Optimising Linked Data Queries in the Presence of Co-reference

Xin Wang, Thanassis Tiropanis, and Hugh C. Davis
University of Southampton
UK

Co-reference in Linked Data

- Co-reference is the phenomenon that a resource is identified by multiple URIs.
- Common in the Linked Data cloud.
  - E.g. Tim Berners-Lee has many different URIs
- Co-referent URIs are presented by *owl:sameAs*
  - E.g. :Tim owl:sameAs :TBL

Why to include co-reference

- Improve the connectivity of Linked Data.
- More results.
- Effective co-reference identification and existing *owl:sameAs* statements.
- Not well supported by existing distributed engines.

Challenges

- A naïve approach has to examine the Cartesian product of the co-reference of every concrete value in a query
- Hard to collect statistics of co-reference
  - Not covered by VoID
- More intermediate results, more optimization pressure

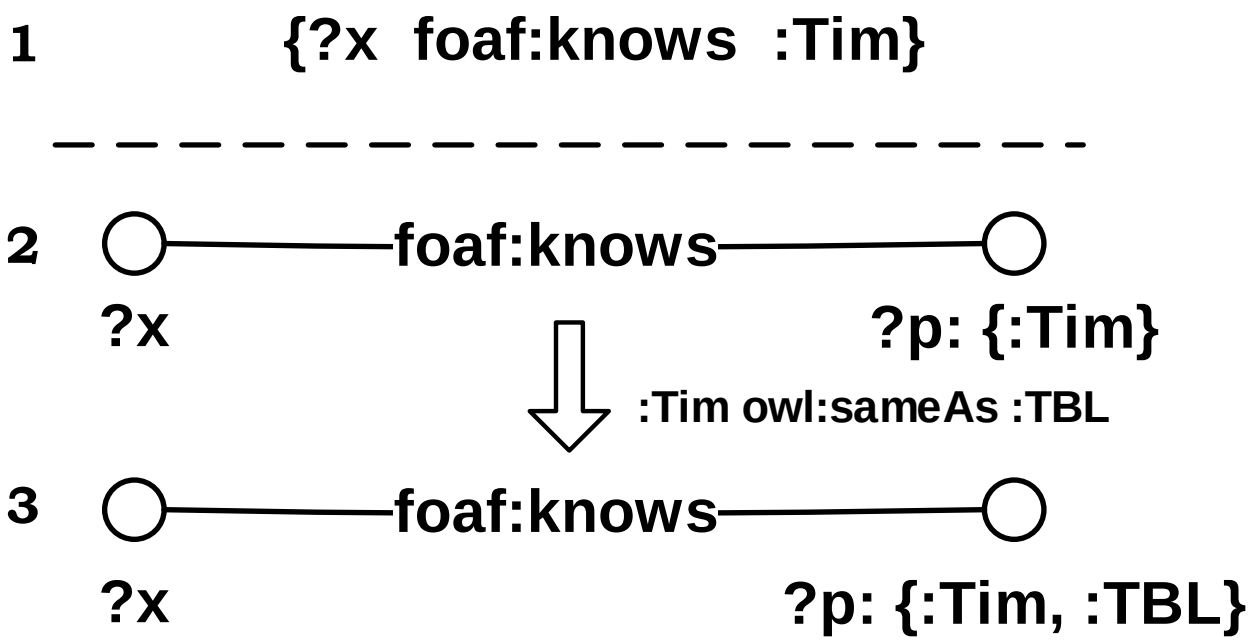
Contributions

- Virtual Graph, merging co-referent URIs into one node.
- Exploiting runtime statistics.
- Algorithm Ψ (**P**arallel **S**ub-graph **I**dentification), breaks a query into sub-queries that can be executed in parallel.
- An co-reference extension of BSBM.

Virtual Graph

- Based on the observation that a concrete node with co-reference can be regarded as a variable having multiple values.
- One query contains all co-reference.
- Enables the optimizer to find the optimal execution plan w.r.t all co-reference.

Virtual Graph



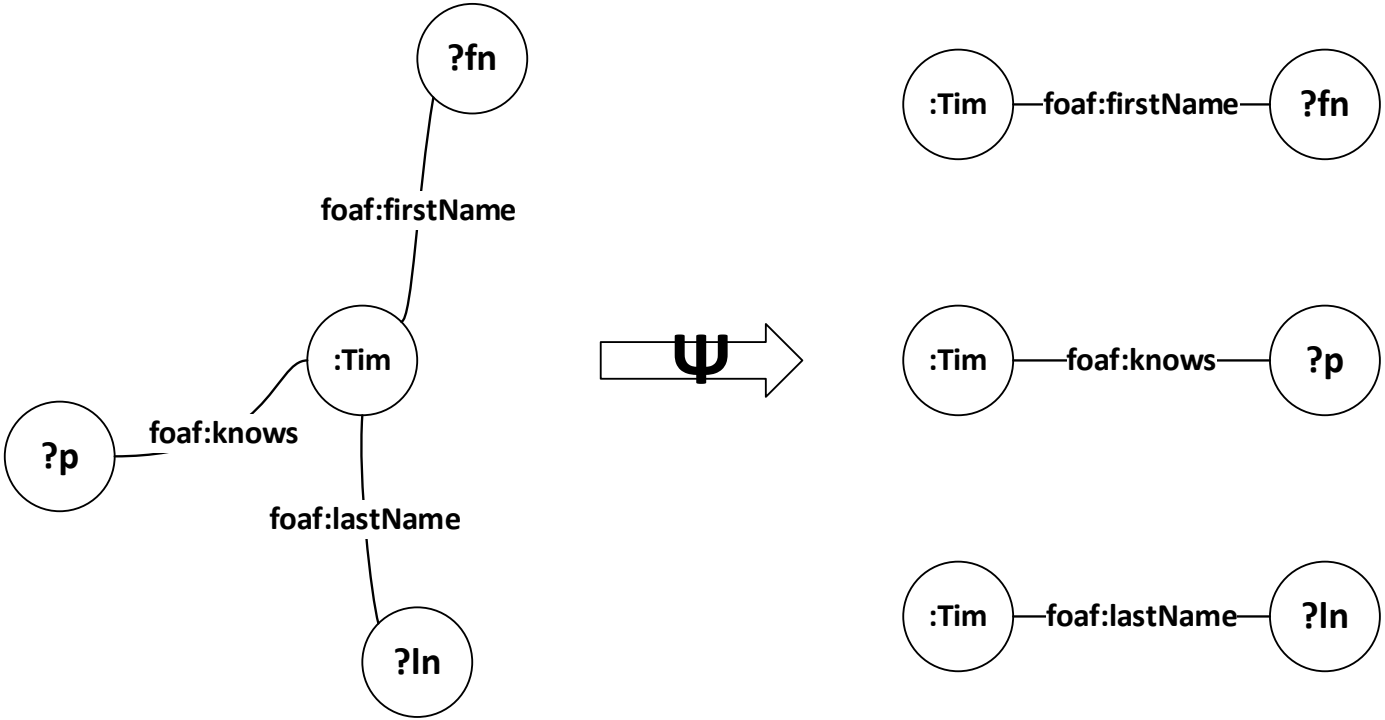
Ψ: Parallel Sub-Query Identification

- Based on the observation that two triple patterns connected by a concrete node can always be executed in parallel.
- Increases parallelization WITHOUT increasing network traffic.

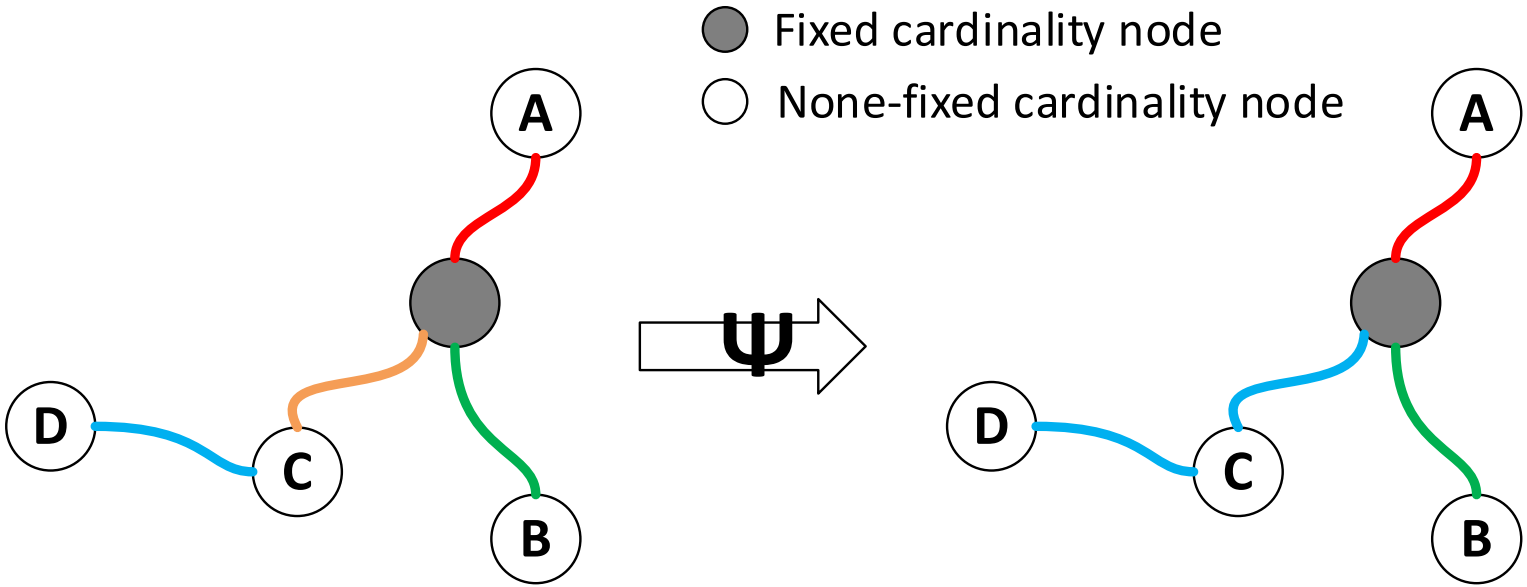
Ψ: Parallel Sub-Query Identification

- A node whose binding size doesn’t change (much) during execution is called a fixed cardinality node.
- A query can be disconnect at fixed cardinality nodes.

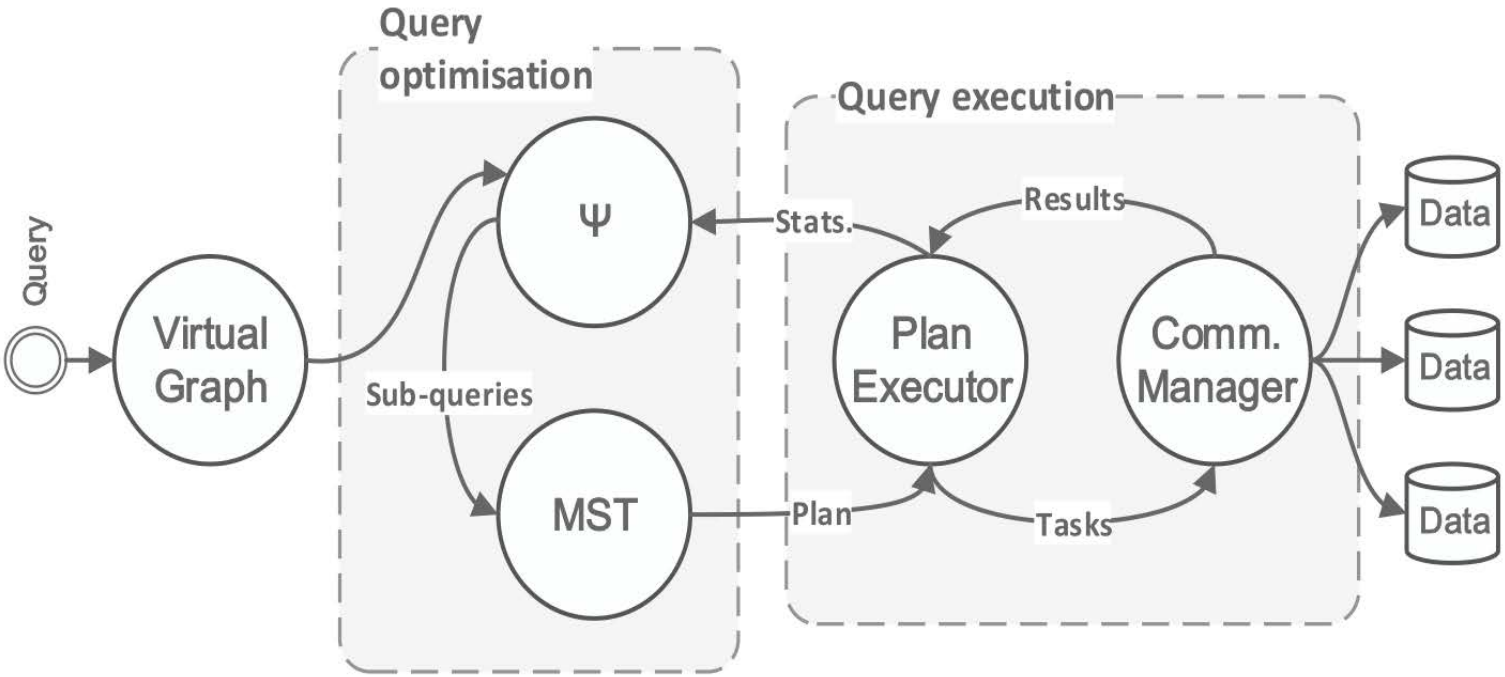
Ψ: Parallel Sub-Query Identification



Ψ: Parallel Sub-Query Identification

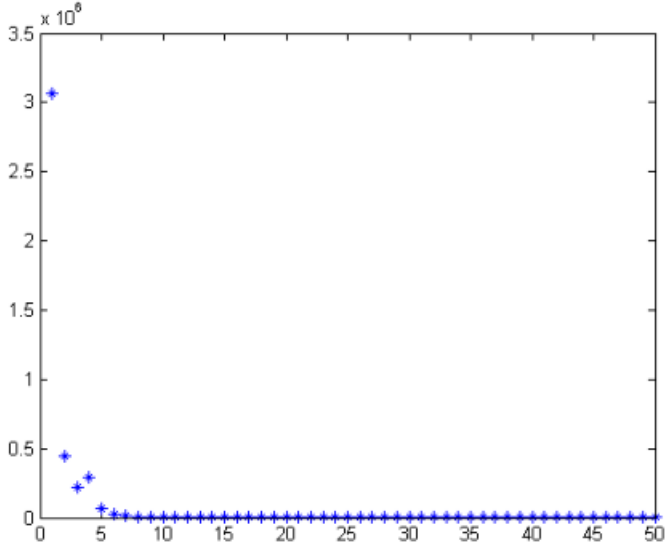


Overview of LHD-d



Evaluation settings

- Simulate co-reference in BSBM based on the statistics of BTC2011 dataset.
  - $p(x) = \alpha x^{-\beta}$ where $\beta = 2.528$



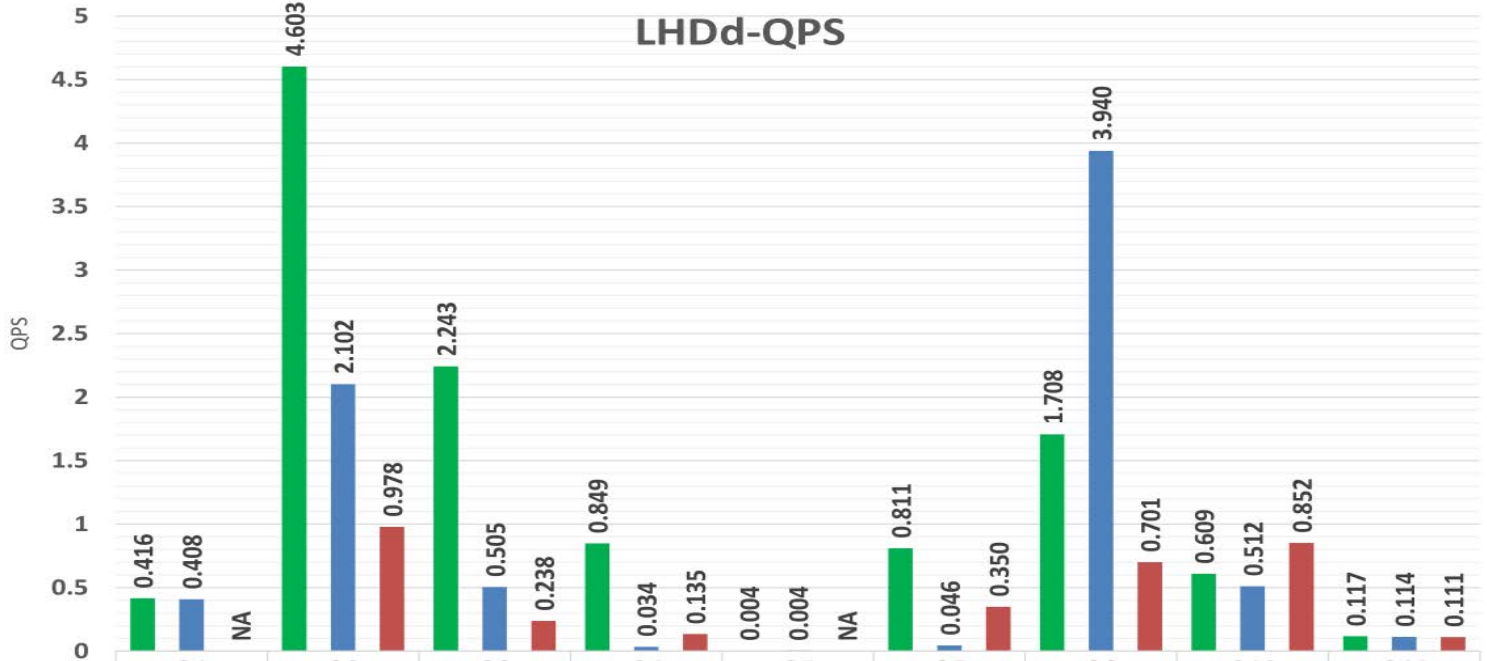
Evaluation of Ψ (no co-ref)

Comparing LHD-d to:

- LHD (referred to as LHD-s here)
  - X. Wang, T. Tiropanis, and H. C. Davis, “LHD: Optimising Linked Data query processing using parallelisation,” LDOW 2013.
- FedX
  - A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt, “FedX: Optimization Techniques for Federated Query Processing on Linked Data,” ISWC 2011.

Not taking co-reference into account

Evaluation of Ψ (no co-ref)



	Q1	Q2	Q3	Q4	Q5	Q7	Q8	Q10	Q11
LHDd	0.4163	4.6033	2.2427	0.8486	0.0042	0.8105	1.7078	0.6088	0.1175
LHDs	0.4084	2.1023	0.5047	0.0345	0.0045	0.0456	3.9397	0.5116	0.1137
FedX	0	0.9784	0.2382	0.1347	0	0.3499	0.7012	0.8520	0.1107

Evaluation of Virtual Graph

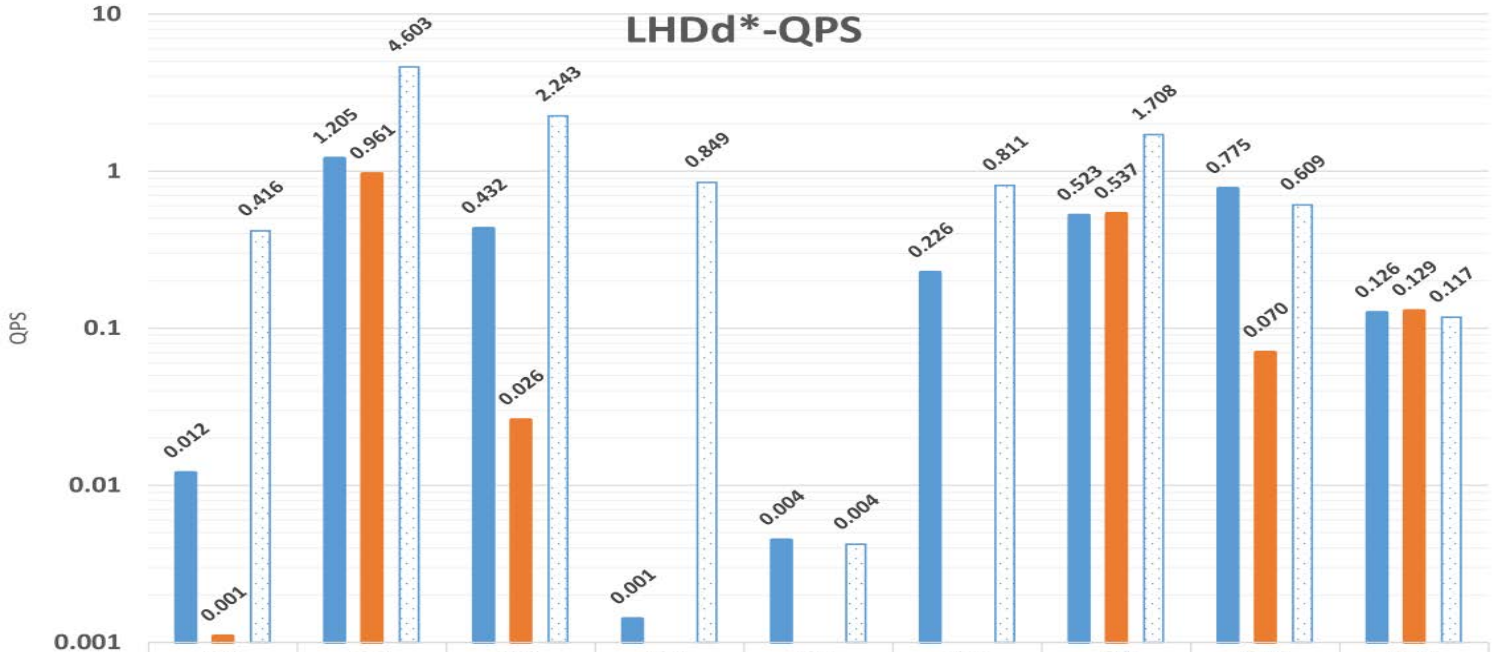
- LHD-d*, LHD-d with Virtual Graph.
- Naïve approach, LHD-d without Virtual Graph, co-referent queries are processed separately.
- LHD-d, without Virtual Graph, not taking co-reference into account.

Impact of co-reference

Table 1: Result sizes of co-reference

	Q1	Q2	Q3	Q4	Q5	Q7	Q8	Q10	Q11
LHD-d*	7397	103	23	65510	14499	1579	101	32	10
Naïve	7397	103	23	NA	NA	NA	101	32	10
LHD-d	53	29	8	29	14499	63	21	12	10

Evaluation of **Virtual Graph**



	Q1	Q2	Q3	Q4	Q5	Q7	Q8	Q10	Q11
LHDd*	0.0120	1.2048	0.4322	0.0014	0.0045	0.2262	0.5230	0.7748	0.1257
Naive	0.0011	0.9607	0.0261	0	0	0	0.5369	0.0702	0.1291
LHDd	0.4163	4.6033	2.2427	0.8486	0.0042	0.8105	1.7078	0.6088	0.1175