Review of "Eddies: Continuously Adaptive Query Processing"

http://www.skorage.org/wp/wp-content/2007/11/nov01.pdf
Nicolas Bonvin

November 1, 2007

Summary In this paper the authors introduced a new query operator called an "eddy". This is a query processing mechanism that contiunously reorders pipelined operators in a query plan on a tuple-by-tuple basis. This allows the system to adapt to fluctuations in computing resources, data characteristics and user preferences.

The eddy operator is used to route tuples to the input of some parent operator that use these tuples from the output of a query plan subtree. When actual query selectivities match the expected selectivities then the eddy operator simply routes tuples as soon as it sees them. But if the selectivities vary the eddy operator has to wait for a moment of symetry and then it will try to change the order of the parents operands. The goal behind this is that the query execution adapts itself in order to get closer to some cheaper plan. The adaptation to quick changes in query selectivity is implemented using a loteries-based algorithm (which perform better than a simple back-pressure algorithm).

Innovative ideas

- Lots of independant threads working on pulling tuples trough query plan trees is complex and no so efficient and simple as routing them. The idea of routing tuples is intuitive and easier to deal with.
- Contiunously reordering pipelined operators in a query plan is a clever concept.
- The concept of moment of symetry is very interesting as it allows to swap the order of processing without loosing work.

Most glaring problems

- The time taken to adapt to changes is not presented in the paper.
- The lotery-based algorithm is not described enough in the paper.
- The author don't provide results about a complete query workload. One may wonder if the eddy will scale when the number of queries to process will increase.
- The eddy architecture does not exploit cache-related benefits.

Possible improvements

- Eddy architecture may repeatedly execute different queries at one operator in order to gain cache-related benefits.
- Maybe increasing the tuple processing granularity can also drives some benefits.
- Would it be possible to use these moments of symetry in other adaptive query processing engines ?

Conclusion The notion of a moment of symetry is very intuitive and clean. It can probably be applied to a variety of query operators. The experimental evaluation section suffers from not providing an evaluation of a full workload.