Avalanche: adaptive query processing over distributed linked data-endpoints

Microdata Computation Centre (MiCoCe) Workshop
Nuremberg 2014

Cosmin Basca, DDIS, University of Zurich
Background

- No or limited control over data distribution,
- No guarantees about data availability and network connectivity of data providers,
- No guarantees about content stability
What is needed?

- **Transparency**: a Semantic Federated Database Management System
- **Adaptivity** (the web is a dynamic environment)
  - during source selection
  - during execution
- **Flexibility** (borrow from p2p systems)
- **Adhere to standards** when possible
  - SPARQL, RDF, HTTP(S), etc
Avalanche

- Does not require detailed fine-grained ex-ante statistics with the query engine
- Makes no assumptions about data distribution, schema, availability, or partitioning
- Provides up-to-date results from distributed indexed endpoints
- Is adaptive during execution adjusting dynamically to external network changes
- Is flexible as it makes limited assumptions about the structure of participating RDF stores
Running Avalanche (Overview)

1) get sources

2) get triple pattern cardinalities

3) execute distributed join

Avalanche
SPARQL endpoint

Search Engine
i.e., http://void.rkbexplorer.com/

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Running Avalanche (Detailed)

1] Source Discovery phase

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX drugbank: <http://www4.wiwi.fu-berlin.de/drugbank/resource/drugbank/> 
3 PREFIX chebi: <http://bio2rdf.org/ns/bio2rdf#>
4 PREFIX dc: <http://purl.org/dc/elements/1.1/>
5
6 SELECT ?drug ?keggUrl ?chebiImage
7 WHERE
8 {
15 }
```
Running Avalanche (Detailed)

1) Source Discovery phase

AVALANCHE endpoints
Search Engine
i.e., http://void.rkbexplorer.com/

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Running Avalanche (Detailed)

1] Source Discovery phase

Query Parser → Source Selector → Statistics Requester

AVALANCHE endpoints
Search Engine
i.e., http://void.rkbexplorer.com/

2] Statistics Gathering phase

Query Monitor / Stopper → Results Queue

Asynchronous Executor Pool → Plan Dispatcher → Plan Generator

Result Source Selector → Query Parser

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Running Avalanche (Detailed)

1] Source Discovery phase

2] Statistics Gathering phase

3] Query Planning and Execution phase

---

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Running Avalanche (Detailed)

**1] Source Discovery phase**

**3] Query Planning and Execution phase**

```sql
1. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2. PREFIX drugbank: <http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/>
3. PREFIX chebi: <http://bio2rdf.org/ns/bio2rdf#>
4. PREFIX dc: <http://purl.org/dc/elements/1.1/>

5. SELECT ?drug ?keggUrl ?chebiImage WHERE {
   6. SERVICE <http://drugbank-endpoint/sparql> {
      9.   ?drug rdf:type drugbank:drugs
   10. }
   11. SERVICE <http://chebi-endpoint/sparql> {
   14. }
   15. SERVICE <http://kegg-endpoint/sparql> {
   17. }
   18. }
```

Requirements for Avalanche endpoints

Common IDs

A requirement for executing joins between any two hosts is that they share a common id space. The natural identity on the web is given by the URI itself. However, some statistical analyses of URIs on the web show that the average length of a URI is 76 characters, while analyses of the Billion Triple Challenge dataset reveal:

- [Average length of a URL](http://www.supermind.org/blog/740/average-length-of-a-url-part-2)
- [Billion Triple Challenge](http://gromgull.net/blog/category/semantic-web/billion-triple-challenge/).
Running Avalanche (Detailed)

1] **Source Discovery** phase

- Query Parser
- Source Selector
- Statistics Requester

AVALANCHE endpoints
Search Engine
i.e., http://void.rkbexplorer.com/

2] **Statistics Gathering** phase

3] **Query Planning and Execution** phase

- Plan Generator
- Plan Dispatcher
- Plan Executor Pool

Asynchronous Executor Pool
Running Avalanche (Detailed)

1] Source Discovery phase

Query Parser → Source Selector → Statistics Requester

AVALANCHE endpoints
Search Engine
i.e., http://void.rkbexplorer.com/

2] Statistics Gathering phase

3] Query Planning and Execution phase

Plan Generator → Plan Dispatcher → Plan Executors

Asynchronous Executor Pool

Results Queue

Query Monitor / Stopper

Result

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Execution in a nutshell

- Exhaustive exploration of all possible non-union plans
  - while incomplete, these plans are simple
- Concurrent execution of all generated plans
- Order of execution is critical (the planner handles this)
  - favour fast plans, low latency connections
- Uncertain termination criteria (user specified):
  - Fast First K results
  - Relative Saturation (no new results seen, stop early)
  - + traditional ones: Limit / Timeout
No free lunch…

- In its purest form Avalanche can lead to a higher system load
- The space of incomplete plans can be very large
- In part a direct consequence of the individual servers’ autonomy, essentially a tradeoff between decentralisation and system load
No free lunch…

- In its purest form Avalanche can lead to a higher system load
  - the space of incomplete plans can be very large
  - in part a direct consequence of the individual servers’ autonomy, essentially a tradeoff between decentralisation and system load

- Good news: we are testing an optimised query planner!
  - offers control over the tradeoff between the number of plans and their complexity
Privacy benefits for Microdata computation

- **The client**: can run a query without knowing which data-store has elements of the answer
- No need for a centralised catalog, query pertinent statistical information is requested at query time (via the same security constraints under which the query is executed)
Privacy benefits for Microdata computation

requests fine-grain statistics, pertinent to the given query only

1] Source Discovery phase

2] Statistics Gathering phase

3] Query Planning and Execution phase

AVALANCHE endpoints
Search Engine
i.e., http://void.rkbexplorer.com/

Avalanche: adaptive query processing over distributed Linked-Data endpoints, Cosmin Basca
Privacy benefits for Microdata computation

requests fine-grain statistics, pertinent to the given query only

... in a privacy ensuring way:
- generalised statistics (i.e., >1000)
- detect if requester can infer about individuals and halt if yes
Privacy benefits for Microdata computation

□ **The data-provider:** can answer queries without having to disclose what parts of the answer it could have
□ same reason as before
Privacy benefits for Microdata computation

☐ The data-provider: can answer queries without having to disclose what parts of the answer it could have
☐ same reason as before

☐ possible extensions:
  ☐ the data-provider gets fine grained control over what kind of answers is willing to give (i.e., only aggregation queries)
  ☐ make use of privacy-preserving joins (i.e., from data-mining approaches)
Future benefits for Microdata computation

- Avalanche could be extended to answer queries while ensuring k-anonymity and l-diversity on each data provider
- Enable the computation of quality bounds on the overall answer
Conclusion (1)

□ Avalanche:
  □ is transparent, adaptive and flexible
  □ provides partial results
  □ completeness guarantees only if all plans are executed
  □ uncertain termination criteria (user specified)
  □ does not break privacy by requiring an external catalog of participating data-providers
Conclusion (2)

- Possible Avalanche extensions:
  - Ensure **k-anonymity** (and **l-diversity**)
  - Report **generalised statistics** for query planning
  - Detect whether the requester can infer about individuals and halt or take action if yes
  - Use **privacy-preserving** joins (i.e., from Data-Mining)
Thank you

Acknowledgements:
The Swiss National Science Foundation