Applying Streaming Algorithms to Data at Rest

Mashhood Ishaque Ligia Nistor Kevin Backhouse
Oracle Corporation, Cambridge, MA, USA
mashhood.ishaque@oracle.com ligia.nistor@oracle.com kevin.backhouse@oracle.com

Guided navigation [5] is fundamental for the Oracle Big Data Discovery product. Our customers need to know the most frequent elements in their datasets, and the number of elements of a particular kind. In Figure 1, for the *winetype* property, we use a query that returns the top 6 kinds of wines ('Red', 'Cabernet Sauvignon', 'Burgundy Cote de Beaune', 'Pinot Noir', 'Merlot', 'White') and a query that counts the number of distinct types of wines ('371 others'). These queries can become extremely expensive, both in terms of space and time. In this paper we propose to use streaming algorithms for speeding up queries and lowering the memory usage. We detail our experience in applying two known streaming algorithms: the SpaceSaving algorithm [4] that has favorable mathematical guarantees [1], and HyperLogLog [2, 3], to data at rest. The first algorithm is normally used for finding the most frequent items in a data stream, while the second is used for counting the number of distinct elements in a data stream. Our innovation is to apply these algorithms to hundreds of gigabytes of data at rest.

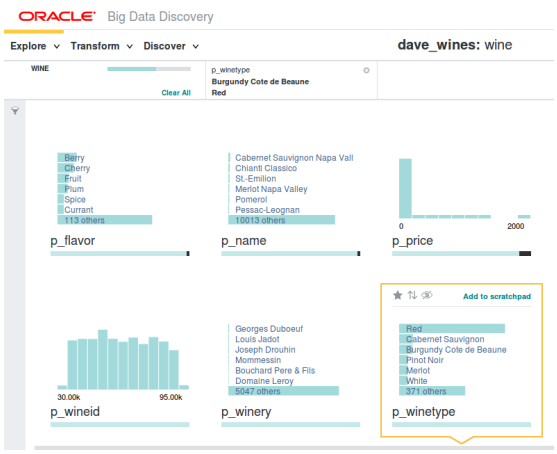

Figure 1. Guided Navigation (multiple properties)

We have implemented streaming algorithms on gigabytes of data at rest, with dramatic performance improvements. Our solution has the following advantages: we only sort a constant number of values (a few thousand); we give accurate counts when there is a small number of distinct values; we return approximate counts when there is a large number of distinct values, but with a provably small error; we use well studied algorithms that have been shown to perform well in practice; and streaming algorithms are usually embarrassingly parallel.

The assumption in the case of streaming algorithms is that the data cannot be stored, and hence that it cannot be sorted. Our insight is that we can avoid sorting by running these algorithms on our data, even though our data can be stored (hence the phrase 'data at rest').

References

[1] Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J. Strauss. Space-optimal heavy hitters with strong error bounds. 2010.

[2] Philippe Flajolet, Eric Fusy, Olivier Gandouet, and Frederic Meunier. Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. 2007.

[3] Stefan Heule, Marc Nunkesser, and Alex Hall. Hyperloglog in practice: Algorithmic engineering of a state of the art cardinality estimation algorithm. 2013.

[4] Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Efficient computation of frequent and top-k elements in data streams. 2005.

[5] Daniel Tunkelang. *Faceted Search. Synthesis Lectures on Information Concepts, Retrieval, and Services*. 2009.