DB vs RDF: structure vs correlation

+ comments on data integration & SW

Peter Boncz
Senior Research Scientist @ CWI
Lecturer @ Vrije Universiteit Amsterdam
Architect & Co-founder MonetDB
Architect & Co-founder VectorWise
Correlations

Real data is highly correlated
- (gender, location) ⇔ firstname
- (profession, age) ⇔ income

But…

database systems assume attribute independence
- wrong assumption leads to suboptimal query plan
Example: co-authorship query

DBLP
“find authors who published in VLDB, SIGMOD and Bioinformatics”

SELECT author
FROM publications p1, p2, p3
WHERE p1.venue = VLDB and p2.venue = SIGMOD and p3.venue = Bioinformatics and p1.author = p2.author and p1.author = p3.author and p2.author = p3.author
Correlations: RDBMS vs RDF

Schematic structure in RDF data hidden in correlations, which are everywhere

SPARQL leads to plans with
- many self-joins
- whose hit-ratio is correlated (with e.g. selections)

Relational query optimizers do not have a clue
- all self-joins look the same to it
- random join order, bad query plans 😞
RDF Engines to the Next Level

**Challenge:** solving the correlation problem.

**Ideas?**
- Interleave optimization and execution
  - Run-time *sampling* to detect true selectivities
  - Run-time query (re-)planning
- Tackling when there are long join chains
  - Creating partial path indexes
  - Graph “cracking”: index build as side-effect of query
Stratos Idreos: database cracking

**Challenge**: solving the correlation problem.

**Ideas?**

- Interleave optimization and execution
  - Run-time sampling to detect true selectivities
  - Run-time query (re-)planning

- Tackling when there are long join chains
  - Creating partial path indexes
  - Graph “cracking”: index build as side-effect of query
RDF Engines to the Next Level

**Challenge:** solving the correlation problem.

**Ideas**?

- Interleave optimization and execution
  - Run-time **sampling** to detect true selectivities
  - Run-time query (re-)planning

- Tackling when there are long join chains
  - Creating partial path indexes
  - Graph “**cracking**”: index build as side-effect of query
  - “**recycling**” intermediate results
RDF Engines to the Next Level

Benchmarking in LOD2:
  - Attempting to engage vendors to collaborate
    “a TPC for RDF”
  - New and more challenging benchmarks
    “Suitably designed benchmarks drive progress”

benchmarks with:
  - complex query patterns on large data
  - geo and text data and queries
  - outside Linked Open Datasets that get joined to synthetic data
  - highly interlinked graph structures and queries on them
  - correlated query predicates
Social Intelligence Benchmark (SIB)

RDF-friendly benchmark simulating a huge **social network**
- Social Graphs have understandable, interesting, scenarios
- Social Graphs are highly connected
  - LOD in the wild not yet

+ Exploiting knowledge bases from interlinked RDF datasets
  => not only synthetic data, also linking out to DBpedia

conversation topics, real world concepts, geographical information, connectivity, social network analysis

Data correlations galore
thoughts on.. **Information Integration**

**Data** integration and **Schema** Integration

- Different applications, organizations, time motives
  hard problem, tens of B$/y

- Has been on the DB R&D menu for 20+ years
  - AI complete, immature tech
  - hard to achieve high precision **automatically**

- Semantic Web does not solve this issue
  - In fact, it is its major hurdle to success
  - **Information integration != \{Reasoning, Inference, Logic\}**
The **schema.org** Approach

choose

- web-addressable, machine-readable schema+data over
- ragged, graph, schema-last RDF data model

Approach:

- schema-first, centralized, controlled
- well-defined use case (web search = ad money)
The **Watson** Approach

Winning **Jeopardy!** is pretty cool

No central role for reasoning, inference there

Recipe:
- Finding statistical evidence in Big Data
- Using semantics for focused sub-tasks (only)
- Intelligently combining multiple approaches
- **Focusing on the Jeopardy! problem at hand**