Eddies: Continuously Adaptive Query Process

Authors: Ron Avnur and Joseph Hellerstein
From University of California, Berkeley

Presented at CPSC 504 2009 by Noreen Kamal

What system are we looking at?
- Query data across a large widely distributed site
  - Distributed processing
- Large shared-nothing clusters
  - Parallel processing

What is the problem?
- Hardware and workload complexity
  - Bursty performance caused by servers and networks (as we saw previously)
- Data complexity
  - No statistics (as we saw previously)
  - Non alphanumeric data types
  - Inaccurate selectivity estimates
- User Interface Complexity
  - Allow users to control properties of the query during execution

Run-Time Fluctuations
The 3 properties that vary during query processing:
1. The cost of operators
2. Selectivities
3. Rates tuples arrive from the input

What is the solution?
- Re-optimize query execution while it is running (i.e. the query is processing)
- Or “re-orderability of plans”
  - Decrease/limit or no synchronization barriers
    - E.g. when in some join algorithm the speed depends on the rate of the slower input
  - Increase moments of symmetry

Eddy: Discussion Question 1
* Their general philosophy is: "we favor adaptability over best-case performance"
1. Does this seem reasonable? In this case? In general?
2. How does this compare with previous approaches that we've looked at?
Moments of Symmetry

- The order of the inputs to the join can be changed without modifying any state in the join.
- Moments of symmetry allow reordering of the inputs to a single binary operator
- Figure: switching the inner and outer loops at the moment of symmetry
- Eddy to be effective favor join algorithm with frequent moments of symmetry, adaptive or non existent barriers and minimal ordering contraints

Moments of Symmetry and Join Algorithms

- What can be ruled out?
  - Hybrid hash joins
  - Need to avoid blocking
  - Merge Joins
  - Minimize ordering constraints and barriers
  - Nested Loops joins
  - Infrequent moments of symmetry and imbalanced barriers
- What are considered?
  - Ripple joins (group of joins with frequent moments of symmetry)
  - Constrained versions of pipelined hash joins
  - Hash Ripple joins
  - Block Ripple joins
  - Index Ripple Joins

Ripple Joins

- keeps the history of all tuples seen from either side (consumes much more memory)

Eddies

- "an eddy is the swirling of a fluid and the reverse current created when the fluid [river] flows past an obstacle" *
- The River (Query Processing Environment)
  - Shared-nothing parallel query processing framework that dynamically adapts to fluctuations in performance and workload.
  - Multi-threaded and can exploit barrier-free algorithms by reading from various inputs at independent rates

* Wikipedia: Eddy (Fluid Dynamics), August 2009

An Eddy

- A module in the River (query processor framework) that contains:
  - An arbitrary number of inputs
  - A number of participating unary and binary modules
  - Single output relation

A word on Routing…

- An eddy module directs the flow of tuples from the inputs through the various operators to the output (yes!)
- We need a routing policy…
  - Prioritize the tuples as they come into the eddy and move through the operators (gaining in priority)
  - Avoid the eddy getting "clogged" with new tuples
**Eddy (con’t)**

- An eddy encapsulates the scheduling of its operators
- The tuples entering the eddy can flow through its operators in any number of orders
- An eddy merges unary and binary operators into ONE n-ary operator (based on moments of symmetry)

**Lottery Scheduling**

- A simple learning algorithm to enhance the priority scheme to track both consumption and production
- Accounts for differences in selectivity
- Runs by “creating lottery tickets” for each operator as a way of determining who gets the next tuple
- Learns an ordering of operators based on efficiency

**Concluding Remarks**

- Done through re-ordering of plan on the fly
  - Changing the operators used while a query is processing
- Eddies work well with algorithms that have frequent moments so symmetry: Ripple joins
- An eddy merges unary and binary operators into ONE n-ary operator (based on moments of symmetry)
- Lottery scheduling can be used to learn which operators are most efficient

**Eddy: Discussion Question 2**

- Which would you rather use: Tukwila or Eddies? Why?