Joins on many machines

csc343, Introduction to Databases
Renée Miller and Diane Horton
Fall 2016

Slides adapted from Suciu & Balazinska

UNIVERSITY OF TORONTO

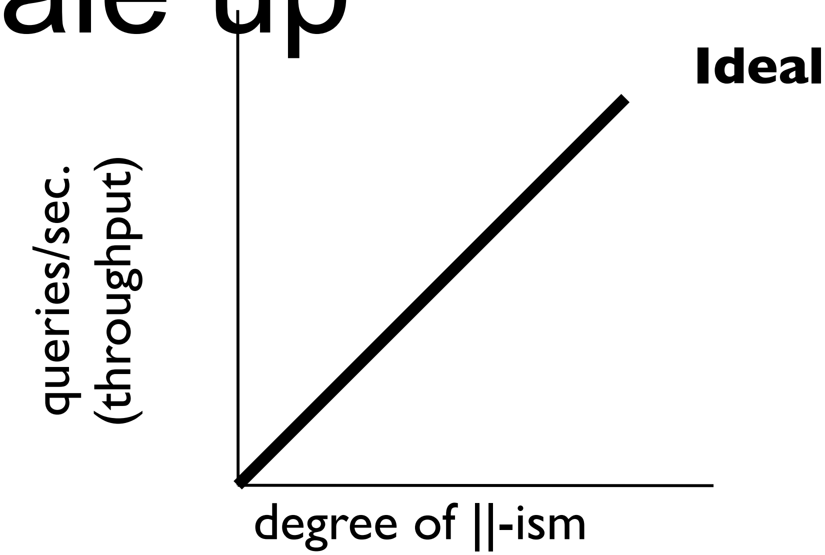
Why Use Many Machines?

2

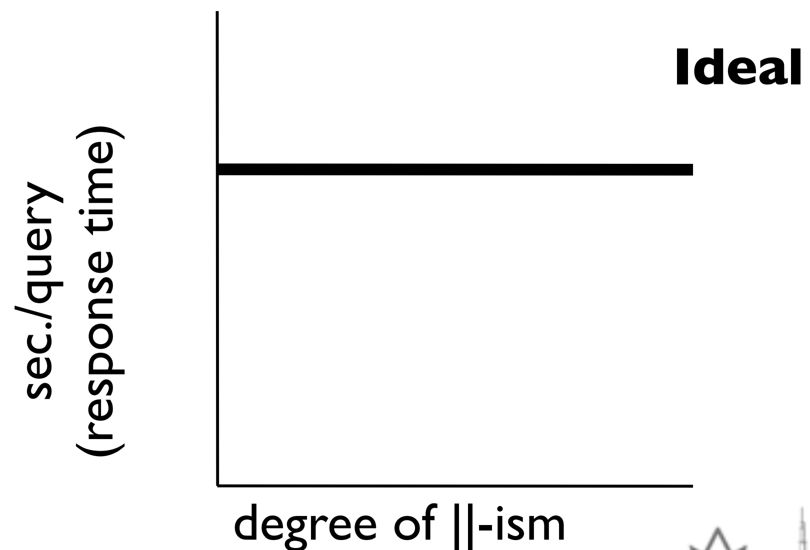
- Improve performance
  - How do we measure performance?
    - **Through-put**
      - Number of queries that can be answered in a unit of time
    - **Response time**
      - Time required per query

Speed up and Scale up

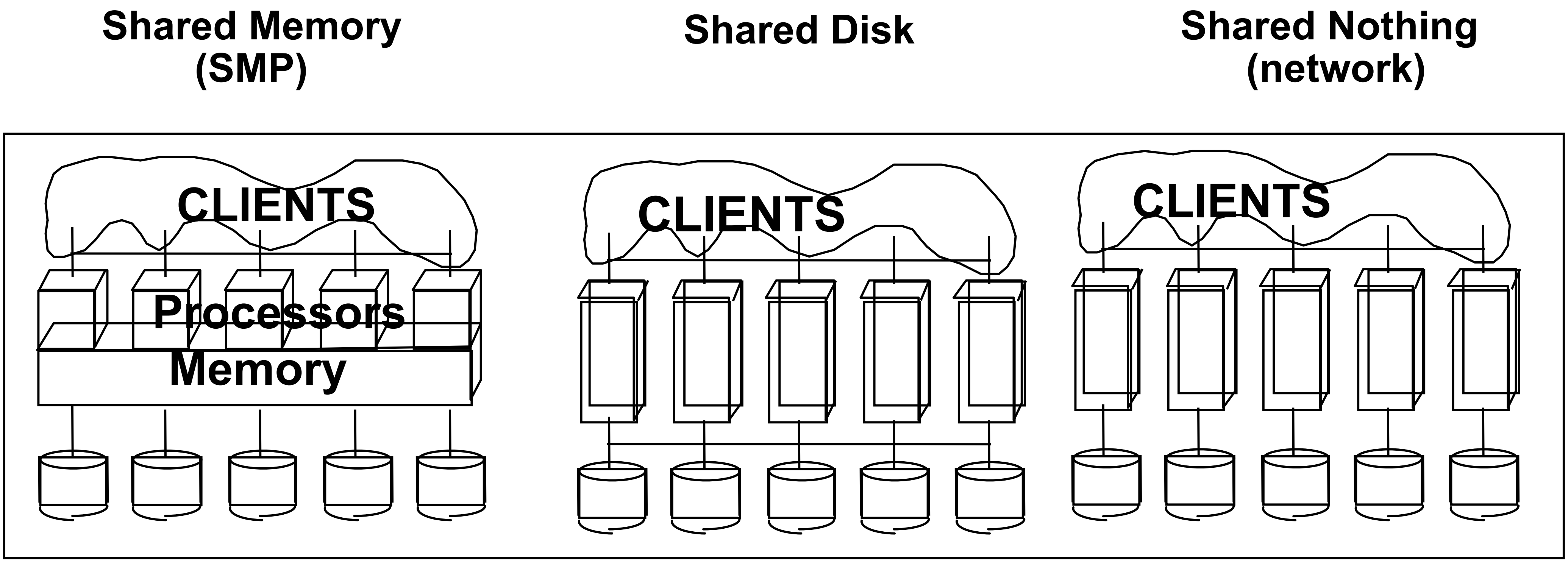
- Speed Up
  - More resources means proportionally less time for given amount of data.



- Scale Up
  - If resources increased in proportion to increase in data size, time is constant.



What does many machines mean?



Easy to program
Expensive to build
Difficult to scaleup

Hard to program
Cheap to build
Easy to scaleup

Parallel DBMS

- Inter-query parallelism
  - Each query runs on one processor
  - Only for OLTP (online transaction processing)
- Inter-operator parallelism
  - A query runs on multiple processors
  - An operator runs on one processor
  - For both OLTP and OLAP (online analytic processing, aka Decision Support)
- Intra-operator parallelism
  - An operator runs on multiple processors
  - For both OLTP and OLAP

Main parallelism in DBMS since 1980's

UofT:DB Group

Horizontal Data Partitioning

- Have a large table R(K, A, B, C)
  - Need to partition on a shared-nothing architecture into P chunks $R_1, \ldots, R_P$, stored at the P nodes
- Partition Size: $size(R_1) \approx \ldots \approx size(R_P)$
- Hash partitioned on attribute A:
  - Tuple t goes to chunk i, where $i = h(t.A) \bmod P + 1$
- Range partitioned on attribute A:
  - Partition the range of A into $-\infty = v_0 < v_1 < \ldots < v_P = \infty$
  - Equiwidth or equidepth
  - Tuple t goes to chunk i, if $v_{i-1} < t.A < v_i$

Parallel GroupBy

- Q: SELECT sum(C)
  FROM R
  GROUP BY X
- If R is partitioned on X, then each node computes the group-by *locally* and computes sum *locally*
- Otherwise, hash-partition R on X
  - each value of X goes to one node
  - then compute group-by locally

Speed up and Scale up

- The runtime is dominated by the time to read the chunks from disk, i.e., size(Ri)

- If we double the number of nodes P, what is the new response time of Q?

- If we double both P and the size of the relation R, what is the new response time?

Uniform Data v.s. Skewed Data

- Uniform partition:
  - For all i, size(R_1) ≈ … ≈ size(R_P) ≈ size(R) / P
  - Linear speed up, constant scale up
- Skewed partition:
  - For some i, size(R_i) >> size(R) / P
  - Speed up and scale up will suffer

UofT:DB Group

Uniform Data v.s. Skewed Data

10

- Let R(K,A,B,C); which of the following partition methods may result in skewed partitions?

• Block partition

Uniform

• Hash-partition

– On the key K

– On the attribute A

Uniform
Text

Assuming perfect uniform hash

May be skewed

• Range-partition

– On the key K

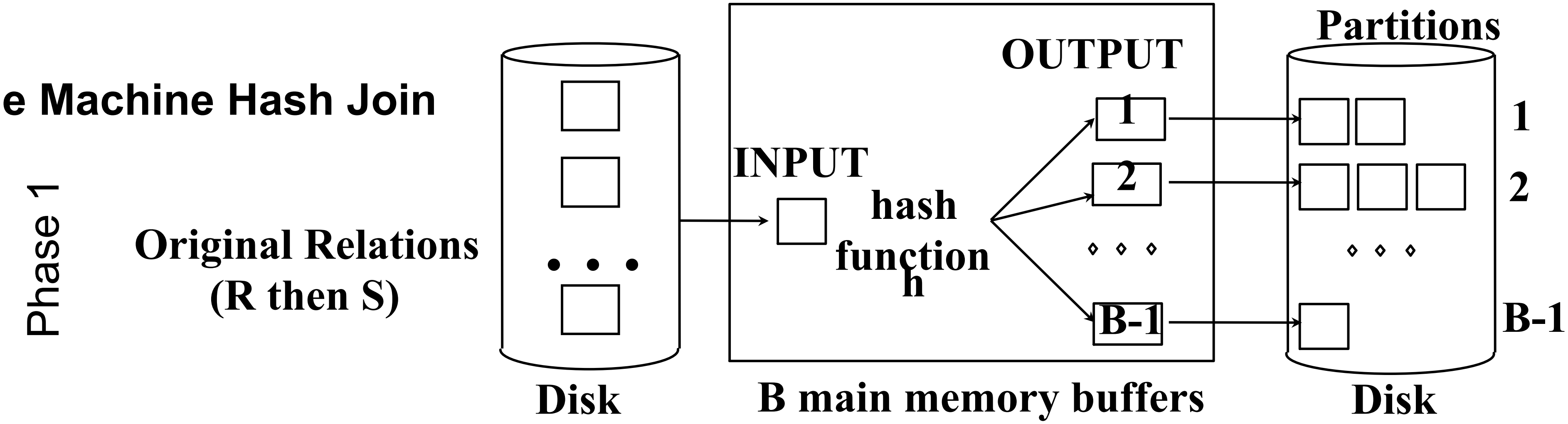
– On the attribute A

May be skewed

Difficult to maintain perfect range-partitioning

Hash Join

One Machine Hash Join

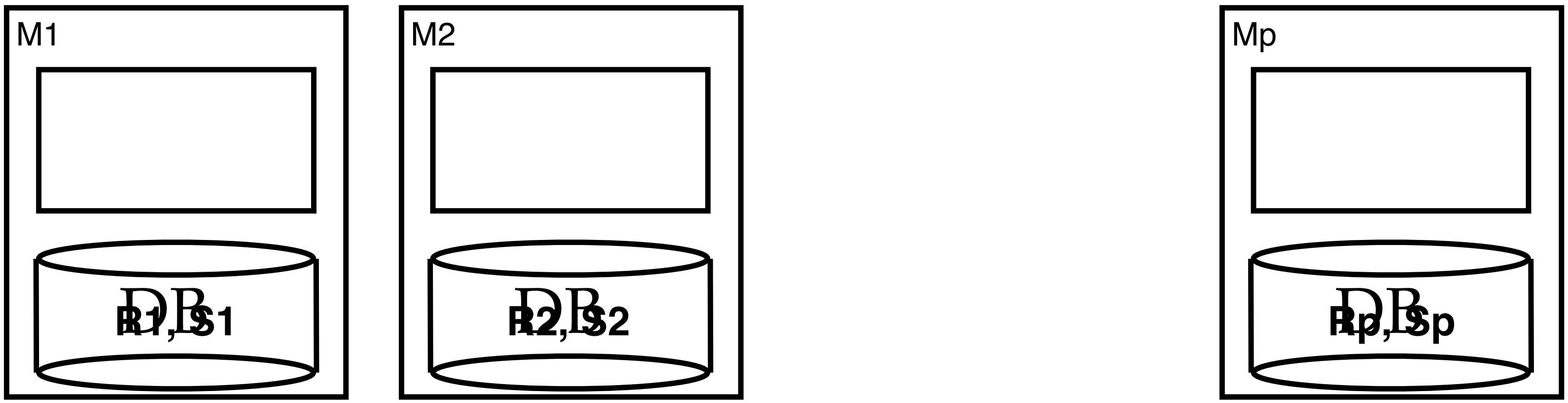


Parallel Hash Join

- In first phase, each machine hashes its data
  - Each hash partition (R & S) sent to a single machine
- Each machine does join on its partition
  - Can use any (non-parallel) join algorithm

Parallel Hash Join $R \bowtie S$

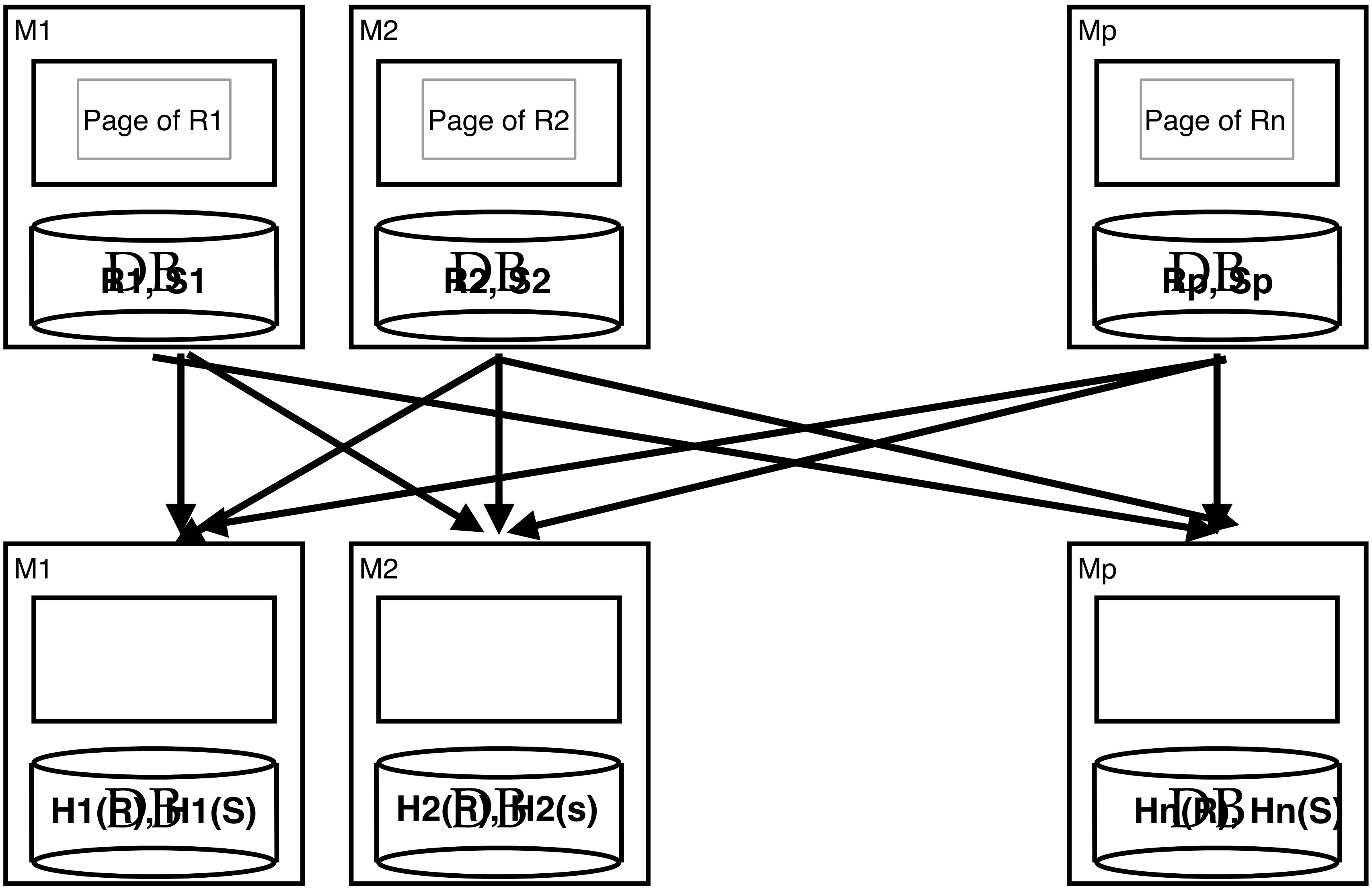
12



- Each machine hashes R on bid
  - tuples that hash to 1 sent to machine 1
  - tuples that hash to 2 sent to machine 2
  - …
  - tuples that hash to p sent to machine

Parallel Hash Join R ⋈ S

13



UofT:DB Group

Parallel Hash Join

- Once each machine contains only same hash partitions of both R and S, then locally any join algorithm can be applied.