# Big Data Analytics: Why Sharing Scans Is Not Enough

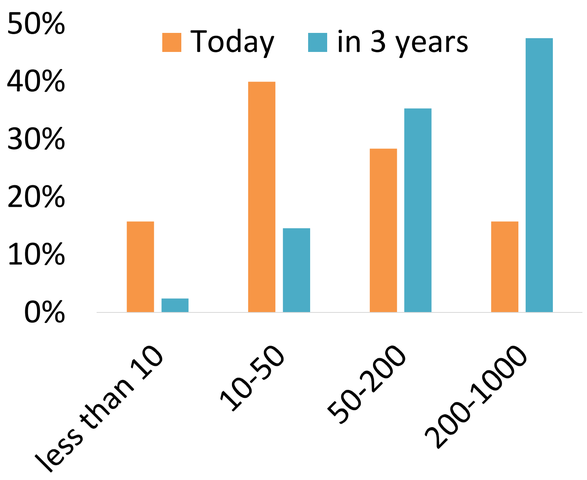
Reactive and Proactive Sharing Across Concurrent Analytical Queries

Iraklis Psaroudakis, Manos Athanassoulis, Matthaios Olma, and Anastasia Ailamaki

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

## Sharing and parallelism: friends or foes?

50% of analytical applications will have 100s-1000s of concurrent clients by 2015*

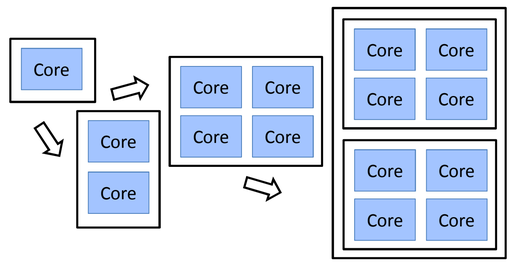



*High-Performance Data Warehousing, P. Russom, 2012

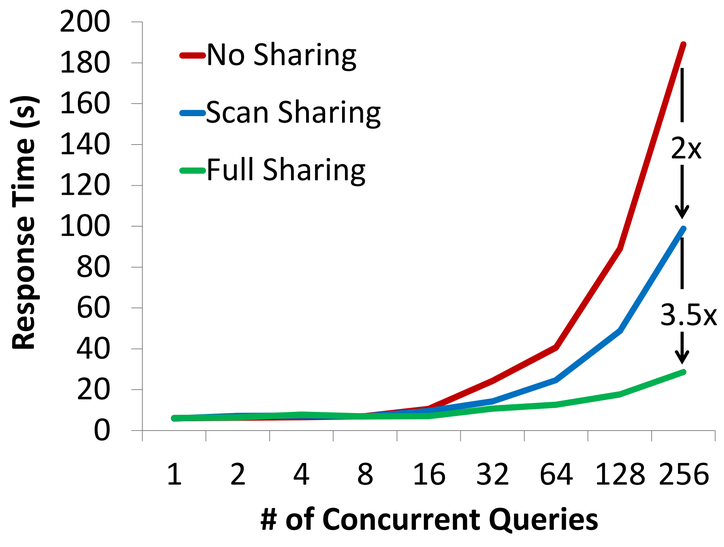
**Sharing only scans exploits 50% of potential improvement ➔**

**➔ Concurrency is increasing**

**#cores and #sockets follow Moore's law**

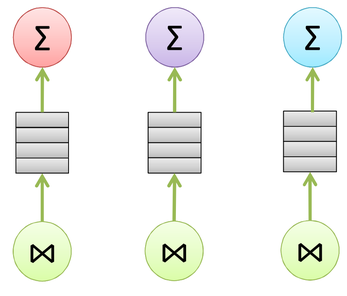


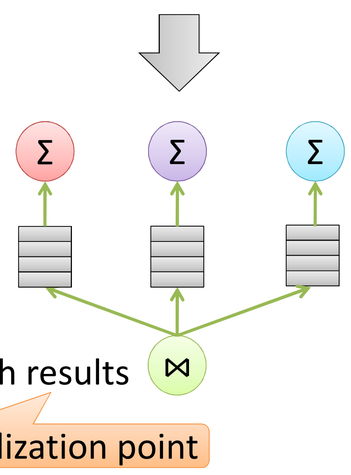

Sharing only scans misses out on performance improvement by a factor of 3.5x




## Sharing Efficiency vs. Parallelism

### Reactive Sharing (push)

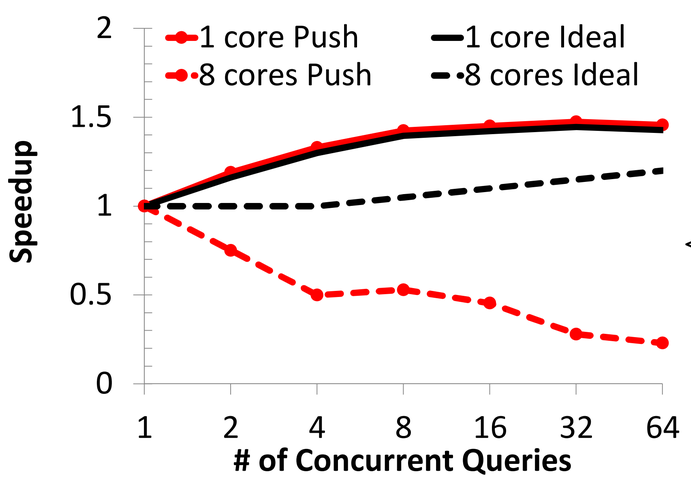


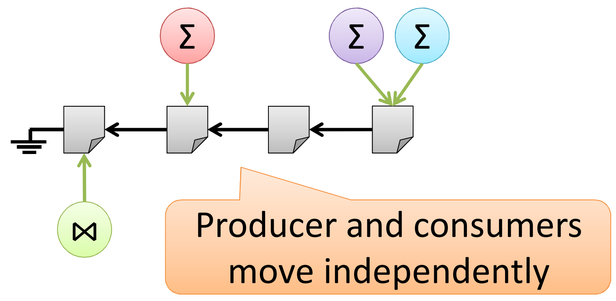

**Optimize for Latency**

✓Opportunistically share common results

✓Minimize sharing cost

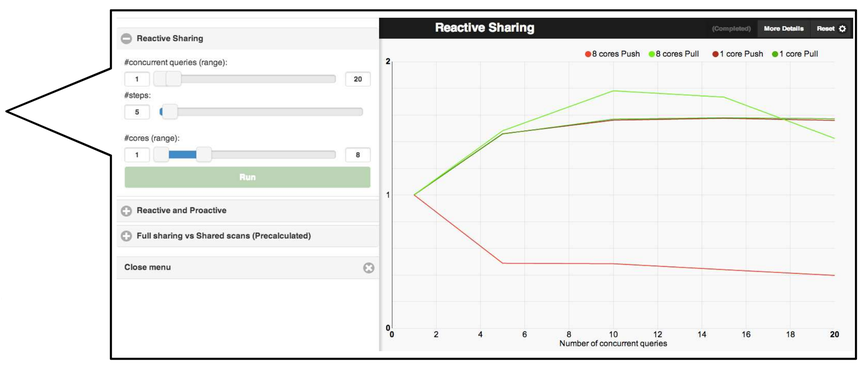



### Reactive Sharing (pull)




**Efficient Sharing in presence of Parallelism**

✓Decouple producer from forwarding

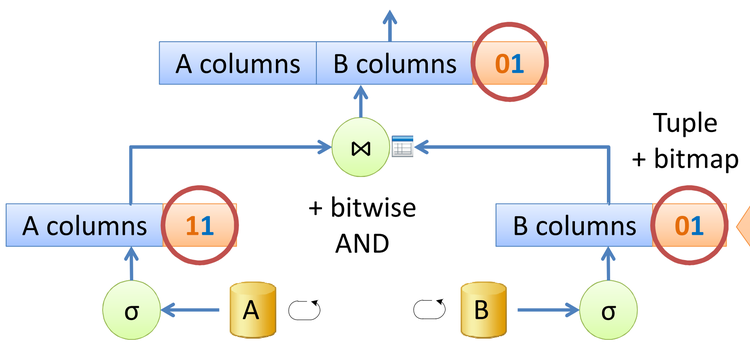



## Optimizing for Latency and Throughput

### Proactive Sharing

$Q_1$ SELECT * FROM *A, B* WHERE $A.c_1 = B.c_1$ AND σ(A) AND σ(B)

$Q_2$ SELECT * FROM *A, B* WHERE $A.c_1 = B.c_1$ AND σ'(A) AND σ'(B)




**Proactive with Reactive**

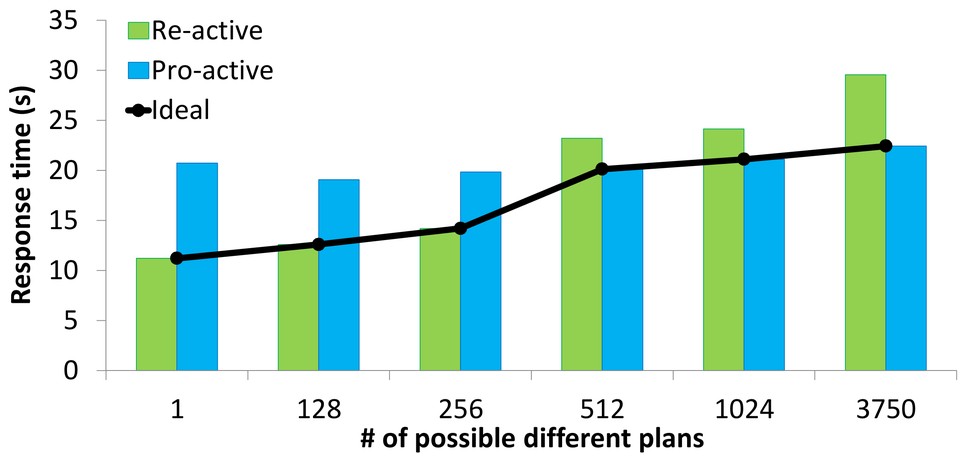
✓Collapses common sub-plans through shared operators

✓Collapses the bits of common sub-plans

✓Eliminates unnecessary book-keeping overhead

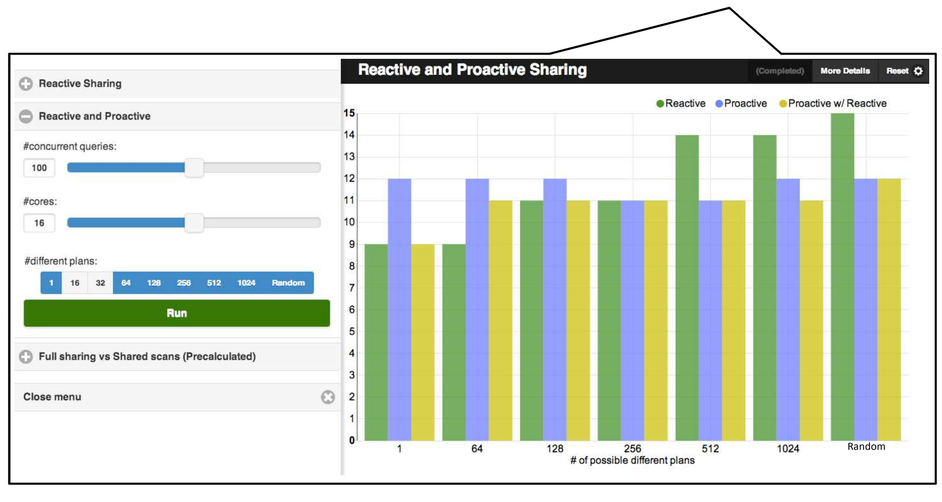
**Machine**: 2x8-core Xeon, 64 GB RAM. **Storage manager**: Shore-MT+Qpipe+CJOIN. **Workload**: Star-schema benchmark

256 concurrent SSB Q3.2 queries




**Optimize for Throughput**

✓Shared operators built up to expect a high number of concurrent queries




DATA-INTENSIVE APPLICATIONS AND SYSTEMS