Query Processing on Clusters

Communication-Cost Model

Multiway Joins

Recursion

Environment

- Computing cluster with distributed file system.
  - E.g., GFS, HDFS.
- Map-reduce implementation.
  - E.g., Hadoop.
  - Or extension to general acyclic workflow, as in Clustera, Hyracks.

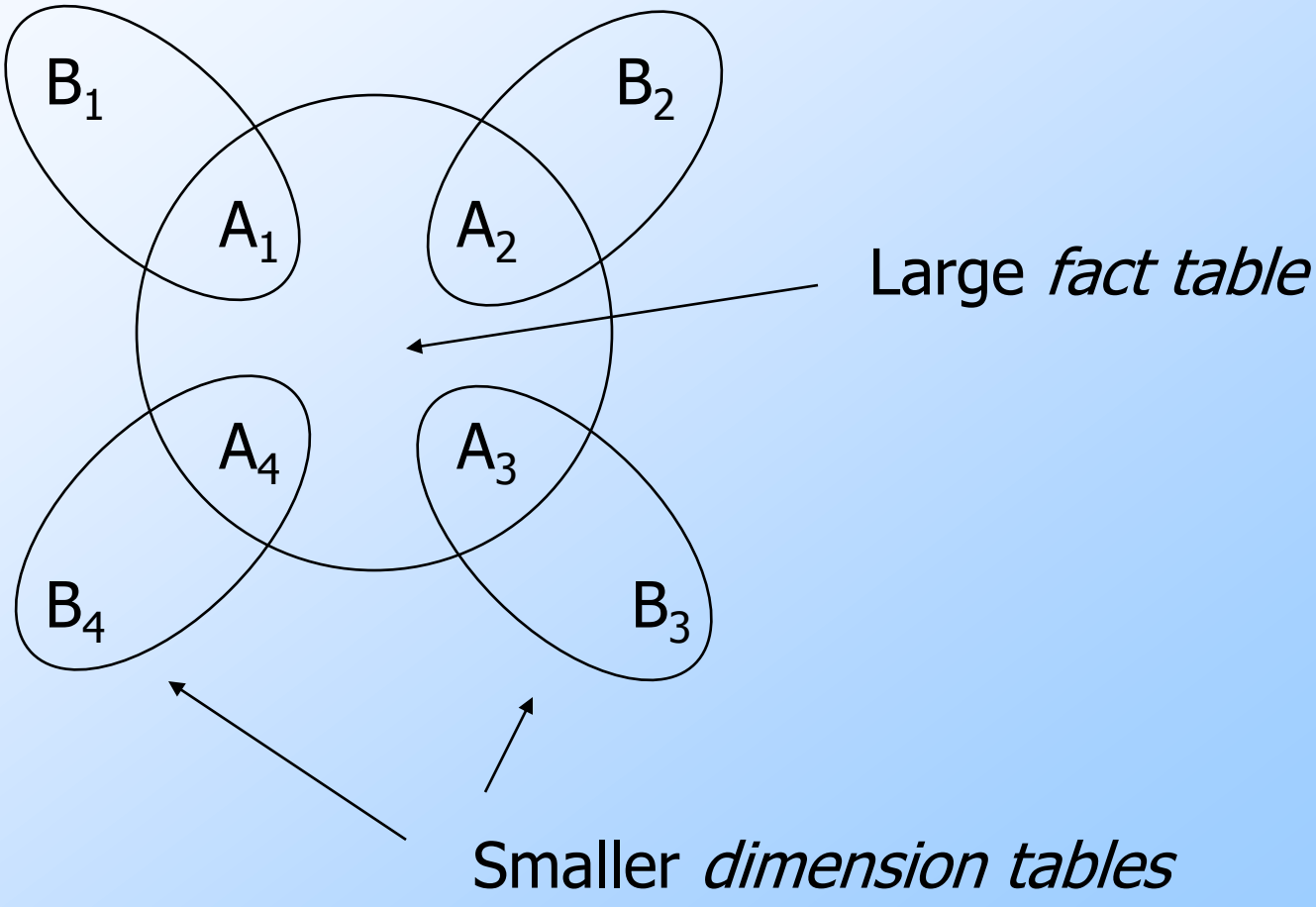
Communication Cost

- Assumption: efficiency of an algorithm is tied to the sum of the input sizes to all tasks.
- Justification:
  1. Typically simple, main-memory operation at each task, e.g., hash-join.
  2. Large outputs are either input to another task or aggregated in final result.

Multiway Join

- Afrati/Ullman EDBT-2010.
- Key idea: sometimes, a cascade of 2-way joins, each implemented by a map-reduce stage, is less efficient than a single multiway join.
- Intuition: if intermediate result is large, communicating it costs more than replicating tuples of the arguments.

Example: Star-Join



Implementing a Star Join

- Use the A's as a hash key, so the Map tasks hash each fact tuple to one Reduce task corresponding to a bucket.
  - Optimization: each A_i is hashed a number of ways inversely proportional to the size of the dimension table (A_i, B_i).
  - A tuple of dimension table *i* is sent to all Reduce tasks corresponding to its value of A_i (and any buckets for the other A's).

Example: Star Join

- Four dimension tables of equal size.
- 256 Reduce tasks.
- Hash each A_i to 4 buckets.
- Tuple (a,b) of dimension table (A_2, B_2) is sent to 64 Reduce tasks corresponding to hash values (*,h(a),*,*).

Aster Data Approach

- Hash and distribute the fact table permanently.
- Replicate the dimension tables as for a join of all relations.
- But they patented a strange approach that is data-dependent.
- We give optimal partitioning independent of data.

Problem with Recursion

- Map-reduce works because task failures can be handled by restarting only the failed task(s).
- Why possible? Because each task delivers output only at the end.
- But recursive tasks must make outputs and then process more input.

Solutions

1. HaLoop: use iterated map-reduce, with attention to avoiding redistribution of intermediate results.
2. Pregel: use recursive tasks but checkpoint after every few rounds.
   - Rollback on failure, but not too far.

Solutions – (2)

- Work of Afrati, Vinayak Borkar, Mike Carey, Alkis Polyzotis, Ullman.
- If operations are idempotent (e.g., Datalog recursions computing sets), then a recursive task can be restarted anyway.

Solutions – (3)

- In general (not idempotent) case, files passed between tasks are replicated anyway.
- Master controller can restart a task, provide previous input files, but throw away output files previously delivered.