Empirically Evaluating Configurations of a Sketch-Based Data Analysis Framework

Tooba Bajwa
The College of New Jersey

Antonis Manousis
Carnegie Mellon University

Vyas Sekar
Carnegie Mellon University

Justine Sherry
Carnegie Mellon University

1 INTRODUCTION

Performing interactive multi-dimensional analytics over massive amounts of data is a priority for modern data analytics frameworks. Computing in real-time various statistical analysis on incoming streaming data can significantly improve the services companies offer to their customers. For instance, in the case of video streaming platforms that analyze video sessions for different clusters of viewers, such capabilities can enable anomaly detection or expose useful content insights which will improve viewer engagement [2].

In this landscape, the queries that analysts need to process are often complex, needing to be performed over many data aggregations. For example, a video streaming company would want to query "For how many sessions, is the bit rate below some threshold, for all viewers in NYC?". Naturally, analysts have a few key requirements for such a framework: it should support a wide range of queries, for many groups of data, in real-time, and should be provably accurate.

Unfortunately, available approaches cannot satisfy all of the analysts' needs. Traditional methods such as SQL cannot support interactive analytics, due to the large time/space overheads involved when processing massive amounts of data [1]. Therefore, approximate data analysis techniques are more suitable. Sampling, while a seemingly natural approach, unfortunately does not offer provable accuracy guarantees. Sketching, on the other hand, is a promising approximate data analytics approach, due to its speed, support of a range of query types, and the accuracy it provides. However, sketching approaches present a scalability concern when it comes to supporting multiple data groups and metrics.

In this work, we introduce HYDRA, a novel sketch that shows promise in addressing the aforementioned scalability concern. We introduce the sketching primitive and present a preliminary empirical performance analysis of the sketch as a function of its large configuration space. We analyze HYDRA Sketch under an anonymized trace of video sessions across three dimensions: query time, update time and accuracy of the sketch's approximation results. Using a data-driven approach, we identify key parameters of HYDRA Sketch structure that are crucial in optimizing the sketch's performance. We also determine a key parameter of HYDRA Sketch that contributes to performance inefficiencies.

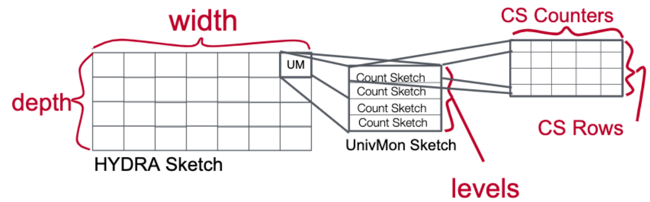

Figure 1: HYDRA **Sketch overview**

2 THE CASE FOR HYDRA SKETCH

A sketch is an approximate data structure for streaming data. At a high-level, a sketch structure can be thought of as a 2D array of counters, with the number of columns and rows known as the width and depth of the sketch. For every incoming data point, the sketch uses independent hash functions (one per row) to identify column locations and update the counters at those columns accordingly.

With such traditional sketching approaches, a scalability concern arises both when (1) the number of data groups and (2) the number of estimation tasks increases. This is because one sketch is needed for every data group, as well as a separate sketch for every estimation task. In total, the estimated number of sketches needed becomes $O(groups \times tasks)$. This rapid growth of complexity appears to complicate the use of sketches in a multi-dimensional setting. Below, we briefly explain how HYDRA addresses these challenges.

Tackling Many Metrics: Prior work has proposed a sketching primitive called UnivMon that is able to handle multiple estimation tasks within a single sketch[3]. Using UnivMon in the context of a sketch-based multi-dimensional analytics approach would reduce its complexity to $O(groups)$.

Tackling Many Data Groups: Our insight to handle all possible aggregations of incoming data without instantiating a sketch per group is to map multiple groups in one (UnivMon) sketch. HYDRA, by collapsing many UnivMon Sketches together in a 2D array, ensures that all possible aggregations of data will be covered by a set of sketches. The actual number of sketches to be used will be determined by the array's parameters (width and depth).

HYDRA is a "sketch of sketches", a 2D array where each cell is itself a UnivMon structure. In HYDRA Sketch, there are five main user-defined parameters shown in Figure 1: 1) the width (number of columns) and 2) depth (number of rows) of the sketch table, 3) the number of rows (CS Rows) and

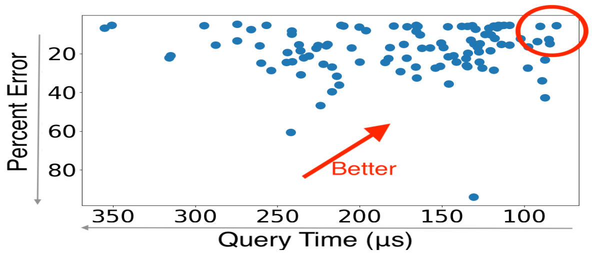

Figure 2: Query time and percent error of different HYDRA **instances. Optimal is top right corner.**

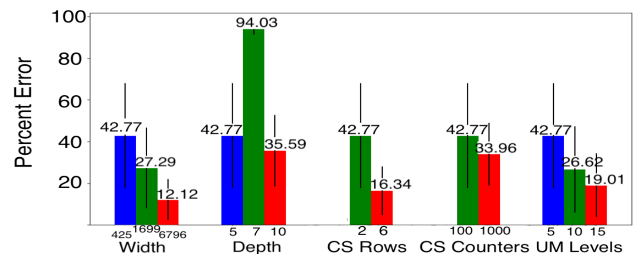

Figure 3: Percent Error vs Increasing Variables.

4) counters per row, for each Count Sketch (CS Counters), and 5) the levels, or parallel instances of Count Sketches, for each UnivMon Sketch (UM Levels).

3 PERFORMANCE EVALUATION

The duty of the programmer is to ensure that the values they assign to the sketch parameters lead to the best performance of the sketch. They must consider trade offs involved with update and query time of the sketch, and the accuracy of the resulting approximation. To achieve this goal, it is necessary to be aware of the effects each of the five parameters has on the behavior of the HYDRA sketch.

Therefore, our goal in this evaluation is to run an empirical data-driven analysis to shed light on what the key parameters (and their values) are that optimize HYDRA's accuracy while minimizing update and query times. To achieve this objective, we execute a trace driven simulation from an anonymized video trace, with distinct video sessions as input. We run 108 instances of the HYDRA Sketch on this input of video sessions. We initialize each instance with a different configuration of the five parameters shown in Figure 1 , with only one parameter value changing between every instance. For each of the instances, we test a large number of data points, for each of which, we record the time it takes to update the sketch with that data and the time it takes to query the sketch in microseconds. We compute the approximate entropy of bitrate, and then calculate the percent error of the sketch's estimation. We take all these data points into account to formulate the final query time, update time and percent error for one instance.

3.1 Results

We follow the Pareto Efficiency concept to find the optimal sketch configurations (those with the lowest query, and update time and the highest accuracy) of the 108 instances of HYDRA.

Which Parameters optimize our objective?:

Figure 2 shows the distribution of the 108 instances of the HydraSketch in relation to percent error and query time. We find that the best configurations are those with highest width and CS counter values, while the impact of the remaining parameters is marginal. For brevity, we omit our results showing the distribution in terms of query time vs update time, and percent error vs update time, however they show similar results, with the three most optimal points showing increased values for width and CS counters alone. This leads us to conclude that the width and CS counters per row are the two parameters that when increased, result in optimal performance of the sketch.

Figure 3 shows all the parameters and their increasing values, in terms of their accuracy measured by percent error of bit rate entropy approximation. A significant drop in percent error is seen for when CS rows is increased from 2 to 6. This improvement in accuracy between two values of CS rows, is greater than the increase in accuracy for any other two consecutive values for any other parameter. This leads to the conclusion that the CS rows parameter is significant in improving the accuracy of a sketch.

Which Parameters Promote Worse Performance of the Sketch?

As seen in Figure 3, there is very minimal net improvement in accuracy when depth increases. When observing the changes in update and query time with increasing depth, we notice a significant growth in update and query time overhead as well. This leads us to believe that depth is not a defining parameter HYDRA's accuracy. Therefore, setting the depth parameter of the sketch at a lower value is optimal for the performance and accuracy of the sketch.

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 *Proceedings of the 8th ACM European Conference on Computer Systems*, pages 29–42, 2013.

[2] J. Jiang, V. Sekar, H. Milner, D. Shepherd, I. Stoica, and H. Zhang. {CFA}: A practical prediction system for video qoe optimization. In *13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16)*, pages 137–150, 2016.

[3] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. Braverman. One sketch to rule them all: Rethinking network flow monitoring with univmon. *Proceedings of the 2016 ACM SIGCOMM Conference*, 2016.