PushdownDB: Accelerating a DBMS using S3 Computation

Idea with PushdownDB

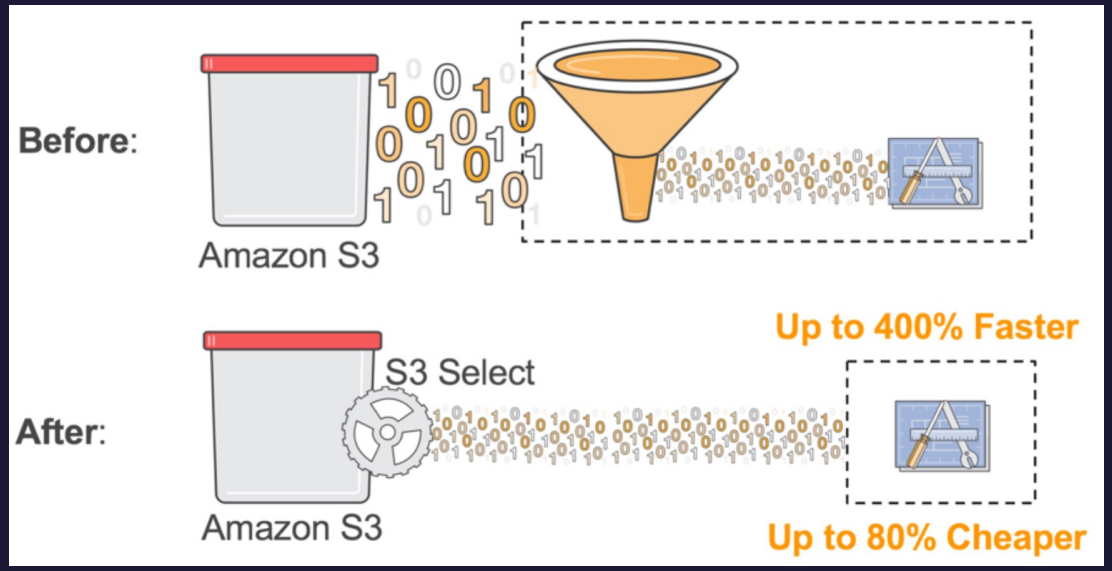
- Storage Disaggregation – Independent scaling, Reliability
- Causes network to be a bottleneck
- **Idea**: Push down some computation to reduce data that has to be transferred

PushdownDB

- Authors create a new DBMS: PushdownDB
- Has query operator implementation both with and without computation pushdown
- Effectively compare differences in performance and cost

S3 Select

- PushdownDB uses S3 for storage
- New feature to Amazon S3 in 2017
- Selections, Projections and Aggregation (over all selected rows)



From AWS Blog

Complex Query Operators

- S3 Select allows us to push selection and projection into S3. Just this could be a good thing.
- What about other more complex query operators?
  - Joins
  - Group By
  - Top K
  - Selection through indexes
- This paper uses S3 Select operations as building blocks to improve performance and cost of these complex operators

Hash Joins

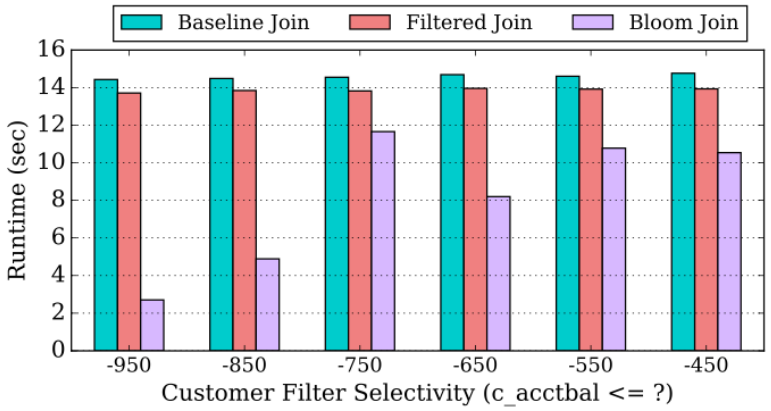
- **Baseline** Load first table as building relation, load second table as probing relation. Join on Compute Node.
- **Filtered** Same as baseline, but push down projection and selection to storage service
- Can we do something more fundamental with the join operator? Filtering based on join key?

Bloom Join

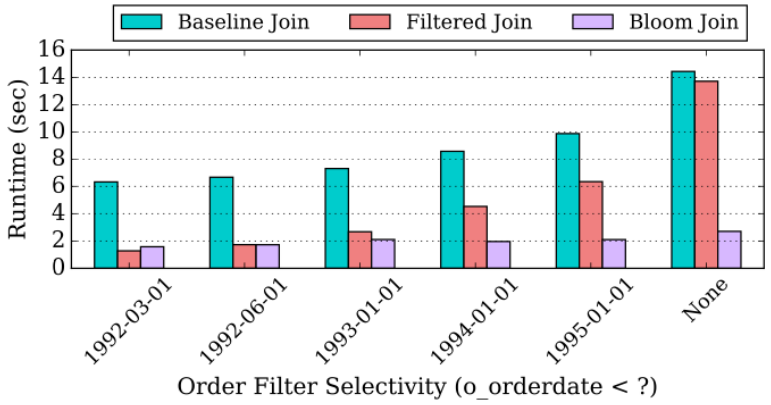
- Load first table as building relation and build the hash table
- Create a **bloom filter** of join keys in building relation
  - Probabilistic. Can have false positives, no false negatives
- Load second table, and use S3 Select selection with the bloom filter
- Because no false negatives, all rows we care about will be returned
- If high selectivity in probing relation, can reduce network traffic

Join Results

```
SELECT
    SUM(O_TOTALPRICE)
FROM
    CUSTOMER, ORDER
WHERE
    O_CUSTKEY = C_CUSTKEY AND
    C_ACCTBAL <= upper_c_acctbal AND
    O_ORDERDATE < upper_o_orderdate
```



(a) Runtime



(a) Runtime

From the paper

Summary

- Push some computation to reduce number of rows to return
- Significantly reduce network traffic

Comments and Discussion

- What should we push down to get benefits of storage disaggregation while still limiting network transfer
- If we push a lot of computation, could the independent scaling property be ruined? If push too much, get a traditional shared nothing system.
- Only pushes computation in a way to respect data partitioning. Don’t require any data shuffling. Also limit computation at storage nodes
- Computation that requires all partitions is done in compute nodes. Get filtered tables and horizontal partitions from the storage layer and assemble at compute layer

Thanks for listening!

Top-K

- Want to select lowest K rows in some attribute
- Approach based on probing. 2 phases
  - Probe S>K rows.
  - Get all rows with value less than the K:th element in the result after the probe