Join and Update Processing in Distributed RDF

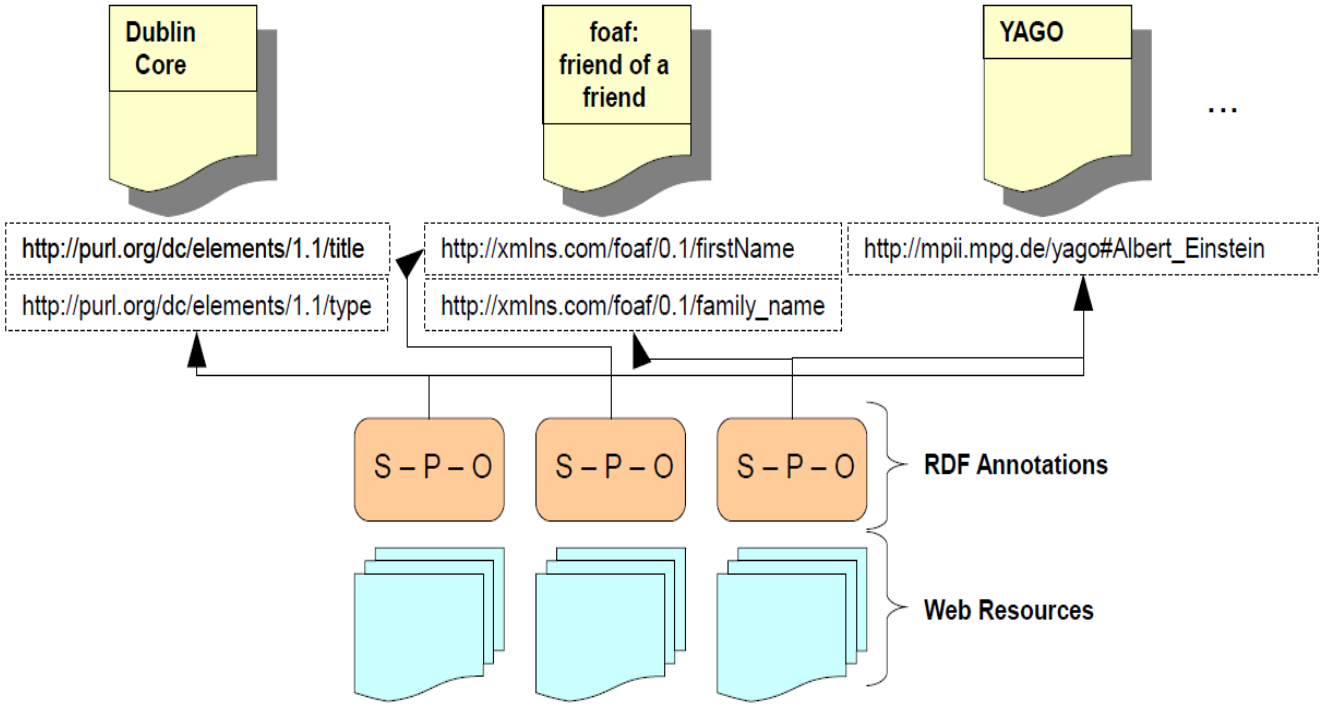
CS 848 - Advanced Topics in Databases: Cloud Data Management

Project Presentation - MatRDF

by
Güneş Aluç

Resource Description Framework (RDF)

- RDF is a World Wide Web Consortium (W3C) standard, with which:
  - the ***semantics*** of (web) resources can be described in a ***machine-processable*** language,
  - (web) resources whose schemas are (i) different or (ii) not well-defined in advance can be ***integrated***



RDF Triples

Subject	Predicate	Object
uwaterloo:cs848	dc:type	yago:course
uwaterloo:cs848	lubm:instructor	foaf:KSalem
uwaterloo:cs848	*hasCapacity*	“30”
foaf:KSalem	foaf:family_name	“Salem”
foaf:KSalem	dblp:hasPublication	*bNode1*
bNode1	...	...

URI

Blank Node

URI

URI

literal

Blank Node

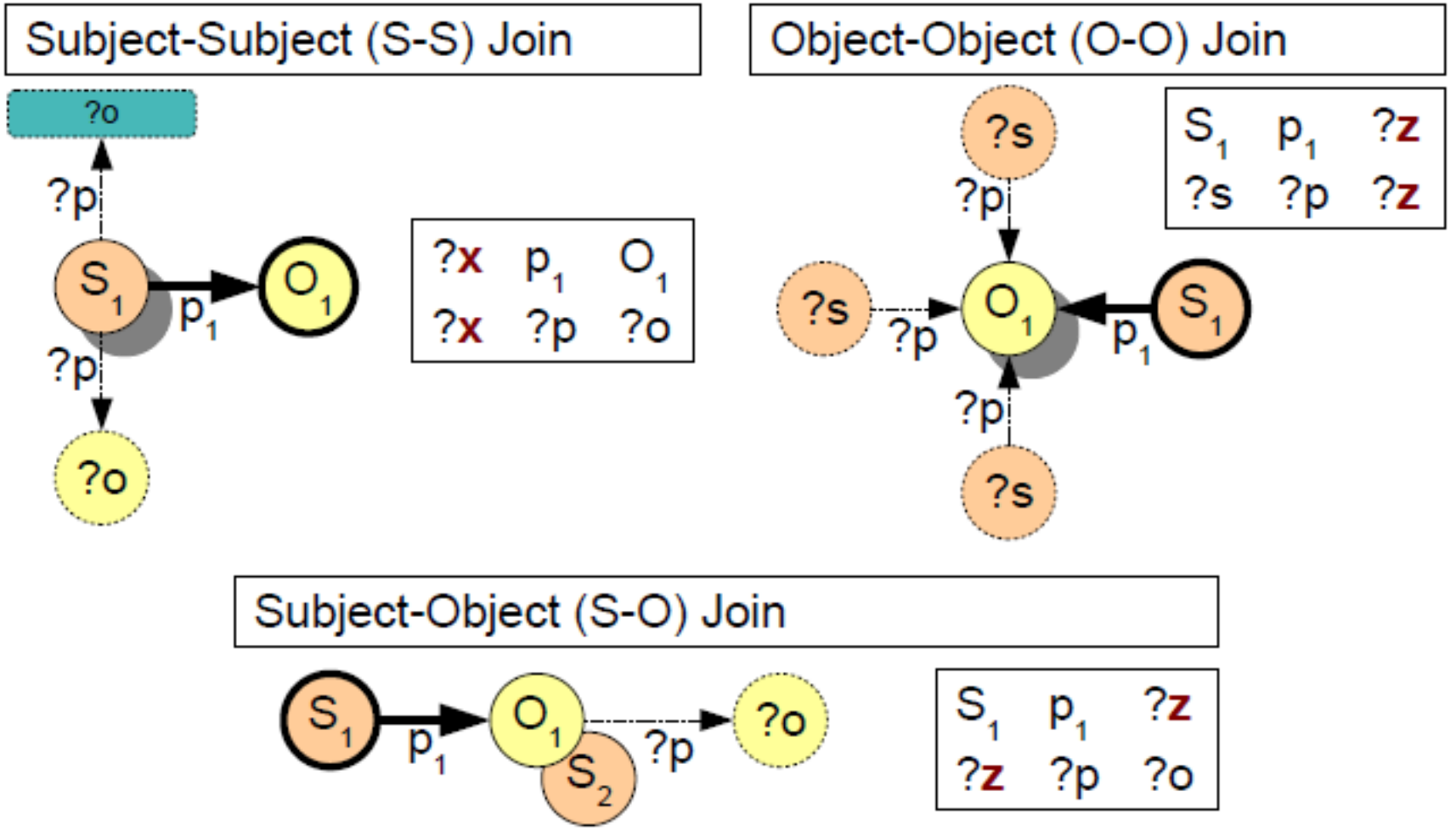
Queries on RDF Triples

- SPARQL (SPARQL Protocol and RDF Query Language) is the W3C recommended query language for RDF:

```
PREFIX dc:    <http://purl.org/dc/elements/1.1/>
PREFIX ...
SELECT ?x
WHERE  {
       ?x    dc:type     yago:course    .
       ?x    ?y          foaf:KSalem    .
}
```

Subject	Predicate	Object
uwaterloo:cs848	dc:type	yago:course
uwaterloo:cs848	lubm:instructor	foaf:KSalem

Join Types


Subject-Subject (S-S) Join
?o
?p
S1
p1
O1
?p
?o
?x p1 O1
?x ?p ?o
Object-Object (O-O) Join
?s
?p
?s
?p
O1
S1
p1
?p
?s
S1 p1 ?z
?s ?p ?z
Subject-Object (S-O) Join
S1
p1
O1
S2
?p
?o
S1 p1 ?z
?z ?p ?o

Triplestores

- As a response to the need to store and efficiently process very large sets of semantic data, RDF has received significant attention from the database community
  - http://challenge.semanticweb.org/ → billion triples!
- A triplestore is a DBMS that supports storage and querying (e.g. SPARQL) of RDF triples
  - Non-distributed approaches:
    - *Challenge*: queries involve multiple star-joins and long-path joins
  - Distributed approaches:
    - *Challenge*: joins require data transfer between multiple nodes
  - Common:
    - *Challenge*: updates are almost always ignored or assumed to be batch operations

Non-distributed Triplestores

S	P	O
#KSalem	#firstName	"Ken"
#KSalem	#lastName	"Salem"
#MTOzsu	#lastName	"Özsu"
#pub1	#hasAuthor	#KSalem
#pub2	#hasAuthor	#KSalem
#pub2	#hasAuthor	#MTOzsu

Statement Table

(+) easy to maintain
(-) necessitates self-joins

[4, 7]

(+) speeds up joins
if there is a schema
(-) NULL values

[23]

URI	*firstName*	*lastName*
#KSalem	"Ken"	"Salem"
#MTOzsu	**NULL**	"Özsu"

URI	...	*hasAuthor*
#pub1	...	{#KSalem}
#pub2	...	{#KSalem, #MTOzsu}

Property Table

lastName

S	O
#KSalem	"Salem"
#MTOzsu	"Özsu"

hasAuthor

S	O
#pub1	#KSalem
#pub2	#KSalem
#pub2	#MTOzsu

(+) fast on column-stores
(-) queries involving
joins on unbound predicates
are problematic

[1]

Vertical Partitioning

Non-distributed Triplestores

(+) equally generous to all query types
(-) aggressive space utilization (compression?)

[22, 16, 17]

***indexed** and **clustered** on P-O*

P	O	S
#firstName	"Ken"	#KSalem
#hasAuthor	#KSalem	#pub1
#hasAuthor	#KSalem	#pub2
#hasAuthor	#MTOzsu	#pub2
#lastName	"Özsu"	#MTOzsu
#lastName	"Salem"	#KSalem

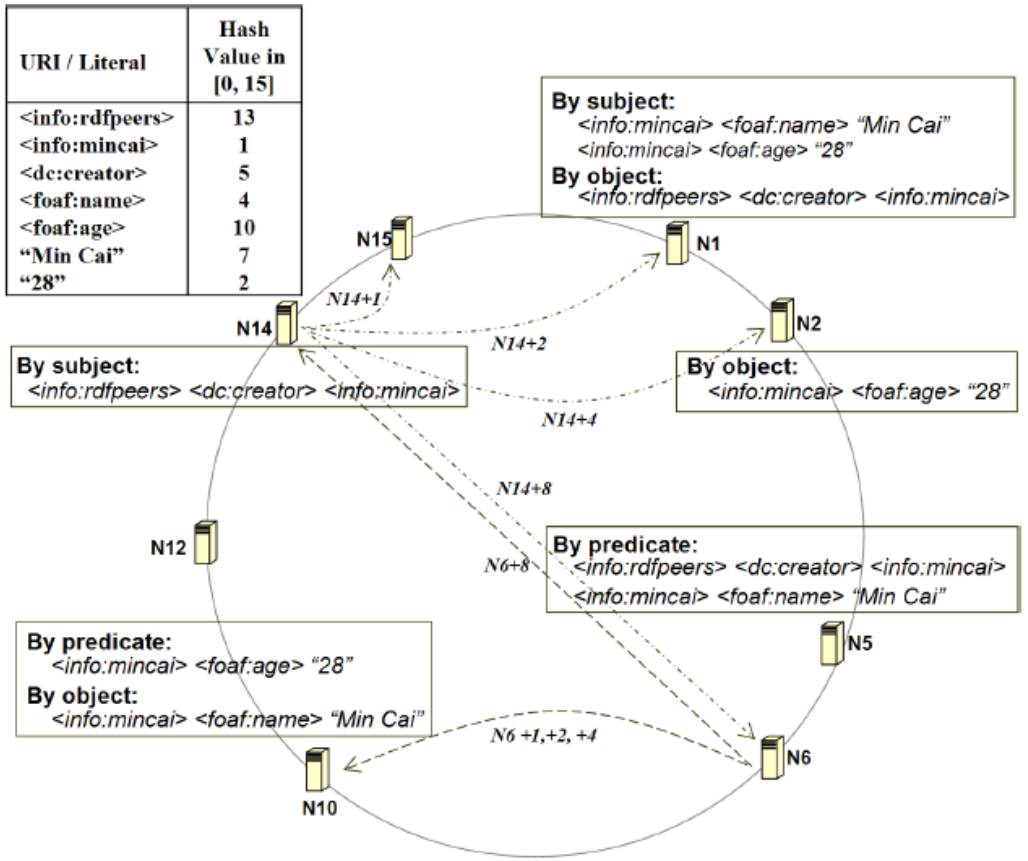
***indexed** and **clustered** on P-S*

P	S	O
#firstName	#KSalem	"Ken"
#hasAuthor	#pub1	#KSalem
#hasAuthor	#pub2	#KSalem
#hasAuthor	#pub2	#MTOzsu
#lastName	#KSalem	"Salem"
#lastName	#MTOzsu	"Özsu"

… and indexed and clustered on O-S, O-P, S-O, S-P

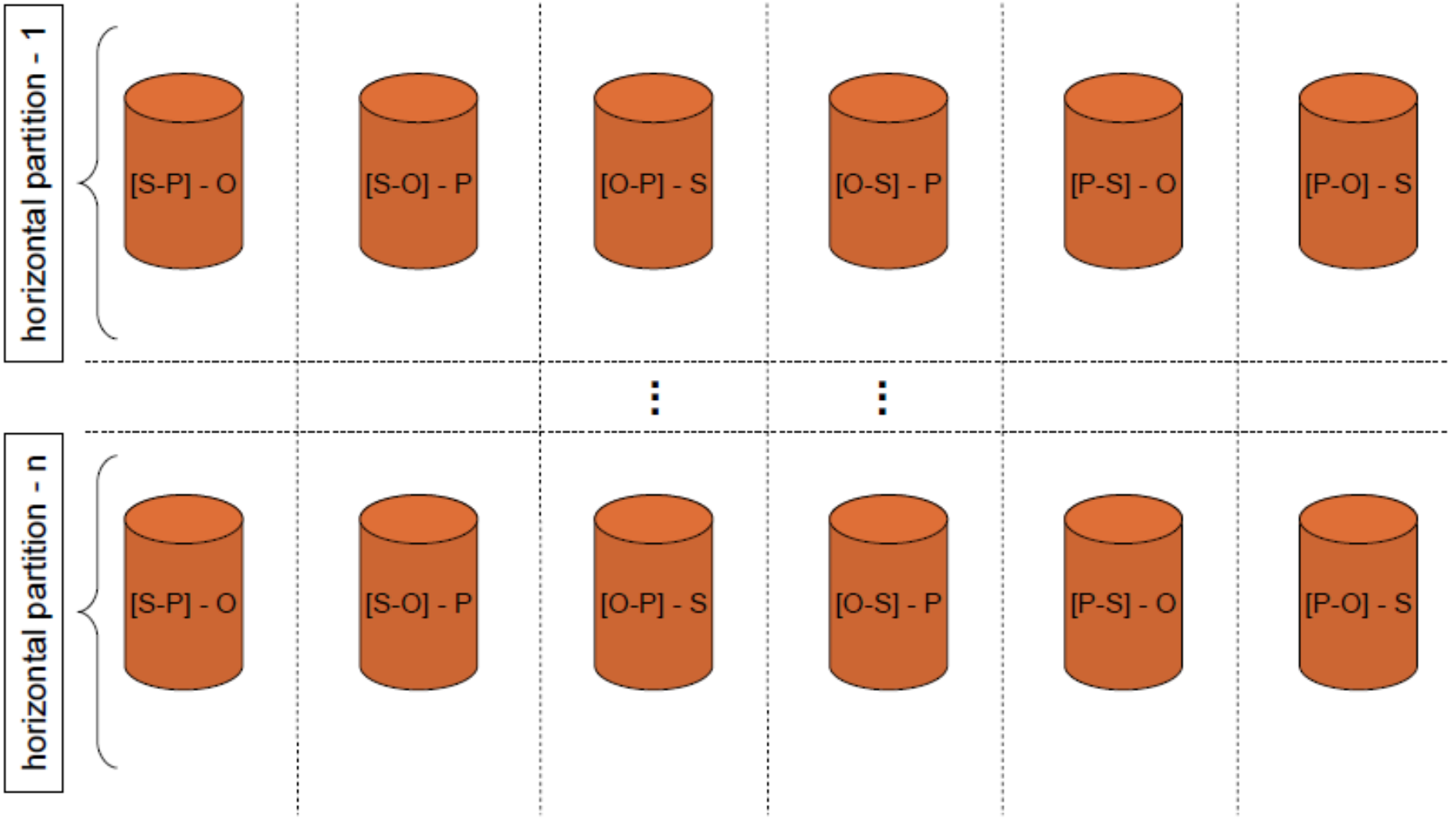
Aggressive Indexing

Distributed Triplestores



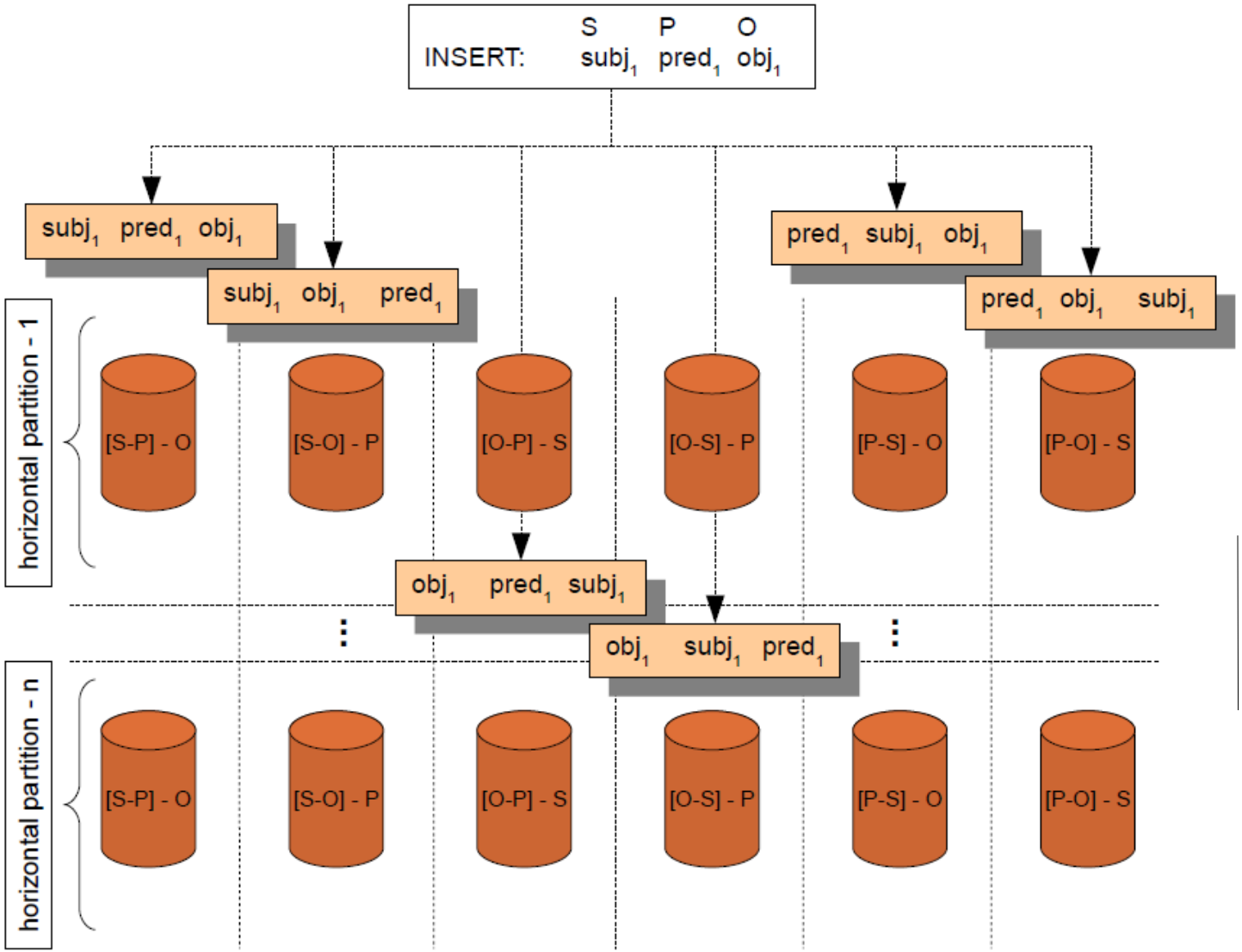
- RDFPeers [5, 15] is a triplestore that uses a P2P infrastructure, namely the multi-attribute addressable network (MAAN), to distribute the triples among the peers.
- Each triple is stored 3 times, on the node hashed by its subject, predicate and object
- Problem:
  - even matching a simple atomic triple pattern (e.g. s_i, ?p, ?o) requires log(N) hops

MatRDF – Partitioning Scheme



[A-B]-C implies that the table is indexed and clustered on attributes A and B, respectively.

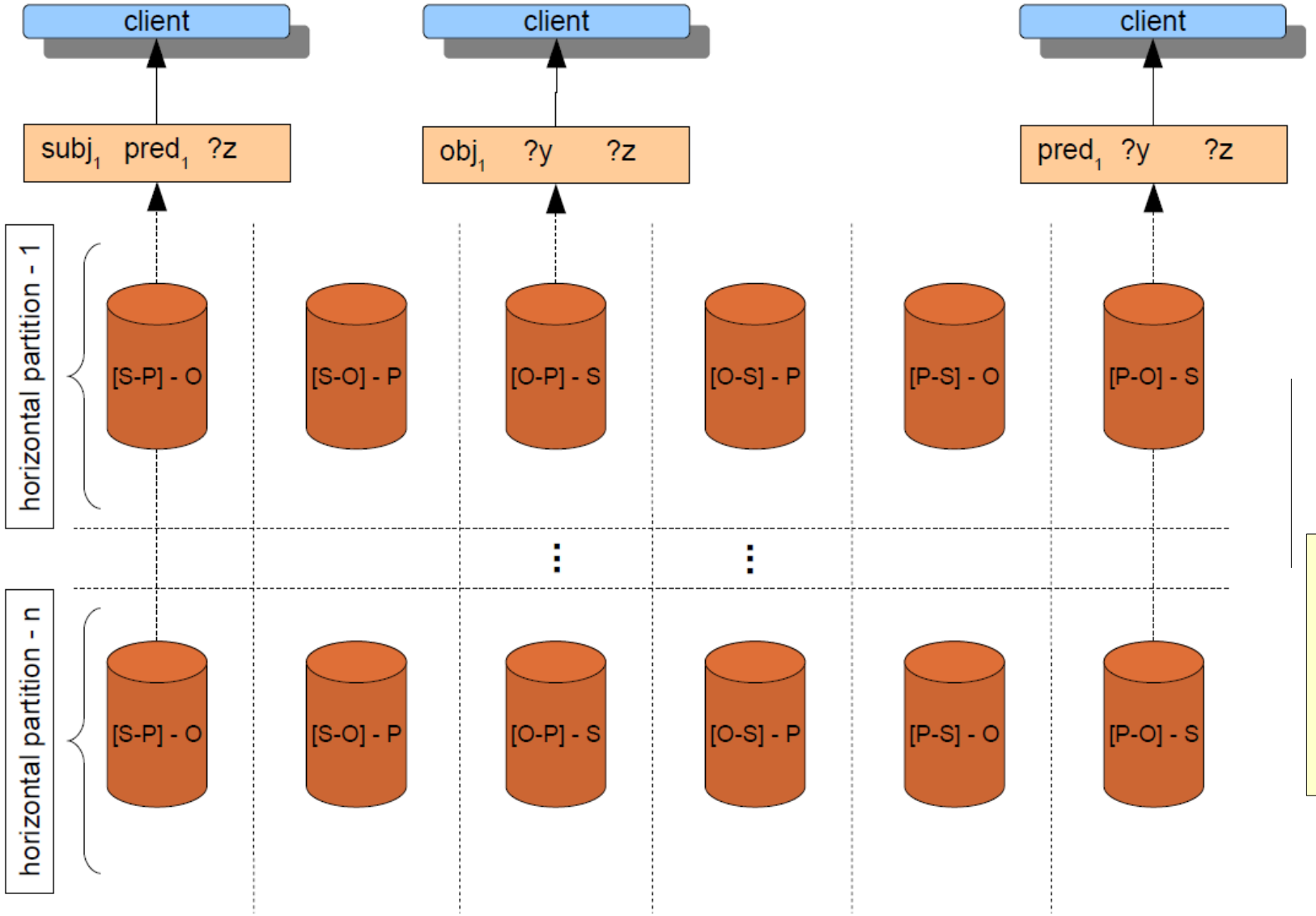
MatRDF – Inserts (and Updates)



[A-B]-C implies that the table is indexed and clustered on attributes A and B, respectively.

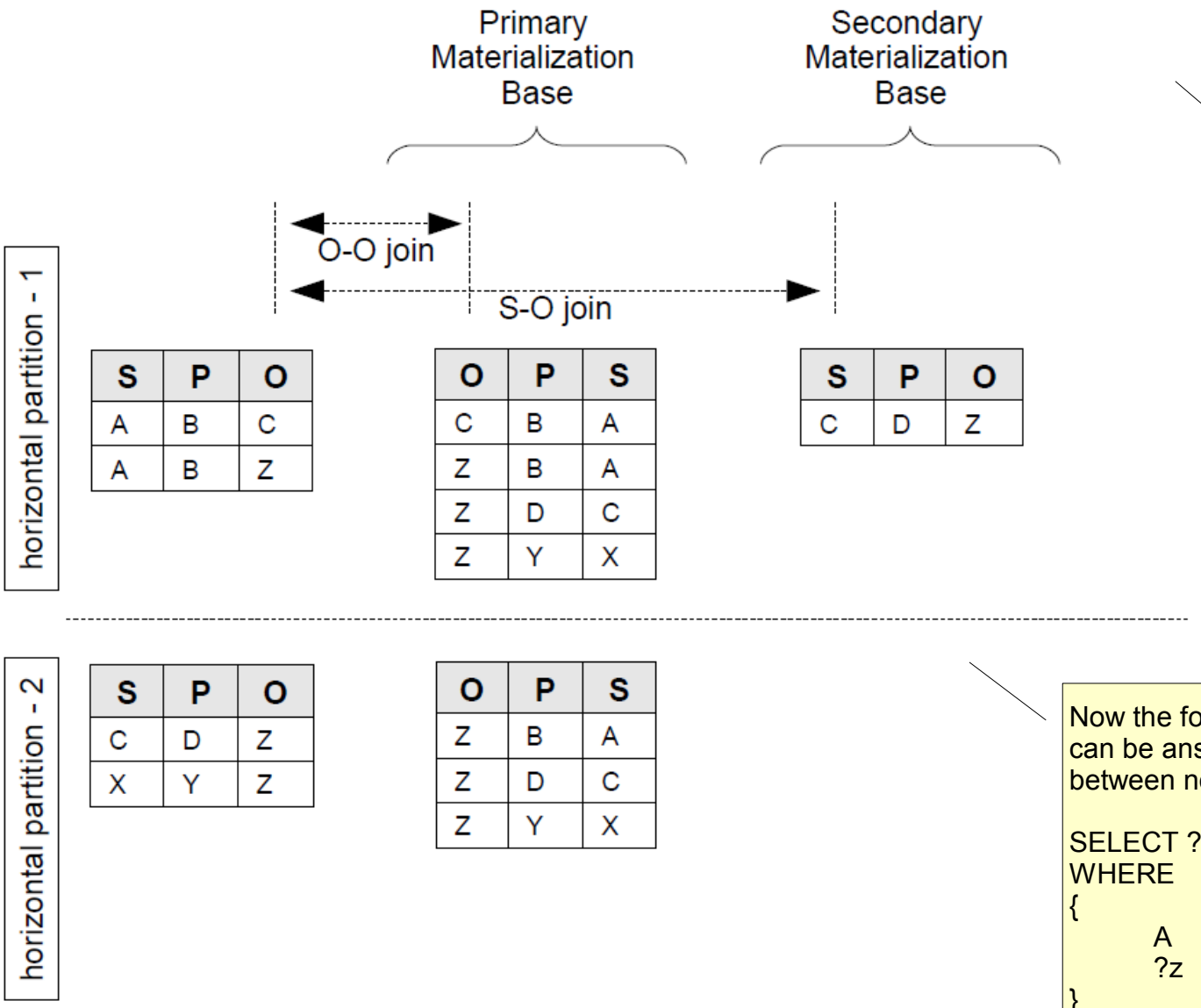
The cost of update is reduced by a factor of 6, since each permutation of the triple is processed in parallel

MatRDF – Atomic Triple Patterns



Given that the queries involve atomic triple patterns only, they can be answered in isolation, concurrently

MatRDF – Star Joins & Long-Path Joins



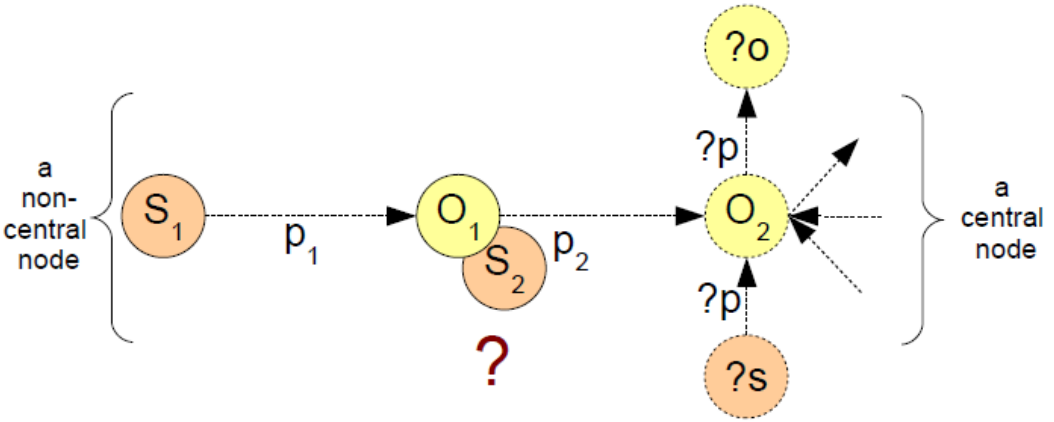
On each node, the triples involved in possible join paths (S-S, S-O, O-O) are stored in additional tables

Now the following SPARQL query can be answered without data transfer between nodes:

```
SELECT ?a  ?b
WHERE
{
        A      B      ?z     .
        ?z     ?a     ?b     .
}
```

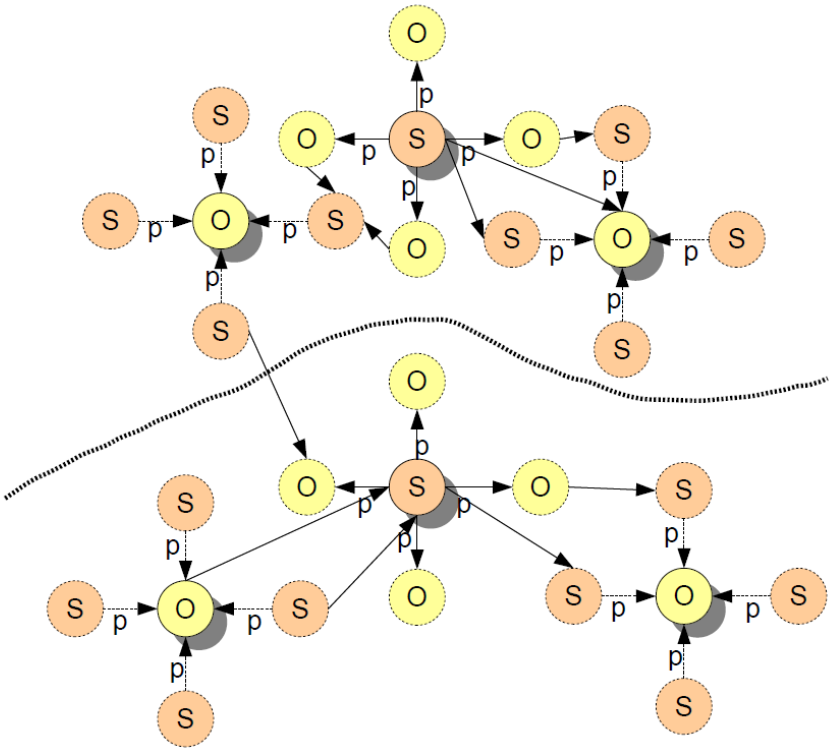
MatRDF- Selecting What to Materialize

- In the worst case, the primary and secondary tables are replicated over each horizontal partition → not scalable!!! We need to restrict what can be stored in the primary and secondary tables:
  - Strategy 1: On each node randomly select a subset of all possible S-S / S-O / O-O join paths
  - Strategy 2: Dynamically populate the primary and secondary tables based on a replacement policy
  - Strategy 3: Incorporate the vertex centrality (in particular degree centrality) of RDF nodes to approximate how valuable it is to store the join paths involving that node



MatRDF - Horizontal Partitioning Scheme

- Strategy 1: Split lexicographically
  - (+) Easy to maintain
  - (-) Hot-spots?
  - (-) Inter-node data transfer?
- Strategy 2: Split based on graph locality
  - (+) Inter-node data transfer reduced
  - (-) Difficult to maintain



Thank you...
Questions?

References

[1] Daniel J. Abadi, Adam Marcus, Samuel R. Madden, and Kate Hollenbach. Scalable semantic web data management using vertical partitioning. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 411–422. VLDB Endowment, 2007.

[2] Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel Abadi, Avi Silberschatz, and Alexander Rasin. Hadoopdb: an architectural hybrid of mapreduce and dbms technologies for analytical workloads. *Proc. VLDB Endow.*, 2(1):922–933, 2009.

[3] Christian Bizer and Andreas Schultz. The berlin sparql benchmark. *Int. J. Semantic Web Inf. Syst.*, 5(2):1–24, 2009.

[4] Jeen Broekstra, Arjohn Kampman, and Frank van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In Ian Horrocks and James Hendler, editors, *Proceedings of the first International Semantic Web Conference (ISWC 2002)*, number 2342 in Lecture Notes in Computer Science, pages 54–68, Sardinia, Italy, June 9 – 12, 2002. Springer Verlag, Heidelberg Germany.

[5] Min Cai and Martin Frank. Rdfpeers: a scalable distributed rdf repository based on a structured peer-to-peer network. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 650–657, New York, NY, USA, 2004. ACM.

[6] Min Cai, Martin Frank, Jinbo Chen, and Pedro Szekely. Maan: A multi-attribute addressable network for grid information services. In *GRID '03: Proceedings of the 4th International Workshop on Grid Computing*, page 184, Washington, DC, USA, 2003. IEEE Computer Society.

[7] J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson. The jena semantic web platform: Architecture and design. Technical Report HPL-2003-146, HP Laboratories, 2003.

[8] Hyunsik Choi, Jihoon Son, YongHyun Cho, Min Kyoung Sung, and Yon Dohn Chung. Spider: a system for scalable, parallel / distributed evaluation of large-scale rdf data. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 2087–2088, New York, NY, USA, 2009. ACM.

[9] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, 2008.

[10] O Erling and I Mikhailov. Towards web-scale rdf. *SSWS, Karlsruhe, Germany*, 2008.

[11] Mohammad Farhan Husain, Pankil Doshi, Latifur Khan, and Bhavani Thuraisingham. Storage and retrieval of large rdf graph using hadoop and mapreduce. In *CloudCom '09: Proceedings of the 1st International Conference on Cloud Computing*, pages 680–686, Berlin, Heidelberg, 2009. Springer-Verlag.

[12] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. Lubm: A benchmark for owl knowledge base systems. *J. Web Sem.*, 3(2-3):158–182, 2005.

References

[13] Andreas Harth, Jürgen Umbrich, Aidan Hogan, and Stefan Decker. Yars2: A federated repository for querying graph structured data from the web. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *ISWC/ASWC*, volume 4825 of *Lecture Notes in Computer Science*, pages 211–224. Springer, 2007.

[14] Chengwen Liu and Hao Chen. A hash partition strategy for distributed query processing. In *EDBT '96: Proceedings of the 5th International Conference on Extending Database Technology*, pages 373–387, London, UK, 1996. Springer-Verlag.

[15] Akiyoshi Matono, Said Pahlevi, and Isao Kojima. Rdfcube: A p2p-based three-dimensional index for structural joins on distributed triple stores. *Databases, Information Systems, and Peer-to-Peer Computing*, pages 323–330, 2007.

[16] Thomas Neumann and Gerhard Weikum. Rdf-3x: a risc-style engine for rdf. *Proc. VLDB Endow.*, 1(1):647–659, 2008.

[17] Thomas Neumann and Gerhard Weikum. Scalable join processing on very large rdf graphs. In *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, pages 627–640, New York, NY, USA, 2009. ACM.

[18] Alisdair Owens, Andy Seaborne, Nick Gibbins, and mc schraefel. Clustered tdb: A clustered triple store for jena. Technical report, Electronics and Computer Science, University of Southampton, 2008.

[19] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker. A comparison of approaches to large-scale data analysis. In *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, pages 165–178, New York, NY, USA, 2009. ACM.

[20] Lefteris Sidirourgos, Romulo Goncalves, Martin Kersten, Niels Nes, and Stefan Manegold. Column-store support for rdf data management: not all swans are white. *Proc. VLDB Endow.*, 1(2):1553–1563, 2008.

[21] Mike Stonebraker, Daniel J. Abadi, Adam Batkin, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amerson Lin, Sam Madden, Elizabeth O'Neil, Pat O'Neil, Alex Rasin, Nga Tran, and Stan Zdonik. C-store: a column-oriented dbms. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 553–564. VLDB Endowment, 2005.

[22] Cathrin Weiss, Panagiotis Karras, and Abraham Bernstein. Hexastore: sextuple indexing for semantic web data management. *Proc. VLDB Endow.*, 1(1):1008–1019, 2008.

[23] Kevin Wilkinson. Jena property table implementation. In *Proc. SSWS*, 2006.