dbis
INSTITUT FÜR INFORMATIK
HUMBOLDT-UNIVERSITÄT ZU BERLIN

HUMBOLDT-UNIVERSITÄT ZU BERLIN

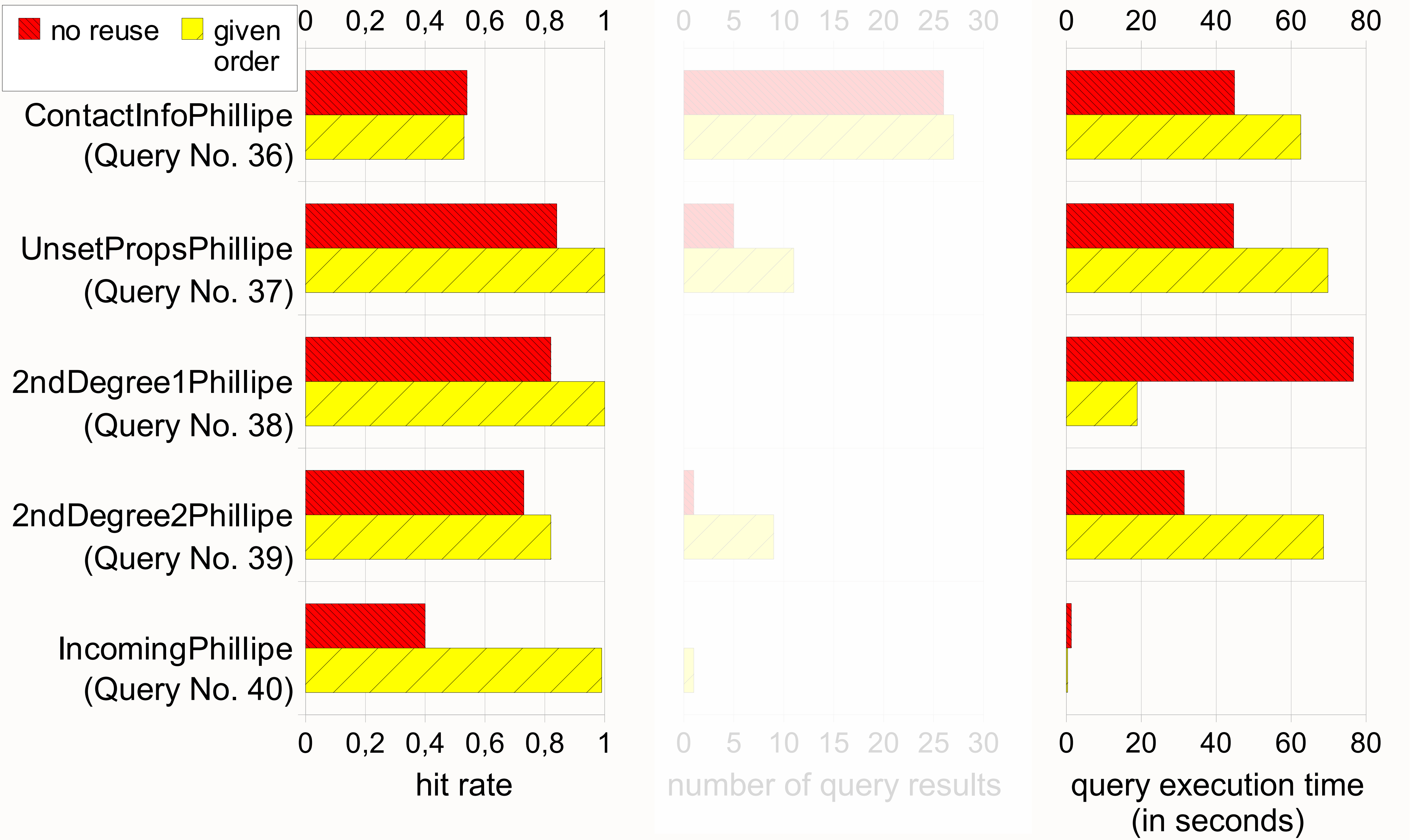
A Main Memory Index Structure to Query Linked Data

Olaf Hartig
http://olafhartig.de/foaf.rdf#olaf
@olafhartig

Frank Huber

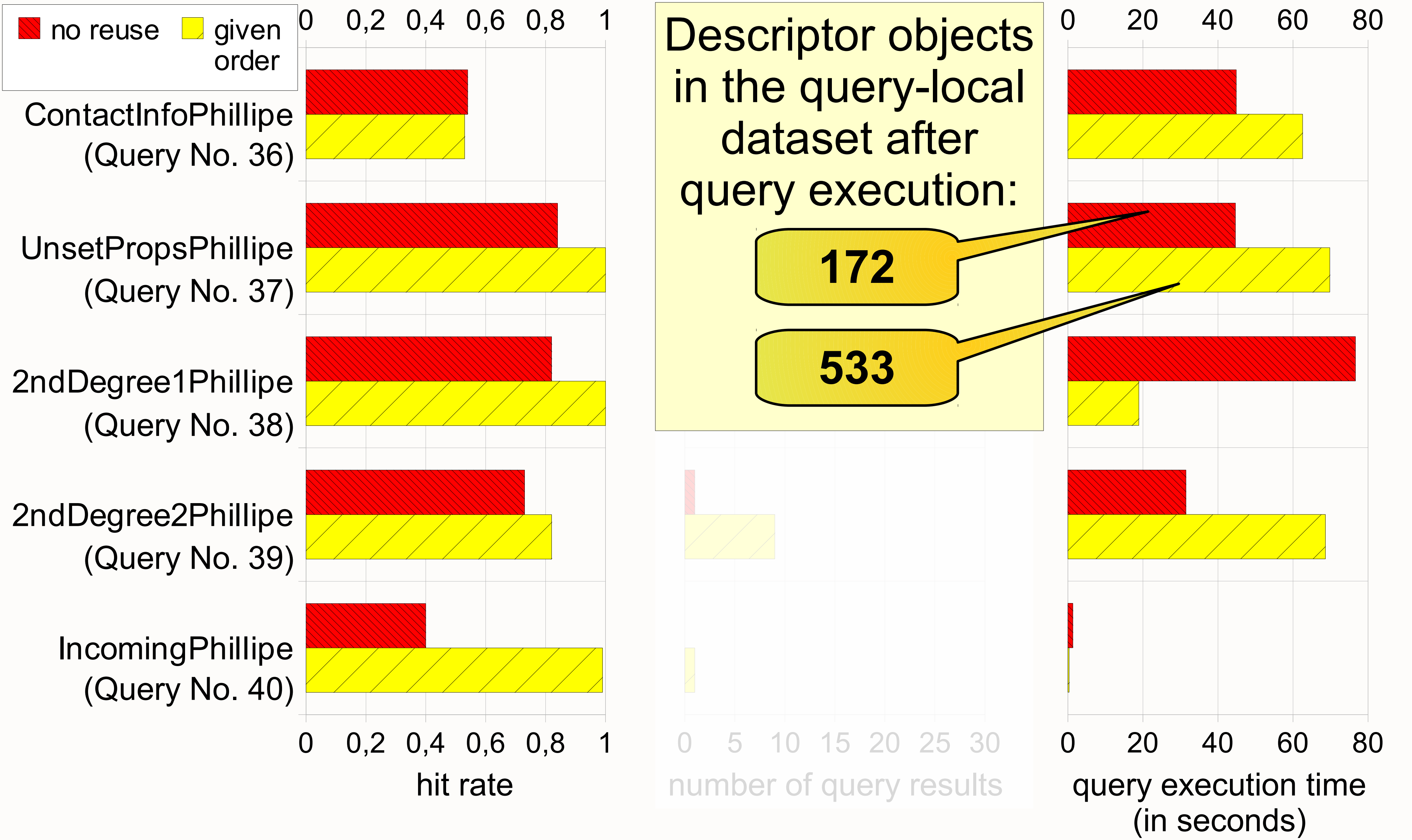
Database and Information Systems Research Group
Humboldt-Universität zu Berlin

The Issue



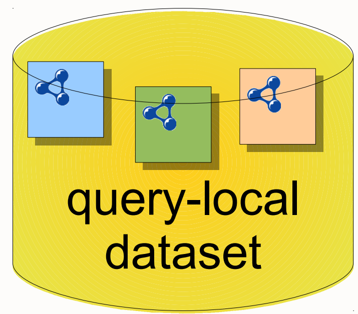
no reuse
given order
ContactInfoPhillipe (Query No. 36)
UnsetPropsPhillipe (Query No. 37)
2ndDegree1Phillipe (Query No. 38)
2ndDegree2Phillipe (Query No. 39)
IncomingPhillipe (Query No. 40)
0 0,2 0,4 0,6 0,8 1
hit rate
0 5 10 15 20 25 30
number of query results
0 20 40 60 80
query execution time (in seconds)

The Issue



no reuse
given order
ContactInfoPhillipe (Query No. 36)
UnsetPropsPhillipe (Query No. 37)
2ndDegree1Phillipe (Query No. 38)
2ndDegree2Phillipe (Query No. 39)
IncomingPhillipe (Query No. 40)
0 0,2 0,4 0,6 0,8 1
hit rate
0 5 10 15 20 25 30
number of query results
0 20 40 60 80
query execution time (in seconds)
Descriptor objects in the query-local dataset after query execution:
172
533

Logical representation of
Linked Data from the Web



Physical representation of
Linked Data from the Web

?

What data structure do we use to physically represent the query-local dataset?

Outline

1. Requirements + Existing Work

2. Data Structures

3. Evaluation

Requirements

- **(Consecutively) build and use ad hoc collections of many small sets of RDF triples**
- **Four main operations:**
  - *Find* … matching triples for a triple pattern in all descriptor objects
  - *Add*, *Remove*, *Replace* … descriptor objects
- **Support of concurrent access (i.e. isolation)**

- ***Non*-relevant properties:**
  - Querying descriptor objects individually is not necessary
  - No need to write data back to the Web
  - ACID properties not required for complete queries

Requirements

- **(Consecutively) build and use ad hoc collections of many small sets of RDF triples**
- **Four main operations:**
  - *Find* … matching triples for a triple pattern in all descriptor objects
  - *Add*, *Remove*, *Replace* … descriptor objects
- **Support of concurrent access (i.e. isolation)**

- ***Non*-relevant properties:**
  - Querying descriptor objects individually is not necessary
  - No need to write data back to the Web
  - ACID properties not required for complete queries

Existing Work

- **Disk based storage solutions for RDF data**
  - Unsuitable due to very costly I/O operations
- **Main memory based data structures in the literature**
  - Focus on a large, single set of RDF triples
  - Optimized for complete graph pattern queries or path queries
- **Main memory based data structures in RDF frameworks**
  - Focus on Jena, ARQ and NG4J
  - Inefficient (see evaluation)

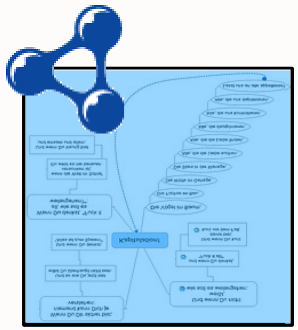
Outline

1. Requirements + Existing Work

2. Data Structures

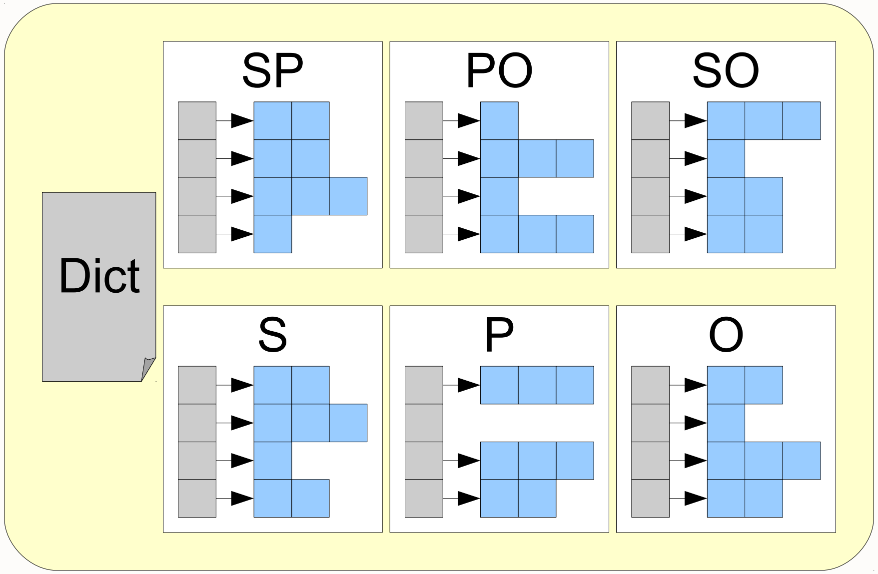
3. Evaluation

Hash-Based Index for RDF Data



Logical representation

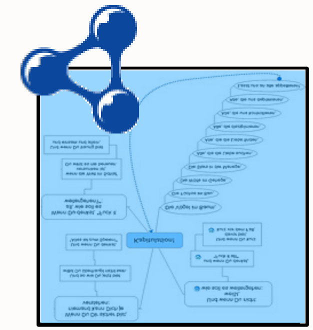
Physical representation



- **Dictionary:**
  - Two-way mapping between RDF terms and numerical identifiers
- **6 hash tables:**
  - Each hash table contains all ID-encoded triples
  - Efficient support for all types of triple patterns

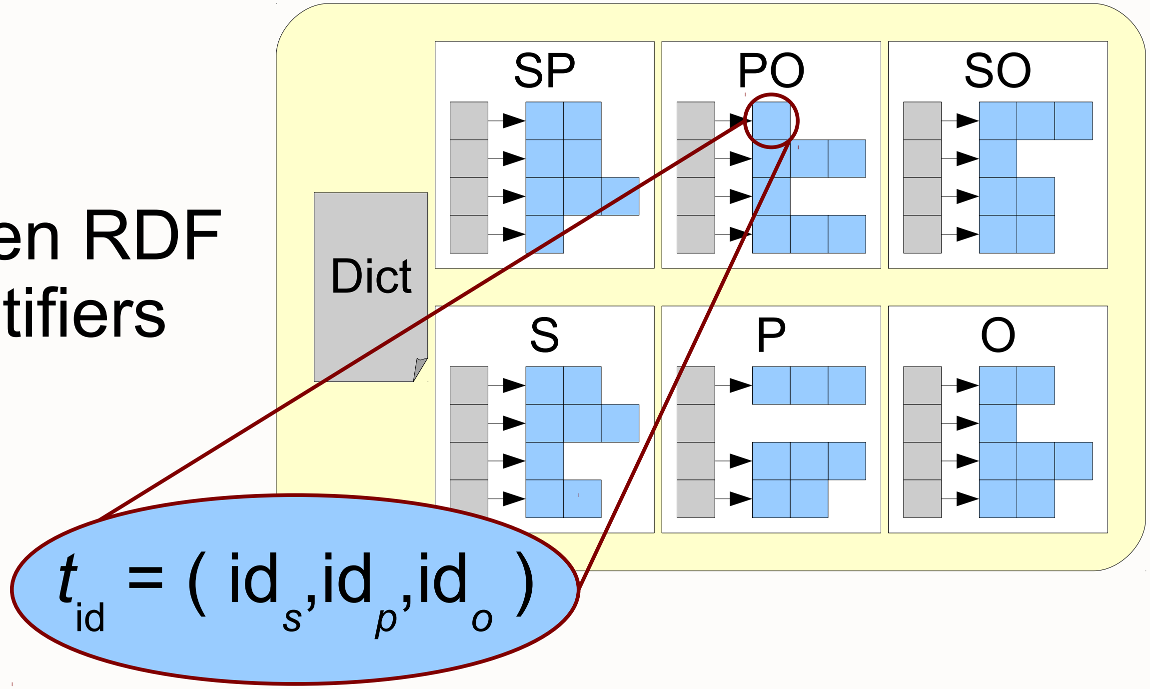
*Similar to Harth and Decker, 2005

Hash-Based Index for RDF Data



Logical representation

Physical representation



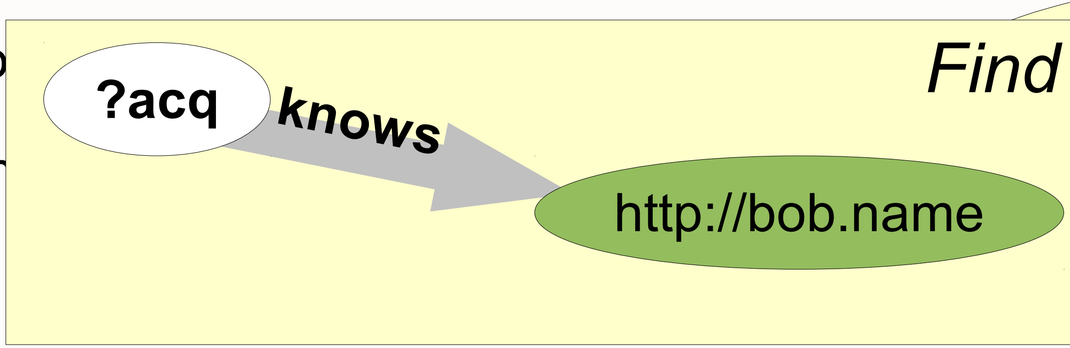
- **Dictionary:**
  - Two-way mapping between RDF terms and numerical identifiers
- **6 hash tables:**
  - Each hash table contains all ID-encoded triples
  - Efficient support for all types of triple patterns

*Similar to Harth and Decker, 2005

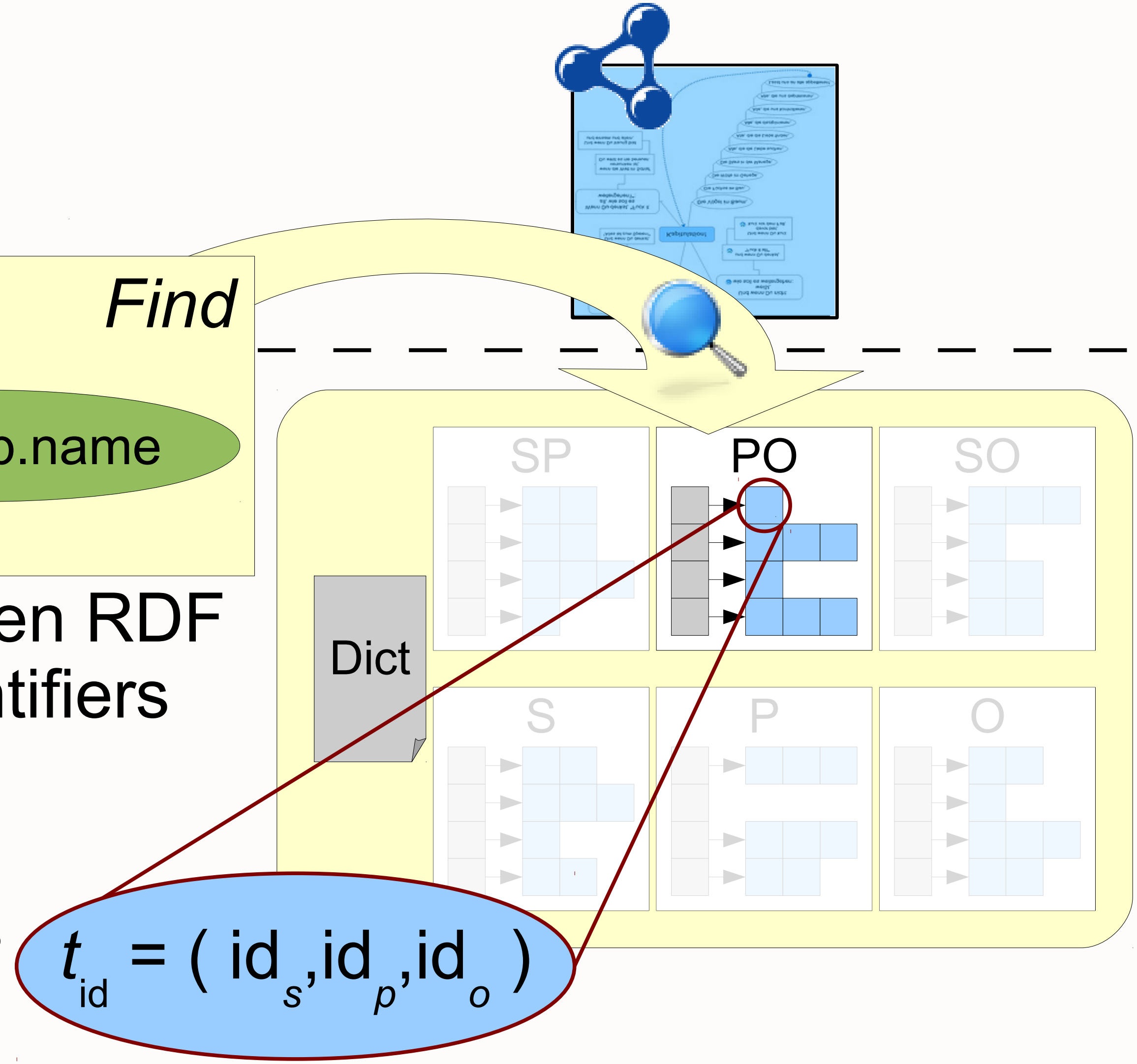
Hash-Based Index for RDF Data

Lo

Ph



- Two-way mapping between RDF terms and numerical identifiers



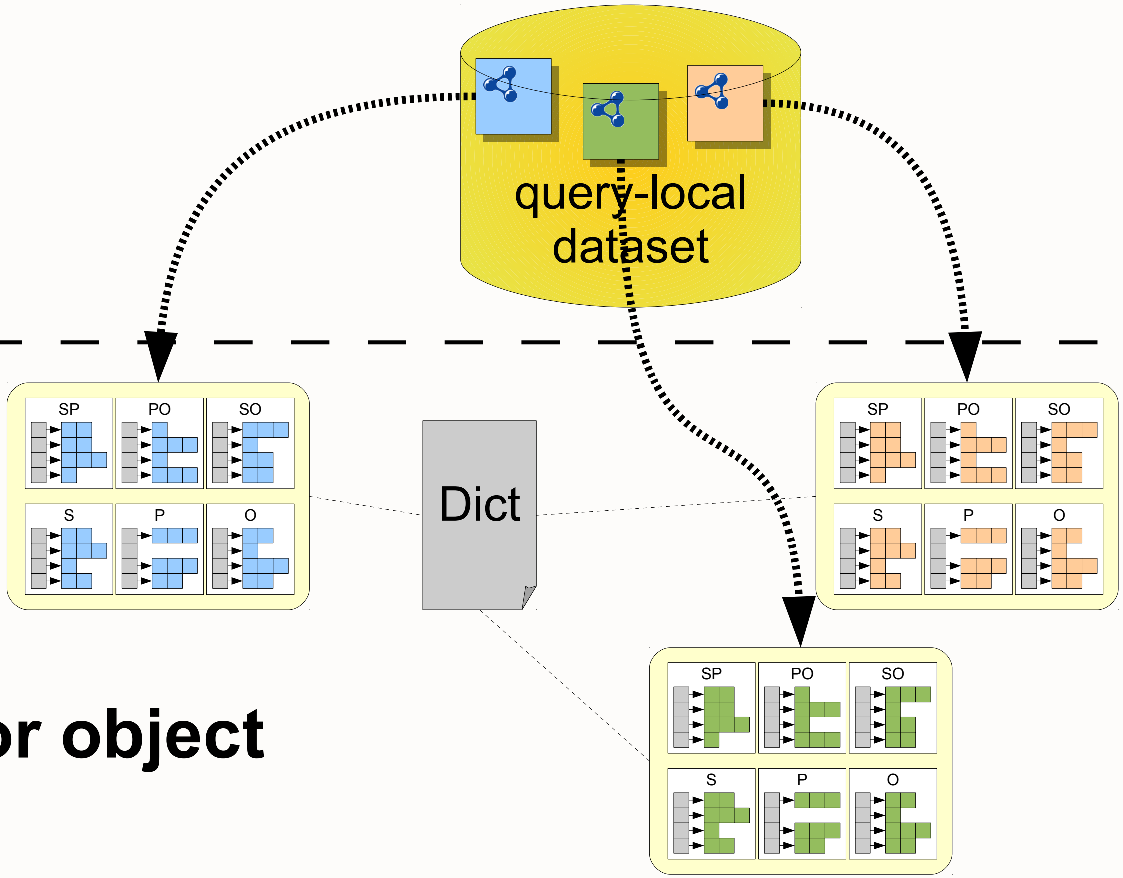
- **6 hash tables:**
  - Each hash table contains all ID-encoded triples
  - Efficient support for all types of triple patterns

*Similar to Harth and Decker, 2005

Individual Indexing

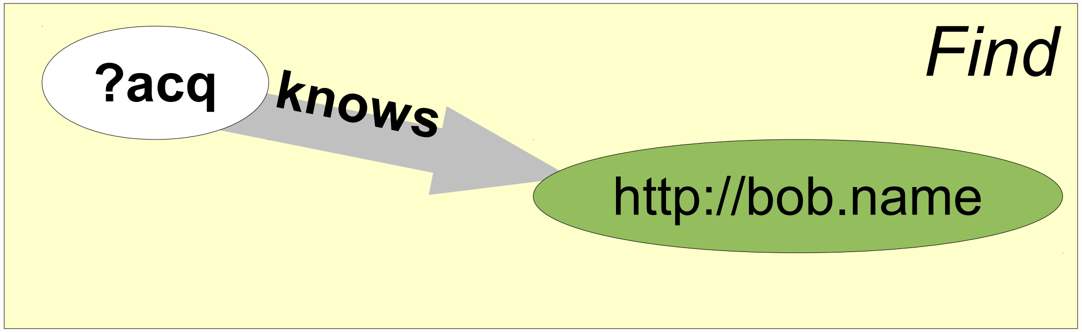
Logical representation

Physical representation

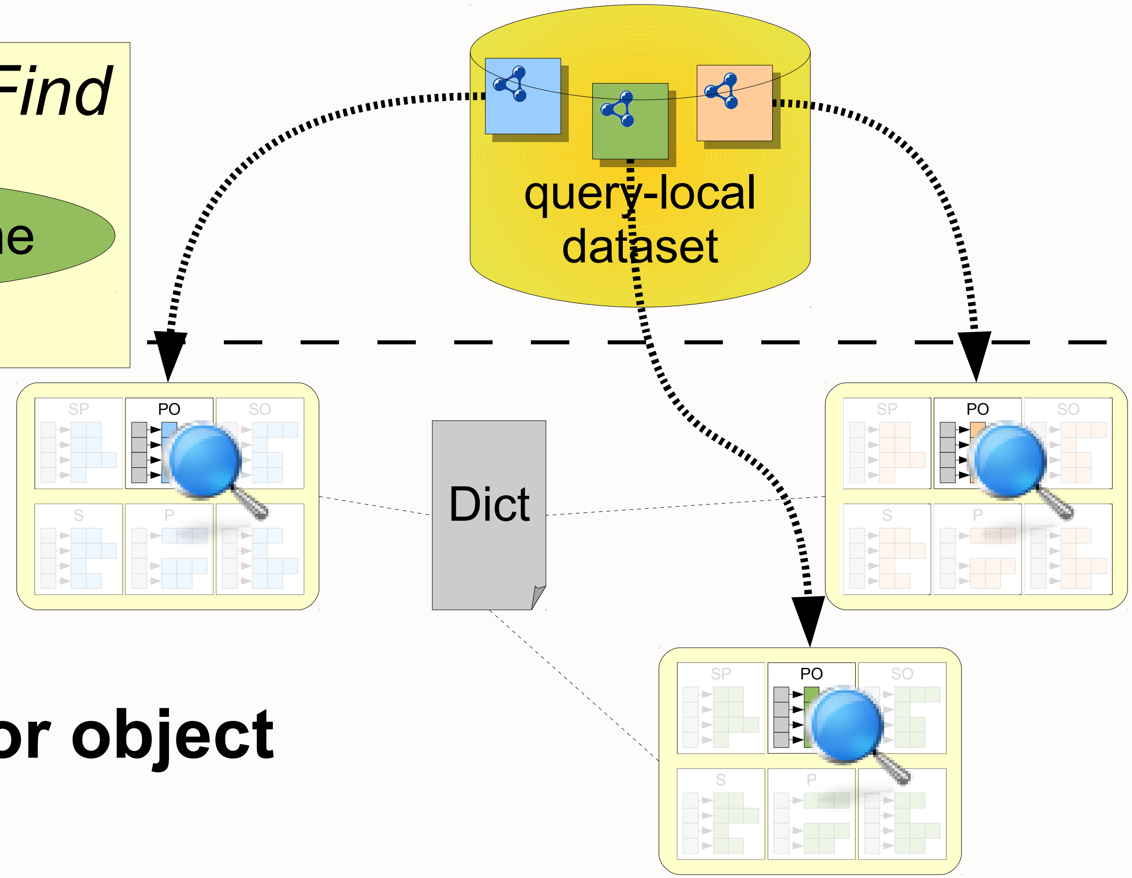


- **Idea: Index each descriptor object separately**
- **Implementation of the four operations:**
  - *Add*, *Remove*, and *Replace* are straightforward
  - *Find* requires iterating over all indexes

Individual Indexing



Physical representation



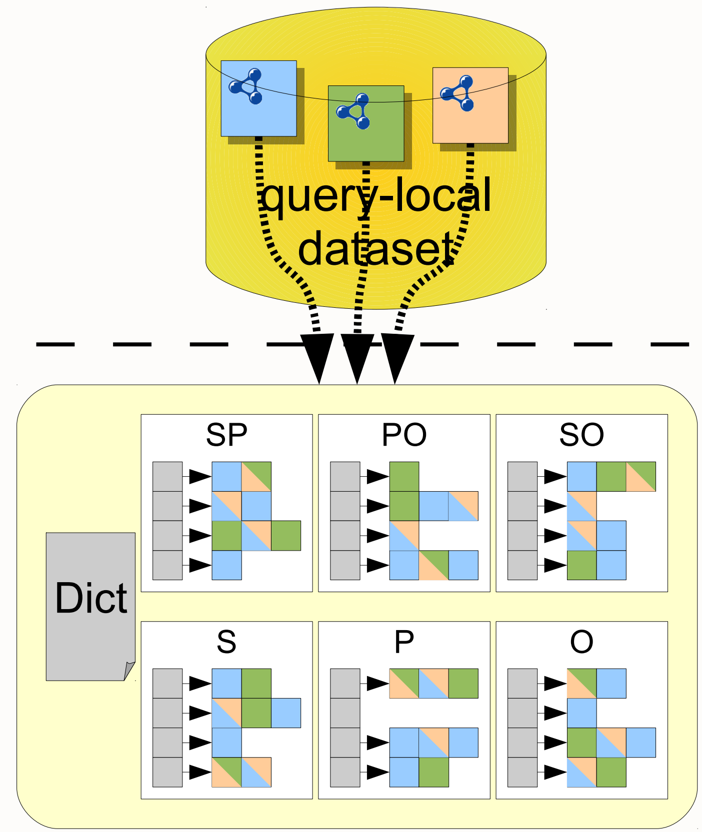
- **Idea: Index each descriptor object separately**
- **Implementation of the four operations:**
  - *Add*, *Remove*, and *Replace* are straightforward
  - *Find* requires iterating over all indexes

Combined Indexing

Logical representation

Physical representation

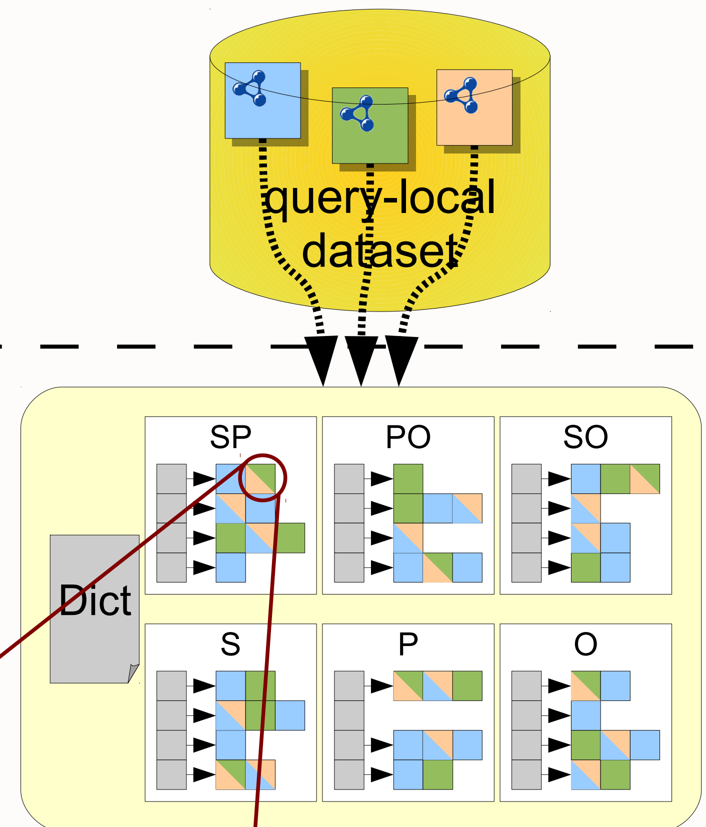
- **Idea: Use a single index for all descriptor objects**
  - *src* – maps each triple to a set of descriptor object IDs



Combined Indexing

Logical representation

Physical representation



- **Idea: Use a single index for all descriptor objects**
  - *src* – maps each triple to a set of descriptor object IDs

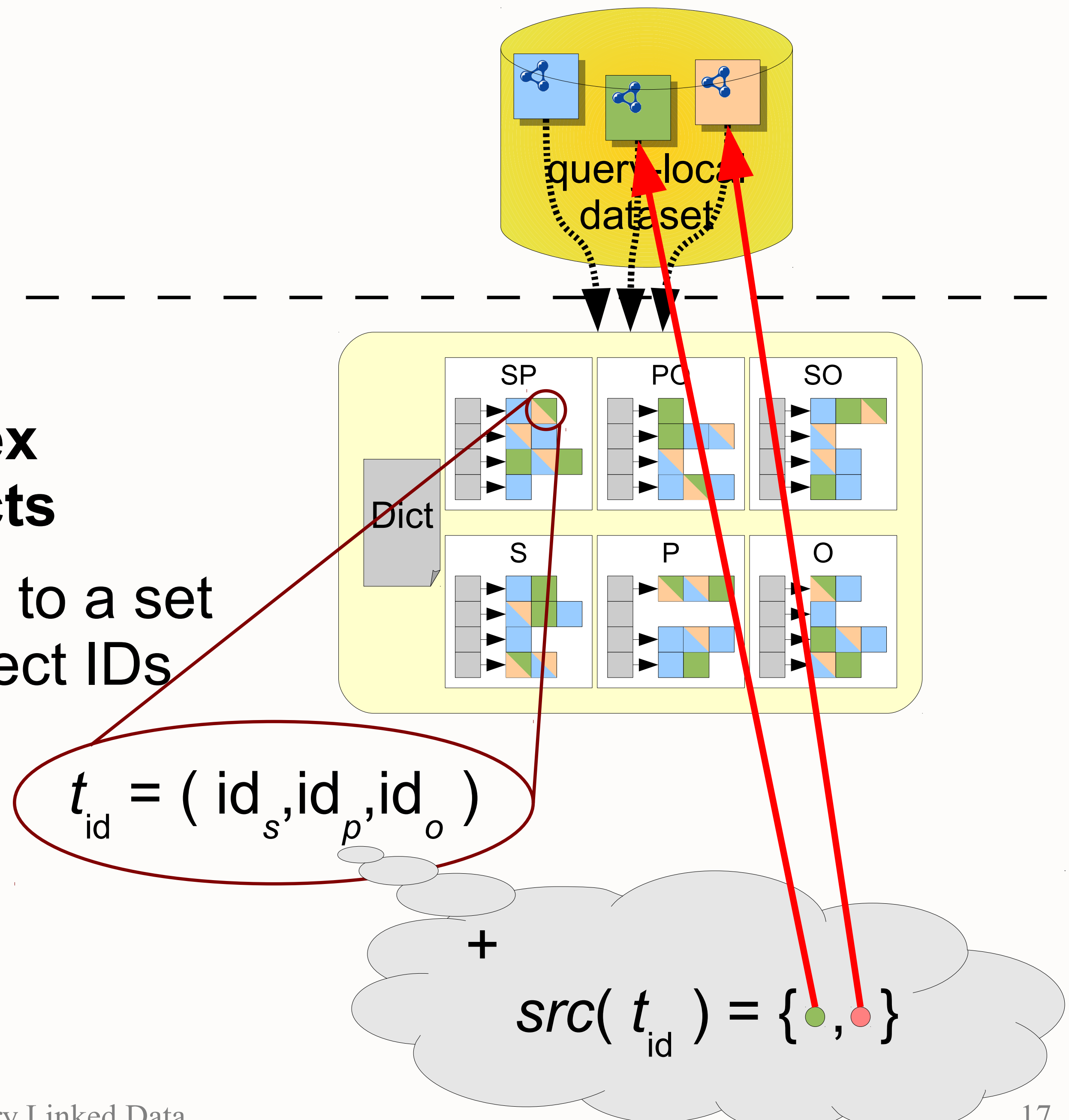
$$t_{id} = (id_s, id_p, id_o)$$

Combined Indexing

Logical representation

Physical representation

- **Idea: Use a single index for all descriptor objects**
  - *src* – maps each triple to a set of descriptor object IDs

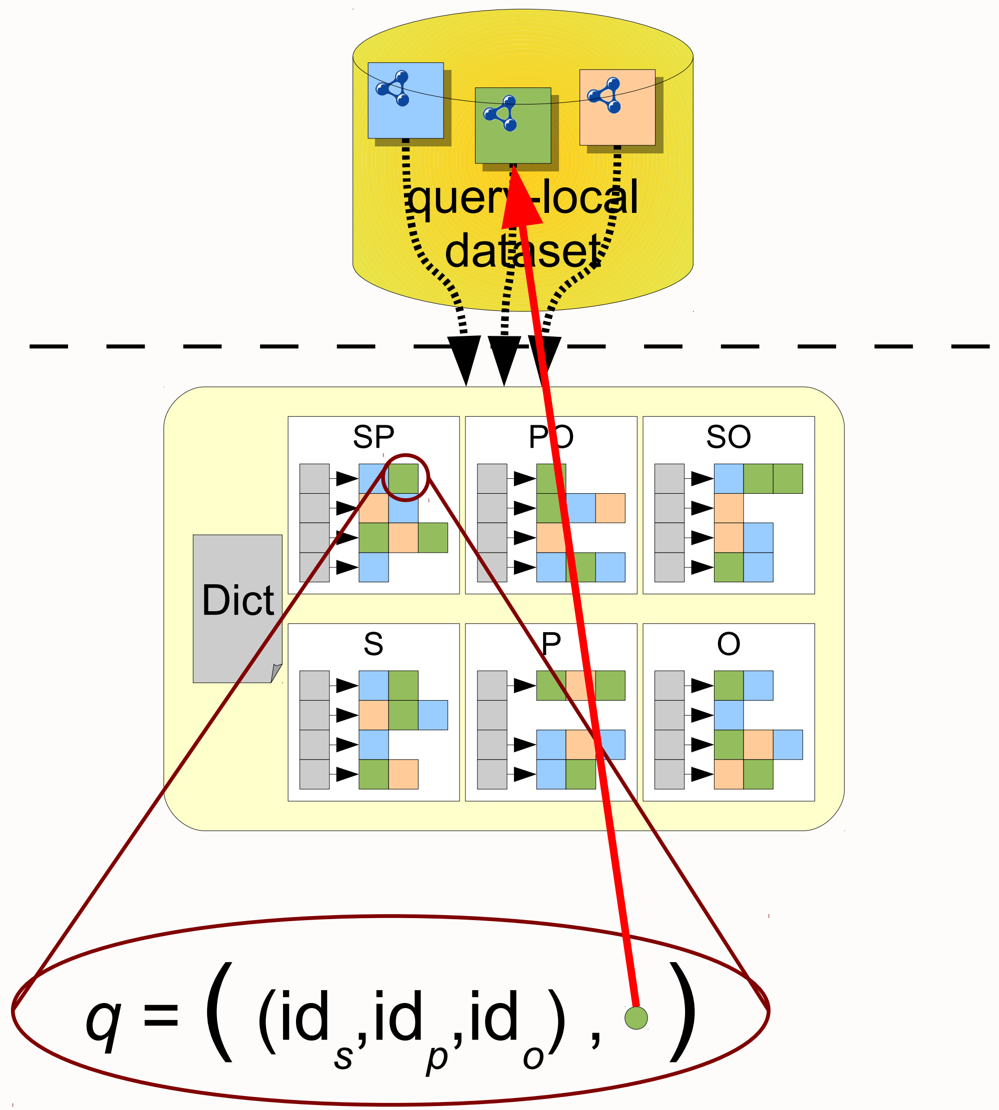


Quad Indexing

Logical representation

Physical representation

- **Idea: Use a single quad index for all descriptor objects**
  - *quad* = ID-encoded triple + descriptor object ID



Outline

1. Requirements + Existing Work

2. Data Structures

3. Evaluation

Experiment Setup

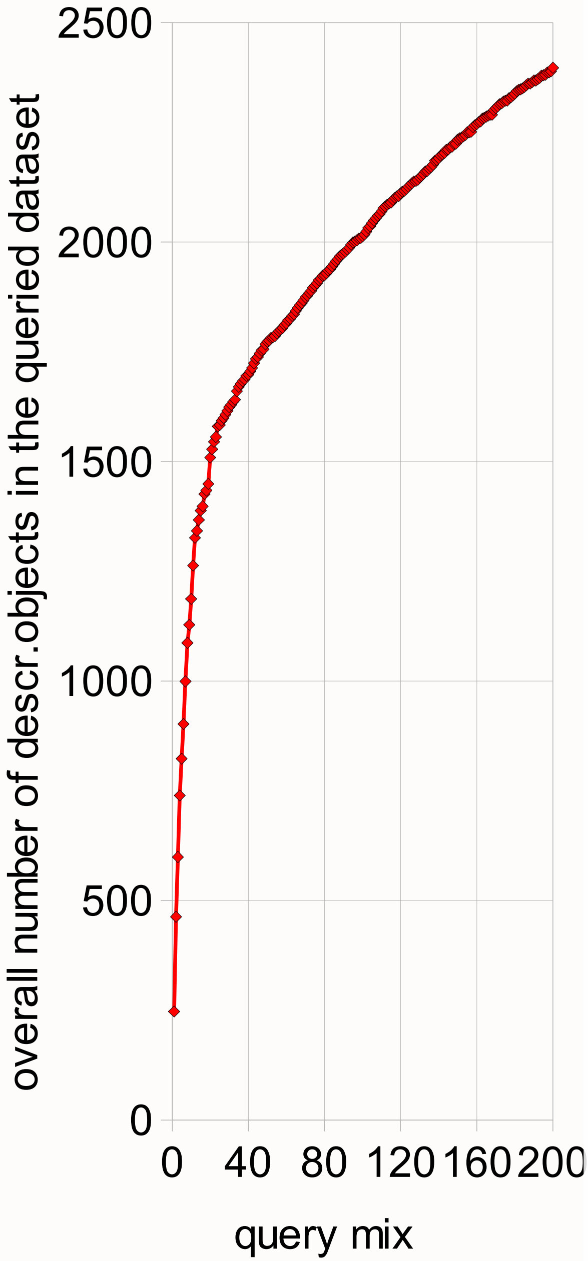
Does this affect the overall execution time for link traversal based query executions ?

Experiment Setup

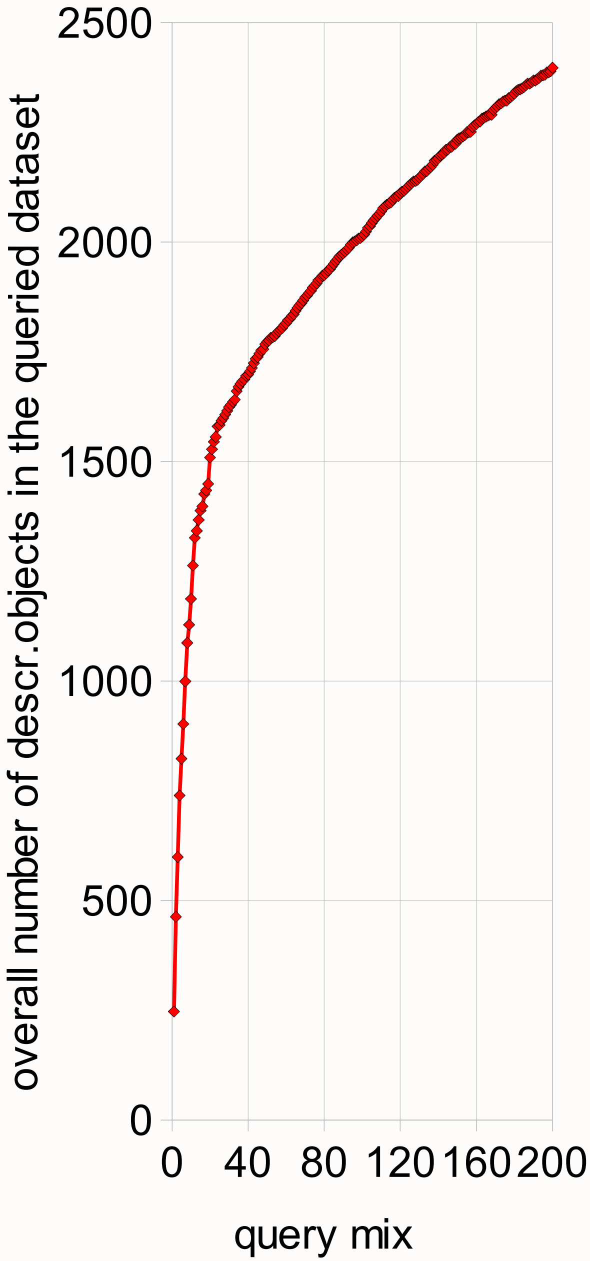
Does this affect the overall execution time for link traversal based query executions ?

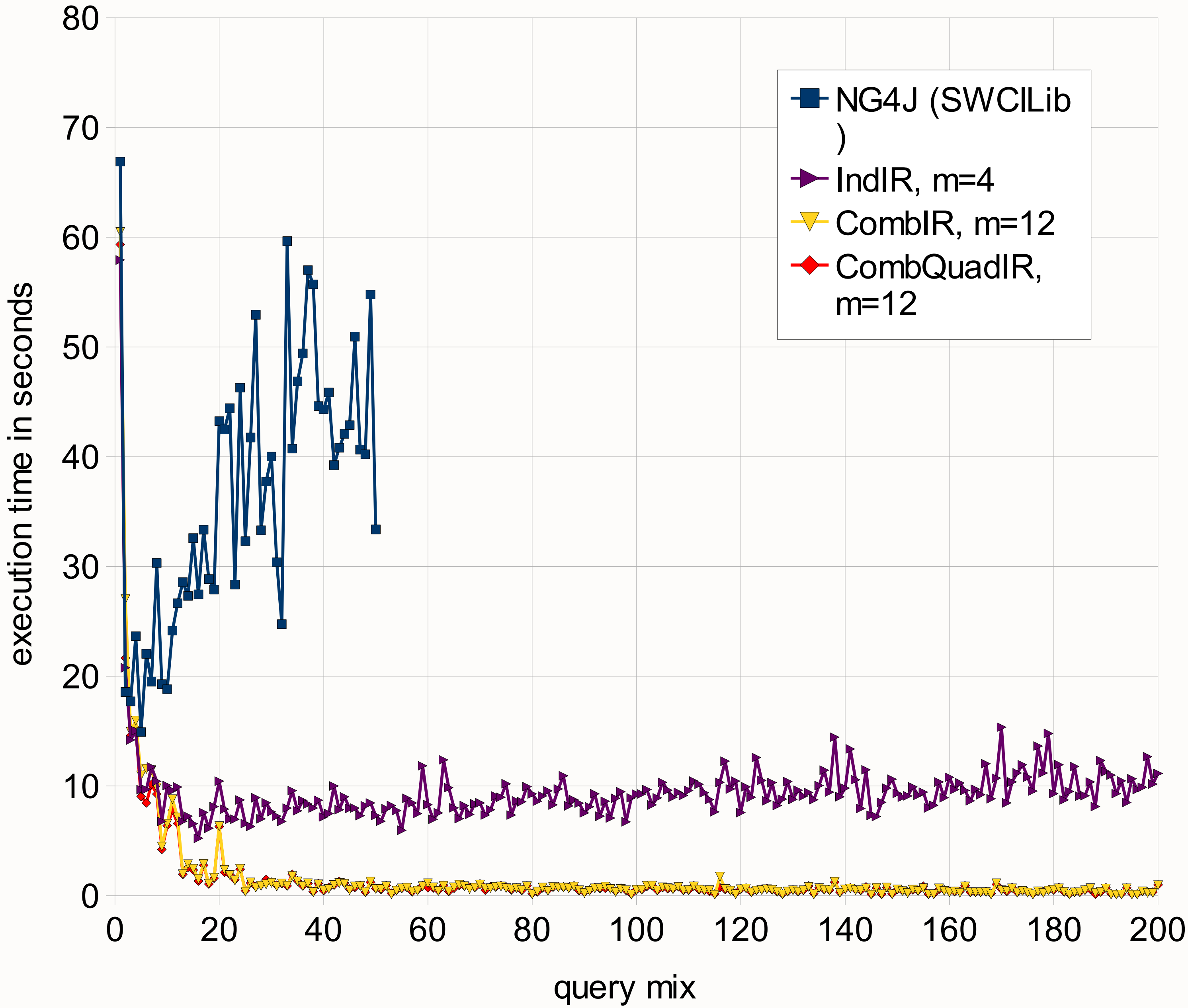
- **Simulation of the Web of Data**
  - Linked Data server publishes BSBM dataset (scal. factor: 50)
  - Adjusted BSBM queries link to the simulation server
- **Experiment:**
  - Sequence of 200 query mixes
  - Reuse of the query-local dataset for the whole sequence
  - IndIR, CombIR, and QuadIR (as presented), engine: SQUIN
  - `NamedGraphSetImpl` (NG4J/Jena), engine: SemWeb Client

Execution Time



Execution Time



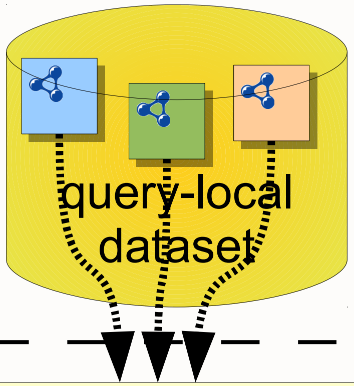


Summary

- **Three hash index based data structures:**
  - Individually indexing
  - Combined indexing
  - Quad indexing
- **Findings:**
  - A single index improves query performance significantly
  - Smaller load times with quads
- **Also for other use cases of ad hoc storing of Linked Data**
  - Consecutively retrieved from remote sources
  - Used for immediate local processing

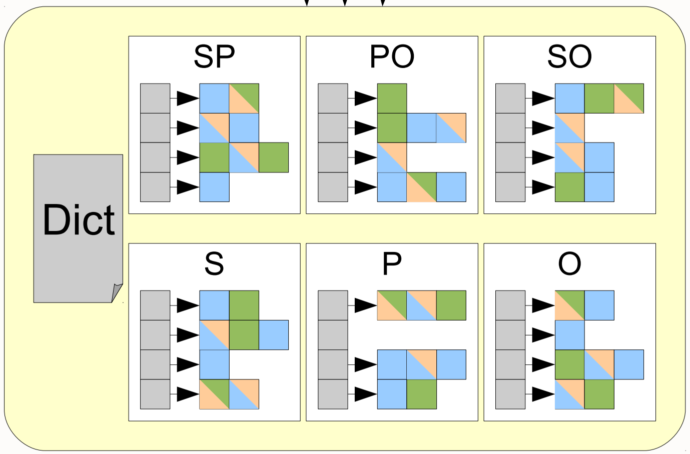
Backup Slides

Combined Indexing



Logical representation

Physical representation



- **Idea: Use a single index for all descriptor objects**
  - *src* – maps each triple to a set of descriptor object IDs
- **Implementation of the four operations requires:**
  - *status* – maps each descriptor object ID to a status (BeingIndexed, Indexed, ToBeRemoved, BeingRemoved)
  - *Find* reports a triple *t* only if: $\exists d \in src(t) : status(d) = \text{Indexed}$

Implementation

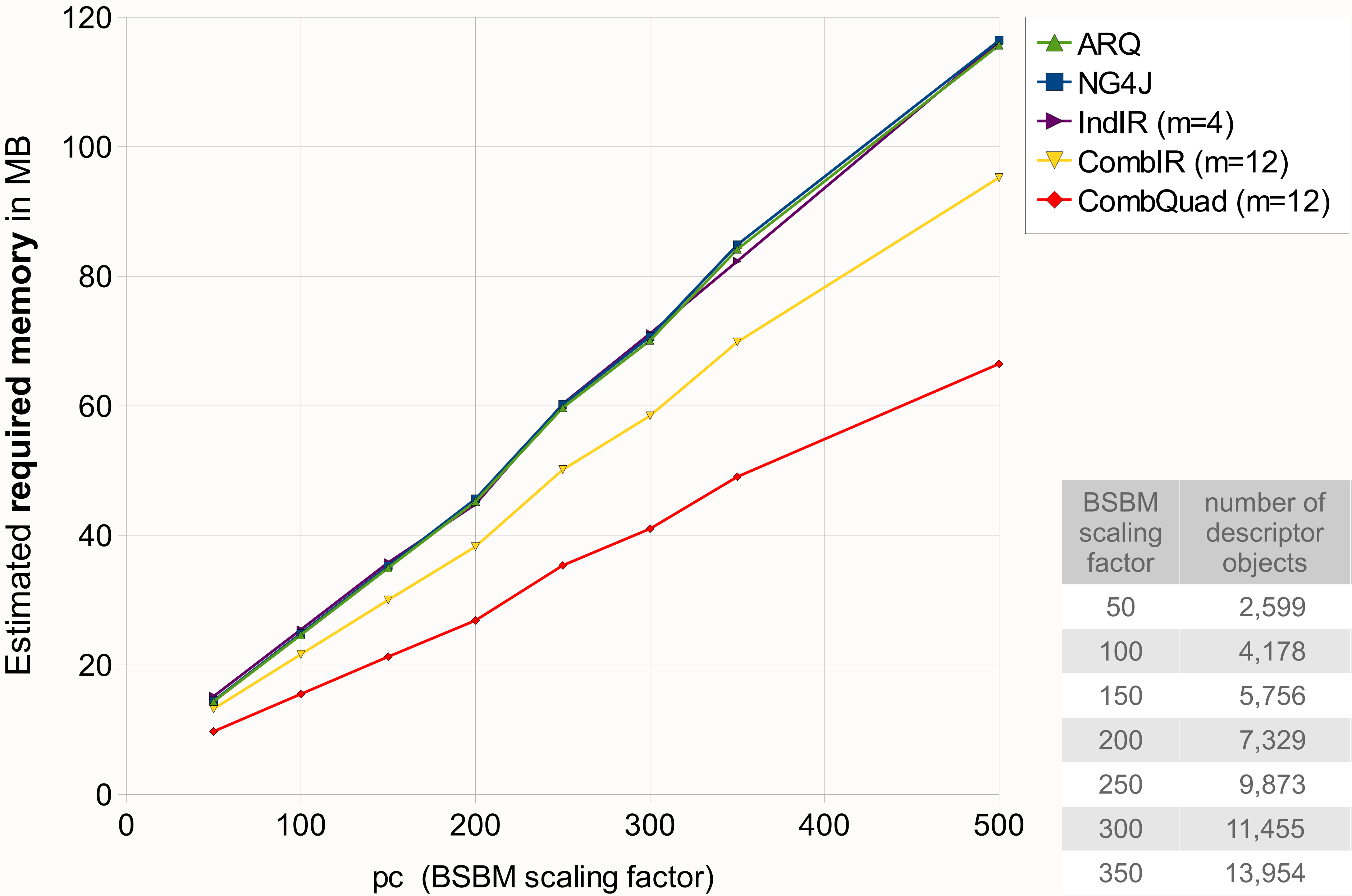
- **Available as Free Software** (http://squin.org)
- **Hash tables with $n = 2^m$ buckets**
- **Hash functions:**
  - $h_S(i_s, i_p, i_o) = i_s \,\&\, \text{bitmask}^{[m]}$
  - $h_{SP}(i_s, i_p, i_o) = (i_s \bullet i_p) \,\&\, \text{bitmask}^{[m]}$
  - etc.
- **Which m ?***
  - $m = 4$ for the individual indexes
  - $m = 12$ for the combined index
  - $m = 12$ for the quad index

*see paper

Experiment Setup

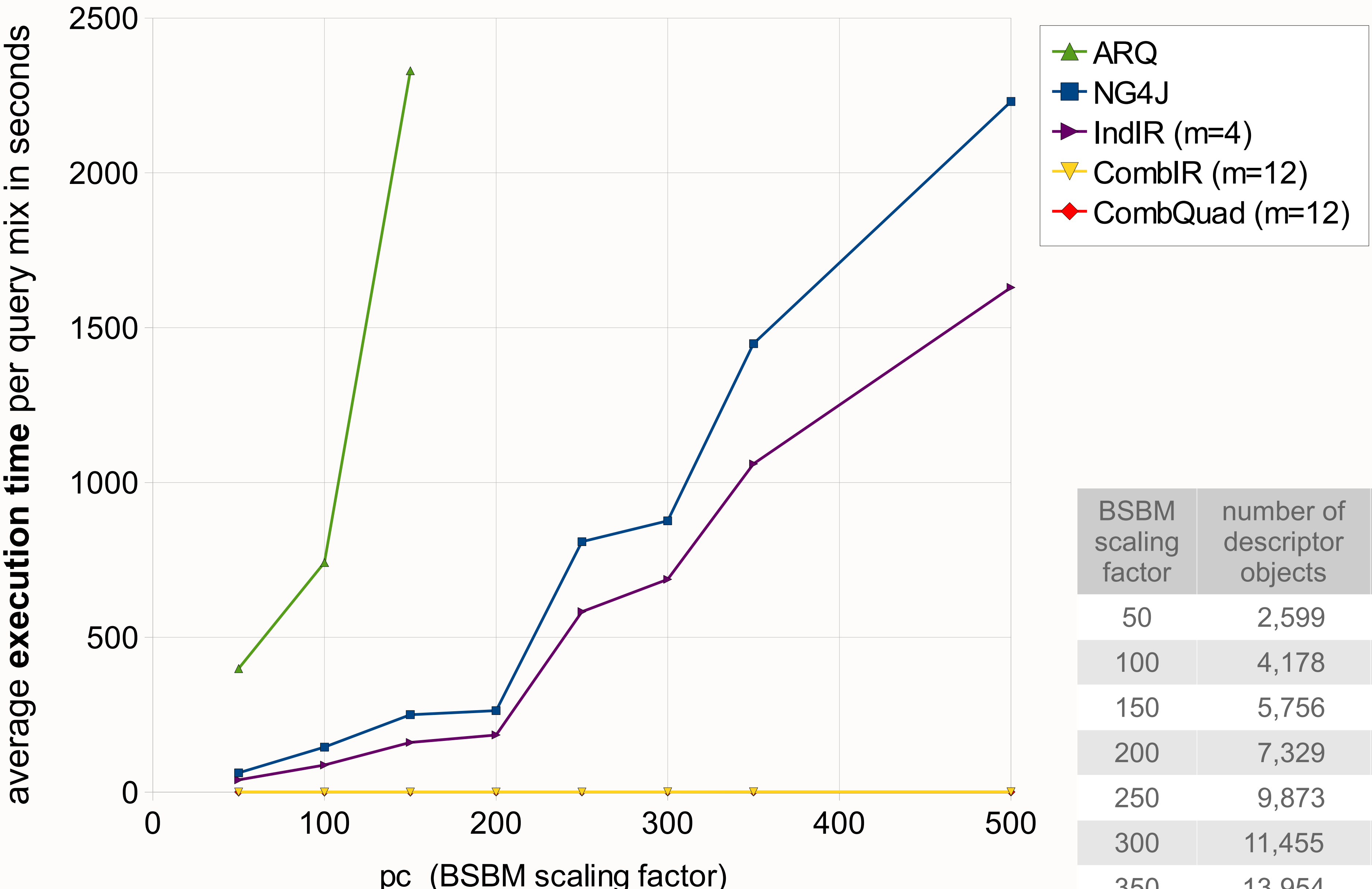
- **Comparison without link traversal based query execution**
- **Compared data structures:**
  - Our implementation of IndIR, CombIR, CombQuadIR
  - `NamedGraphSetImpl` in NG4J (Jena)
  - `DatasetGraphMap` in ARQ (Jena)
- **Berlin SPARQL Benchmark (BSBM)**
  - BSBM datasets partitioned into query-local datasets
  - BSBM (v2.0) query mixes executed over these datasets

Required Memory



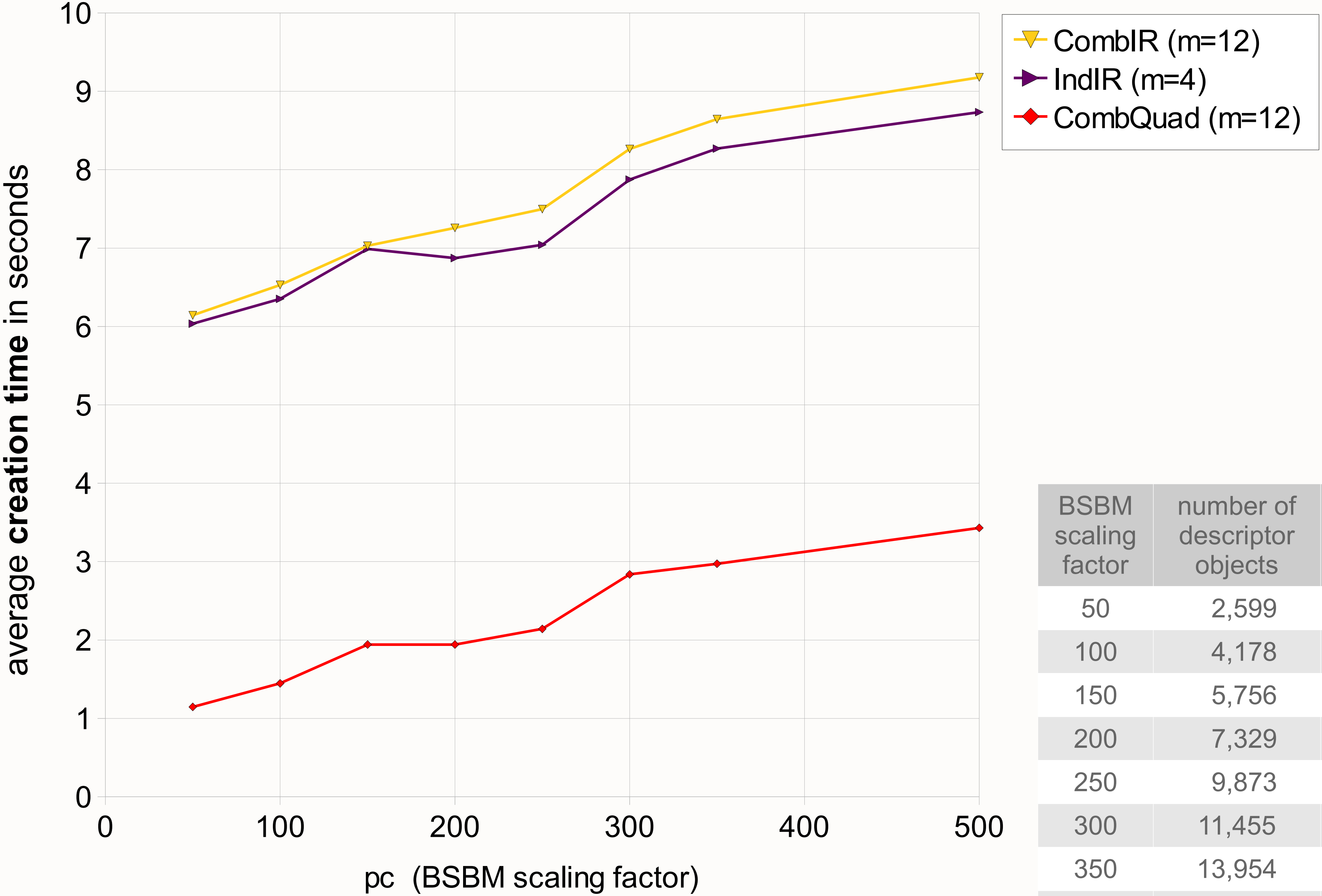
BSBM scaling factor	number of descriptor objects	overall number of triples
50	2,599	22,616
100	4,178	40,133
150	5,756	57,524
200	7,329	75,062
250	9,873	97,613
300	11,455	115,217
350	13,954	137,567
500	18,687	190,502

Execution Time



BSBM scaling factor	number of descriptor objects	overall number of triples
50	2,599	22,616
100	4,178	40,133
150	5,756	57,524
200	7,329	75,062
250	9,873	97,613
300	11,455	115,217
350	13,954	137,567
500	18,687	190,502

Load Time



BSBM scaling factor	number of descriptor objects	overall number of triples
50	2,599	22,616
100	4,178	40,133
150	5,756	57,524
200	7,329	75,062
250	9,873	97,613
300	11,455	115,217
350	13,954	137,567
500	18,687	190,502

Execution Time