Distributed Aggregation for Data-Parallel Computing: Interfaces and Implementations

Yuan Yu, Pradeep Gunda, Michael Isard

presented by
Risi Thonangi

Topic of discussion

• Groupby-Aggregation

  SELECT avg(income) ... GROUP BY zip

  – Aggregation with user-defined functions
• Compare various distributed executing systems for Groupby-Aggregation
  – MapReduce, Parallel DB, Dryad

Distributed Aggregation (example)

• Top layer distributes input
• Below layer aggregates
• Disadvantage
  • Lot of data in network

Partial aggregation for improved performance

• Save on
  • Network traffic
• Distribute computation
• Costs?
  • more nodes in the pipeline
  • Slightly more computation

Decomposable Functions

• Not all aggregation functions are decomposable
  – Median, ...
• Function H is decomposable if
  – \( H = C(I(x_1 + x_2)) = C(I(x_1) + I(x_2)) \)
  – C and I are commutative
• H is associative-decomposable if
  – H is decomposable and
  – \( C(C(x_1 + x_2) + C(x_3)) = C(C(x_1) + C(x_2 + x_3)) \)

Next...

• Programming interfaces
  – Hadoop, Parallel DB, DryadLINQ
• Applications which need Groupby-Aggregation
• Implementations for distributed aggregation
• Experimental evaluation
Hadoop's interface

- Hadoop's is an Iterator interface
- Comments
  - Programmer has to marshal data to/from system types
  - Interface is restricted

Oracle's interface

- Oracle's interface
  - Accumulator-based interface
  - Comments
    - Not much difference between iterator and accumulator-based interfaces
    - Weak point
      - Complex aggregation functions and data types are difficult to express

DryadLINQ

- Dryad + LINQ framework
- Allows tight integration between programming language and execution engine
  - No overhead due to type casts/convertion
  - Use a native programming language (as in Hadoop)
  - Dryad implements both iterator and accumulator-based interfaces

Decomposable aggregation functions

- Aggregation function is decomposable if every leaf in its expression tree is
  - A constant
  - g.key
  - Or H(g) for a decomposable function H

- Examples
  - Average, standard deviation, ...
DryadLINQ Implementations for Groupby-Aggregate (contd.)

- Memory at level-1
- Full Hash at level-2
- No pipelining
- no sorting
+ good reduction

Experimental Evaluation

- 3 aggregation functions
  - Word statistics
  - Word top documents
  - PageRank
- Execution Environment
  - 236 computers
  - Two-level network

Word statistics (& popularity)

- Intermediate aggregation is not helping
  - Extra node in the pipeline
  - Extra Disk I/O & node initialization costs
- Sort implementations are slow
- sort complexity
- Acc-FullHash is worse than Acc-PartialHash
  - May be because of the small partial hash table

Summary

- DryadLINQ has much richer interface than Hadoop
- DryadLINQ has better integration with programming languages than parallel DBs
- 6 plans for groupby-Aggregate in DryadLINQ

Questions?