CS 443
Parallel DB

Winter 2013

Adapted from Suciu & Balazinska

Parallel DBMS

- Inter-query parallelism
  - Each query runs on one processor
  - Only for OLTP queries
- Inter-operator parallelism
  - A query runs on multiple processors
  - An operator runs on one processor
  - For both OLTP and Decision Support
- Intra-operator parallelism
  - An operator runs on multiple processors
  - For both OLTP and Decision Support
  - Main parallelism used in parallal DBMS since 1980's

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
- Block Partition: $size(R_1) \approx \ldots \approx size(R_P)$
- 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

- R(K,A,B,C), how could we compute these GroupBy's, for each of the partitions
  - $\gamma_{A,sum(C)}(R)$
- If R is partitioned on A, then each node computes the group-by locally
- Otherwise, hash-partition R(K,A,B,C) on A, then compute group-by locally

Performance Metric: Parallel DBMS

- P = the number of nodes (processors, computers)
- Speedup:
  - More nodes, same data leads to higher speed
- Scaleup:
  - More nodes, more data leads to same speed

- OLTP: “Speed” = transactions per second (TPS)
- Decision Support: “Speed” = query time

Speedup and Scaleup

- The runtime is dominated by the time to read the chunks from disk, i.e. size(R_i)
- If we double the number of nodes P, what is the new running time of $\gamma_{A,sum(C)}(R)$?
- If we double both P and the size of the relation R, what is the new running time?

Uniform Data v.s. Skewed Data

- Uniform partition:
  - size(R_1) ≈ … ≈ size(R_P) ≈ size(R) / P
  - Linear speedup, constant scaleup

- Skewed partition:
  - For some i, size(R_i) >> size(R) / P
  - Speedup and scaleup will suffer

Uniform Data v.s. Skewed Data

- Let R(K,A,B,C); which of the following partition methods may result in skewed partitions?
  - Block partition — Uniform
  - Hash-partition — Uniform
    - On the key K — Text
    - On the attribute A — May be skewed

Assuming perfect uniform hash

  - Range-partition
    - On the key K — May be skewed
    - On the attribute A

Difficult to maintain perfect range-partitioning

Parallel Join?

- R(A,B) join on B with S(B,C)