Cardinality Estimation Done Right: Index-Based Join Sampling

A short summary by Platon Karageorgis

The following summary is focused on the paper of 2017 by Viktor Leis, Bernhard Radke, Andrey Gubichev, Alfons Kemper and Thomas Neumann that introduces an innovating method, on how to do cardinality estimation with lower cost and better results. The authors' first attempt, is to explain the common problems that the modern techniques face so far. They also explain that the random sampling method has not been implemented earlier, due to main memory capacity limits in previous decades and because it cannot guarantee practical solutions every time. Subsequently, they introduce their idea that does not use random sampling, but it adds a sampling operator, which provides better results and is also cheaper. Moreover, there are references to other implementations. Some examples are sampling-based query re-optimization, which takes an independent sample from each relation of the query and then performs a join between them or the ROX system, that depends on existing indices, in order to do the calculations with a lower cost. In the next part, the extensive analyses of the authors' proposal begins. For a start, the addition of the index-based join operator is critical. The operator manages to be effective in a short time schedule, since it produces exactly n samples after the join, by drawing n random materialized tuples from the list of matching tuples and therefore avoids large samples that delay the calculations. Furthermore, it is also quite effective in intermediate computations as well, being able to sample every one of them in up to 8-way joins, in contrast with the ROX algorithm that might give accurate estimates fast, but does not sample small intermediate relations. That leads to the traditional estimation causing the "fleeing from knowledge to ignorance" occurrence, which refers to the optimizer being misled by the inaccurate low cost estimations of the traditional method, and therefore building an insufficient plan. This problem is solved via the bottom-up solution, which exhaustively calculates the smaller joins, until it depletes all the available resources. In addition, the enumeration algorithm is mentioned, which mostly functions as a budget saver, by setting a time limit in order to skip bad plans. This is important because index sampling is firmly linked with time, due to the fact that the authors suppose it takes little time. This algorithm has also two optimizations, it deals with small samples by revisiting the same values multiple times, and it immediately joins small tables if there is an absence of an index. This implementation is also beneficial, because it does not interfere with the query optimizer and this means that it treats every query without bias. Subsequently, it achieves faster results with the index-based join operator but does not delay results of queries with different orientation. It should be noted that the operator is "turned on" optionally. In the following part, the authors attempt to support their

claims by evaluating their model. They first set the environment and the variables that the evaluation is based on. For example, the clarification that for the most experiments there is an index generation on all primary key and foreign key columns, as it is the worst case due to the trickiness of choosing the optimal plan. In the evaluation it is proven via graphic representations, that the index-based join sampling has more accurate estimations than PostgreSQL, especially with a higher number of samples. Moreover, it is confirmed that the index-based join sampling has lower cost than any other method like PostgreSQL or sampling based re-optimization and it is closer to the optimal query plan more than any other technique, in low and high budget cases. Last but not least, the technique proposed is also providing efficient query plans even in cases that the indices limited. In the final chapter, the authors discuss some other ideas suggested, such as the detection of cardinality estimation errors at runtime, the merging of the phases of optimization and execution and the arrangement of the query operators in a manner that they do not depend on cardinality estimates.

Which questions aren't answered by this text?

In general the purpose of this paper is to "advertise" a new implementation that is more efficient than the existing ones. From that perspective, there is nothing missing since the authors check every box. They mention a lot of times the other implementations, they explain methodically their idea and they evaluate it with very clarifying graphs. They also defend their opinion that their model is better, by including the other models in these evaluations and performing them under relatively fair conditions. On the other hand, it should be mentioned that the reader should be qualified to understand the biggest part of the paper, since the authors consider many terms known.

What has changed since this was written?

The text was written in 2017 which is very recent so I do not believe that anything has changed.