Expressive and Scalable Query-based Faceted Search over SPARQL Endpoints

Sébastien Ferré
Team LIS, Data and Knowledge Management, IRISA/Univ. Rennes 1

Semantic Web (ISWC)
22 October 2014, Riva del Garda

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTEMES ALÉATOIRES

UMR IRISA LIS

Semantic Search

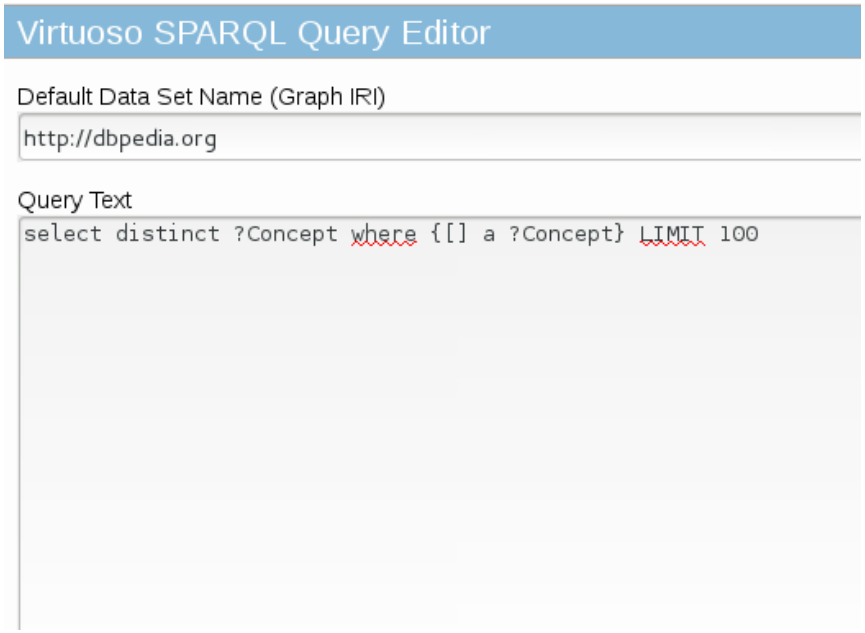
Semantic search is one of the cornerstone of the Semantic Web

- *What's the point building all those ontologies and linked data if one cannot explore and query them effectively?*
- Good news: we now have loads of data
  - public SPARQL endpoints, Linked Open Data, RDB wrappers
  - no more a "chicken-and-egg problem": *We have the chickens and we need to get more eggs out of them!*
- Bad news: user-side semantic search remains difficult
  - mostly canned queries sent programmatically by apps
  - SPARQL editors remain the tools of choice
    - ★ SPARQL is expressive, scalable, a standard

Difficulties with SPARQL for Users (1/2)

SPARQL has a formal syntax and semantics that needs to be learned

- like for programming languages



- with arcane error messages

```
Virtuoso 37000 Error SP030: SPARQL compiler, line 4: syntax error at '.' before '}'
```

We've just lost $>$ 90% *of potential users !*

Difficulties with SPARQL for Users (2/2)

The vocabulary of the target endpoint must be known

- precise URIs of classes and properties

```
SELECT DISTINCT ?film ?country
WHERE { ?film a dbpedia-owl:Film ;
            dbpedia-owl:actor dbpedia:Johnny_Depp ;
            dbpedia-owl:country ?country }
LIMIT 100
```

- any vocabulary error generally entails empty results with no explanation

film	country

This makes semantic search tedious, and causes frustration

Alternatives to SPARQL

How to make semantic search more usable ?

- Interactivity
  - semantic browsers: from entity to entity
    - ★ e.g. Fluidops
  - query builders: syntax-based (graphical) editors
    - ★ e.g. SemanticCrystal, Atomate it!
  - faceted search: guided filtering of sets of entities
    - ★ e.g. Ontogator, mSpace, /facet, gFacet, VisiNav, SemFacet, OpenLink
- Natural language (NL)
  - spontaneous NL + parsing + mapping
    - ★ e.g. PowerAqua, CASIA
  - controlled NL (CNL) + autocompletion
    - ★ e.g. Ginseng, ACEWiki

Required Properties and Existing Limits

- expressivity = covering users' information needs
  - generally, *the more usable, the less expressive*
  - only a small subset of SPARQL 1.1 is covered
- guidance = assisting users in their search
  - most often concept-based (coarse-grained)
  - instance-based (fine-grained) $\Rightarrow$ lower expressivity
- readability = speaking users' language
  - high expressivity $\Rightarrow$ only NL is readable
- scalability = scaling to largest endpoints
  - interactivity $\wedge$ fine-grained guidance $\Rightarrow$ lower scalability
- portability = working on new users' endpoints
  - many require manual configuration

We need them all for an effective semantic search !

Our Approach (implemented as SPARKLIS)

Our approach combines:

1. faceted search + structured queries
   - to reconcile guidance and expressivity
   - QFS = Query-based Faceted Search [ISWC'11]
2. requests to SPARQL endpoints for results and increments
   - to provide scalability and portability
3. NL verbalization of queries
   - to reconcile expressivity and readability

Not to be confused with

- query builders:
  - + fine-grained guidance, + NL verbalization
- NL interfaces:
  - no free input: − parsing, − mapping
  - more flexible guidance than auto-completion

Our Approach (implemented as SPARKLIS)

Our approach combines:

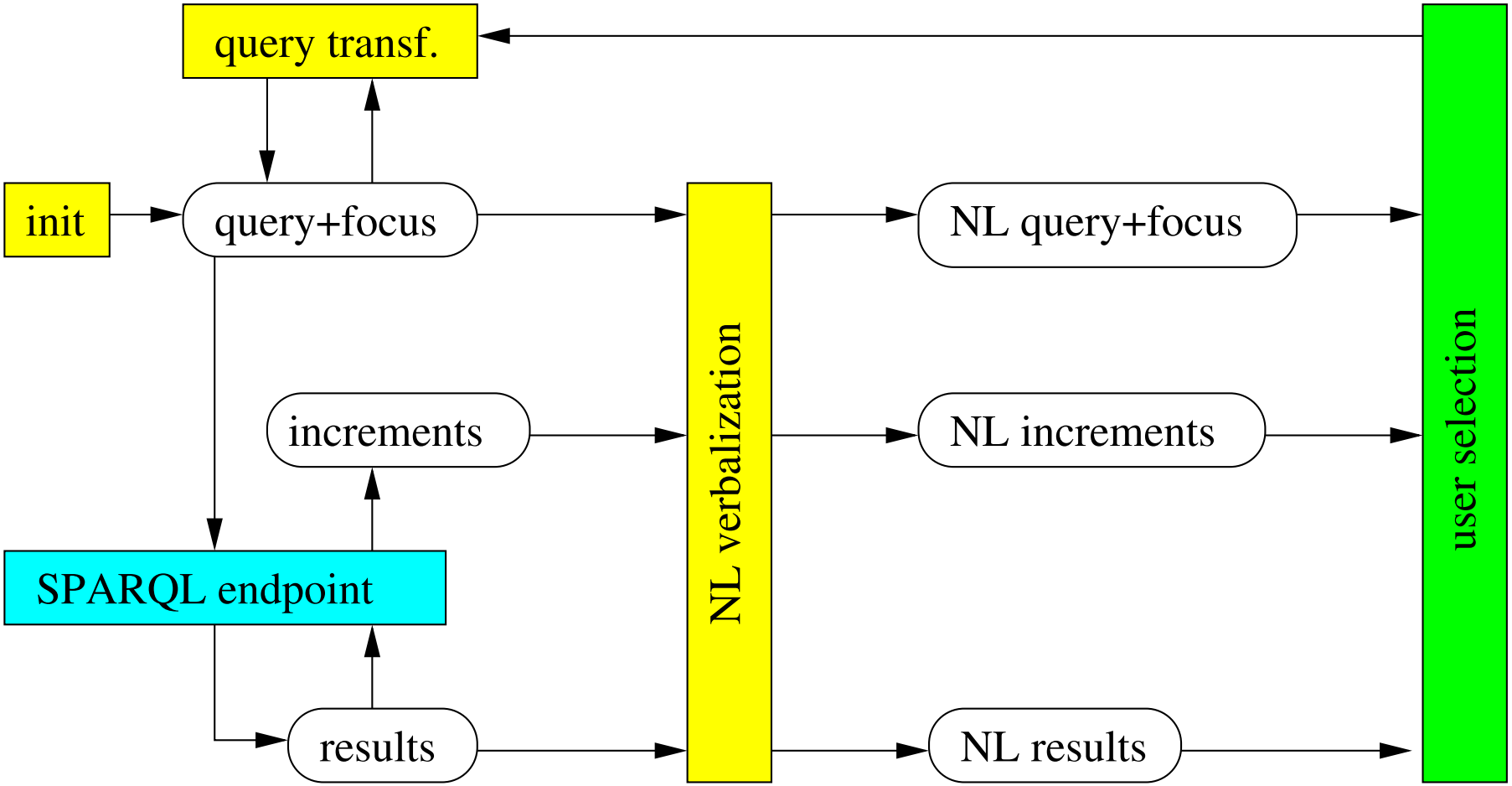
1. faceted search + structured queries
   - to reconcile guidance and expressivity
   - QFS = Query-based Faceted Search [ISWC'11]
2. requests to SPARQL endpoints for results and increments
   - to provide scalability and portability
3. NL verbalization of queries
   - to reconcile expressivity and readability

Not to be confused with

- query builders:
  - + fine-grained guidance, + NL verbalization
- NL interfaces:
  - no free input: − parsing, − mapping
  - more flexible guidance than auto-completion

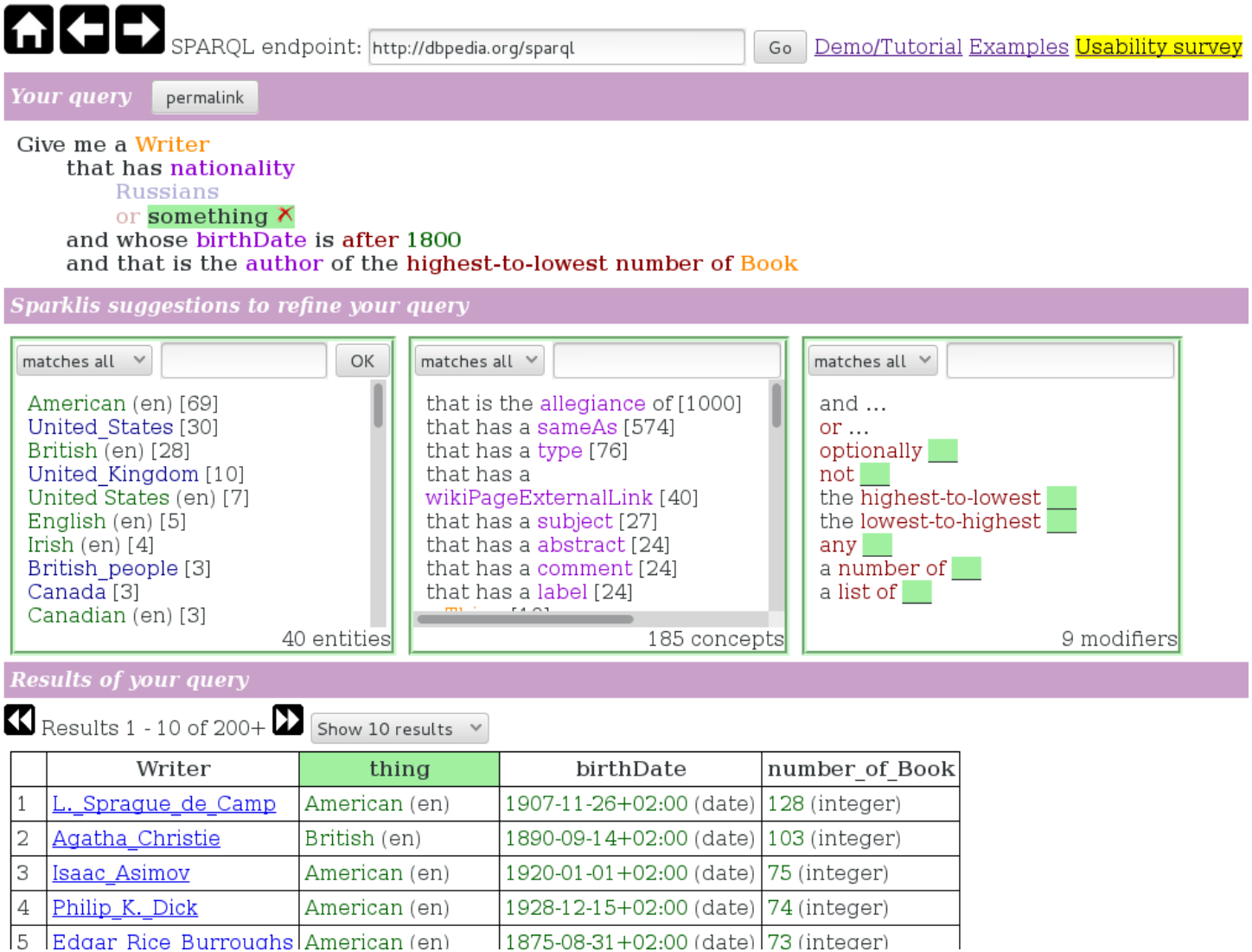
SPARKLIS' Architecture

How does it work ?



Note: the focus is an "insertion point" in the query

SPARKLIS' Screenshot



	Writer	thing	birthDate	number_of_Book
1	L._Sprague_de_Camp	American (en)	1907-11-26+02:00 (date)	128 (integer)
2	Agatha_Christie	British (en)	1890-09-14+02:00 (date)	103 (integer)
3	Isaac_Asimov	American (en)	1920-01-01+02:00 (date)	75 (integer)
4	Philip_K._Dick	American (en)	1928-12-15+02:00 (date)	74 (integer)
5	Edgar_Rice_Burroughs	American (en)	1875-08-31+02:00 (date)	73 (integer)

A Few Technical Hints

"The devil is in the details"

1. the internal representation of queries is intermediate between NL and SPARQL
   - a compiler technique to simplify translations (both ways) and query transformations
2. only partial results for the sake of scalability (`LIMIT 200`)
   - increments are computed over those partial results
   - an intelligent autocompletion is used to ensure guidance completeness
3. special increment computation when blank nodes in results
   - because blank nodes cannot be injected into SPARQL queries

Evaluations

Using SPARKLIS in Firefox 19.0.2 on a Linux laptop

- expressivity
  - covers SPARQL 1.1 – expressions – named graphs
  - covers all QALD questions
- scalability
  - responsive on `http://dbpedia.org/sparql`
  - QALD questions can be answered in 30s (avg) by a trained user
- guidance and readability (i.e. usability)
  - training is necessary, but is usually short (online video = tutorial)
  - survey (12 questions): SPARQL-aware (10/12) vs non-IT (7/12)
  - most complex question: *Give me all bridges crossing the Saint Laurence river, ordered by decreasing length, and with an optional depiction.* (success for 4/6 subjects)
- portability
  - only standards: HTML5/CSS3/JS + SPARQL HTTP requests (280kb)
  - main limit: half endpoints do not allow cross-domain requests

Kinds of Questions Covered by SPARKLIS

- fact retrieval
  - *Give me the homepage of Forbes.*
- result lists
  - *Which rivers flow into a German lake ?*
- tables
  - *Give me Tim Burton's films with their optional release date, and their budget in decreasing order.*
- analytics (OLAP)
  - *Give me the average runtime of films per European country, and per release date.*
- overviews
  - *What about songs from the sixties ?*

Conclusion

In Summary:

- Query-based Faceted Search (QFS) over SPARQL endpoints
  - expressivity of SPARQL
  - guidance of faceted seach
  - readability of NL
  - scalability and portability of SPARQL endpoints

Available as a mature Web app:

- SPARKLIS – `http://www.irisa.fr/LIS/ferre/sparklis/`
  - used by *GenOuest bioinformatics platform* (France), *Database Center for Life Science* (Japan), *XSB* (USA) on their own endpoints

Help!

Help me improve it by filling in the survey
...and by asking non-IT people to do so !