Technical Perspective: Efficient and Reusable Lazy Sampling

Thomas Neumann
TUM
neumann@in.tum.de

When interactively working with data, query latency is very important. In particular when ad-hoc queries are written in an explorative manner, it is essential to quickly get feedback in order to refine and correct the query based upon result values. This interactive use case is difficult to support if the underlying data is large, as analyzing large volumes of data is inherently expensive.

An attractive way to tackle this problem is to use approximate query processing (AQP). Instead of computing the exact query result, the system produces an approximate answer to the query, which is often good enough when still interactively exploring the data, and sometimes even good enough as the final answer [1]. The advantage of using approximate answers is that these can be computing much more efficiently, sometimes orders of magnitude faster than the exact result. And if the user is only interested in a rough overview over the data the full precision of, e.g., aggregate values is not required anyway.

Approximate query processing is usually based upon sampling techniques, that is the query is evaluated not on the full data set but on a random sample of data [2], which is much smaller but which exhibits the same data distribution as the original data. For simple queries like

```
select avg(x) from R
```

that is straight forward, the query will produce roughly the same result when executing on a random sample of `R` instead of the full table. But when the query contains filter predicates like

```
select avg(x) from R where y<4
```

the situation becomes more difficult, as a random sample might contain no or only a few tuples that satisfy the filter condition.

To alleviate that systems have mainly two options: Either they use larger samples, which makes it less likely that they are unable to answer the query, but which increases the AQP evaluation time and the storage costs. Or they use stratified sampling, which means that they maintain samples for a given predicate (or a given set of values).

Which allows for answering a query if a suitable sample is present, even for selective predicates, but which makes samples less versatile. For example a sample for the condition `y<4` can also be used to answer queries with a predicate `y<3`, but not to answer queries with `y<6`. For very predictable and repetitive queries that is less of an issue, but for the interactive use cases that is a severe problem, as queries can vary greatly.

Having one large sample over everything does not work well for selective predicates, but computing a sample for every predicate that we see in a query is not very practical, as the number of combinations is very large and eagerly computing samples is expensive.

The next paper tackles this problem by an interesting observation: We can construct a larger uniform random sample from two smaller uniform random samples over the same domain if 1) both samples come from disjoint parts of the original relation, and 2) we know how large the original data partitions were. Basically we can union two existing samples into a larger one, similar to a reservoir sampling strategy, but we have to take into account the selection probabilities for the elements were different. This allows for flexible stitching together available samples such that the query can be answered with the available data, even if the samples do not perfectly match the query.

There are more technical hurdles that have to be overcome, and the system must make a decision which sampling should be maintained given both the query and the already existing sample, but for that read the paper.

1. REFERENCES

[1] S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica. Blinkdb: queries with bounded errors and bounded response times on very large data. In Z. Hanzálek, H. Härtig, M. Castro, and M. F. Kaashoek, editors, *Eighth Eurosys Conference 2013, EuroSys '13, Prague, Czech Republic, April 14-17, 2013*, pages 29–42. ACM, 2013.

[2] A. Birler, B. Radke, and T. Neumann. Concurrent online sampling for all, for free. In D. Porobic and T. Neumann, editors, *16th International Workshop on Data Management on New Hardware, DaMoN 2020, Portland, Oregon, USA, June 15, 2020*, pages 5:1–5:8. ACM, 2020.