Linked Data Query Processing

Tutorial at the 22nd International World Wide Web Conference (WWW 2013)

May 14, 2013

http://db.uwaterloo.ca/LDQTut2013/

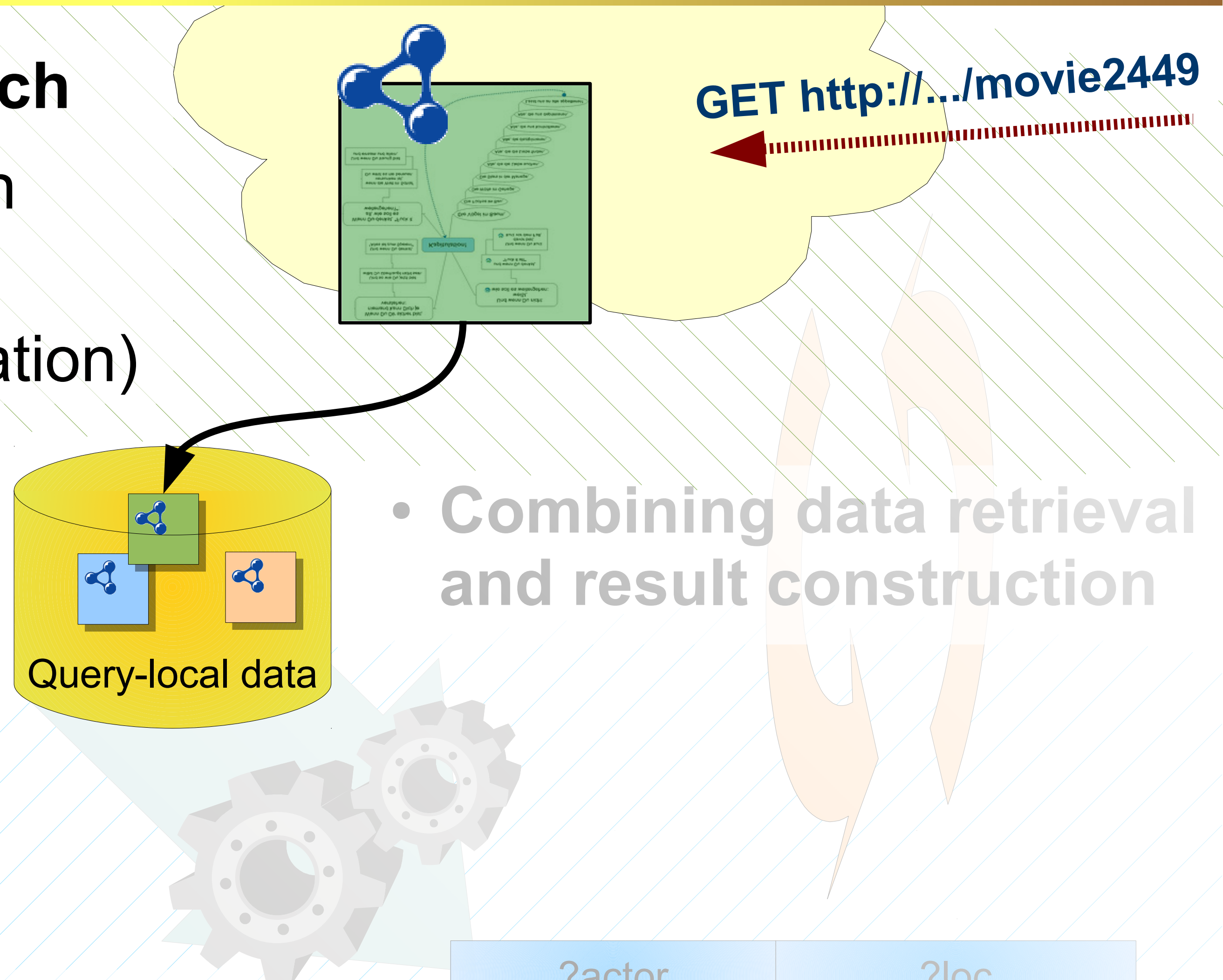
3. Source Selection

Olaf Hartig

University of Waterloo

“Ingredients” for LD Query Execution

- **Data retrieval approach**
  - Data source selection
  - Data source ranking (optional, for optimization)



- **Combining data retrieval and result construction**

- **Result construction approach**
  - i.e., query-local data processing

?actor	?loc
http://mdb.../Paul	http://geo.../Berlin
http://mdb.../Ric	http://geo.../Rome

Query-Specific Relevance of URIs

- **Definition: A URI is relevant for a given query if looking up this URI gives us data that contributes to the query result.**
- **Example:**
  - Conjunctive query (BGP): { (*Bob*, *lives in*, *?x*) , (*?y*, *lives in*, *?x*) }
  - Looking up URI *Bob* gives us: { (*Bob*, *lives in*, *Berlin*) , ... }
  - Looking up URI *Alice* gives us: { (*Alice*, *lives in*, *Berlin*) , ... }
  - Hence, $\mu = \{ ?x \rightarrow Berlin \text{ , } ?y \rightarrow Alice \}$ is a solution
  - Thus, URIs *Bob* and *Alice* are relevant for the query
- **Simply contributing a matching triple is not sufficient:**
  - Suppose, URI *Charles* gives us { (*Charles*, *lives in*, *London*) , ... }
  - Since the matching triple cannot be used for computing a solution, URI *Charles* is not relevant.

Objective of Source Selection

- **Source selection: Given a Linked Data query, determine a set of URIs to look up**

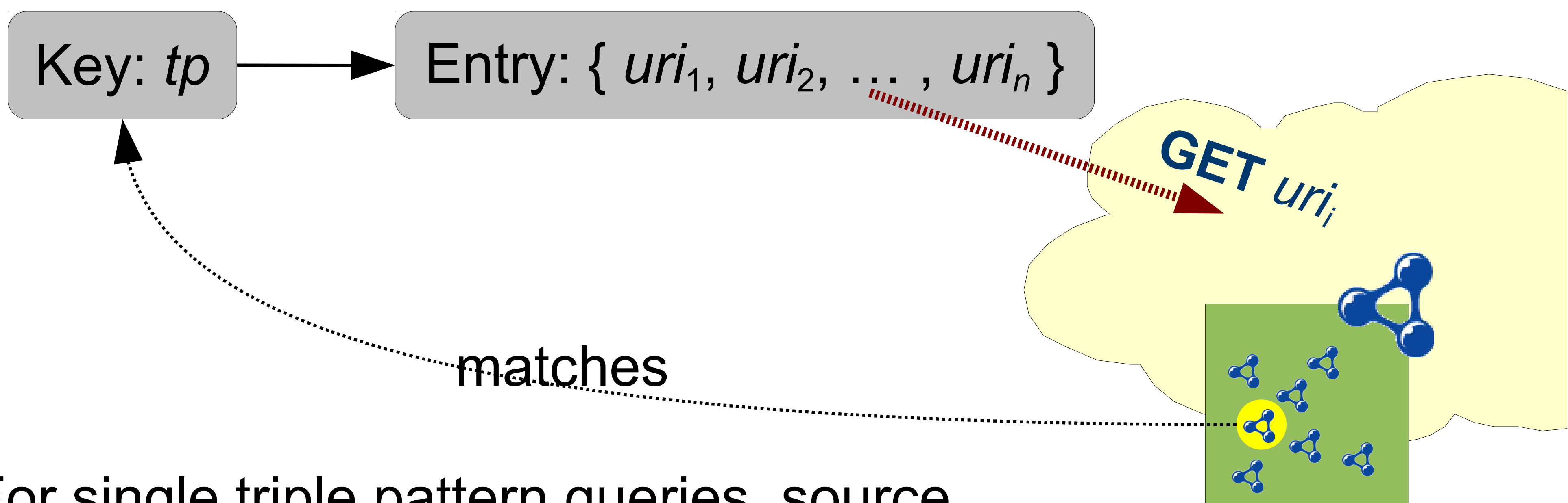
- **Ideal source selection approach:**
  - For any query, selects all relevant URIs
  - For any query, selects relevant URIs only
- **Irrelevant URIs are not required to answer the query**
  - Avoiding their lookup reduces cost of query executions significantly!
- **Caveat:**
  - What URIs are relevant (resp. irrelevant) is unknown before the query execution has been completed.

Outline

- Objectives of Source Selection √
- **Index-Based Strategy**
  - General Idea
  - Possible Index Structures
- **Live Exploration Strategy**
- **Comparison of both Strategies**
- **Combining both Strategies**

Idea of Index-Based Source Selection

- **Use a pre-populated index structure to determine relevant URIs (and to avoid as many irrelevant ones as possible)**
- **Example: triple-pattern-based indexes**



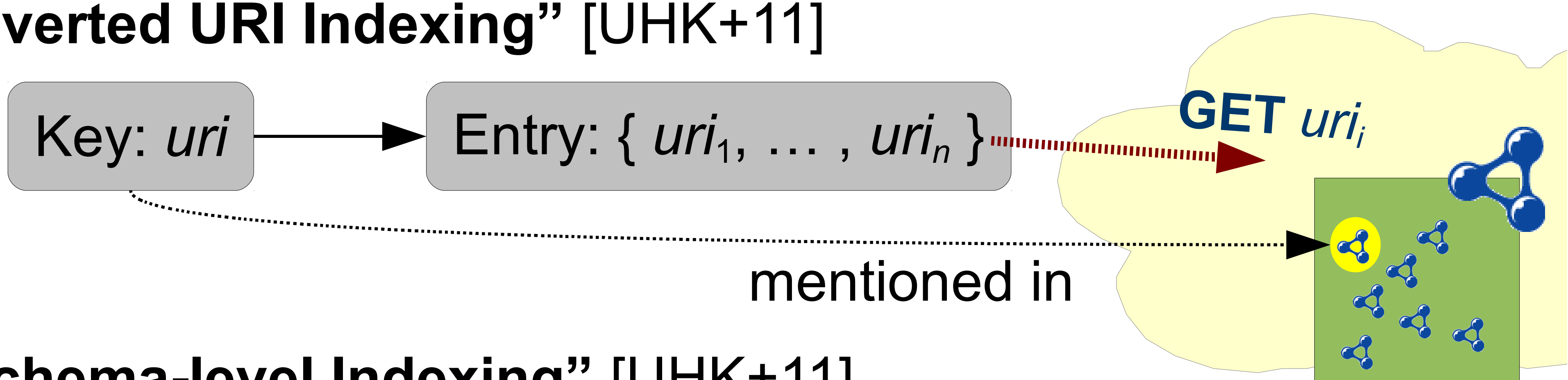
- For single triple pattern queries, source selection using such an index structure is sound and complete (w.r.t. the indexed URIs)

General Properties of Lookup Indexes

- **Index entries:**
  - Usually, a set of URIs
  - Each URI in such an entry may be paired with a cardinality (utilized for source ranking)
  - Indexed URIs may appear multiple times (i.e., associated with multiple index keys)
- **Type of index keys depends on the particular index structure used**
  - e.g., triple patterns
- **Represent a summary of the data from all indexed URIs**
  - Perfect summary: index keys are individual elements
  - Approximate summary: index keys may range over elements

Perfect Summaries

- **Triple-pattern-based indexes**
- **"Inverted URI Indexing"** [UHK+11]



- **"Schema-level Indexing"** [UHK+11]
  - Index keys: schema elements
  - Like a triple-pattern-based index that considers only two types of triple patterns: (*?s*, *property*, *?o*) and (*?s*, rdf:type, *class*)
- **Tian et al.** [TUY11]
  - Index keys: Unique encodings of combinations of triple patterns (i.e., BGPs) frequently found in a query workload

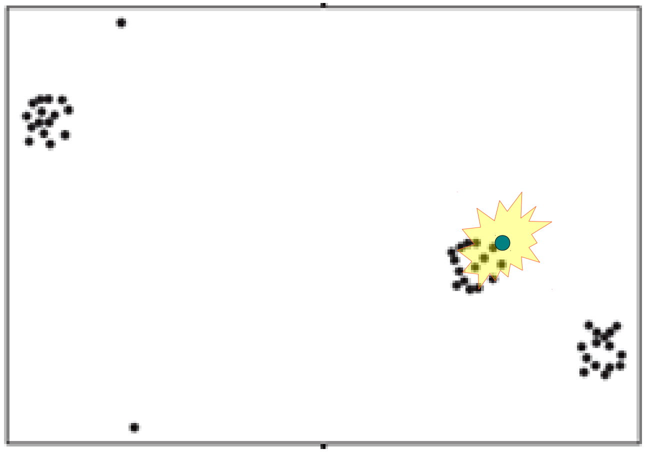
Approximate Summaries

- **Recall, index keys may range over elements**
- **Advantage: approximation reduces index size**
- **Disadvantage: index lookup may return false positives**
- **Examples of data structures used:**
  - Multidimensional histogram [UHK+11]
  - QTree [HHK+10, UHK+11]

Multidimensional Histograms

- **Transform RDF triples to points in a 3-dimensional space**

  (*Bob*, *lives in*, *Berlin*) → hash function → (422, 247, 143)



Multidimensional Histograms

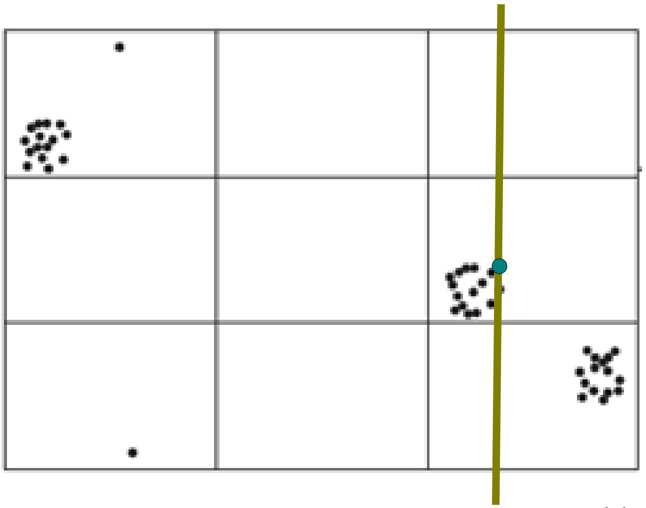
- **Transform RDF triples to points in a 3-dimensional space**

  (*Bob*, *lives in*, *Berlin*) → hash function → (422, 247, 143)

- **Buckets partition that space into disjoint regions**
- **Indexing: Each bucket contains entries for all URIs whose data includes an RDF triple in the corresponding region**
- **Source selection:**
  - Transform triple patterns to lines / planes in the space

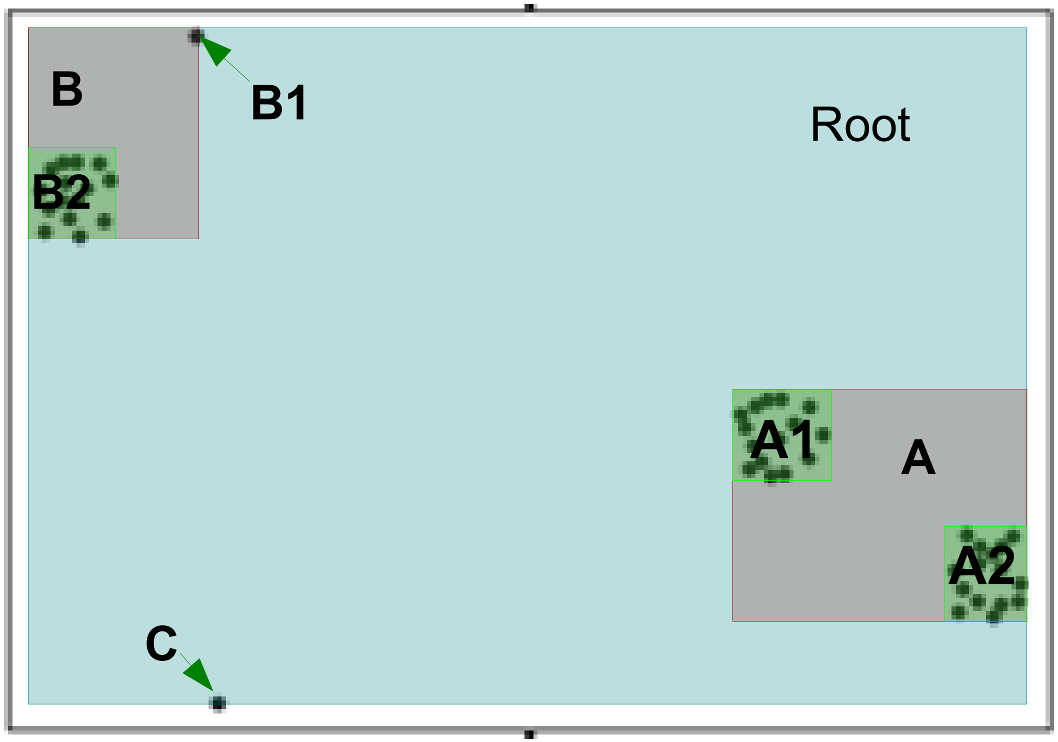
    (*Bob*, *lives in*, *?x*) → (422, 247, ?)

  - Any URI relevant for the triple pattern may only be contained in buckets whose region is touched by the line / plane
  - Pruning due to non-overlapping regions



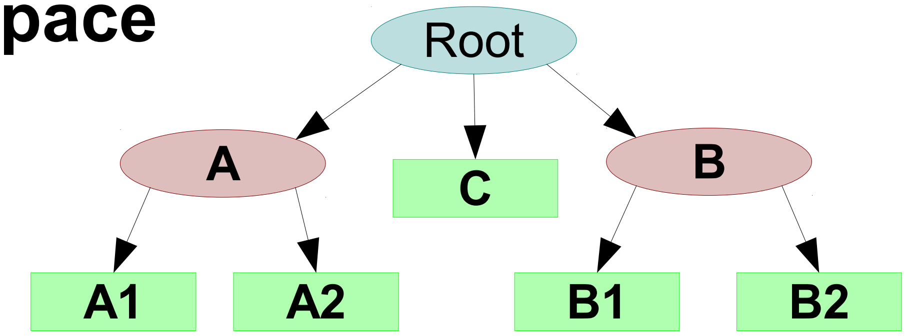
QTree

- **Combination of histograms and R-trees (i.e., hierarchical)**



- **Leaf nodes are the buckets**
  - Different buckets may represent regions of different size
    (in contrast to fixed-sized regions used for MDH)
  - Non-populated regions are ignored

- **Deals more efficiently with a space that is populated sparsely or contains many clusters**



Index Construction

- Given a set of URIs to index, each of these URIs needs to be looked up and its data needs to be retrieved
- Alternative: crawl the Web to obtain URIs and their data
- Alternative: populate index as a by-product of executing queries using live-exploration-based source selection

Index Maintenance

- **Adding additionally discovered URIs**
- **Keeping the index in sync with original data**
  - Still an open research problem
  - Similar to index maintenance in information retrieval and view maintenance in database systems

Outline

- Objectives of Source Selection √
- Index-Based Strategy
  - General Idea
  - Possible Index Structures √
- **Live Exploration Strategy**
- Comparison of both Strategies
- Combining both Strategies

Live Exploration

- **General idea: Perform a recursive URI lookup process at query execution runtime**
  - Start from a set of seed URIs
  - Explore the queried Web by traversing data links
- **Retrieved data serves two purposes:**
  (1) Discover further URIs
  (2) Construct query result
- **Lookup of URIs may be constrained (i.e., not all links need be traversed)**
  - Natural support of reachability-based query semantics

Comparison to Focused Crawling

Focused Crawling **vs.** **Live Exploration**

- **URIs qualify for lookup because of their high relevance for a topic**
- **Separate pre-runtime (or background) process**
  - Crawler populates a search index or a local database

- **Relevance of URIs related to the query at hand**
- **Essential part of the query execution process itself**
  - Live exploration aims to discover data for answering a particular query

Outline

- **Objectives of Source Selection** √
- **Index-Based Strategy** √
  - General Idea
  - Possible Index Structures
- **Live Exploration Strategy** √
- **Comparison of both Strategies**
- **Combining both Strategies**

Live Exploration – *vs.* – Index-Based

- **Possibilities for parallelized data retrieval are limited**
  - Data retrieval adds to query execution time significantly
- **Usable immediately**
  - Most suitable for "on-demand" querying scenario
- **Depends on the structure of the network of data links**

- **Data retrieval can be fully parallelized**
  - Reduces the impact of data retrieval on query exec. time
- **Usable only after initialization phase**
- **Depends on what has been selected for the index**
- **May miss new data sources**

None of both strategies is superior over the other w.r.t. result completeness (under full-Web query semantics).

- Both strategies may miss (different) solutions for a query

Hybrid Source Selection

Why not get the best of both strategies by combining them?

- **Ideas:**
  - Use index to obtain seed URIs for live exploration (e.g., “mixed strategy” [LT10])
  - Feed back information discovered by live exploration to update, to expand, or to reorganize the index
  - Use data summary for controlling a live exploration process (e.g., by prioritizing the URIs scheduled for lookup)

Outline

- **Objectives of Source Selection** √
- **Index-Based Strategy** √
  - General Idea
  - Possible Index Structures
- **Live Exploration Strategy** √
- **Comparison of both Strategies** √
- **Combining both Strategies** √

Next part: 4. Execution Process ...

These slides have been created by

Olaf Hartig

for the

WWW 2013 tutorial on

Link Data Query Processing

Tutorial Website: http://db.uwaterloo.ca/LDQTut2013/

(Slides 10,11, and 12 are inspired by slides from Andreas Harth [HHK+10] – Thanks!)