MS-III. Implementation

Ali Hashaam, Ali Memon, Guzel Mussilova, Pavlo Shevchenko

Scientific Project: Databases for Multi-Dimensional Data, Genomics and Modern Hardware

June 13, 2017

Table of Contents

Our Goal

To provide a **framework** that gives user a chance to act as *Holistic SV Optimizer* like in OctopusDB

Add **Approximate Query Processing (AQP)** techniques

Evaluate performance depending on choice of SV

Building a Blinktopus. Recall

First, the Octopus:

- Store incoming data in logs.
- Query the logs (just a filter query).
- Allow users to create views (row, column) over certain logs.
- List all views and logs.
- Launch the query over views or over logs, see the changes in performance.

Building a Blinktopus. Recall

Enters Approximate Query Processing (AQP):

- Which synopsis will we choose to test? (Samples, histograms, sketches?)
- Do Octopuses and AQP match well together?
- Build the selected synopsis on the whole data, after data insertions.
- Using the synopsis, answer the user queries by reconstructing the approximate data.

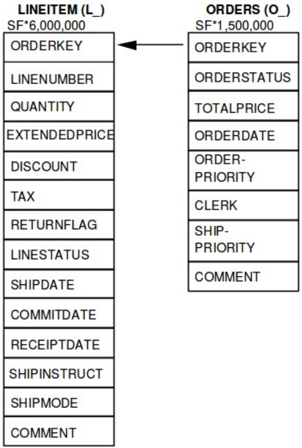
Building a Blinktopus. Implementation

Building a Blinktopus. IDE

- Back end
- Front end [1]

[1] Sources: http://jupyter.org/
http://honstain.com/new-dropwizard-1-0-5-java-service/

Schema



Selectivity Factors = 1,5,10,15
SF 1 = 1.2 M Records

OctopusDB. Customization

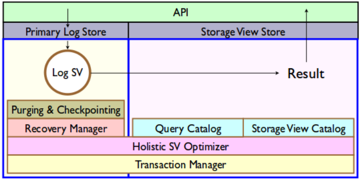


Figure 2: OctopusDB Architecture.

[2] A. Jindal. The Mimicking Octopus: Towards a one-size-fits-all Database Architecture, 2010

OctopusDB. Customization

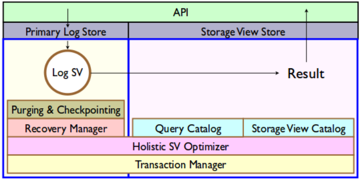


Figure 2: OctopusDB Architecture.

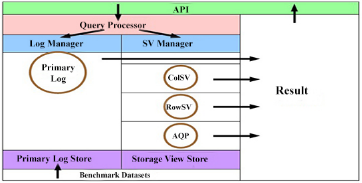


Figure 3: Blinktopus.

2

2 A. Jindal. The Mimicking Octopus: Towards a one-size-fits-all Database Architecture, 2010

OctopusDB. Evaluation

Log	190138277,2	8,279069555
RowSV	7001209,38	6,845173066
ColSV	4613638,51	6,664043563

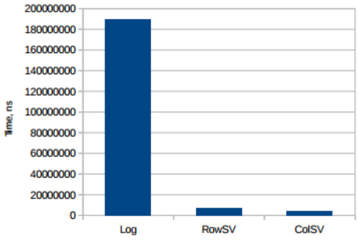


Figure 4: Storage View Type.

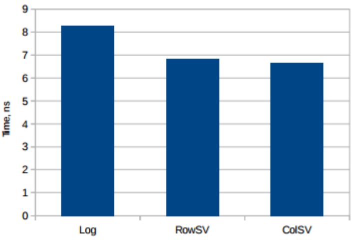


Figure 5: Storage View Type (Log Scale).

Evaluation result for 100 runs over Totalprice Column in Orders with Range from 50,000 to 200,000.

AQP. Synopses

4 main families of synopses[3]:

- Samples
- Histograms ✓
- Wavelets
- Sketches ✓

[3] Cormode, Graham, Minos Garofalakis, Peter J. Haas, and Chris Jermaine. "Synopses for massive data: Samples, histograms, wavelets, sketches." Foundations and Trends in Databases 4, no. 13 (2012): 1-294.

AQP. Histograms

In histogram's development, main cornerstones are:

- Partition the dataset into buckets.
  Number of buckets 'k': $k = 2n^{1/3}$ (RICE RULE)
- Store summary statistics for each bucket about the data values in the it.
- Store information about the buckets themselves, like bucket boundaries.

At query time, the summary and bucket information is used to approximately answer the query.

AQP. Histograms

Vital Points to consider:

- Bucketing Scheme
- Statistics Stored per Bucket
- Approximation Scheme
- Class of queries answered
- Efficiency
- Accuracy & Error Estimates
- Incremental Maintenance

AQP. Histograms

D = {1.61,1.72,2.23,2.33,2.71,2.90,3.41,4.21,4.70,4.82,4.85,4.91}

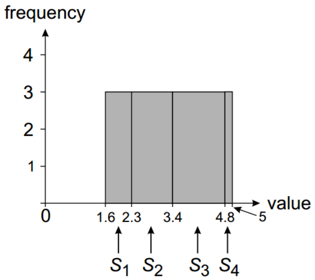


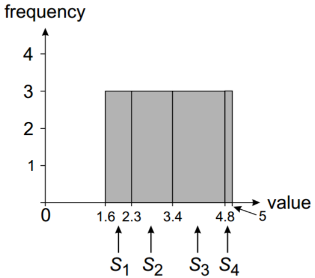
Figure 6: Equi-Depth Histogram [4]

What if the count of the values between 1.1 and 4.5 is required?

[4] Cormode, Graham, Minos Garofalakis, Peter J. Haas, and Chris Jermaine. "Synopses for massive data: Samples, histograms, wavelets, sketches."

AQP. Histograms

Continuous value assumption allows the estimation of values inside a bucket via interpolation.



Actual query : N $= 8$

$AQP : N = 3 + 3 + ((4.53.4)/(4.83.4))3 = 8.4$

Overestimation Error $= 5$

AQP. Sketches

- Sketches, approximately answer queries by creating small summary data structures that approximately resemble the original data.
- Appropriate in scenarios involving streaming of big data or analysis of higher dimensional data is required.
- Other synopsis (Samples, histograms, Wavelets) can be extracted from it.

AQP. Sketches

Phases of Sketching mechanism:

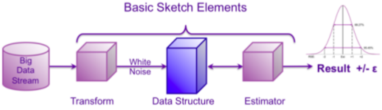


Figure 7: Sketching Phases [5]

Case Under Scrutiny: Count-Distinct

[5] Source: https://yahooeng.tumblr.com/post/135390948446/data-sketches

AQP. HyperLogLog (HLL)

HLL algorithm estimates the number of distinct elements in large datasets i.e. cardinality, in a single pass, and using a very small amount of memory.[6]

4 billion distinct elements $= \log_2 \log_2(2^{32}) = 5$ bits required

[6] Flajolet, Philippe, et al. "Understanding the HyperLogLog: a Near-Optimal Cardinality Estimation Algorithm."

AQP. HyperLogLog (HLL)

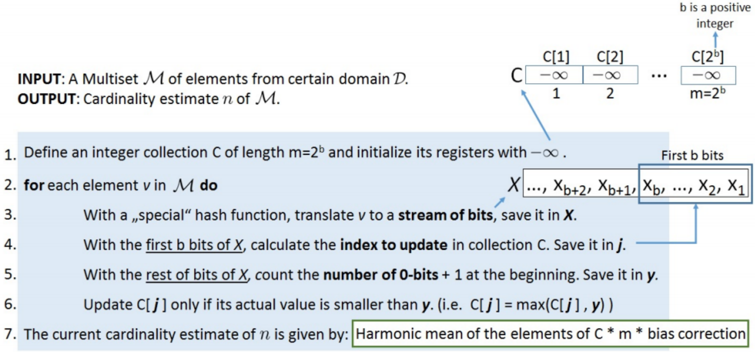


Figure 8: The HyperLogLog algorithm [7]

[7] Flajolet, Philippe, et al. "Understanding the HyperLogLog: a Near-Optimal Cardinality Estimation Algorithm."

AQP. HyperLogLog (HLL)

- Divide the n distinct elements of the input multiset into m number of buckets.
- Each bucket must comprises approximately the same number of elements, $\frac{n}{m}$.
- transform our input multiset M into an ideal multiset via a special Hash function.
- Hash function will transform the values to a stream of 0s and 1s,

AQP. HyperLogLog (HLL)

- Take first b bits of the hashed value as the index of the bucket.
- In each bucket only save the longest run of starting 0-bits+1 among all the hashed values of the elements that belong to that bucket.

n = length of longest run of starting 0-bits + 1

Number of distinct elements in bucket $= 2^n$

bits required $= \log_2 \log_2(2^n)$

AQP. HyperLogLog (HLL)

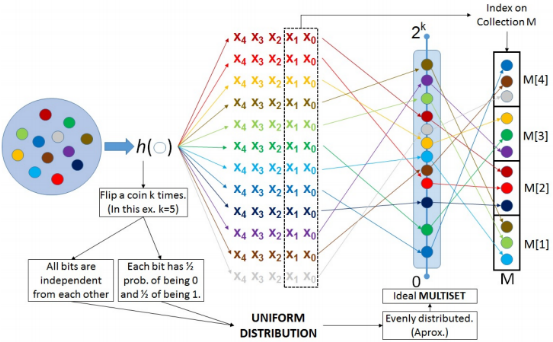


Figure 9: Randomization of elements and division in buckets [8]

[8] Flajolet, Philippe, et al. "Understanding the HyperLogLog: a Near-Optimal Cardinality Estimation Algorithm."

Project Organisation.Roles

Team:

Guzel - Team Leader-Researcher
Pavlo - Developer (Backend - OctopusDB)
Ali H. - Developer (Backend - AQP)
Ali M. - Developer (Frontend - User Views)

Supervisor:

Gabriel Campero Durand

Changing roles after each milestone.

Meetings:

Team Meetings: Mo 14-15
Meetings with supervisor: We 10-11

Thank you! Any questions?

Literature

1. Jindal, Alekh. "The mimicking octopus: Towards a one-size-fits-all database architecture." VLDB PhD Workshop. 2010.
2. Dittrich, Jens, and Alekh Jindal. "Towards a One Size Fits All Database Architecture." CIDR. 2011.
3. Jindal, Alekh. "OctopusDB: flexible and scalable storage management for arbitrary database engines." (2012).
4. Mozafari, Barzan, and Ning Niu. "A Handbook for Building an Approximate Query Engine." IEEE Data Eng. Bull. 38, no. 3 (2015): 3-29.
5. Cormode, Graham, Minos Garofalakis, Peter J. Haas, and Chris Jermaine. "Synopses for massive data: Samples, histograms, wavelets, sketches." Foundations and Trends in Databases 4, no. 13 (2012): 1-294.