)):- V4(X)

cite(f1(X),X)):- V4(X)

sameTopic(X,Y):- V5(X,Y)

cite(X,f2(X,Y)):- V6(X,Y)

cite(f2(X,Y),X)):- V6(X,Y)

sameTopic(X,f2(X,Y)):- V6(X,Y)

Query unfolding - illustration

q(U):- cite(U,V),cite(V,U),sameTopic(U,V)

s={ X/U, V/f2(U,Y) } and rule 4

q”$_1$(U):- V6(U,Y),cite(f2(U,Y),U), sameTopic(U,f2(U,Y))

s= { X/U, Y/Y } and rule 5

q”$_2$(U):- V6(U,U),V6(U,U),sameTopic(U,f2(U,U))

s= { X/U, Y/U } and rule 6

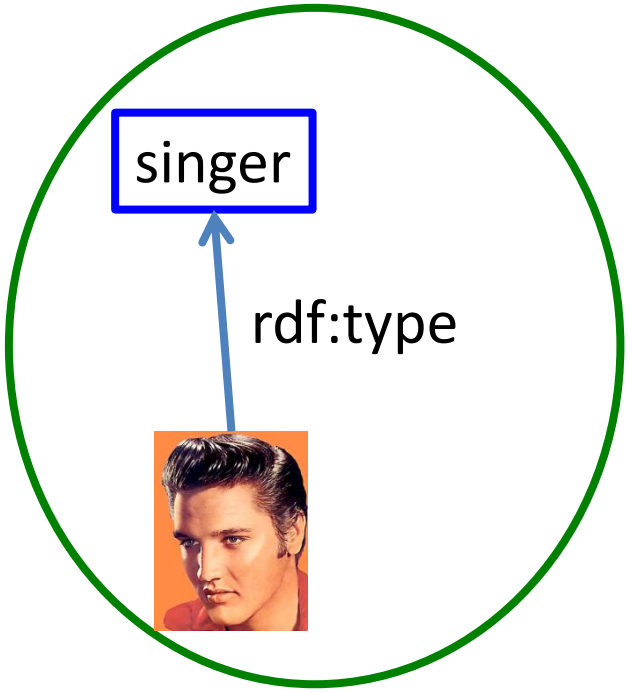
q”$_3$(U):- V6(U,U),V6(U,U), V6(U,U)

simplified to: q”$_4$(U):- V6(U,U) ⇒ a valid query plan

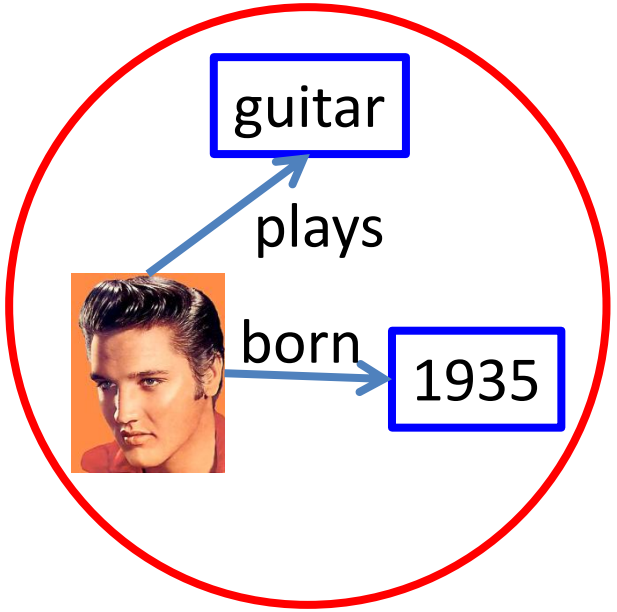
Data integration and linked data problem

Many ontologies talk about the same entity with different URIs

This is bad, because we cannot join the information



Elvisopedia
(http://elvisopedia.org/)

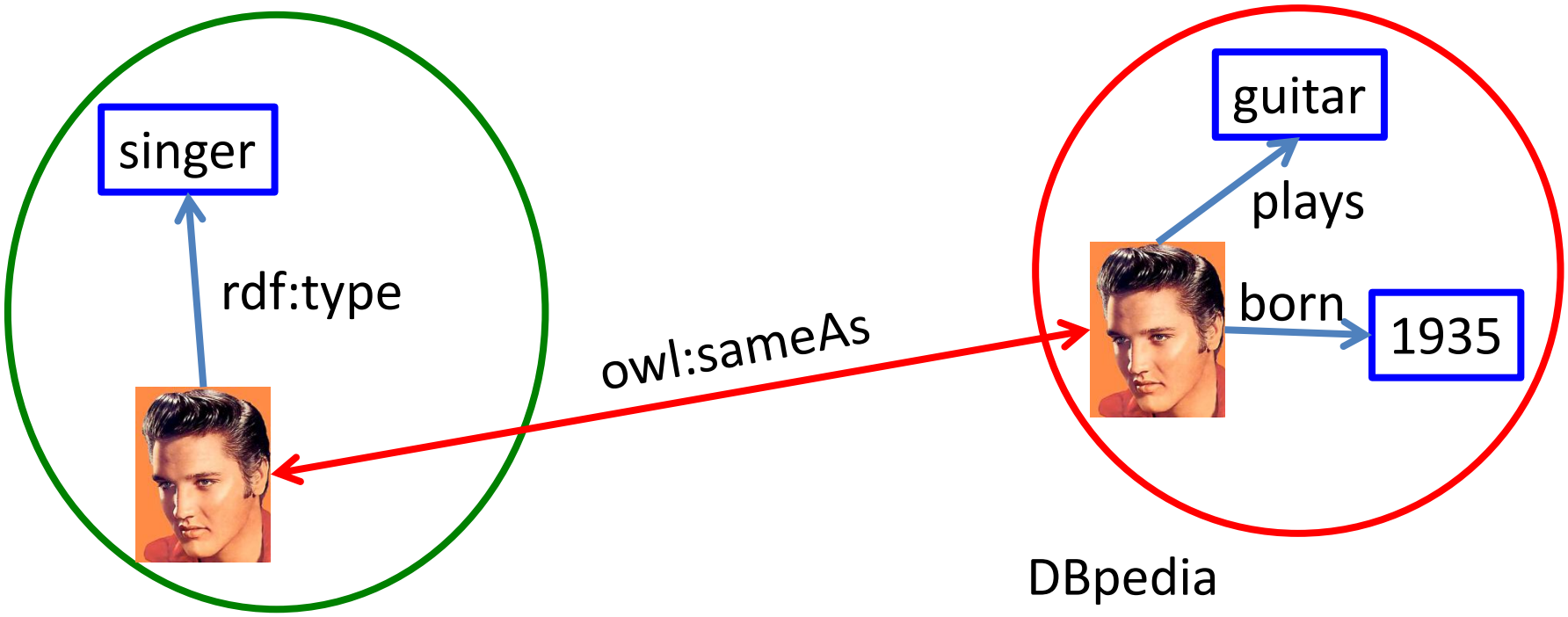


DBpedia
(http://dbpedia.org/)

Linked data solution

OWL provides vocabulary to link equivalent entities

http://elvisopedia.org/Elvis owl:sameAs http://dpbedia.org/Elvis



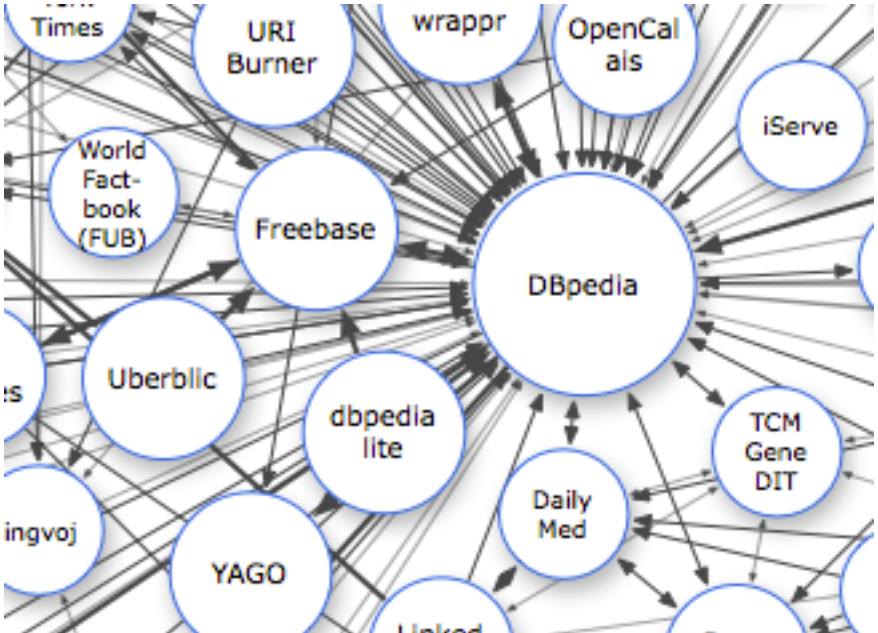
Elvisopedia
(http://elvisopedia.org/)

DBpedia
(http://dbpedia.org/)

The Linking Data Project

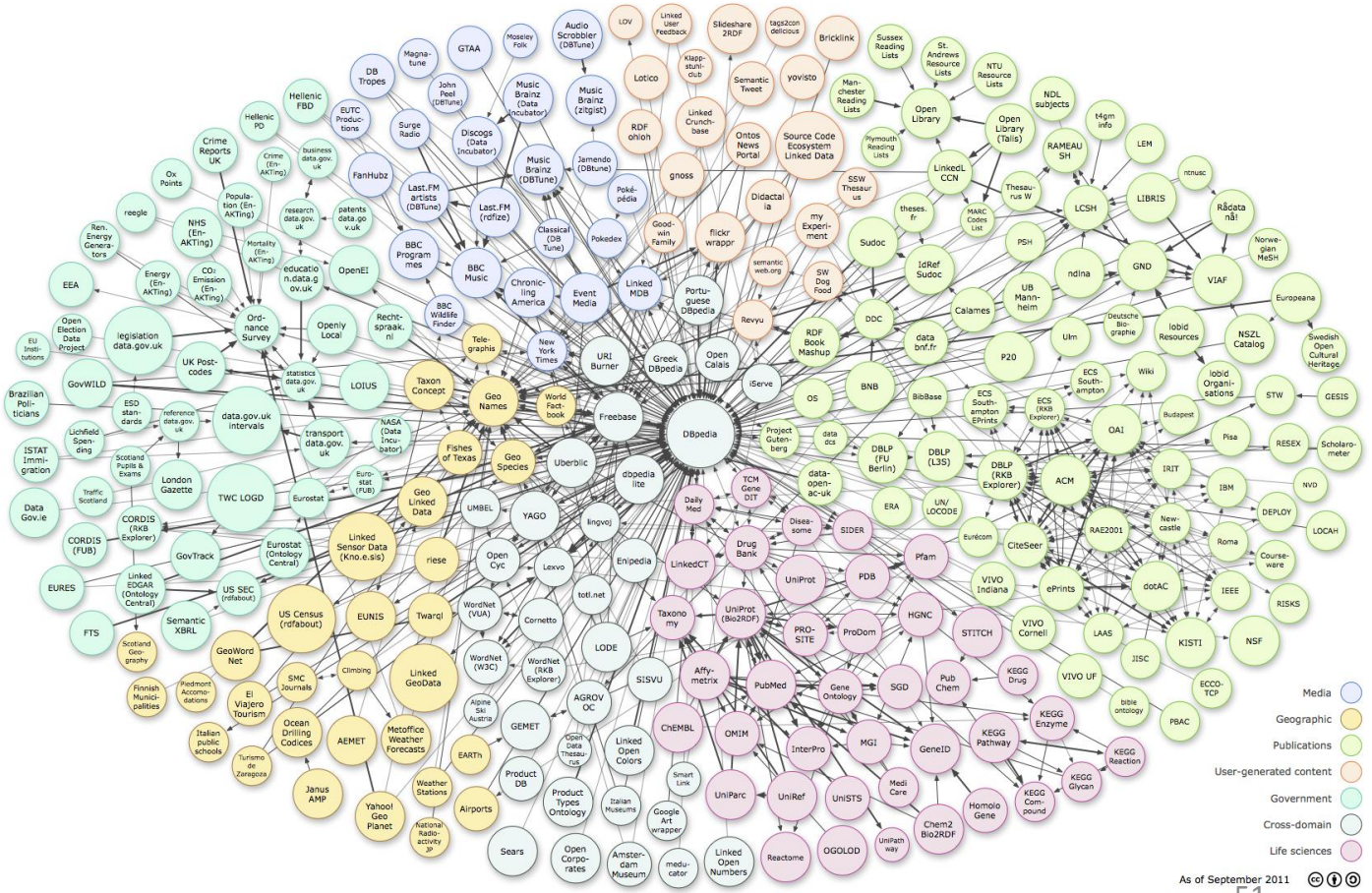
The **Linking Open Data Project** aims to interlink all open RDF data sources into one gigantic RDF graph

LINKINGOPENDATA
W3C SWEO Community Project



The Linked Data Cloud

As of 2011: 295 ontologies, 25 billion triples, 400m links



http://richard.cyganiak.de/2007/10/lod/imagemap.html

Data on the Web

Data on the Web

Potentially the largest possible source

- On Google: birthday of Alan Turing

  Best guess for Alan Turing Date of birth is June 23, 1912
- Wikipedia: very large source of information with some limited use of ontologies

Lots of tables in HTML or pdf

Lots of data in deep Web behind forms

More and more structured data published notably public

Many ontologies: e.g., DBPedia or Yago

Need: tools for search, visualization, linking, integration

Extraction from text

Difficult

Natural language processing is complex and error prone

Simpler if we already have an ontology with lots of instances

- Possible to semi-automatically wrap new data sources using overlaps with already known data
- Use specific techniques adapted to the particular domain
- Heavy use of statistics

Building ontologies

Extract one from existing text sources

- Yago built from Wikipedia

Have humans collaborate to build it

- Freebase: Freebase is an open, Creative Commons licensed graph database with millions of entities
- Linked data: publish RDF links between Web data

Integrate different ontologies by aligning their concepts and relations

- Paris [SuchanekAbiteboulSenellart]

Conclusion

Conclusion

The scalability of reasoning on Web data requires light-weight ontologies

- Reasoning should be feasible – polynomial
- Preferable if query answering can be performed with a relational database engines

RDFS is OK but too limited?

Full OWL is too complex

The Semantic Web

More and more semantic on the Web

- E.g., the UK government makes much of its data available online in RDF – by law

Enriching the standard Web

- Publishing semantic descriptions of Web services/pages
- **Microdata** an upcoming W3C standard to annotate HTML pages with RDF data

Web applications and search engines will more and more rely on such semantic annotations

- The DBpedia Mobile App retrieves data from the Linked Open Data Cloud to show places of interest around you

References

Web data management, Abiteboul, Manolescu, Rigaux, Rousset, Senellart

webdam.inria.fr/Jorge

Semantic Web, Fabian Suchanek,

suchanek.name/work/teaching/inf347/inf347_sw.ppt


Merci !
FRAPAR.