Experimental Comparison of Relational and NoSQL Systems: the Case of Decision Support

Tomás F. Llano-Ríos
tfllan01@louisville.edu
University of Louisville,
Louisville, KY 40208,
USA

Mohamed Khalefa
khalefam@oldwestbury.edu
SUNY College Old
Westbury,
Old Westbury, NY
11568, USA

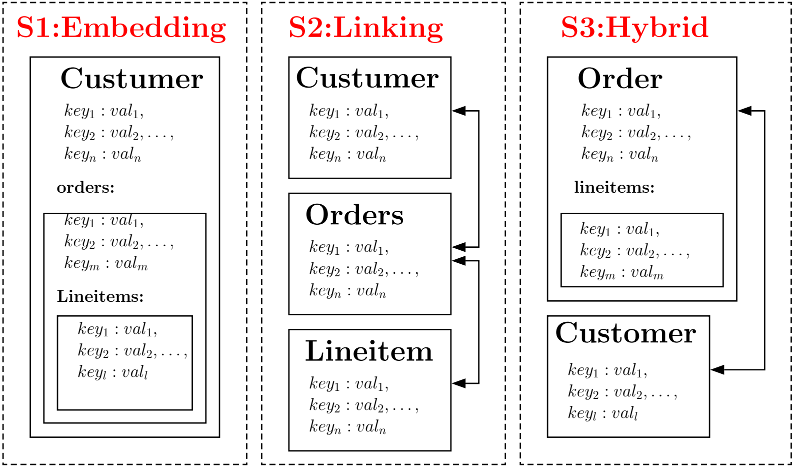
Antonio Badia
abadia@louisville.edu
University of Louisville,
Louisville, KY 40208,
USA

Contribution

- **Experimental comparison**
  - of relational and document-oriented NoSQL systems (PostgreSQL, MongoDB and Couchbase)
  - focused on Decision Support (we use TPC-H)
  - limited to a single-node setting
  - limited to hierarchical data (customer, orders and lineitem tables)
- **Analysis**
  - Query language influence on query optimization
  - Data model's influence on query optimization
  - Possibilities for improvement
  - Is ongoing: We are still doing experiments

TPC-H Schema

translated to document stores in 3 different schemas:



Databases

- **Relational:** PostgreSQL v10.6
- **NoSQL:** MongoDB v4.0, Couchbase CE v6.5

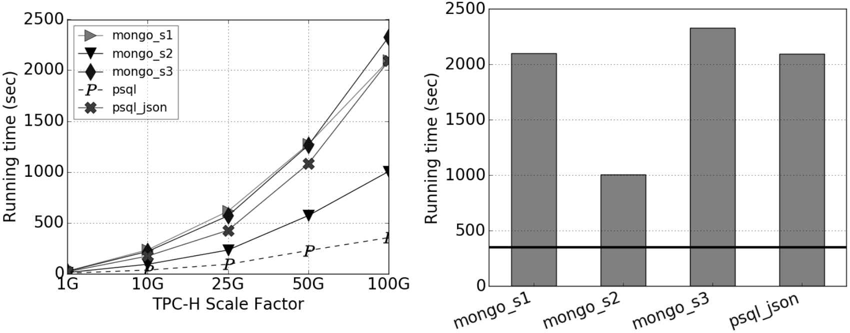
Queries

- Taken from the TPC-H benchmark
- Only involving Customer, Orders, and Lineitem
- Total query versions: 38
- Each version was run 5 times, average taken
- Time limit of 24 hours per query
- Couchbase queries were run at scale factor (SF) 1 only due to poor performance

Results

Query 1: Scan over Lineitem only, S1/S3 at a disadvantage.

S2 is superior because it does not need joins nor unnest.

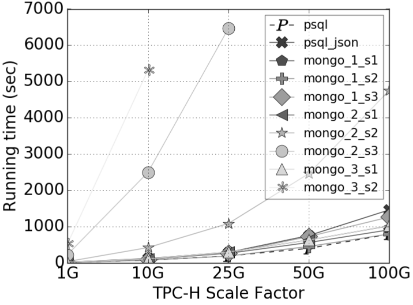


(a)

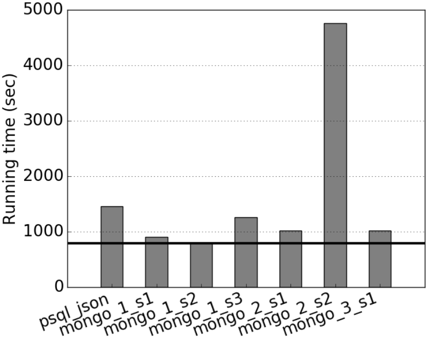
(b) at 100G

Figure: Running times of query 1 on MongoDB and PostgreSQL

Query 3: Join order matters greatly, S2 is superior if orders are filtered first.



(a)

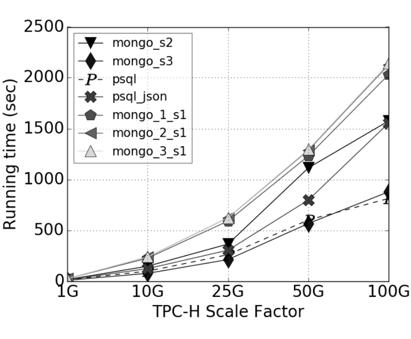


(b) at 100G

Figure: Running times of query 3 on MongoDB and PostgreSQL

Query 4: No lookups in S1 or S3, but S1 has extra unnest.

Filter array and unnest is faster than unnest and then filter.



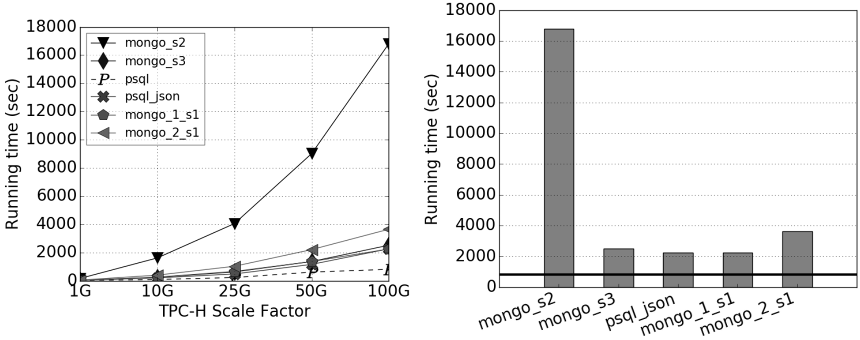
(a)

(b) at 100G

Figure: Running times of query 4 on MongoDB and PostgreSQL

Query 12: Orders ⋈ Lineitem makes S2 the slowest.

No lookups in S1 or S3, but S1 has extra unnest.

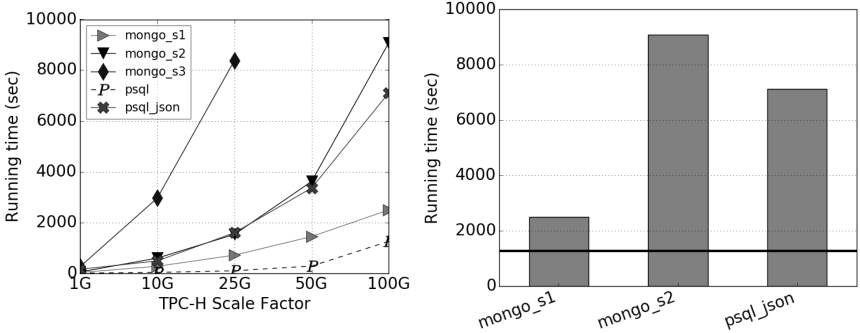


(a)

(b) at 100G

Figure: Running times of query 12 on MongoDB and PostgreSQL

Query 13: Lookup slow with sub-queries (issue [SERVER-41171]); cannot effectively convert $\sigma(\mathcal{C} \sqsupset\!\!\bowtie \mathcal{O})$ into $\mathcal{C} \sqsupset\!\!\bowtie \sigma(\mathcal{O}) | C = \text{Customer} \wedge O = \text{Orders}$ in S2 nor S3. No lookups in S1.



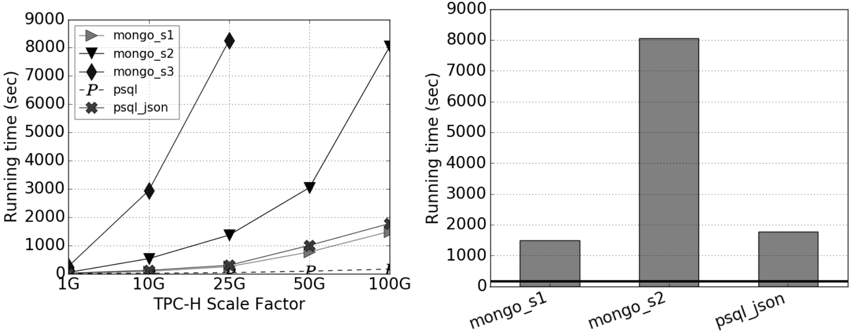
(a)

(b) at 100G

Figure: Running times of query 13 on MongoDB and PostgreSQL

Experiment Results

Query 22: Sub-query translates to self-lookup on S1, ordering of operators depends on schema, sub-queries in lookup are slow.



(a)

(b) at 100G

Figure: Running times of query 22 on MongoDB and PostgreSQL

High Cost

- Scan single collection with large documents
- Unnesting large documents

Main document store limitations

- Join reordering
    - Must be done manually in MongoDB
    - Couchbase CE cannot express correlation as join
- Optimizer
    - No cost-based optimizer
    - Selectivity of conditions not considered

Conclusions

- Typical document store design (one or a few collections with complex documents that use embedding) is not always a good fit for DSS environments.
- Schema-less does not imply schema-free. Schema design matters in document stores for DSS environments.
- Navigational languages should be supported by an optimizer that is able to rewrite and reorder operations in a query.

Future work

- Extending comparison to column-oriented DBs.
- Exploring document storage as multi-dimensional arrays.
- Expanding further schemas and query sets (all TPC-H queries).
- Explore a distributed setup.

Thank you!

Data, Queries and Code

www.github.com/tllano11/dss-sql-vs-nosql-experiments

Questions?

Please contact us:

tfllan01@louisville.edu,

khalefam@oldwestbury.edu,

abadia@louisville.edu