Adaptive Data Skipping in Main-Memory Systems

Wilson Qin
Harvard University
wilsonqin@college.harvard.edu

Stratos Idreos
Harvard University
stratos@seas.harvard.edu

ABSTRACT

As modern main-memory optimized data systems increasingly rely on fast scans, lightweight indexes that allow for data skipping play a crucial role in data filtering to reduce system I/O. Scans benefit from data skipping when the data order is sorted, semi-sorted, or comprised of clustered values. However data skipping loses effectiveness over arbitrary data distributions. Applying data skipping techniques over non-sorted data can significantly decrease query performance since the extra cost of metadata reads result in no corresponding scan performance gains. We introduce *adaptive data skipping* as a framework for structures and techniques that respond to a vast array of data distributions and query workloads. We reveal an *adaptive zonemaps* design and implementation on a main-memory column store prototype to demonstrate that adaptive data skipping has potential for 1.4X speedup.

1. INTRODUCTION

Data Skipping. Modern main-memory optimized systems use scans with lightweight data skipping methods for fast data filtering. These systems maintain lightweight statistics such as minima and maxima values describing virtual zones - contiguous regions of a data column. A scan can then use the metadata to determine the relevance of a zone for a query and decide whether to skip or scan its underlying data. When used appropriately in conjunction with a priori user knowledge about query workloads and datasets, data skipping techniques can drastically improve scan performance by reducing the magnitude of records scanned at a smaller space and maintenance overhead than traditional indexing techniques [6, 8]. Data skipping techniques have been implemented for many modern systems including Netezza [3], Oracle [2], IBM DB2 [7], MySQL [1], Cloudera Impala [5], Hadoop [4], Shark [9], and Brighthouse [8].

Problem with Generalization. Current data skipping techniques do not generalize to all combinations of query workloads and datasets. Techniques typically work well on data columns that are sorted, semi-sorted, or locally clustered, but are ineffective when operating over arbitrary data distributions. Consider for example an attribute with a two value domain. With a column on this random uniform attribute, many zones will contain both values. Minima and maxima metadata that indicate which values are included in each zone are of no use to the query workload. In these cases gains from data skipping disappear as the system is penalized with the extra cost of metadata scan without resulting gains from skipping underlying data. The result is even worse performance than a full scan that uses no additional structure.

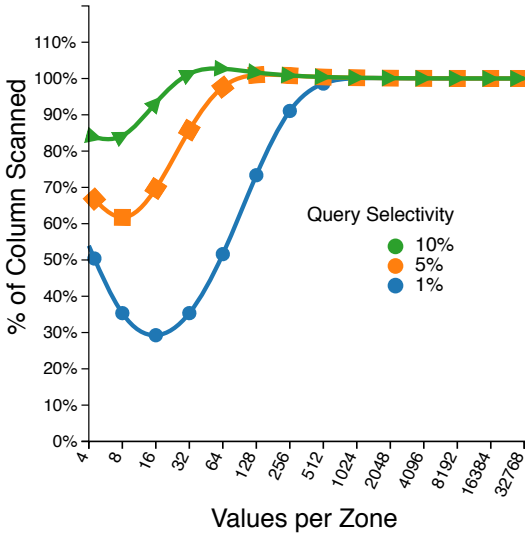

Figure 1: No general data skipping "sweet spot".

End of Static Layout Fits All. The effectiveness of data skipping layouts is strongly affected by variation in data distributions and query workloads. In Figure 1 we illustrate the performance effects of varying zone sizes when a random uniform dataset limited to a domain of 100 values is queried by a workload targeting low values at varying levels of selectivity. The data layout is unsorted and maintains zone minima and maxima. Figure 1 reveals that queries are not equally responsive to the same zonemap layouts; a layout that works for one distribution and workload can lose effectiveness in another case. This variation in zonemap effectiveness indicates the potential for dynamic structures beyond static layouts that can adapt to a changing query workload over time.

Our Contribution. In this paper we study data skipping on columnar main-memory data systems. We introduce adaptive data skipping as a framework for generalizing data skipping to a wide variety of scenarios. Adaptive data skipping leverages dynamic data layout to skip more data for future queries even when there is no order in the data. We show an adaptive zonemap design and implementation with potential for 1.4X speedup over standard data skipping techniques.

SIGMOD/PODS'16 June 26 - July 01, 2016, San Francisco, CA, USA

ACM ISBN 978-1-4503-3531-7/16/06.

DOI: http://dx.doi.org/10.1145/2882903.2914836

2. ADAPTIVE DATA SKIPPING

Adaptive data skipping is a robust data skipping framework applicable to a much wider array of data distributions and ad-hoc query workloads than standard data skipping methods. The core idea behind adaptive data skipping is incremental and on demand refinement of the underlying data (and metadata) during query processing. Metadata statistics can be created and improved on by manipulating a column's underlying data. Adaptive data skipping improves data zones at query time by reorganizing the data layout to maintain useful zone statistics with tighter domain bounds. This adaptive reorganization operates on hot zones during scan time, dynamically dividing or merging zones as needed.

Adaptive Zonemap Prototype. Adaptive zonemaps operate by gradually reducing the amount of excess non-qualifying values that must be scanned for any query. Data layout reorganization occurs at both the underlying data and metadata levels. Data values are relocated to create more order within zones. Large hot zones incur splitting into smaller zones to tune zone minima and maxima. This allows adaptive improvement which makes metadata relevant throughout the lifetime of any query workload, enabling data skipping methods with self-improving performance. Additionally adaptive data skipping allows for all columns to have a zonemap, but only the columns that are targeted by queries incur the costs of materialization. Thus, adaptive data skipping eliminates the previous need for a priori knowledge of which columns to build a zonemap on. By constantly improving the zone layout as well as metadata relevance adaptive data skipping addresses generalized data skipping performance in the long term.

Analysis. We now provide a proof of concept performance analysis. We use a main-memory column-store prototype on a 4-way Intel Xeon E7-4820 configuration with 64 hardware threads and 1 TB of main memory. The system uses bulk processing and late materialization for queries, storing columns as fixed-width and dense arrays. Column and metadata values are assumed to fit in memory. For experimentation, we use a synthetic dataset of 8-byte integers generated with random uniform distribution on a domain of 10^5, producing columns of 10^9 values. We evaluate a synthetic query workload on the system comparing the performance of a scan, zonemap (zone size 2048), and adaptive zonemap. We run 150 select queries of random selectivity over a single column and report the performance of the select operator.

Figure 2 shows the results from the last five queries of the workload which specifically target outlier values. The standard zonemap has large zones of size 2048, but its performance is plagued by an interspersion of outliers that appear in nearly all zones (at a frequency of 1 in 500 values). The adaptive zonemap begins with zones of size 2048, but periodically splits hot zones into smaller zones (size 32) and recomputes metadata for new zones. While the standard zonemap struggles to improve on the plain scan, we see that our adaptive zonemap clearly outperforms both scan and the standard zonemap by nearly 1.4X. By continuously refining the zones to match the access patterns, adaptive zonemaps minimize the amount of data they need to touch.

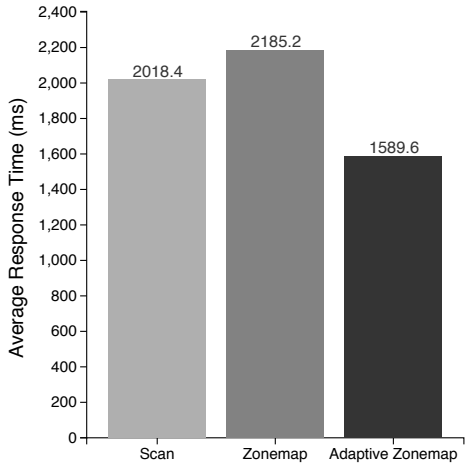

Figure 2: Adaptive Zonemaps provide 1.4X benefit in average response time speedup.

3. SUMMARY AND FUTURE WORK

In this paper, we demonstrate an initial step towards adaptive data skipping to solve the problem of lightweight zonemap-style indexes being useful only for specific data workloads and data orders. We reveal adaptive zonemaps as an implementation of adaptive data skipping. Adaptive zonemaps refine the underlying data incrementally and on demand to match the incoming workload with relevant data zones. Early results show 1.4X performance improvement over standard zonemaps in a main-memory column-store for handling arbitrary data workloads where no initial value order exists. We are currently working on several open topics towards fully materializing adaptive data skipping. For example, a sample of important open topics include studying alternate data reorganization strategies, utilizing adaptive data skipping across numerous columns, and finding the right balance of column optimization at system initialization (given certain idle time) to provide robust future performance.

4. ACKNOWLEDGMENTS

This work is partially supported by NSF Grant No. IIS-1452595. The authors would like to thank Michael S. Kester, Zezhou Alex Liu, and other members of Harvard DASlab for their help and feedback on this project.

5. REFERENCES

[1] Mysql 5.5 reference manual.

[2] Oracle database data warehousing guide.

[3] The Netezza FAST Engines Framework: A Powerful Framework for High-Performance Analytics. *Netezza Corporation*, 2007.

[4] M. Y. Eltabakh, F. Özcan, Y. Sismanis, P. J. Haas, H. Pirahesh, and J. Vondrak. Eagle-eyed elephant: Split-oriented indexing in hadoop. In *Proceedings of the International Conference on Extending Database Technology (EDBT)*, pages 89–100, 2013.

[5] M. Kornacker, A. Behm, V. Bittorf, T. Bobrovytsky, C. Ching, A. Choi, J. Erickson, M. Grund, D. Hecht, M. Jacobs, et al. Impala: A modern, open-source sql engine for hadoop. *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, 2015.

[6] J. K. Metzger, B. M. Zane, and F. D. Hinshaw. Limiting scans of loosely ordered and/or grouped relations using nearly ordered maps. *US Patent US6973452 B2*, 2005.

[7] V. Raman, G. M. Lohman, T. Malkemus, R. Mueller, I. Pandis, B. Schiefer, D. Sharpe, R. Sidle, A. Storm, L. Zhang, G. Attaluri, R. Barber, N. Chainani, D. Kalmuk, V. KulandaiSamy, J. Leenstra, S. Lightstone, and S. Liu. DB2 with BLU acceleration: so much more than just a column store. *Proceedings of the VLDB Endowment*, 6(11):1080–1091, 2013.

[8] D. Slezak, J. Wroblewski, V. Eastwood, and P. Synak. Brighthouse: an analytic data warehouse for ad-hoc queries. *Proceedings of the VLDB Endowment*, 1(2):1337–1345, 2008.

[9] R. S. Xin, J. Rosen, M. Zaharia, M. J. Franklin, S. Shenker, and I. Stoica. Shark: SQL and Rich Analytics at Scale. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 13–24, 2013.