On Data Placement Strategies in Distributed RDF Stores

Int. Workshop on Semantic Big Data (SBD 2017)

Daniel Janke, Steffen Staab, Matthias Thimm

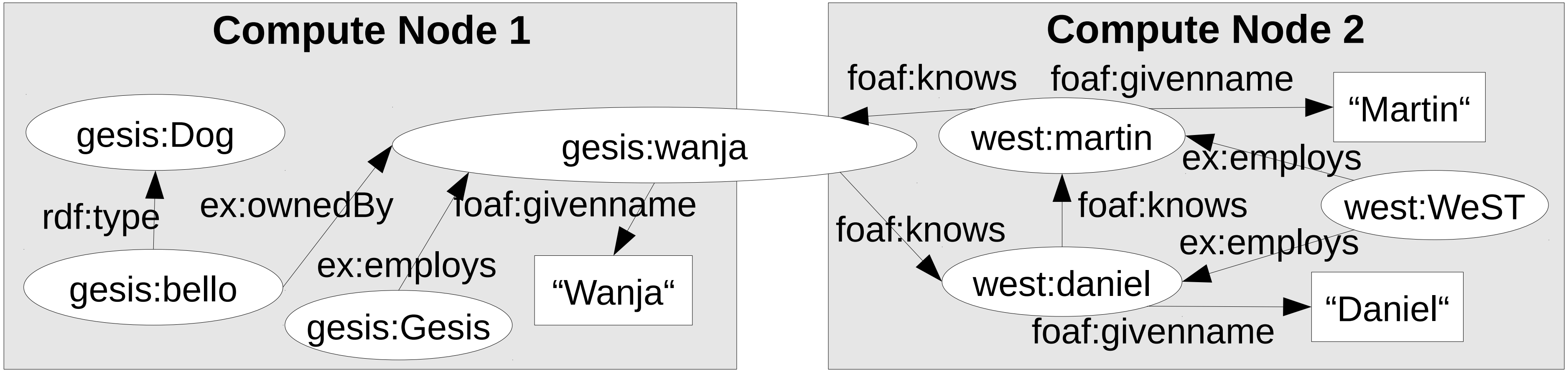
19.05.2017

WeST People and Knowledge Networks

Institute for Web Science and Technologies · University of Koblenz-Landau, Germany

Distributed RDF Stores

- Requirement for trillion triples stores arose in the last years
- Scalable RDF stores in the cloud

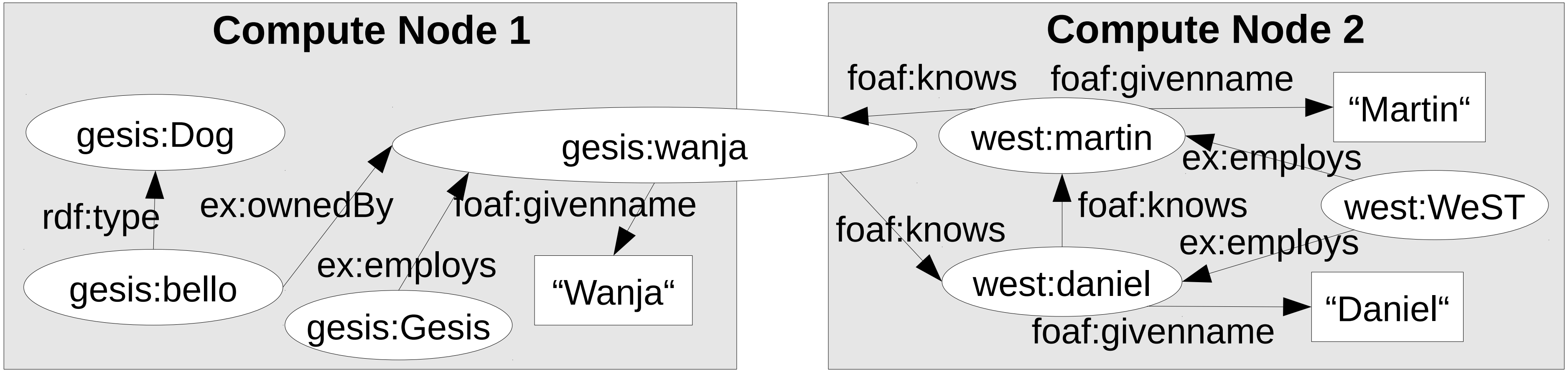


Challenges:

- Data placement strategies
- Distributed query processing
- Handling failures of compute nodes

Distributed RDF Stores

- Requirement for trillion triples stores arose in the last years
- Scalable RDF stores in the cloud

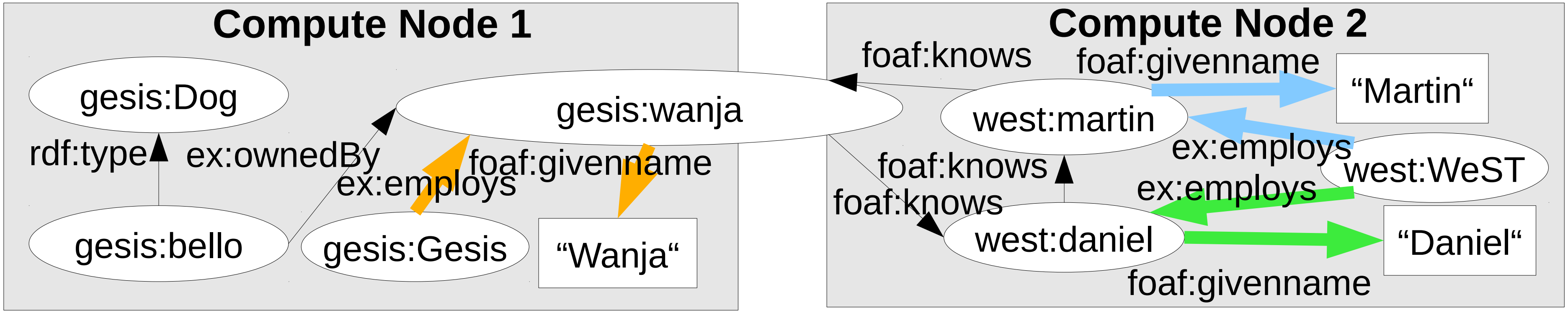


Challenges:

- Data placement strategies ← Focus of our research
- Distributed query processing
- Handling failures of compute nodes

Data Placement Strategies and Scalability

```
SELECT ?org ?name WHERE
    {?org ex:employs ?pers . ?pers foaf:givenname ?name}
```



Horizontal containment

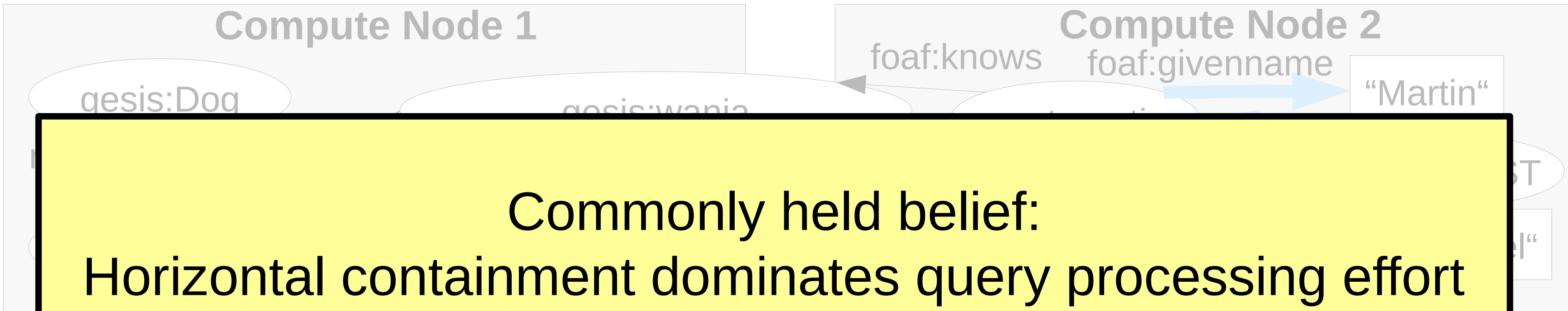
- Computation of individual query results on local data
- Indicator for robust query processing when scaling horizontally

Vertical parallelization

- Parallel computation of different query results on different compute nodes
- Indicator for query processing scaling with growing result set sizes when scaling horizontally

Data Placement Strategies and Scalability

```
SELECT ?org ?name WHERE
    {?org ex:employs ?pers . ?pers foaf:givenname ?name}
```



Commonly held belief:
Horizontal containment dominates query processing effort
(cf. [Huang2011SSQ, Lee2013EDP, Zhang2013ETS, …])

- Computation of individual query results on local data
- Indicator for robust query processing when scaling horizontally

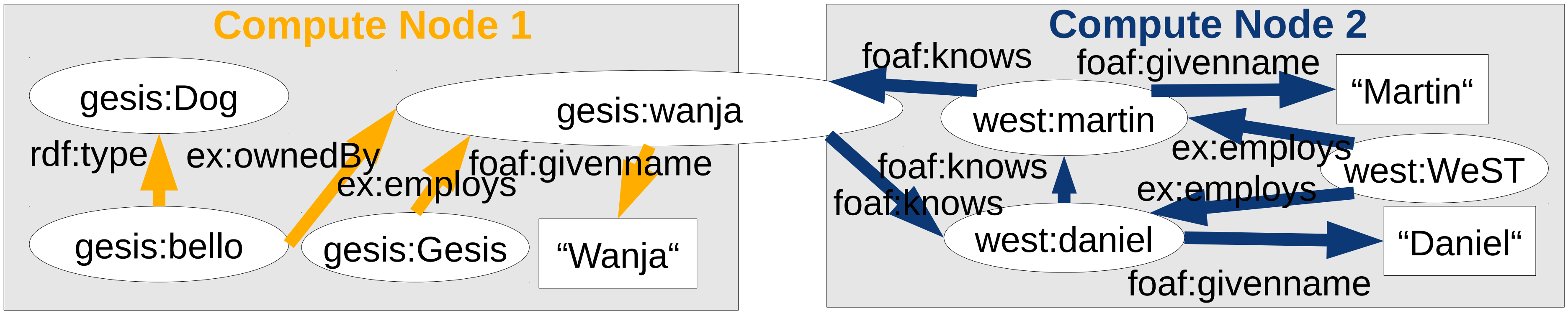
Vertical parallelization

- Parallel computation of different query results on different compute nodes
- Indicator for query processing scaling with growing result set sizes when scaling horizontally

Outline

1) Data Placement Strategies
2) Benchmark methodology showing the interdependencies of data placement strategies and query processing
3) Analysis indicating that vertical parallelization may dominate horizontal containment
4) Conclusion

Graph Cover



Graph cover

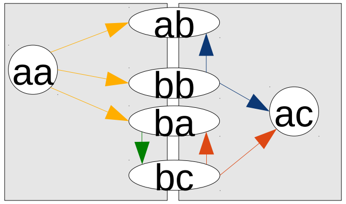
Assignment of each triple to at least one compute node

Graph chunk

Set of triples assigned to a single compute node

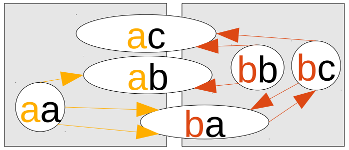
Common Graph Cover Strategies

Hash cover [e.g. Harth2007YAF]

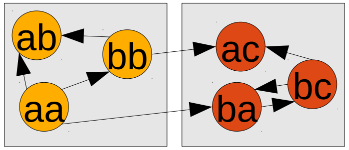


- Triple placement bases on subject hash modulo number of compute nodes

Hierarchical cover [Lee2013SQO]



- Triple placement bases on hash of subject IRI prefixes

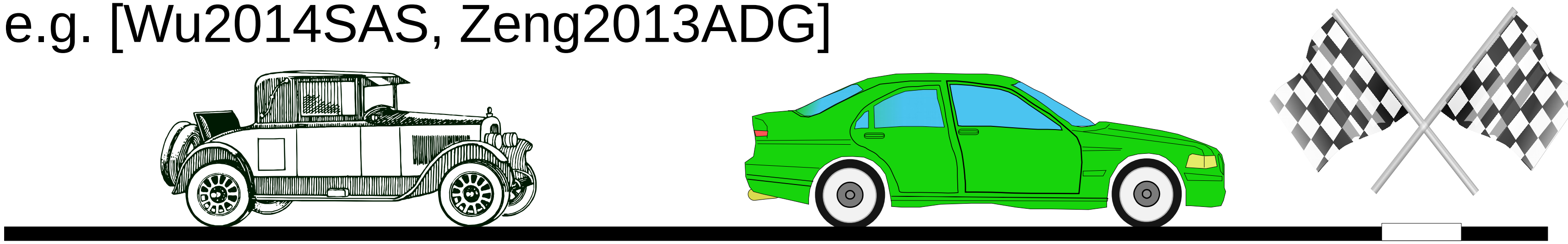


Minimal edge-cut cover [Karypis1998AFA]

- Assign vertices (subjects and objects) to partitions such that
  - Number of edges between vertices of different partitions is minimized and
  - Each partition contains approximately $\frac{|V_G|}{|C|}$ vertices

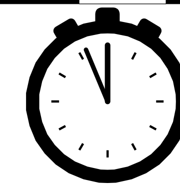
Common Evaluation Strategies

1) Evaluations of graph cover strategies using different databases
   => other components might bias evaluation results
   e.g. [Wu2014SAS, Zeng2013ADG]



Car 1 using fuel A

Car 2 using fuel B

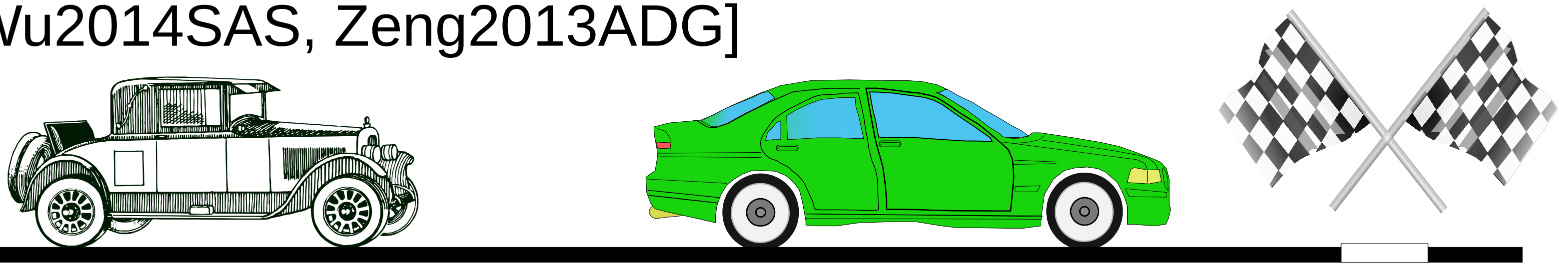


Does fuel A or B allow for a higher speed?

Images from https://openclipart.org

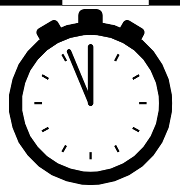
Common Evaluation Strategies

1) Evaluations of graph cover strategies using different databases
=> other components might bias evaluation results
e.g. [Wu2014SAS, Zeng2013ADG]



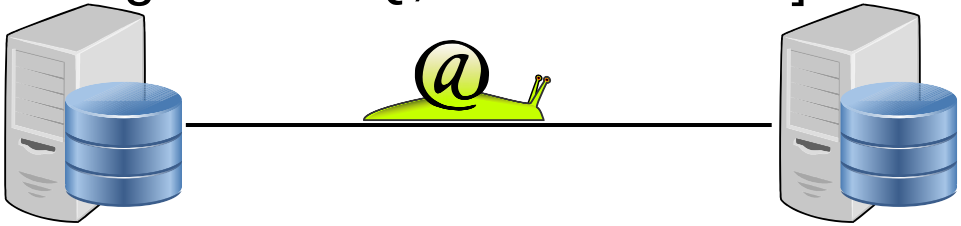
Car 1 using fuel A

Car 2 using fuel B



Does fuel A or B allow for a higher speed?

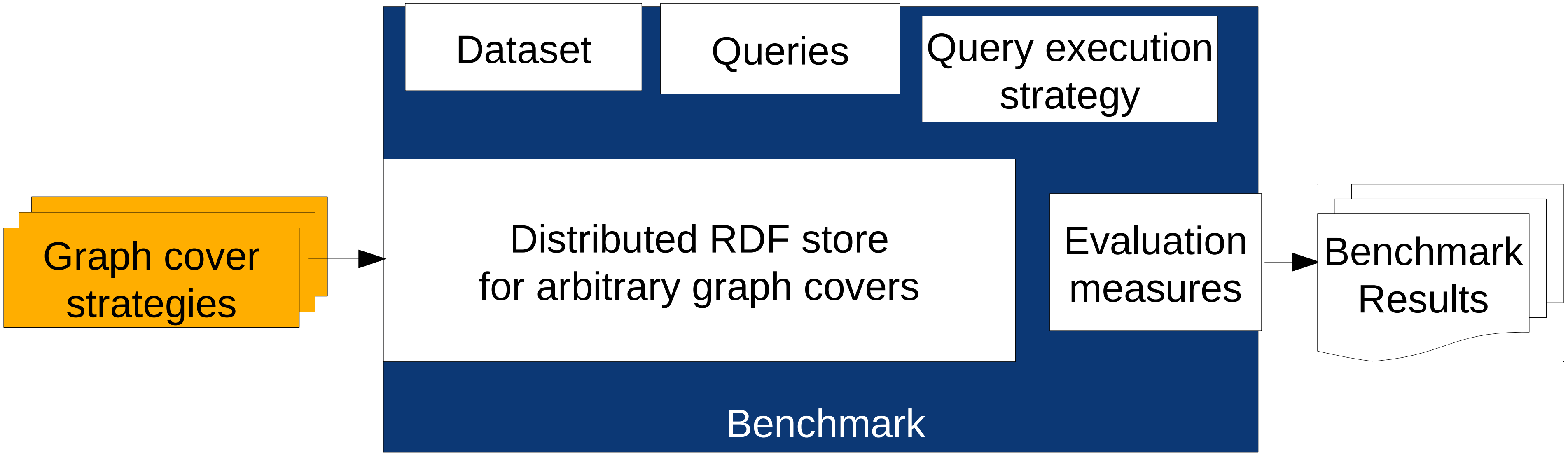
2) Usage of slow communication means like Hadoop File System
=> Increased importance of horizontal containment
e.g. [Huang2011SSQ, Lee2013EDP]



Images from https://openclipart.org

Benchmark Methodology

Goal: Investigating effect of graph cover on the scalability



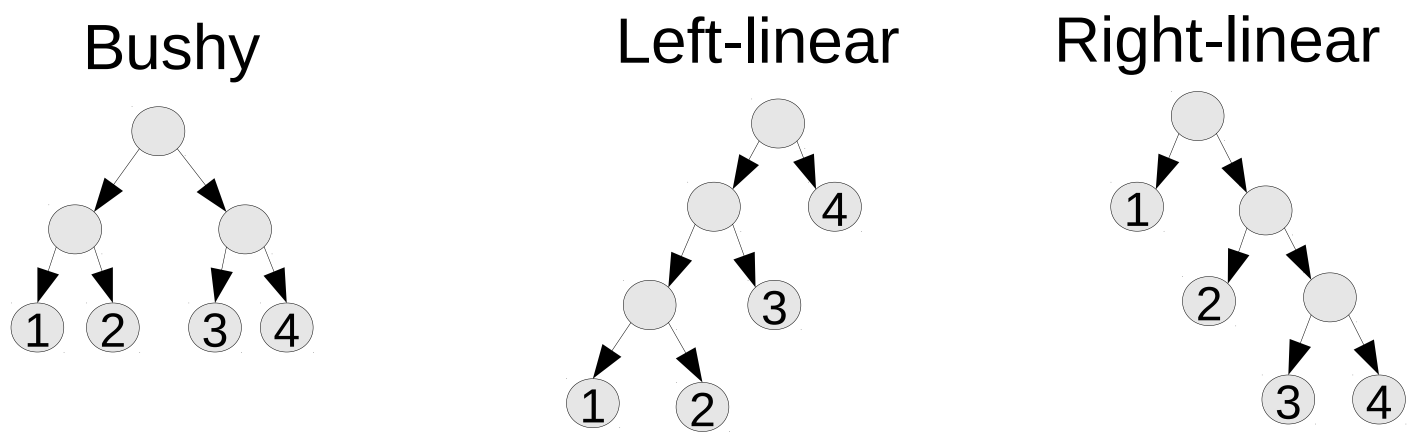
Strategy for Generating Queries

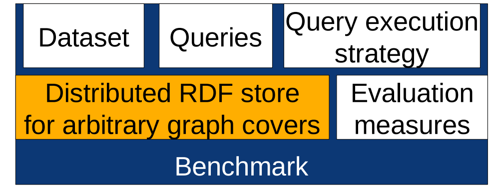
Query Generator: **SPLODGE** [Görlitz2012SSG]

- Generates SPARQL queries for arbitrary datasets
- Generates queries based on
  - Number of joins
  - Join pattern
  - Selectivity
  - Number of data sources

Query Execution Strategy

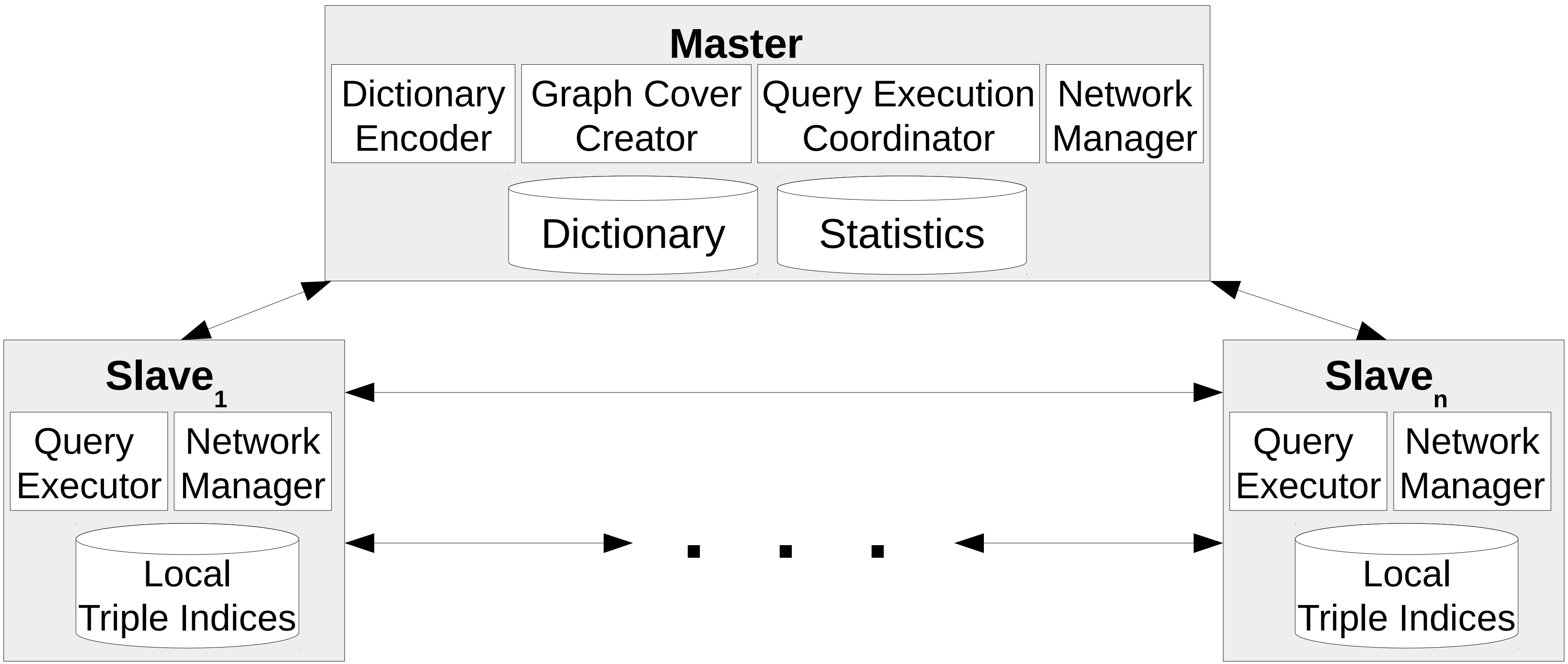
- Query optimizers fitting for arbitrary graph covers difficult
- Execution of several query execution trees:





Koral

- Graph cover independent distributed RDF store
- Inspired by TriAD [GurajadaTheobald2014TAD]



Evaluation Measures

Overall performance

- Query execution time

Horizontal Containment

- Data transfer T :
  variable bindings transferred between compute nodes

Vertical Parallelization (VP)

- Workload Entropy W :
  entropy of join comparisons on each compute node

	T low	T high
W low	low VP	low VP
W high	high VP	low-medium VP

Experimental Setup

Compared graph cover strategies:

- Hash, hierarchical hash and minimal edge-cut cover

Dataset:

- 1 trillion triple subset of BTC2014 [Käfer2014BTC]

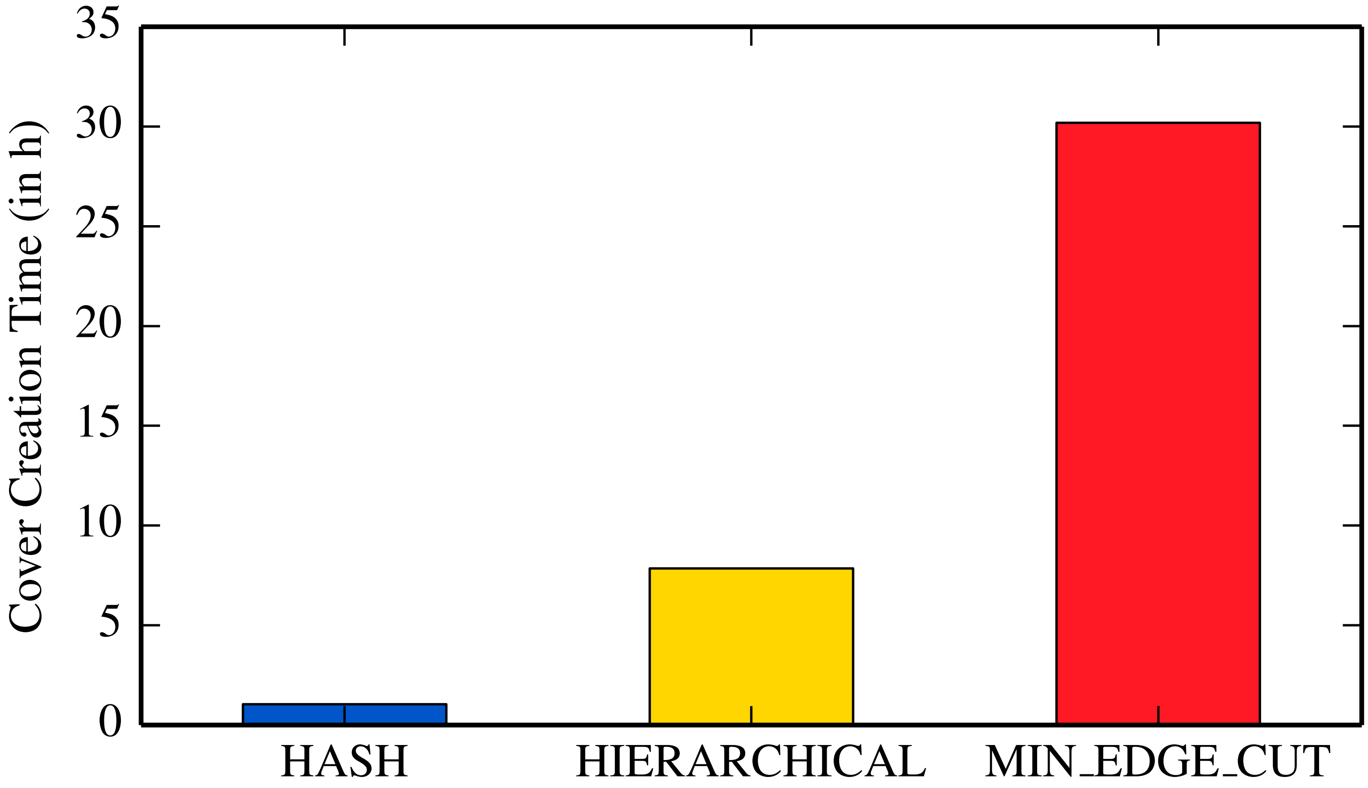
Queries:

- *Number of joins*: 2 and 8 triple patterns
- *Join pattern*: path-shaped and star-shaped
- *Selectivity*: 0.001% and 0.01% (1M and 10M triples)
- *Number of data sources*: 1 and 3

Computer environment:

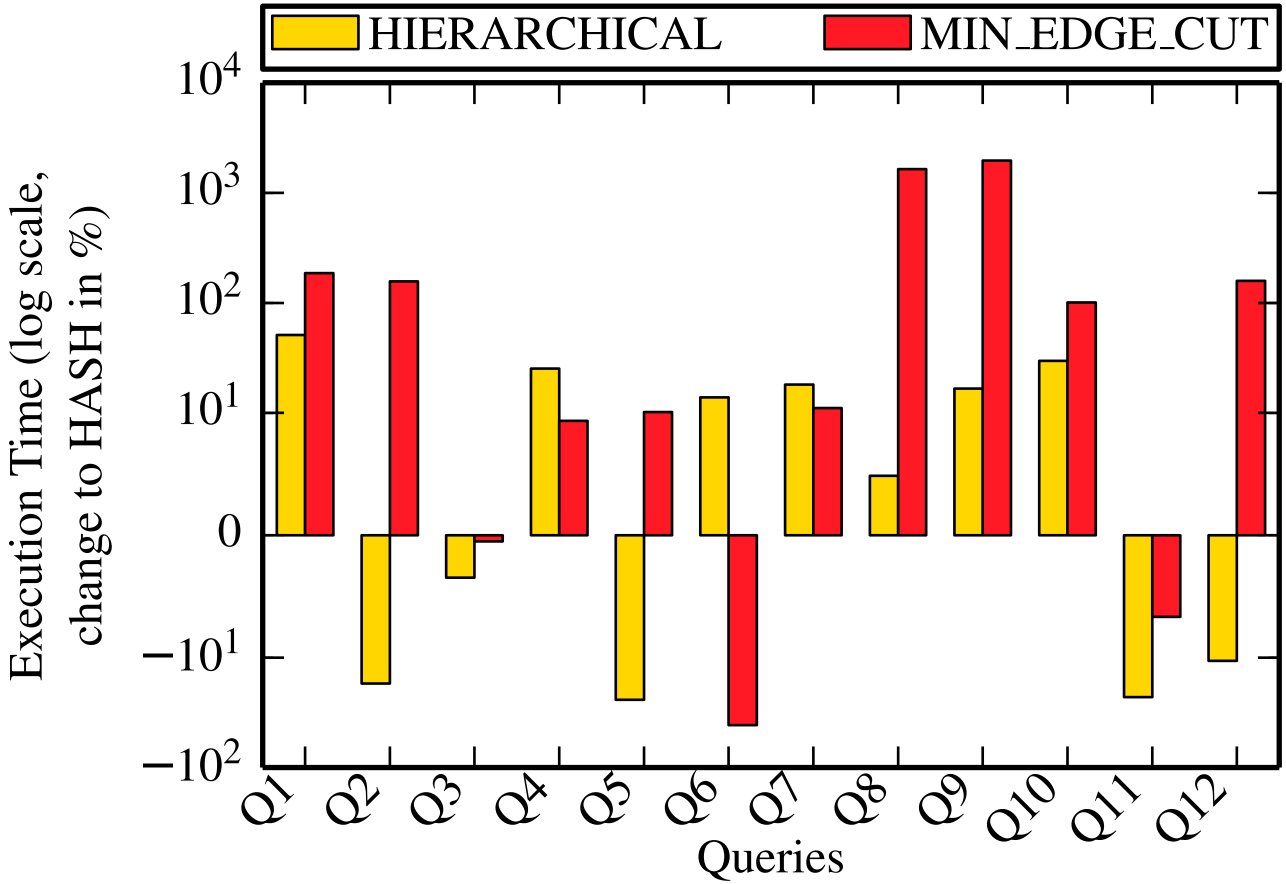
- 1 Master à 4 cores, 8 GB RAM, 1 TB HDD
- 20 Slaves à 1 core, 2 GB RAM, 300 GB HDD
- 1 Gbit ethernet

Graph Cover Creation Time



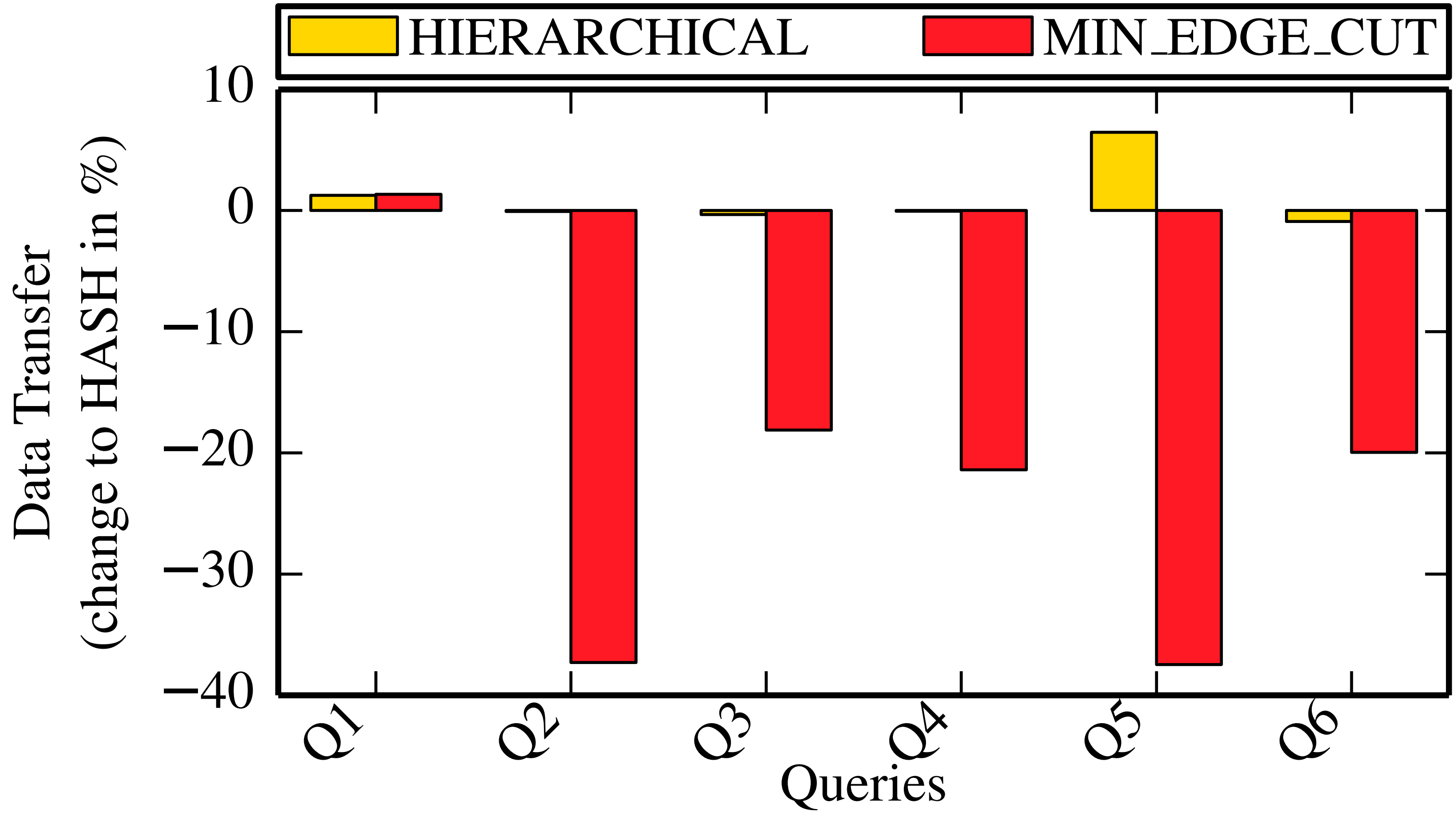
- Minimal edge-cut cover requires most time for creation
- Hash cover is created the fastest

Overall Query Performance



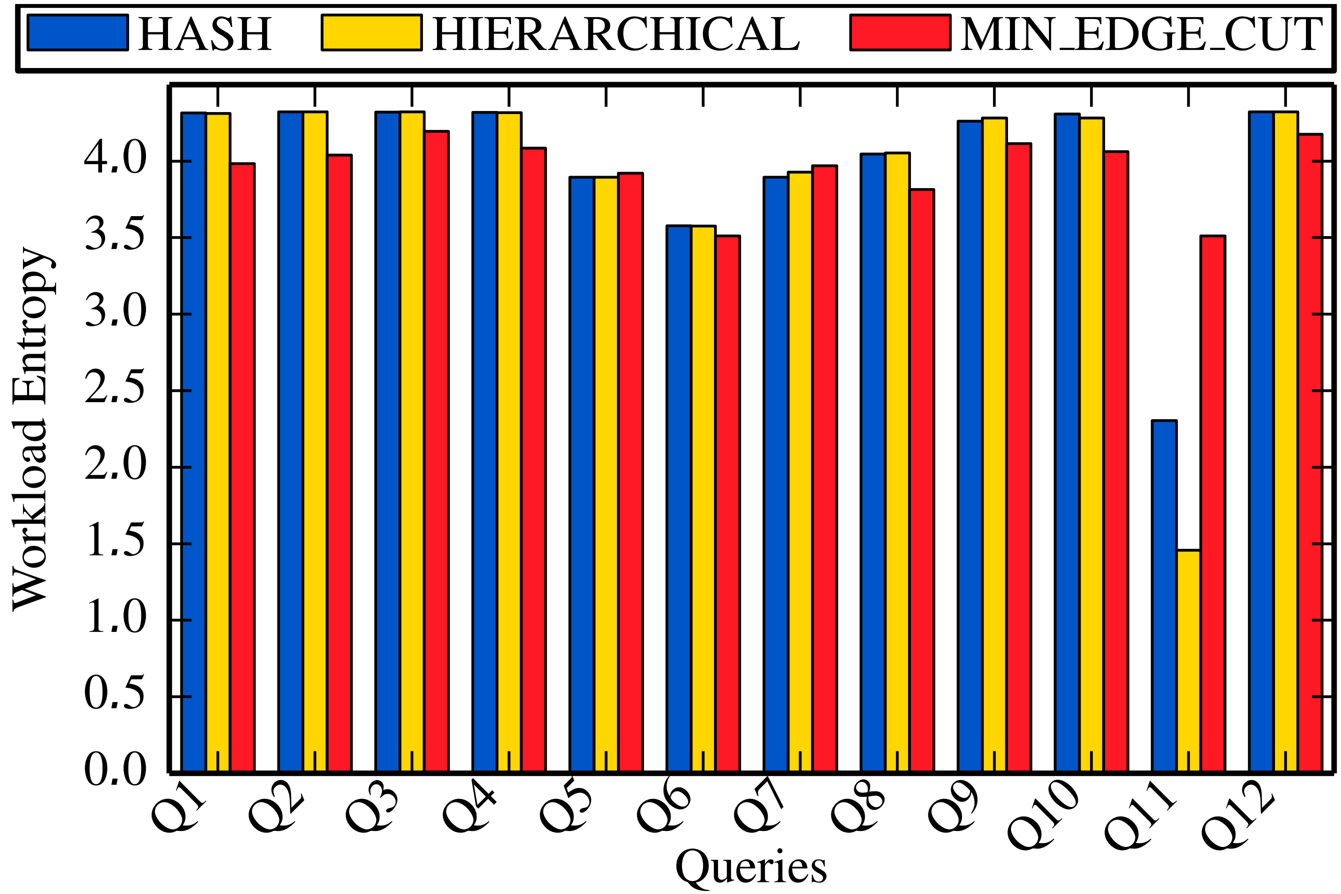
- Bushy query execution outperforms other execution strategies
- Minimal edge-cut causes slowest query execution in most cases
- None of the hash-based covers is faster in general

Horizontal Containment



- Star-shaped queries produce no data transfer
- Minimal edge-cut covers produces less data transfer
- Hash-based covers similar data transfer

Vertical Parallelization



- Minimal edge-cut cover has the least balanced workload
- Hash-based covers have similar balanced workloads

Conclusion

- Minimal edge-cut cover
  - Longest cover creation time
  - Lowest data transfer => high horizontal containment
  - Lowest workload balance => low vertical parallelization
  - Overall performance worse than hash-based covers
- Hash-based covers have similar performance
- Vertical parallelization might be more important than horizontal containment

Future work:
Benchmarking of workload-aware graph cover strategies

Thank you for your Attention!

On Data Placement Strategies in Distributed RDF Stores

Daniel Janke, Steffen Staab, Matthias Thimm

Contributions:

1) Benchmark methodology showing the interdependencies of graph cover strategies and query processing
2) A flexible open-source platform for performing the benchmark
3) Analysis indicating that vertical parallelization may dominate horizontal containment

WeST People and Knowledge Networks

References

[Görlitz2012SSG]

Görlitz, O., Thimm, M., & Staab, S. (2012). Splodge: Systematic generation of sparql benchmark queries for linked open data. The Semantic Web–ISWC 2012, 116–132.

[GurajadaTheobald2014TAD]

Gurajada, S., Seufert, S., Miliaraki, I., & Theobald, M. (2014). TriAD: A Distributed Shared-nothing RDF Engine Based on Asynchronous Message Passing. In Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (pp. 289–300). New York, NY, USA: ACM.

[Harth2007YAF]

Harth, A., Umbrich, J., Hogan, A., & Decker, S. (2007). YARS2: A Federated Repository for Querying Graph Structured Data from the Web. In K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, … P. Cudré-Mauroux (Eds.), The Semantic Web (Vol. 4825, pp. 211–224). Springer Berlin Heidelberg.

[Huang2011SSQ]

Huang, J., Abadi, D. J., & Ren, K. (2011). Scalable SPARQL Querying of Large RDF Graphs. PVLDB, 4(11), 1123–1134.

References

[Käfer2014BTC]

Käfer, T., & Harth, A. (2014). Billion Triples Challenge data set.

[Karypis1998AFA]

Karypis, G., & Kumar, V. (1998). A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. SIAM J. Sci. Comput., 20(1), 359–392.

[Lee2013EDP]

Lee, K., & Liu, L. (2013). Efficient Data Partitioning Model for Heterogeneous Graphs in the Cloud. In Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (p. 46:1--46:12). New York, NY, USA: ACM.

[Lee2013SQO]

Lee, K., & Liu, L. (2013). Scaling Queries over Big RDF Graphs with Semantic Hash Partitioning. Proc. VLDB Endow., 6(14), 1894–1905.

[Wu2014SAS]

Wu, B., Zhou, Y., Yuan, P., Jin, H., & Liu, L. (2014). SemStore: A Semantic-Preserving Distributed RDF Triple Store. In 23rd ACM International Conference on Information and Knowledge Management (CIKM). Shanghai.

References

[Zeng2013ADG]

Zeng, K., Yang, J., Wang, H., Shao, B., & Wang, Z. (2013). A Distributed Graph Engine for Web Scale RDF Data. Proc. VLDB Endow., 6(4), 265–276.

[Zhang2013ETS]

Zhang, X., Chen, L., Tong, Y., & Wang, M. (2013). EAGRE: Towards scalable I/O efficient SPARQL query evaluation on the cloud. In Data Engineering (ICDE), 2013 IEEE 29th International Conference on (pp. 565–576).

Hash Cover [e.g. Harth2007YAF]

Triple placement bases on subject hash modulo number of compute nodes

$$\text{hashCover}(< s, p, o >) := \text{hash}(s) \mod |C|$$

Hierarchical Hash Cover [Lee2013SQO]

Triple placement bases on prefixes of subject IRIs

$$\text{hashCover}(< s, p, o >) := \begin{cases} \text{hash}(\text{prefix}(s)) \mod |C| & \text{, if } s \in I \\ \text{hash}(s) \mod |C| & \text{, otherwise} \end{cases}$$

- IRI: `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`
- Path hierarchy: `org/w3/www/1999/02/22-rdf-syntax-ns/type`
- Determine path hierarchy prefix such that
  - There exist at least $|C|$ hierarchy prefixes
  - That are shared by at least $\omega\%$ of all triples

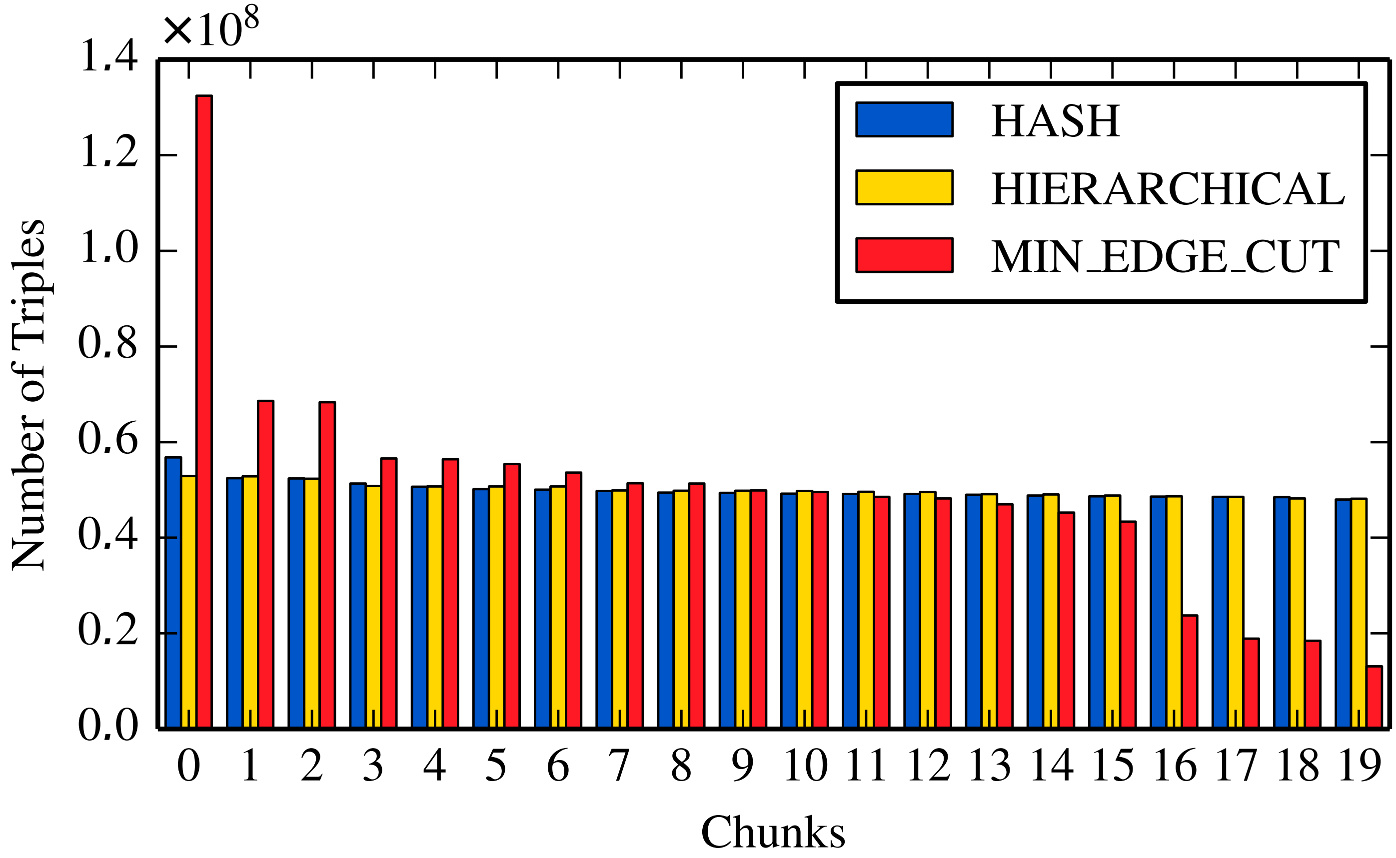
Minimal Edge-Cut Cover

Tries to solve the k-way graph partitioning problem
[Karypis1998AFA]

1) Assign vertices (subjects and objects) to partitions such that
   - Number of edges between vertices of different partitions is minimized and
   - Each partition contains approximately $\frac{|V_G|}{|C|}$ vertices

2) Assign triple to the partition its subject is assigned to

Chunk Sizes



- Minimal edge-cut cover has most unbalanced chunks
- Hash-based covers have equally-sized chunks