Join Processing in Database Systems with Large Main Memories

Leonard D Shapiro

Abhishek Srivastava
Student ID: 861307778

February 9, 2017
CS 236, Winter 2017

The problem:
The paper notes that Join operations done using traditional methods such as sort-merge are memory intensive and other methods can be used to make it more memory inexpensive. This problem usually occurs when merging relations are very large and cannot be accommodated in the main memory leading to more disc read and write.

The contribution:
The author propose that different hash-based methods can be used instead of sort-merge to perform join operations efficiently with given sufficient main memory. Authors also presents optimal ways to handle the cases where main memory is inadequate.

The method:
Classic Hashing algorithm which is extended by author works by constructing hash table in the main memory, scan sequentially for tuple with matching key values in other relation, If match output the pair, If not drop the tuple and continue scanning.

Hash based methods presented in the paper are 2 phase, first is to construct the partitions so hash table of smaller relation can be stored in the main memory and second perform join.

- Simple hash: Partitions is performed by dividing both relations into disjoint sets and joining corresponding sets. It performs badly when main memory is small.
- GRACE hash: Partition both relations in corresponding subsets but divide one relation into equal size subsets. Join is performed by using hash based methods. Fixing size of subset reduces disk read/write and works well in little main memory.
- Hybrid hash: Combines the property of both hash methods by doing partitioning and hashing in the first phase over both relations and doing just join in second phase. Not flushing the memory after phase one improves the efficiency.

One of the main challenges in the paper is handling "Partition Overflow", it occurs when subset size is guessed incorrectly and the partitions overflows main memory available. It can be handled by starting with smaller size and merging into larger partitions until overflow is detected.

The hybrid hash dominates the performance with other hash based methods in both when main memory available is either high or low. It also outperforms sort-merge and also consumes less main memory for same join operations. It can also take advantage of virtual memory with some modifications to reduce the page faults and I/O cost. Filters, Babb-array and semi-join operations can also improve hybrid-hash performance.

Comments:
The paper presents a extended version of classic hashing technique to solve the issues in join operation using sort-merge technique. However, it has some sort coming:

- Depends significantly on main memory available to the process. It becomes an optimization issue in multi-process system.
- Parallel programming mechanism is not considered that how different join methods will perform.