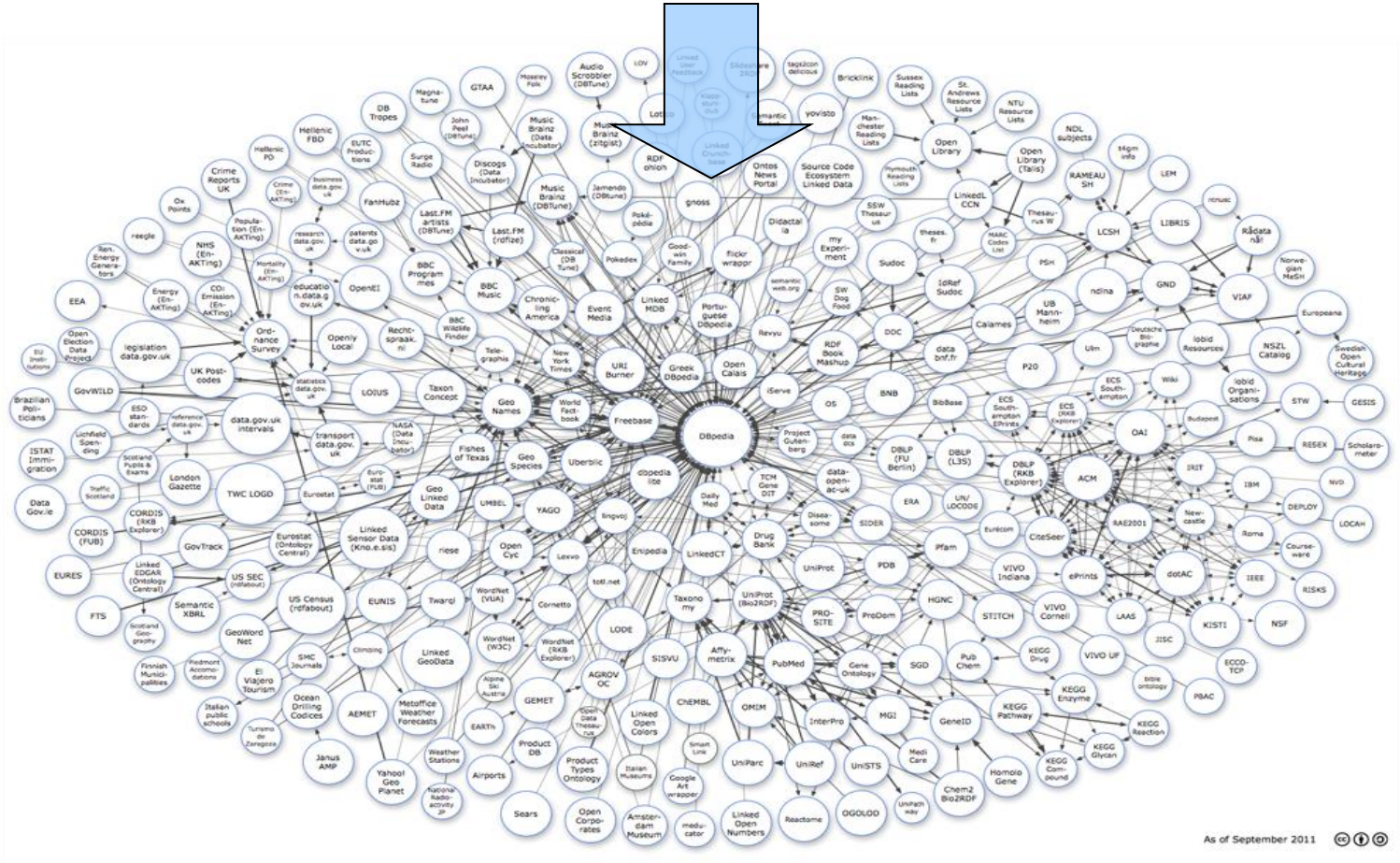
Systematic Generation of SPARQL Benchmark Queries for Linked Open Data

Olaf Görlitz, Matthias Thimm, Steffen Staab

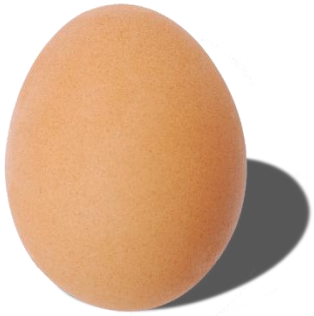
UNIVERSITÄT KOBLENZ · LANDAU

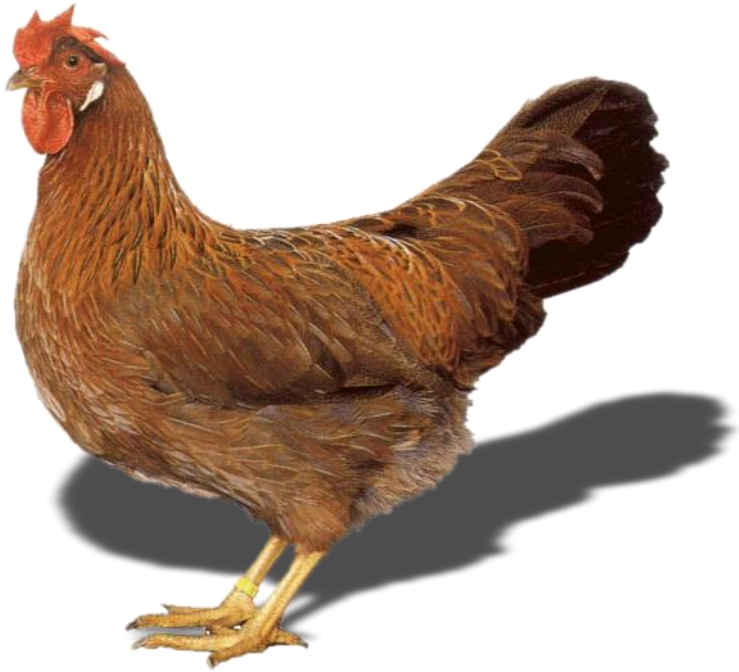
SPARQL Queries on the Linked Data Cloud



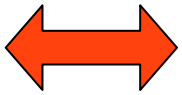
The Problem

Why not use benchmark queries?





distributed queries



federation implementation

RDF Benchmarks

LUBM, BSBM, SP²B, ...

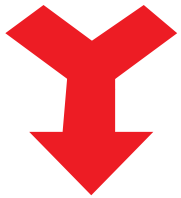
- Synthetic datasets
- Domain-specific
- Highly structured
- Sophisticated queries

Centralized

FedBench (ISWC'11)

- 10 Linked Data sets (~170M triples)
- 25 handpicked distributed queries

Fixed



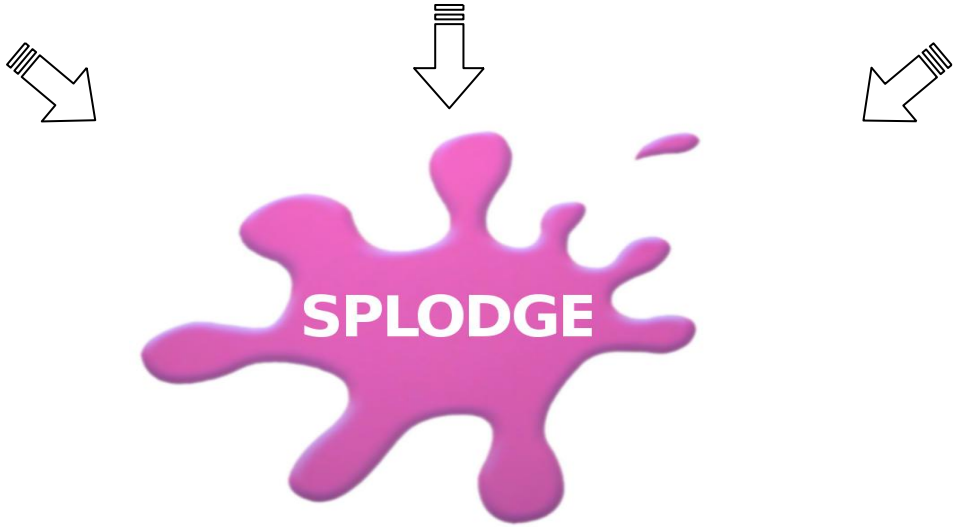
Scalable, Flexible, Expressive Linked Data Benchmark

Overview

- Benchmark Idea
- Methodology
- Evaluation

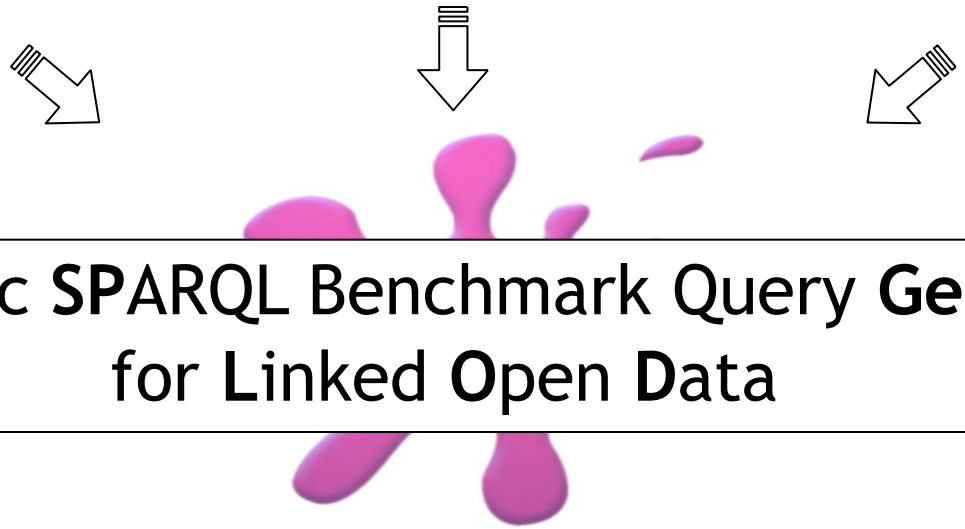
Linked Data Benchmark Features

Real Linked Data Sets Customization Typical+Complex Queries



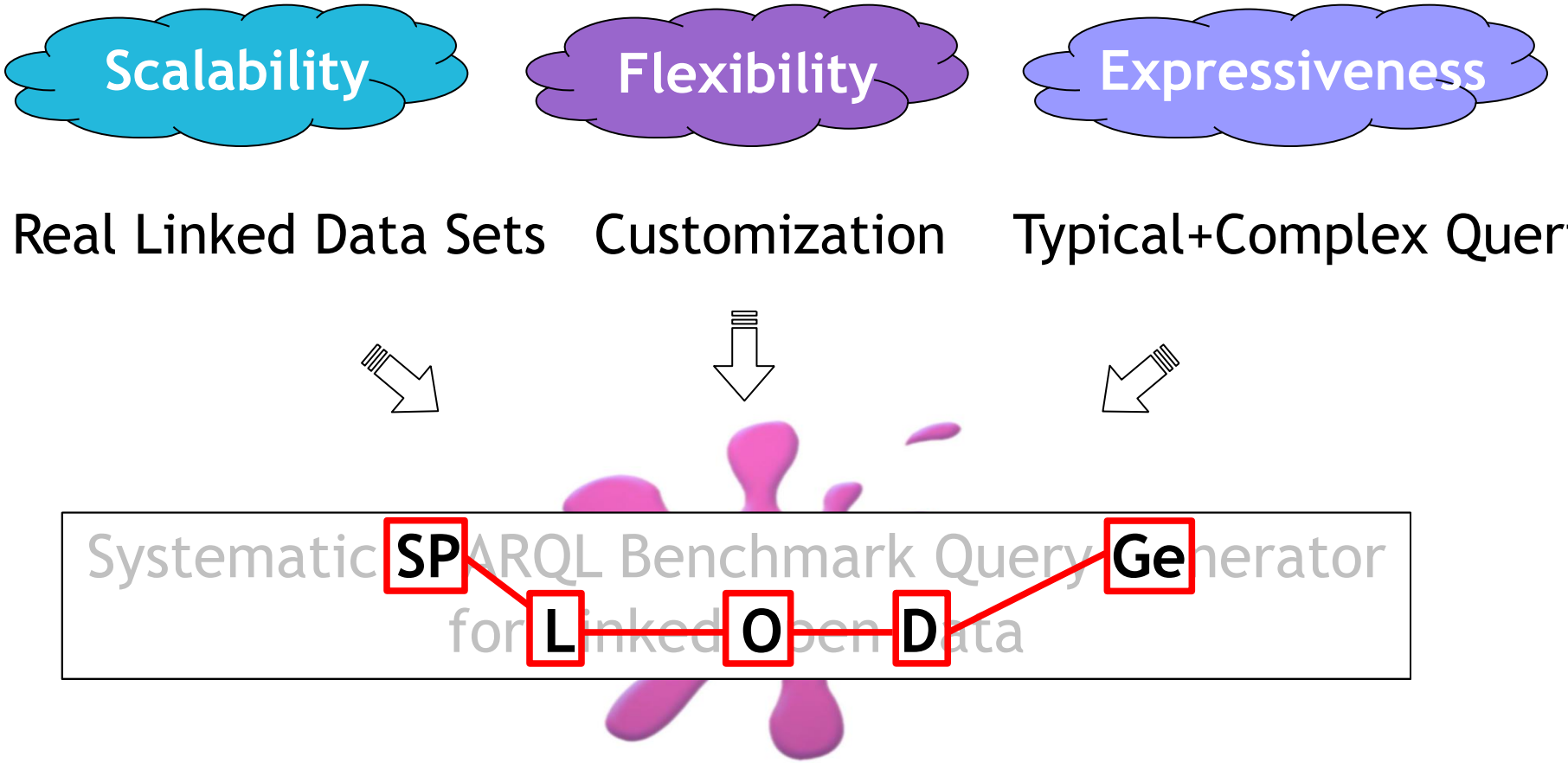
Linked Data Benchmark Features

Real Linked Data Sets Customization Typical+Complex Queries



Systematic **SP**ARQL Benchmark Query **Ge**nerator for **L**inked **O**pen **D**ata

Linked Data Benchmark Features

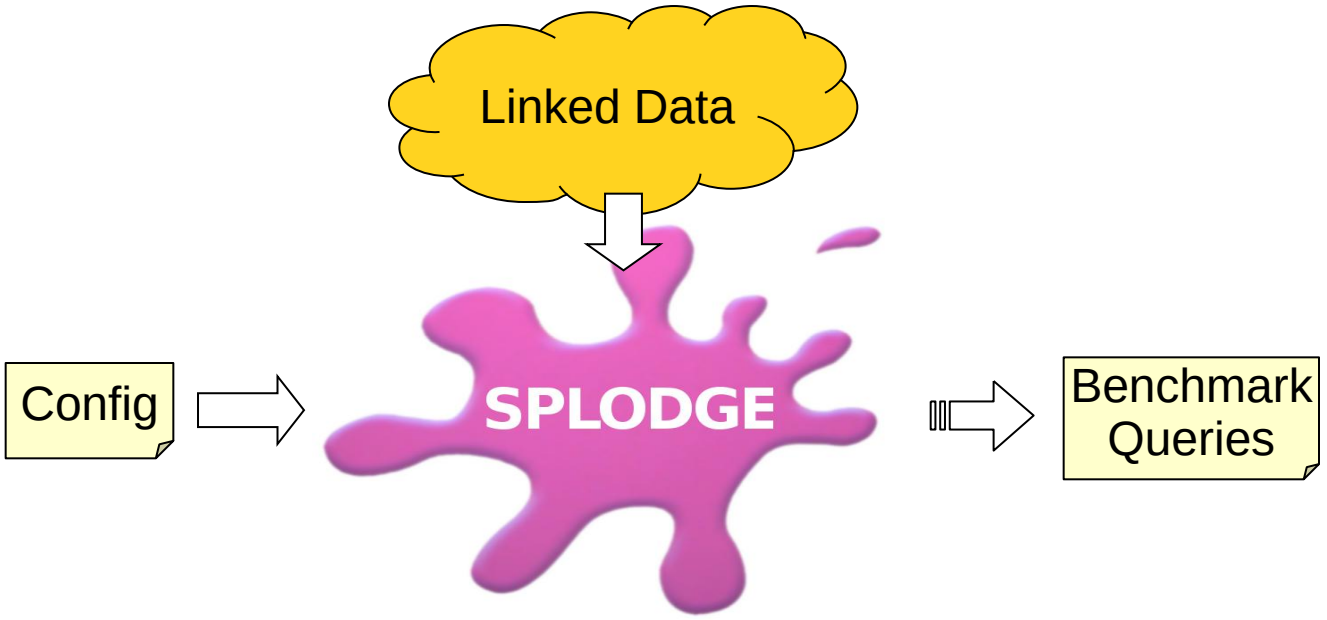


What we want:

1. Define Query Characteristics → Customize Benchmark
2. Automatic Query Generation → Random Queries
3. Query Validation → #results > 0

Contribution

Methodology and toolset for systematic query generation



Parameterization Query Generation Query Validation

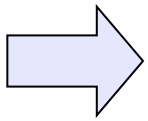
Overview

- Benchmark Idea
- Methodology
- Evaluation

SPLODGE Methodology

Define typical + challenging distributed queries

No federation query logs available



Analyze queries of benchmarks

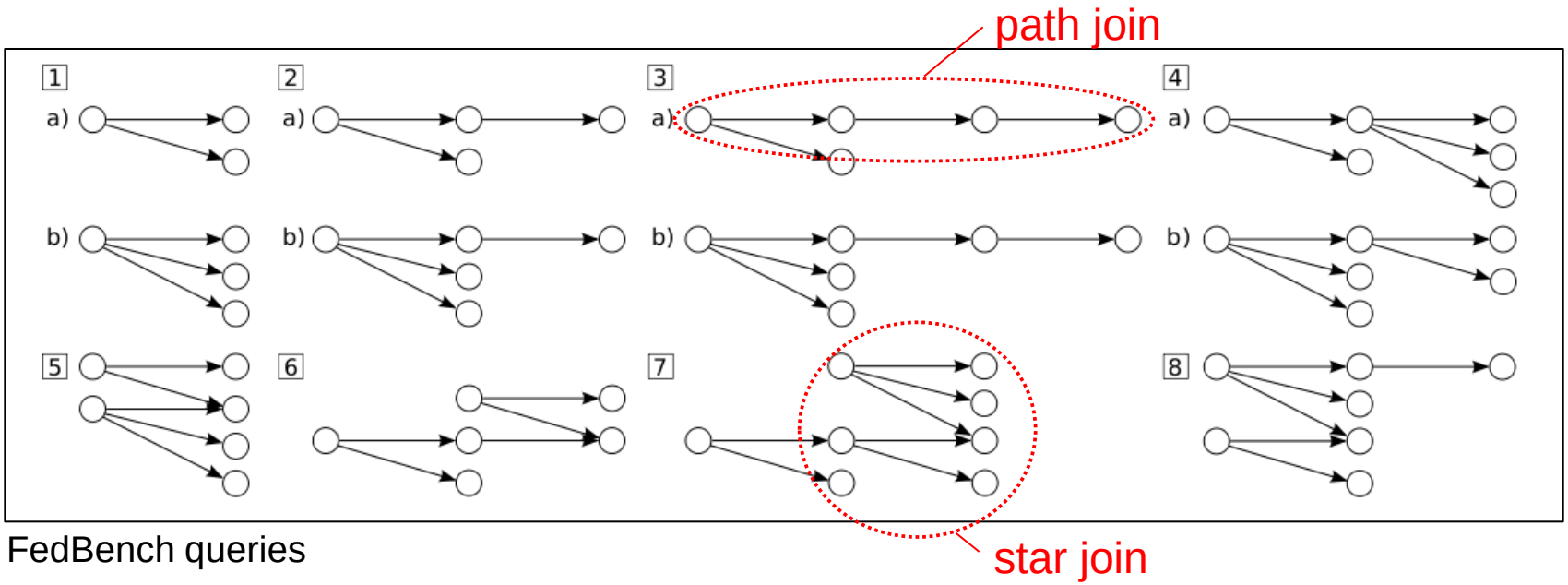
```
SELECT ?drug ?keggUrl ?chebiImage WHERE {
  ?drug rdf:type drugbank:drugs .
  ?drug drugbank:keggCompoundId ?keggDrug .
  ?keggDrug bio2rdf:url ?keggUrl .
  ?drug drugbank:genericName ?drugBankName .
  ?chebiDrug purl:title ?drugBankName .
  ?chebiDrug chebi:image ?chebiImage . }
```

FedBench/LifeScience#5

SPLODGE Methodology

Algebra	Structure	Cardinality
• Query Form *(Select, Construct, ...)*	• Variable Patterns *(s, o, s+o, ...)*	• # Data Sources
• Join Type *(conj. / disj. / left-join)*	• Join Patterns *(star, path)*	• # Joins/ Patterns
• Result Modifiers *(limit, offs, order by)*	• Cross Product	• # Results

Main query parameter: join structure



FedBench queries

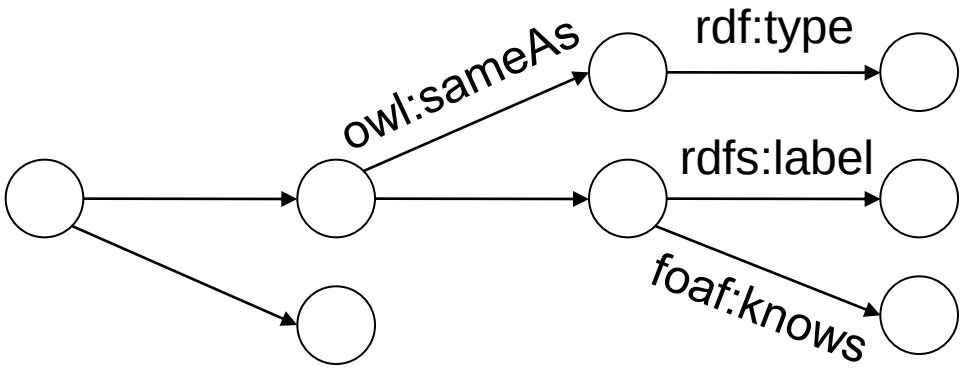
SPLODGE Methodology

Additional query parameters: # triple patterns
data sources
result size
...

Path-join: m sources — n triple patterns, (m≤n)

Star-join: anchor node — n triple pattern, (s/o)

SPLODGE Methodology



Iteratively add random triple pattern ⇐ #results > 0 ?

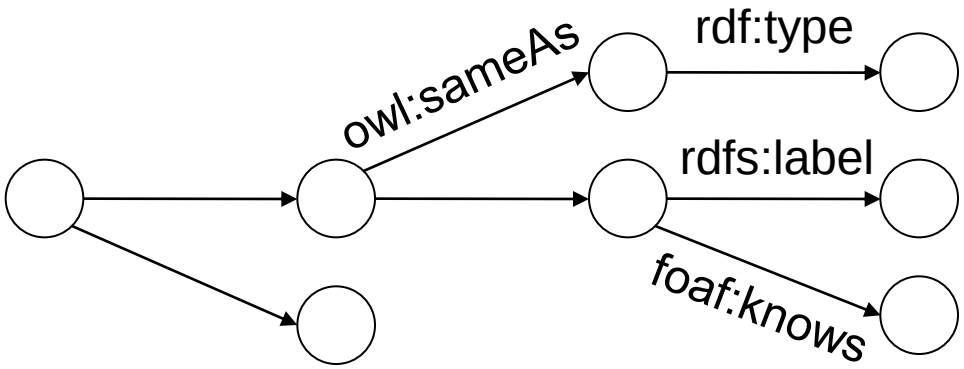
⇓

Need background knowledge ⇐ level of detail?

⇓

Predicate combinations ⇐ how provided?

SPLODGE Methodology



Linked Predicates

(owl:sameAs → rdf:type)

DBpedia → geonames (43, 58)
freebase → DBpedia (86, 72)
...

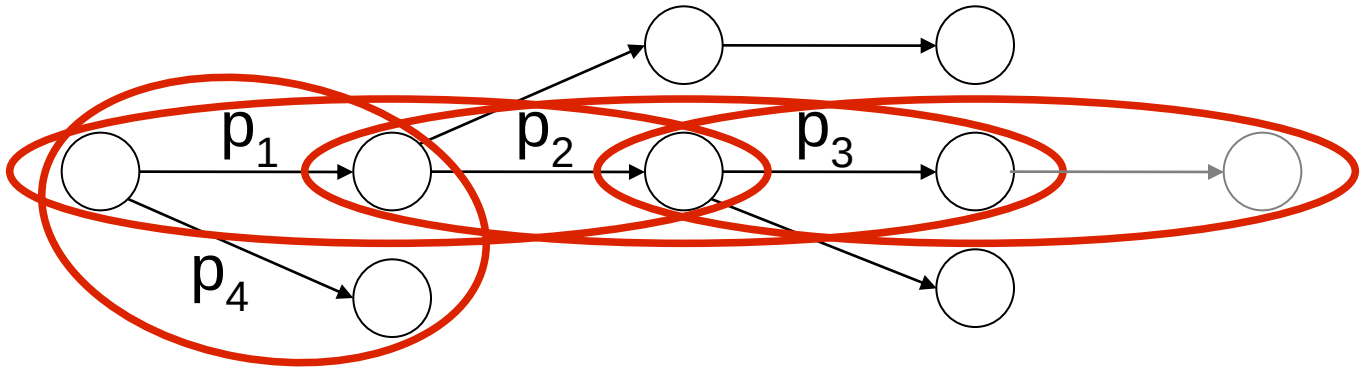
Characteristics Sets*

{rdfs:label, foaf:knows, …}

DBpedia (322), rdfs:label (437)
foaf:knows (322)
...

*[Neumann, Moerkotte, ICDE 2011]

SPLODGE Methodology



Linked Predicates	Characteristics Sets
$(p_1 \rightarrow p_2) \otimes (p_2 \rightarrow p_3)$	$\{p_1, p_4\}$
$\otimes (p_3 \rightarrow p_i)$	$\{p_1, p_4, ...\}$

SPLODGE Methodology

Verify generated queries (#results >0)

How to evaluate? ⇒ Compute confidence value

minimum join selectivity > e

Overview

- Benchmark Idea
- Methodology
- Evaluation

Evaluation Objective

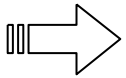
- Verify generation of valid queries (#results >0)
- Compare variations of query generation algorithms

Baseline	SPLODGE*lite*	SPLODGE
"random" predicate	background knowlege	+ minimum join selectivity ($> 10^{-4}/10^{-3}/10^{-2}$)

- Metrics:
  - #queries with non-empty results
  - #result per query

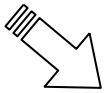
Evaluation Setup

- Real Linked Data
- Random queries
- Triple Store



Billion Triple Challenge Dataset

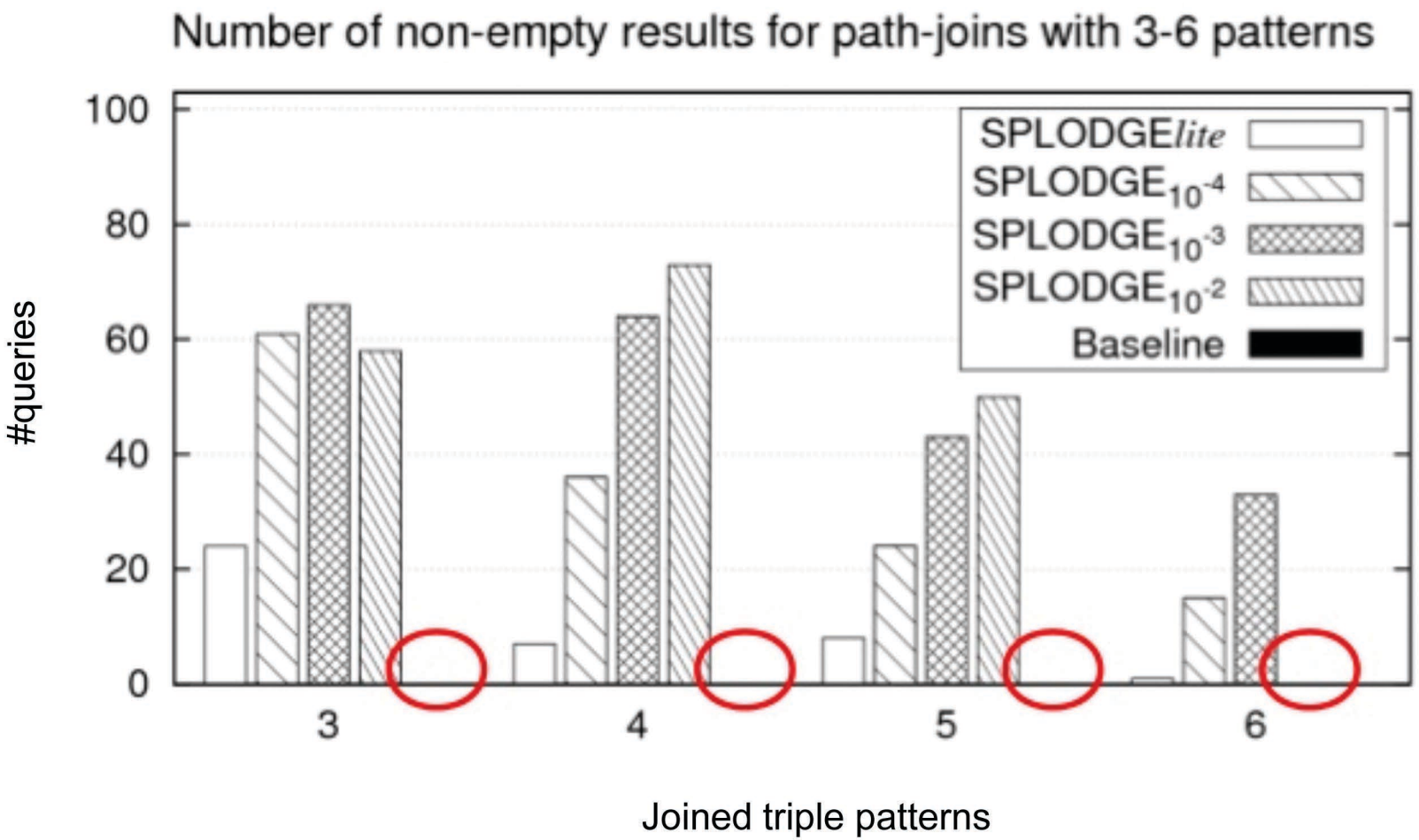
- Path-joins across data sources
- 3-6 patterns, bound predicates
- 100 queries per batch



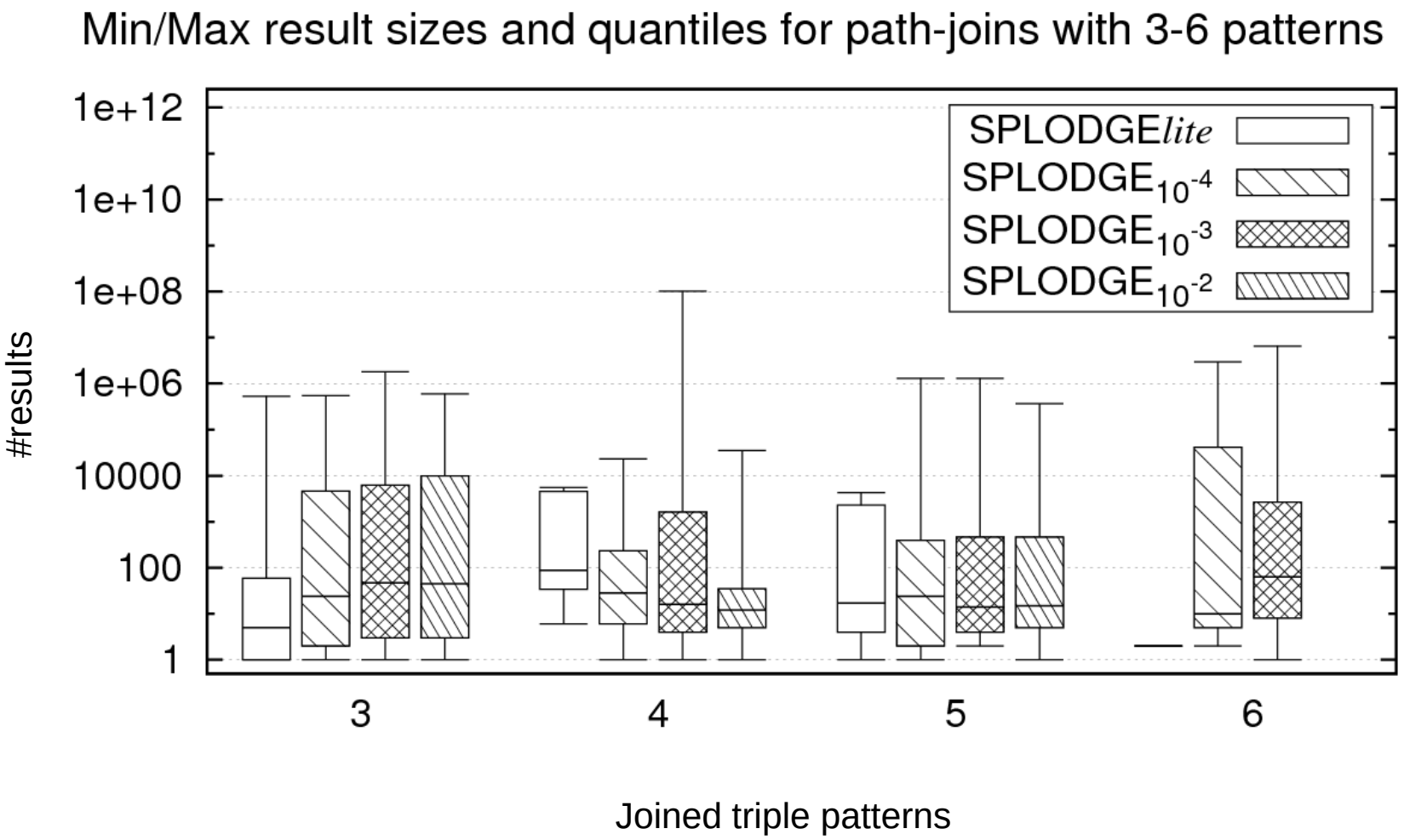
RDF3X

```
SELECT * WHERE {
    ?var1 <http://dbpedia.org/property/description> ?var2 .
    ?var2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> ?var3 .
    ?var3 <http://www.w3.org/2002/07/owl#disjointWith> ?var4 .
    ?var4 <http://www.w3.org/2002/07/owl#disjointWith> ?var5 .
    ?var5 <http://semantic-mediawiki.org/swivt/1.0#wikiPageModificationDate> ?var6
}
```

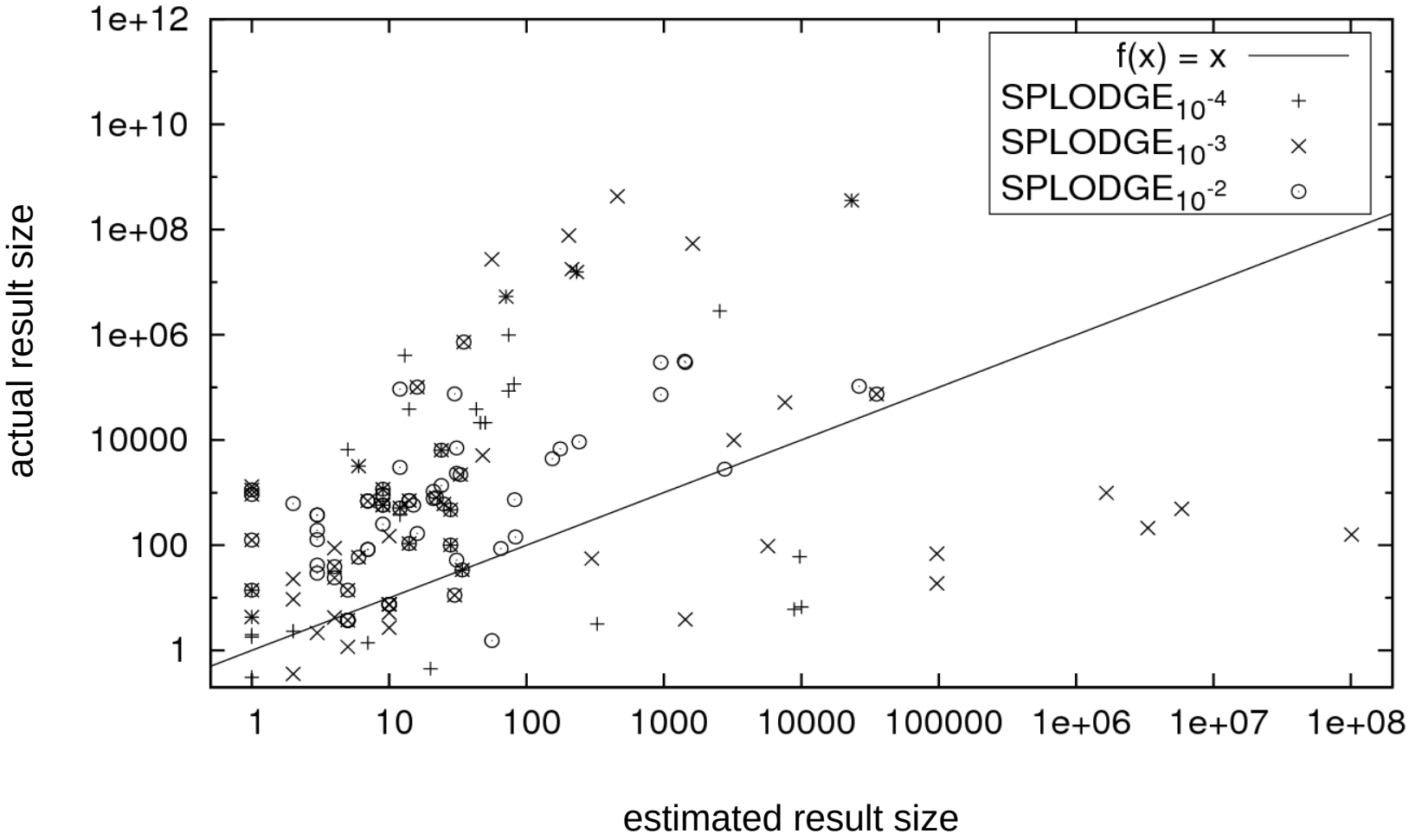
Evaluation Results



Evaluation Results



Estimated vs. actual results size



Predicate Occurrence in Queries

	all	SPLODGE*lite*	SPLODGE$_{10^{-4}}$	SPLODGE$_{10^{-3}}$	SPLODGE$_{10^{-2}}$
rdf:type	19.1 %	15.7 % (2)	18.6 % (1)	21.8 % (1)	21.0 % (1)
rdfs:seeAlso	8.4 %	16.3 % (1)	6.2 % (3)	5.3 % (4)	4.6 % (3)
owl:disjointWith	5.4 %	2.6 % (6)	3.7 % (5)	7.0 % (2)	9.5 % (2)
rdfs:subClassOf	3.9 %	3.1 % (5)	2.4 % (8)	6.5 % (3)	3.5 % (5)
rdfs:isDefinedBy	3.7 %	1.7 % (9)	6.5 % (2)	4.0 % (5)	1.9 % (10)
owl:equivalentClass	2.7 %	2.3 % (7)	2.7 % (7)	3.2 % (6)	2.7 % (8)
foaf:primaryTopic	2.4 %	3.9 % (4)	0.8 % (16)	1.9 % (7)	3.1 % (7)
rdfs:label	1.9 %	1.5 % (10)	1.6 % (10)	1.7 % (9)	3.2 % (6)
skos:exactMatch	1.8 %	1.3 % (11)	4.3 % (4)	0.9 % (18)	–
owl:sameAs	1.7 %	1.9 % (8)	3.2 % (6)	0.7 % (21)	0.8 % (14)
...					
zemanta:targetType	1.2 %	0.1 % (91)	0.5 % (29)	1.3 % (14)	3.8 % (4)
distinct predicates	687	402	353	283	191
total predicates	6600	1800	1800	1800	1200

Conclusion

SPLODGE provides

- Flexible query characterization + parameterization
- Methodology for Systematic & Scalable Query Generation
- Toolset as Open Source (http://code.google.com/p/splodge/)

Future Work:

- Create a LOD Federation Benchmark
- Interactive SPARQL query construction

Questions?

SPLODGE Evaluation Setup

BTC 2011 dataset in RDF3X

- pure triples, no context
- 160 GB repository file

(14h loading, 200 GB tmp mem)

	BTC 2010	BTC 2011	BTC 2012
Total Size	624 GB	450 GB	303 GB
Total Quads	3.171.793.030	2.178.395.469	1.436.545.545
Unique Quads	3.154.896.097	2.145.122.248	1.311.765.894
Unique Triples	1.426.831.520	1.968.347.976	1.056.184.911
Contexts/Documents	8.132.721	7.423.477	9.283.829
Common Domains	22.299	789	837
Types	168.482	314.448	296.607
Predicates	95.589	47.738	57.257

SPLODGE Pre-Processing for BTC data