CWI
Centrum Wiskunde & Informatica

COMMIT/

Distribution-Oblivious Databases

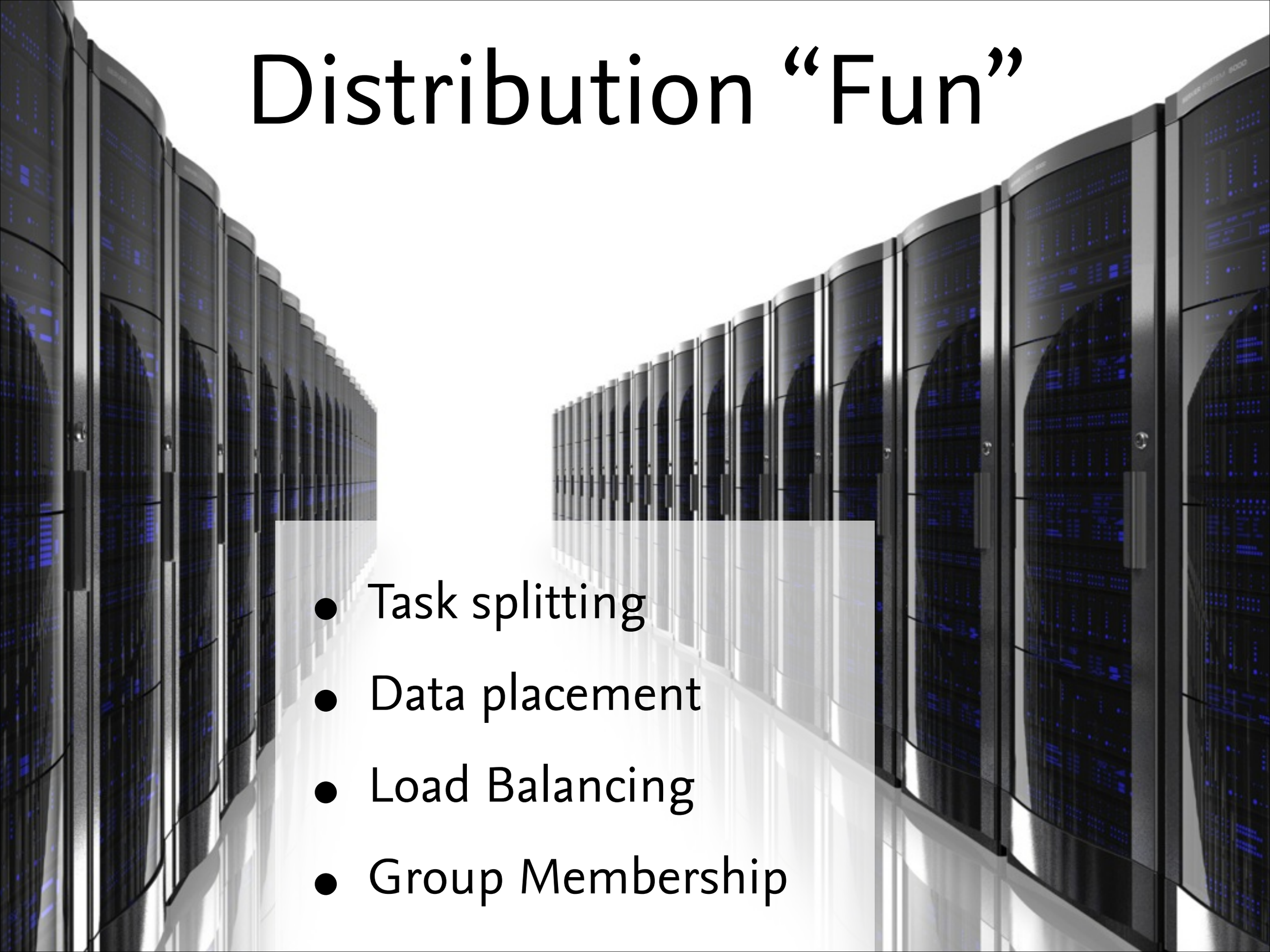
DBDBD, 2013-11-29

Hannes Mühleisen

Environment

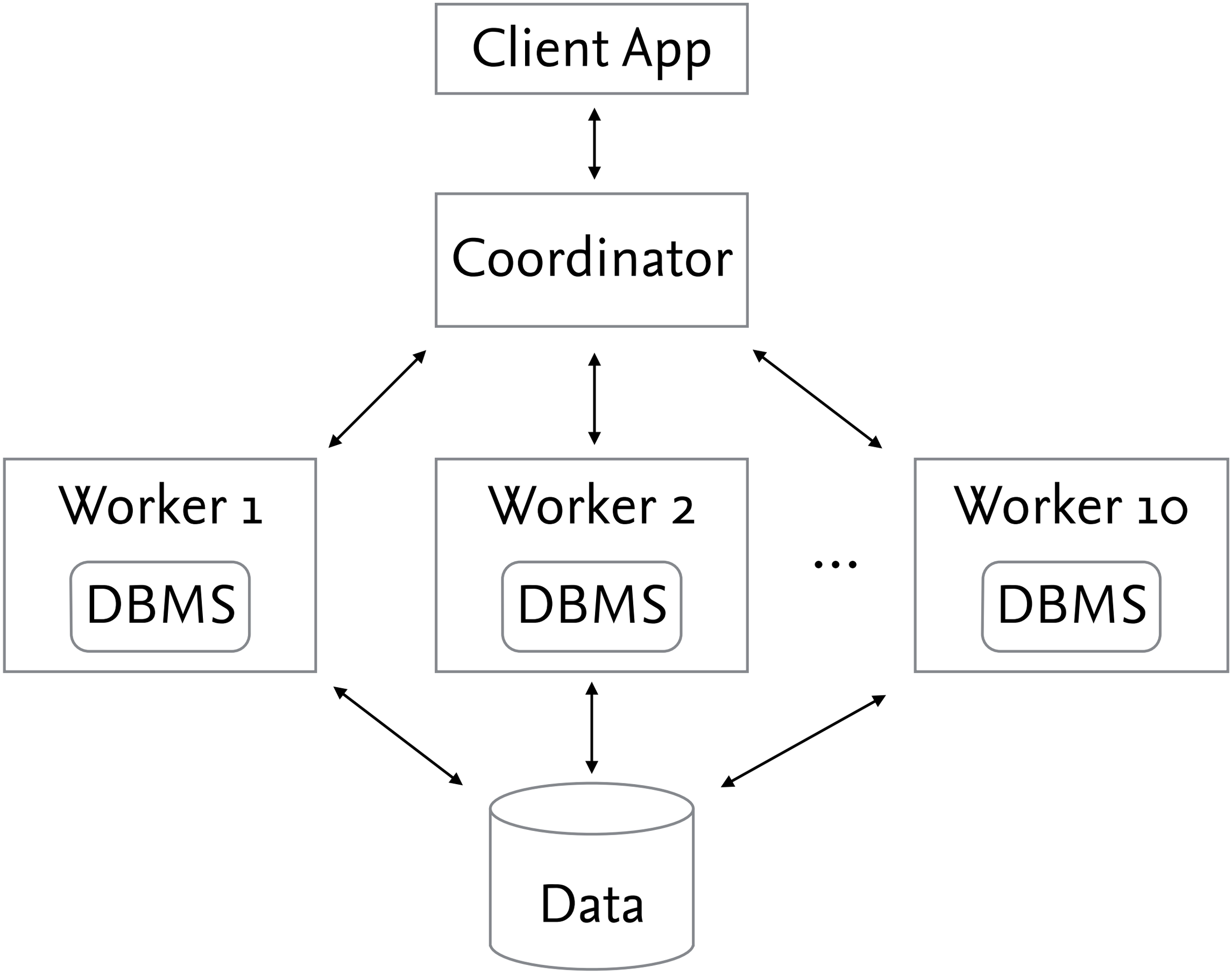
- Analytical queries and growing data volume
  - but scale-up (bigger box) limited ($$$)
- SQL scale-out (more boxes)?
  - a.k.a. “making complex pieces of software even more complex”

Distribution “Fun”



- Task splitting
- Data placement
- Load Balancing
- Group Membership

Can we run SQL databases oblivous to distributed processing and still get scale-out efficiency benefits?


Client App
Coordinator
Worker 1
DBMS
Worker 2
DBMS
...
Worker 10
DBMS
Data

Client App

```
SELECT SUM(l_extendedprice *
 l_discount) FROM lineitem;
```

Coordinator

```
SELECT
SUM(l_extendedprice *
 l_discount) FROM
  lineitem WHERE
l_orderkey >= 0 AND
l_orderkey <= 6000;
```

```
SELECT SUM(l_extendedprice
* l_discount) FROM lineitem
WHERE l_orderkey > 6000 AND
    l_orderkey <= 12000;
```

```
SELECT SUM(l_extendedprice
* l_discount) FROM lineitem
 WHERE l_orderkey > 54000
 AND l_orderkey <= 60000;
```

Worker 1

DBMS

Worker 2

DBMS

...

Worker 10

DBMS

Client App

SUM
114698450836.4234

Coordinator

SUM
11363474.9494

SUM
11474692.5211

SUM
11600867.7359

Worker 1

DBMS

Worker 2

DBMS

...

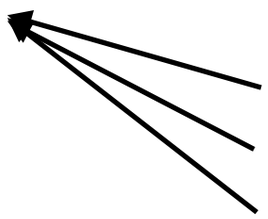
Worker 10

DBMS

```
SELECT l_returnflag,
AVG(l_extendedprice *
l_discount) FROM lineitem
GROUP BY l_returnflag;
```

```
SELECT l_returnflag,
AVG(l_extendedprice * l_discount),
COUNT(*) AS __groupcount FROM lineitem
WHERE l_orderkey >= 0 AND l_orderkey <=
6000 GROUP BY l_returnflag;
```

l_returnflag	SUM
N	1911.9013
R	1912.4650
A	1912.0829



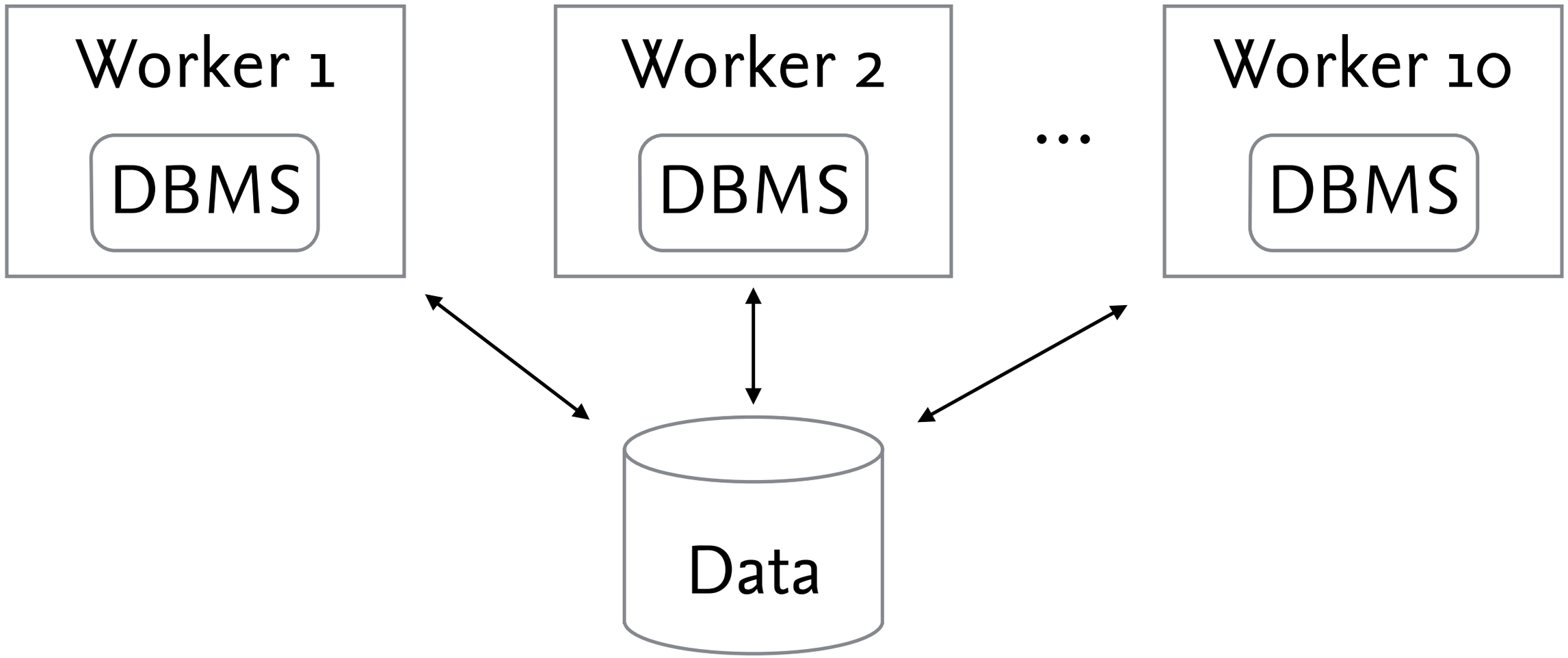
l_returnflag	SUM	__groupcount
N	1885.3028	3077
R	1867.0046	1459
A	1915.2755	1482

SQL level parallelism

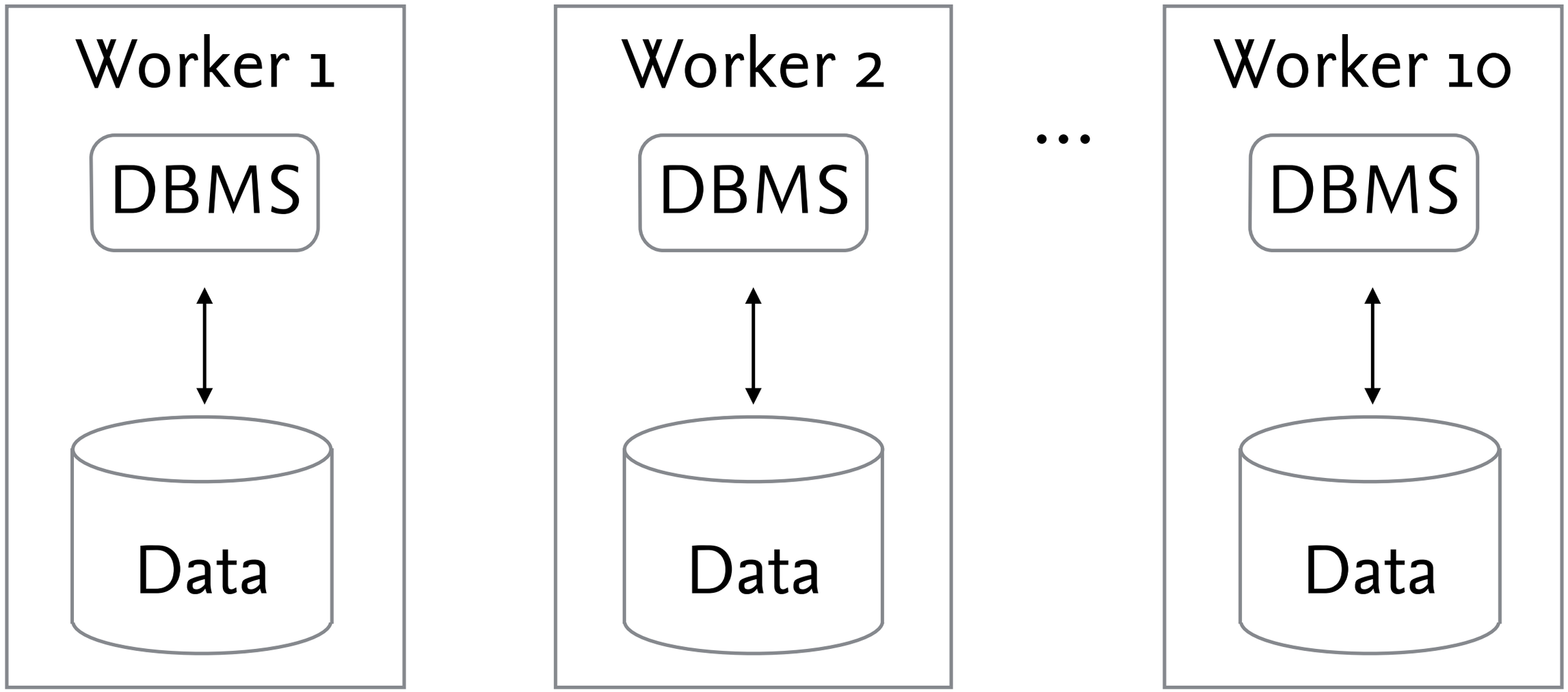
- Coordinator recieves queries, splits them up into smaller subqueries
- Workers calculate result for subqueries, send result to coordinator
- Coordinator merges and delivers results

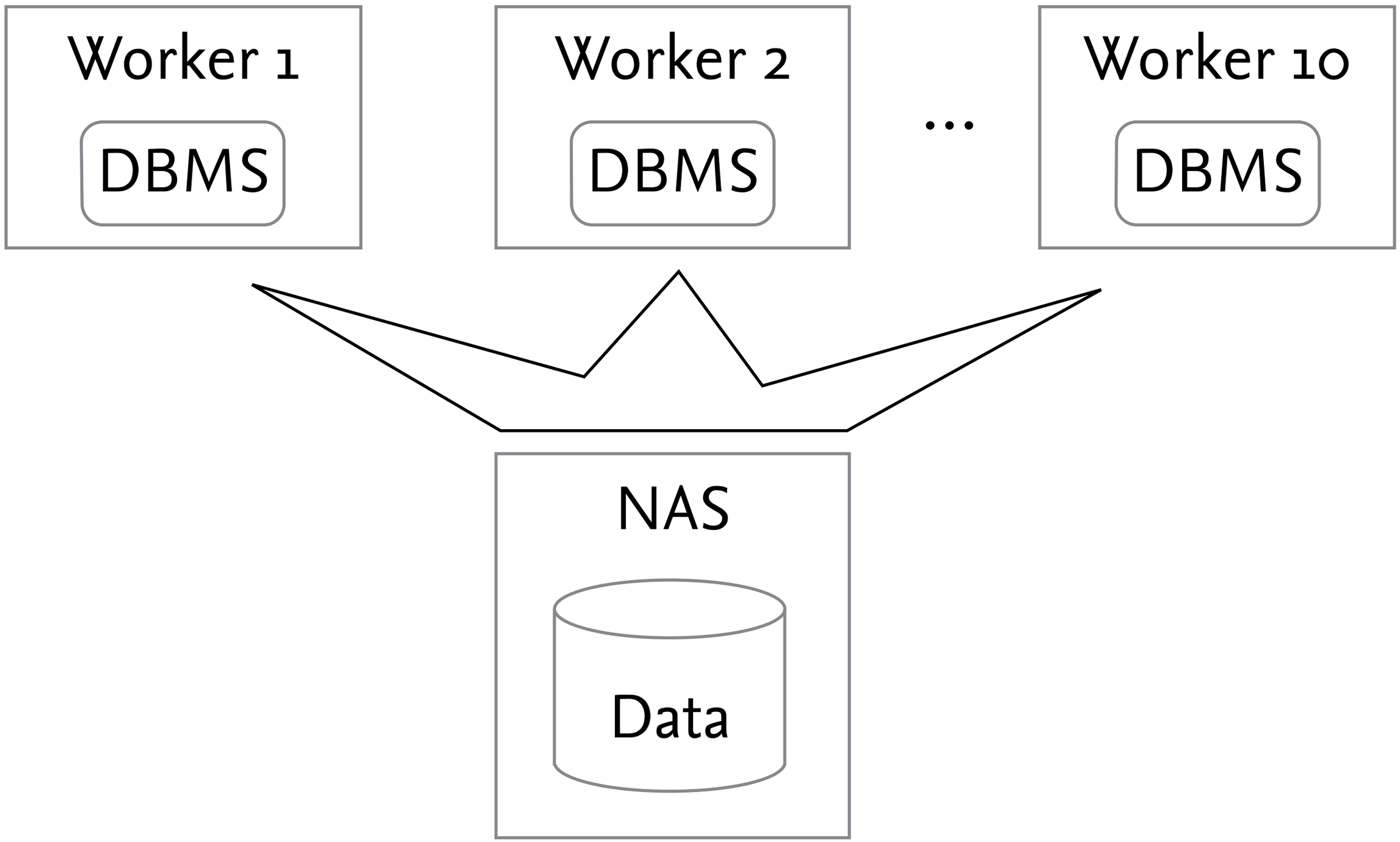
Yang, Yen, Tan & Madden:
Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database

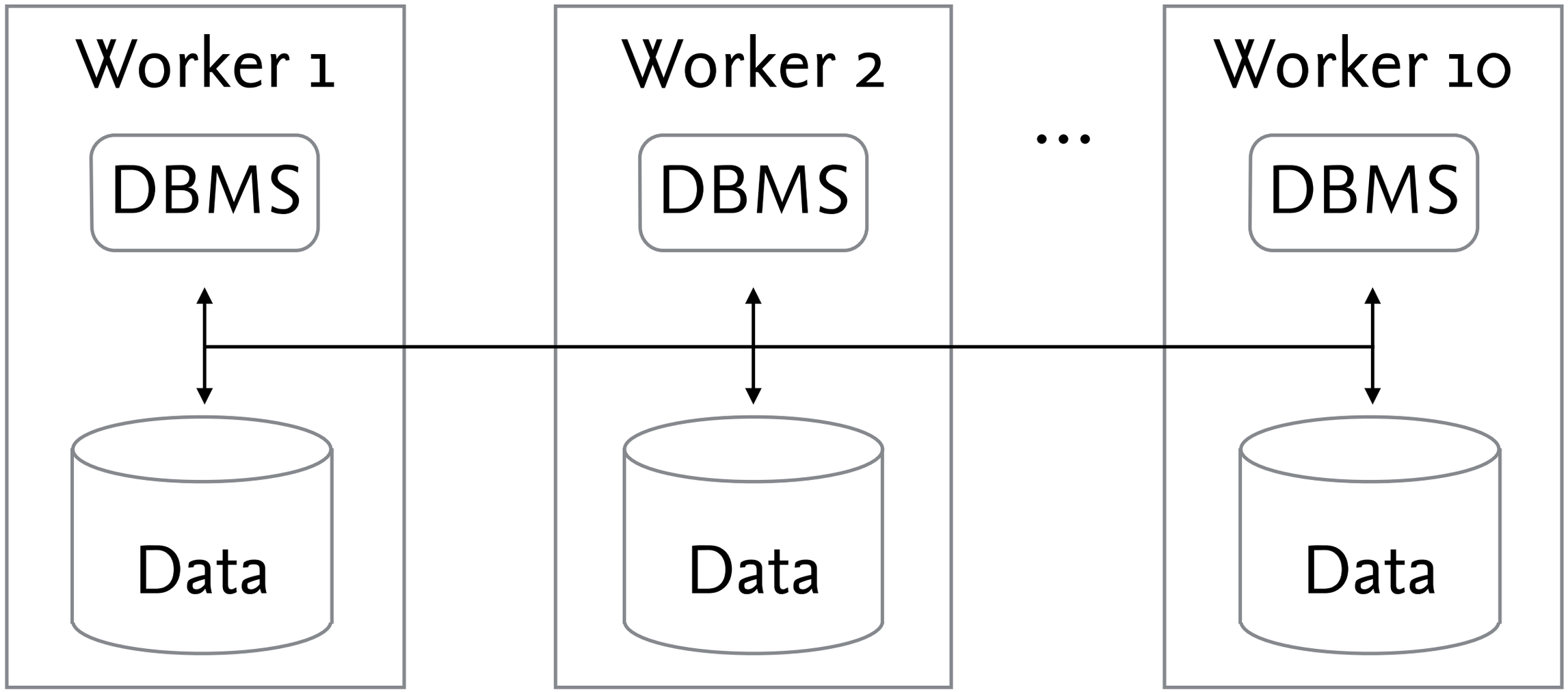
Lima, Mattoso & Valduriez:
Adaptive Virtual Partitioning for OLAP Query Processing in a Database Cluster



Stonebraker:
The Case for Shared Nothing


Worker 1
DBMS
Data
Worker 2
DBMS
Data
...
Worker 10
DBMS
Data


Worker 1
DBMS
Worker 2
DBMS
...
Worker 10
DBMS
NAS
Data


Worker 1
DBMS
Data
Worker 2
DBMS
Data
...
Worker 10
DBMS
Data

Caches?

- What about subquery locality?
- DB Buffer Mgr/OS implement LRU caches for DB data
- Can we predict FS IO from SQL queries?
  - We do not have to! All we need is mapping consistency.

Q1

```
SELECT MIN(C2) FROM T1,T2
    WHERE C5 > 42;
```

T1

C1	C2	C3	C4

T2

C2	C5

Q2

```
SELECT MAX(C2) FROM T1,T3
    WHERE C6 < 84;
```

T1

C1	C2	C3	C4

T3

C2	C6

Q1/3

```
SELECT MIN(C2) FROM T1,T2
        WHERE C5 > 42
AND C1 >= 1000 AND C1 < 2000;
```

T1

C1	C2	C3	C4

T2

C2	C5

Q2/3

```
SELECT MAX(C2) FROM T1,T3
        WHERE C6 < 84
AND C1 >= 1000 AND C1 < 2000;
```

T1

C1	C2	C3	C4

T3

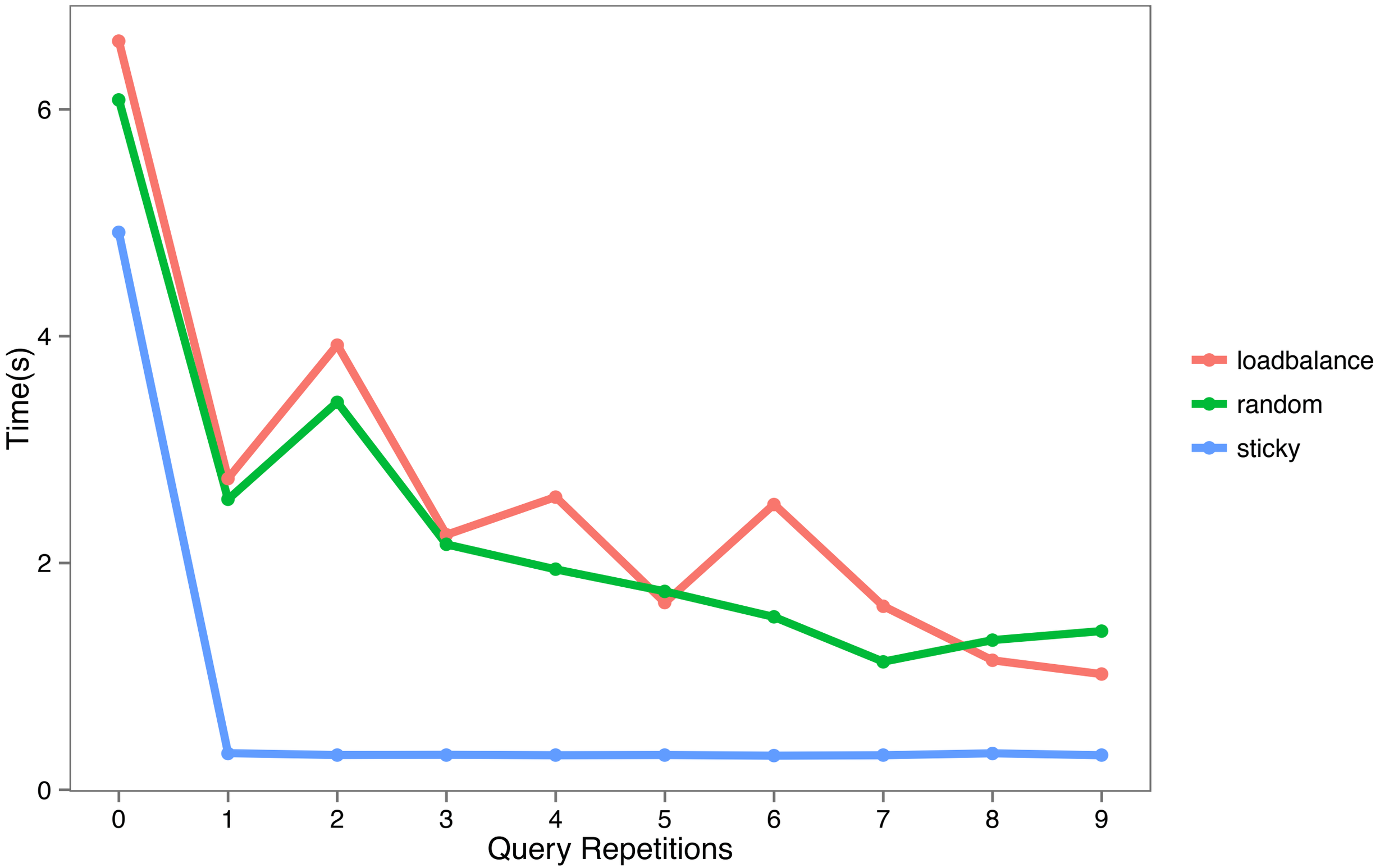
C2	C6

* Assuming some DBMS sanity

Subquery Assignment

- Hope: If we assign subqueries with equal fact table restrictions to same worker, we can make max. use of cached data
- Experiments on 10 nodes, SSBM Q11, 10 times from cold start
  - Sticky
  - Random
  - Balance

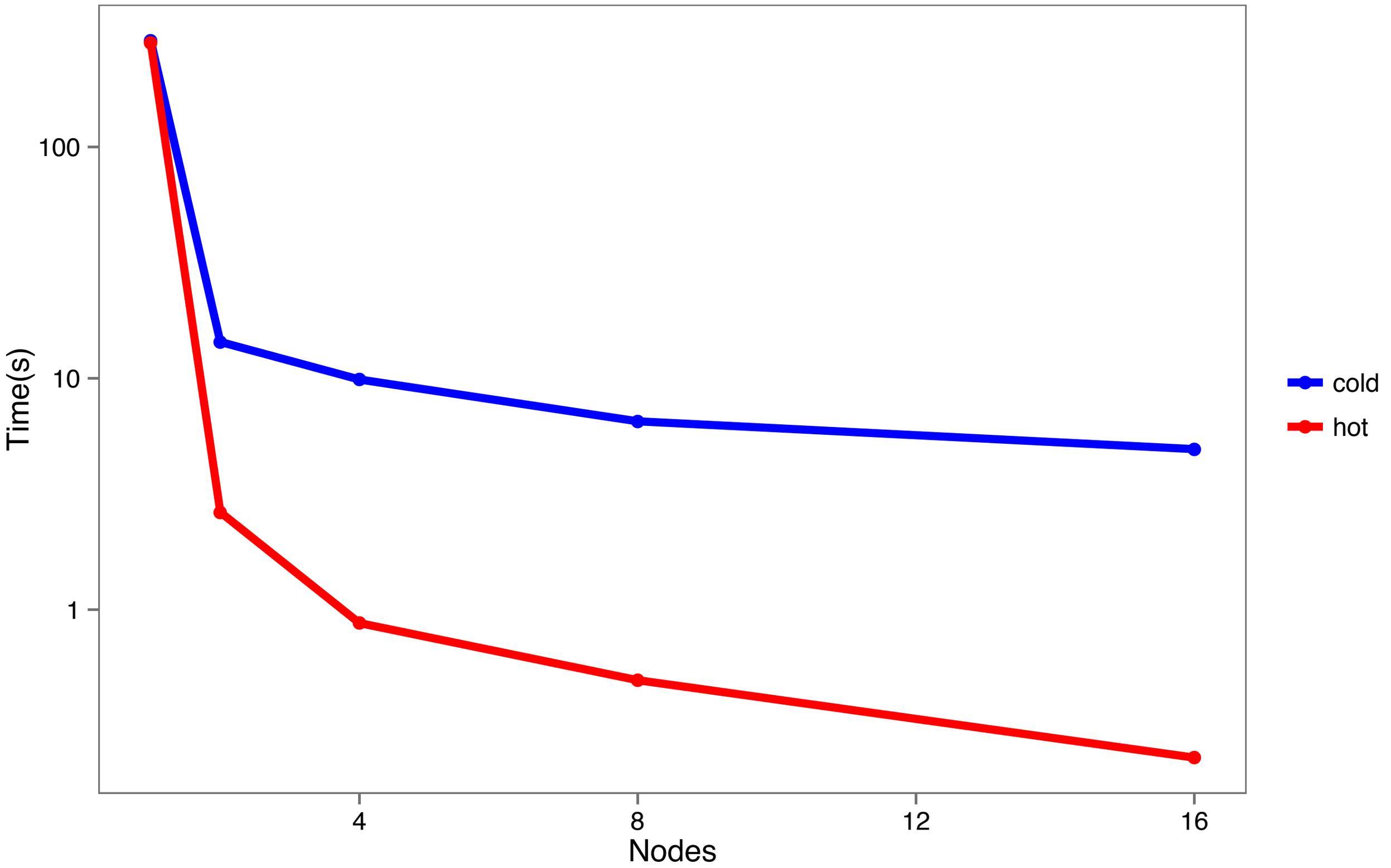
Subquery Assignment, SSBM Q11, 10 Nodes



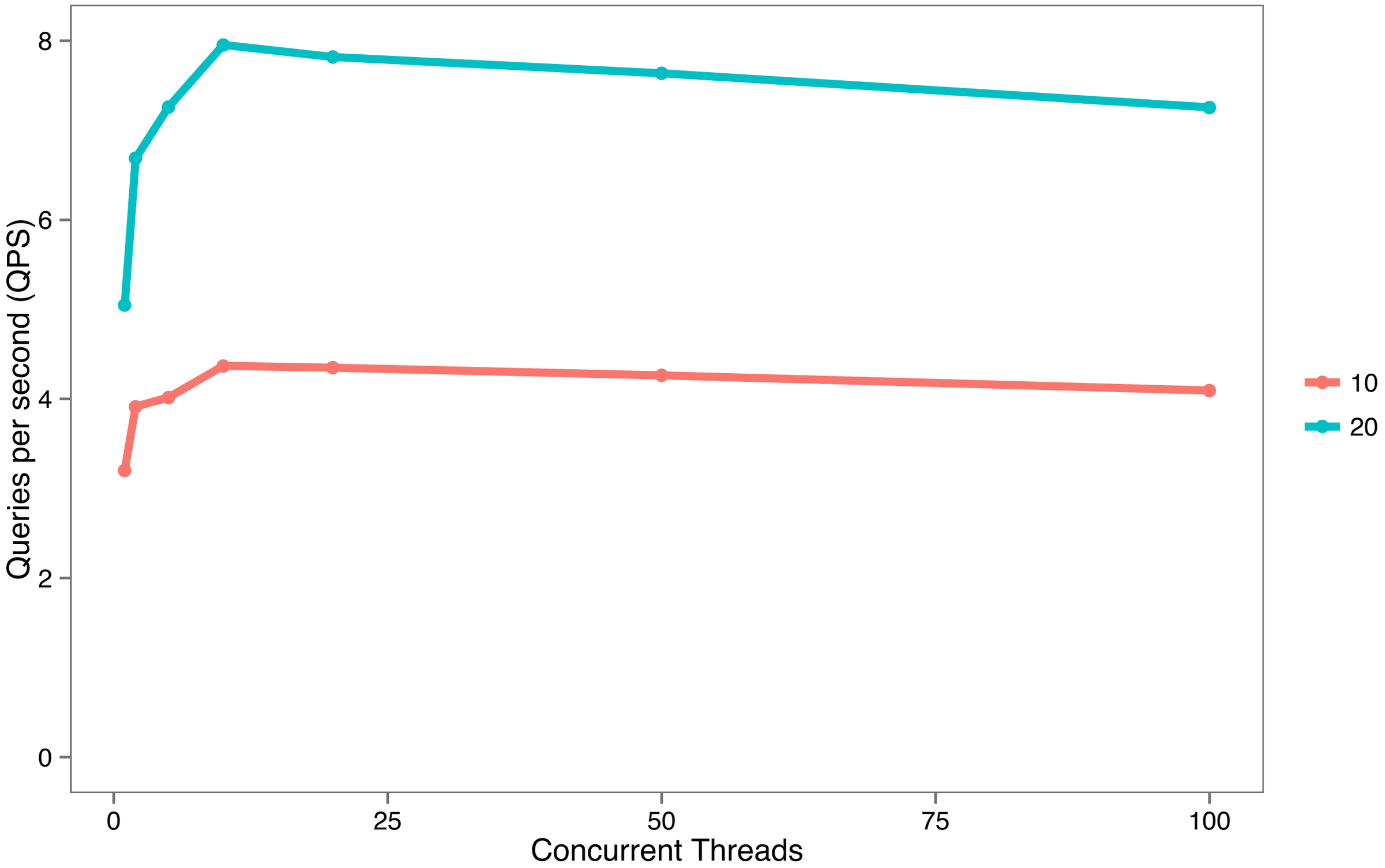
Scaleup

- Latency: More nodes, less query latency
- Throughput: More nodes, more parallel queries

Scaleup Latency, SSBM Q11



Scaleup Throughput, SSBM Q11, Hot



Thank You!

Questions?

Shard Size Impact – SSBM SF100, 10 rocks

10 50 100 200 1000

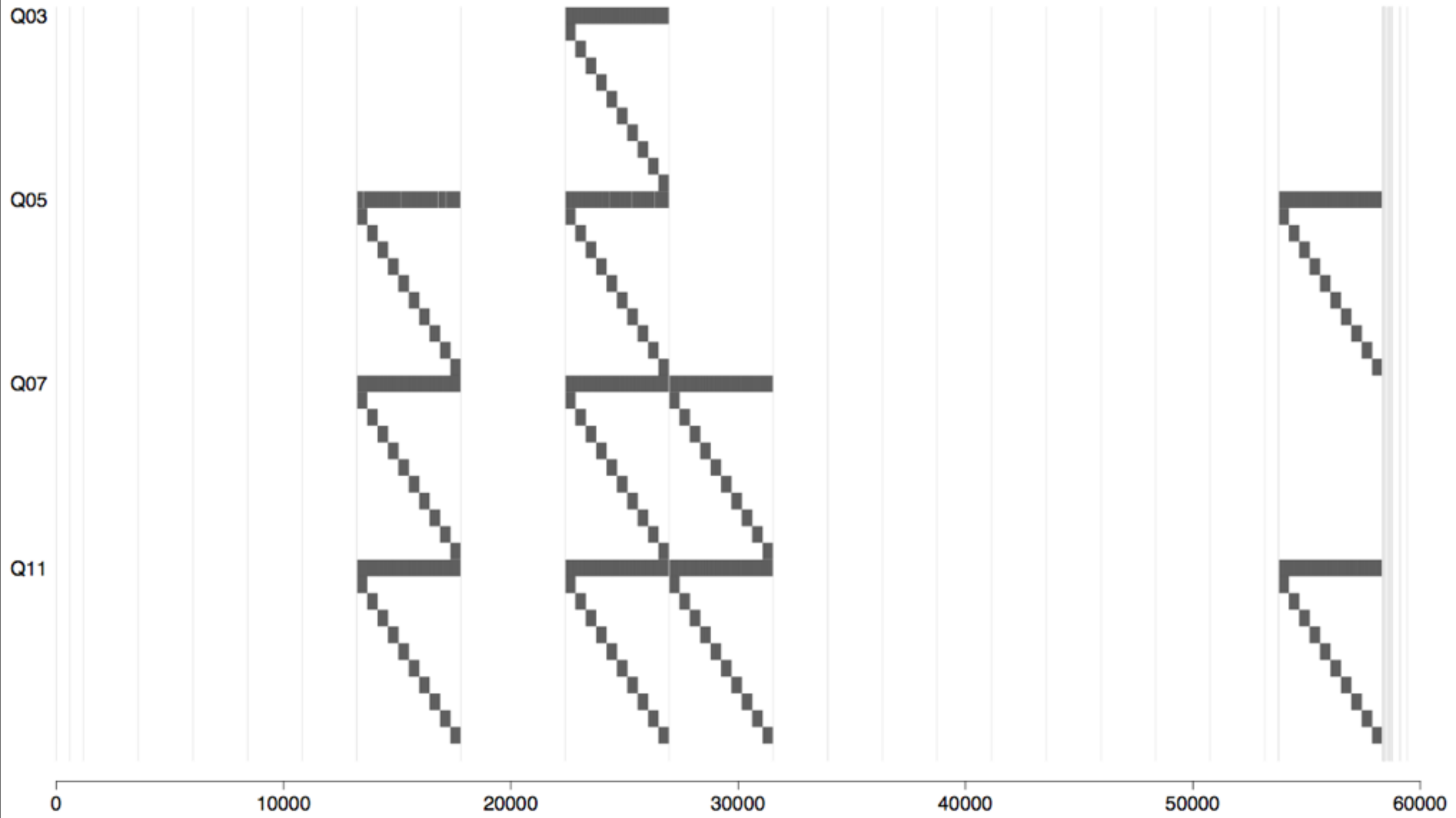
Response Time (s)

10.0
7.5
5.0
2.5
0.0

Q01 Q02 Q03 Q04 Q05 Q06 Q08 Q09 Q10 Q01 Q02 Q03 Q04 Q05 Q06 Q08 Q09 Q10 Q01 Q02 Q03 Q04 Q05 Q06 Q08 Q09 Q10 Q01 Q02 Q03 Q04 Q05 Q06 Q08 Q09 Q10 Q01 Q02 Q03 Q04 Q05 Q06 Q08 Q09 Q10

SSBM Query/Shardsize

File IO SSBM SF-100, MonetDB



File IO SSBM SF-100, PostgreSQL