Wave Computing in the Cloud

Bingsheng He
Microsoft Research Asia
Joint work with Mao Yang, Zhenyu Guo, Rishan Chen,
Wei Lin, Bing Su, Hongyi Wang, Lidong Zhou
My Dream Wave Computing
But, Today, Wave Computing is Actually...

The Wave model is a new paradigm for cloud computing.
State-of-the-art in the Cloud

- We provide scalability and fault-tolerance on thousands of machines.
- We provide the query interference using high level languages.

(MapReduce and its brothers: G. Y. M.)
Are G.Y.M.’s Executions Optimal?

- We looked at a query trace from a production system (20 thousand queries, 29 million machine hours).
- We focused on the I/O and computation efficiency.

(Mr. Leopard)
Our Finding: “Far From Ideal”

- Current Production System: 46% of total I/O
- Ideal System: 0% of total I/O

Bar chart and pie chart showing the comparison between the current production system and the ideal system in terms of I/O and computation steps. The bar chart displays the normalized total I/O, while the pie charts break down the I/O into redundant I/O on input data (33%), distinct I/O (67%), common computation steps (30%), and other computation steps (70%).
I/O Redundancy

- Two sample workloads
  - Obtaining the top ten hottest **Chinese** pages daily
  - Obtaining the top ten hottest **English** pages daily
Computation Redundancy

- Two sample workloads
  - Obtaining the top ten hottest Chinese pages daily
  - Obtaining the top ten hottest Chinese pages weekly

Every day:
- Extract
- Filter: “Chinese”
- Compute Top Ten

Every week:
- Extract
- Filter: “Chinese”
- Compute Top Ten

Common computation on per-day log (Ideally)
Why?

Correlations among queries
  – Temporal correlations among queries
    (A series of queries with recurrent computation)
Why?

Correlations among queries

– Spatial correlations among queries

(Input data are targeted by multiple individual queries)
How To Exploit the Correlations?

Err... This is a little tricky. What about developing these?
- a probabilistic model on scheduling the input data access
- a predictive cache server
- a speculative query decomposer.

No... Let’s K.I.S.S.:
- Since correlations are inherent, we need a notion to capture them.
- Our solution is the Wave model to capture the correlation for both the user and the system.
The Wave Model

• Key concepts capturing the correlation among queries
  – Data: not a static file, but a stream with periodically updated (append-only)
  – Query: computation on the input stream
  – Query series: recurrent computation on the stream
Optimization Opportunities in Waves

• Shared scan
  – Identifies the same input stream accesses among queries

• Shared computation
  – Identifies common computation steps among queries

• Query decomposition
  – Decomposes a query into a series of smaller queries
  – Uncovers more opportunities for shared scan and computation
Query Optimizations in Wave Computing

- Decomposition
- Form jumbo queries
- Optimizations on jumbo queries
  - Shared scan and computation

Query series 1: Obtaining the top ten hottest Chinese pages daily;
Query series 2: Obtaining the top ten hottest English pages daily;
Query series 3: Obtaining the top ten hottest Chinese pages weekly;
Ultimate (Wave+Cloud)

Individual query series

Time

Jumbo queries
Comet: Integration into DryadLINQ

Translation: query to logical representation (expression tree)

Transformation: logical->physical

Encapsulation: physical->Dryad execution graph

Code generation

Cost model

Query normalization
More rules; Views
Shared scan/partitioning
An Example of Query Decomposition in DryadLINQ

// Q2: weekly histogram aggregation grouping on (A,B)
q2 = env.Extractor("log?today-6...today")
  .Select(x => new { x.A, x.B })
  .Where(x => x.A != "qb")
  .GroupBy(x => x) //grouping on (A,B)
  .Select(x => new { x.Key, a = x.Count() });

Q → seven daily queries + one combining query

Daily query = env.Extractor("log?today")
  .Select(x => new { x.A, x.B })
  .Where(x => x.A != "qb")
  .GroupBy(x => x) //grouping on (A,B)
  .Select(x => new { x.Key, c = x.Count() });
  .ToDryadPartitionedTableLazy("q2dview?today");

Combining = env.Extractor2("q2dview?today-6...today")
  .AssumeHashPartition(x => x)
  .GroupBy(x => x)
  .Select(x => new { x.Key, c = x.Sum(y => y.c) });

Decompose an operator
Views (Cost estimation)
Combine all the views

Automatic query decomposition is challenging.
Micro Benchmark

• Overall effectiveness
  – Logical optimization of Comet reduces 12.3% of total I/O.
  – Full (Logical + Physical optimizations) of Comet reduces 42.3% of total I/O.

(Running three sample queries on one week data of around 120 GB; A cluster of 40 machine)
Summary

• The Wave model is a new paradigm for capturing the query correlations in the cloud.
• The Wave model enables significant opportunities in improving performance and resource utilization.
• Comet: our ongoing project integrating Wave computing into DryadLINQ.