SEeSAW – Similarity Exploiting Storage for Accelerating Analytics Workflows

Kalapriya Kannan, Suparna Bhattacharya, Raj Kumar, Muthukumar Murugan, Doug Voigt
Hewlett Packard Enterprise
USENIX HotStorage, June 20, 2016
Digital data generation in an always-on world …

Got to see some great sand sculpting yesterday #sandsculptures #sandsculpting #beach #hamptonbeach #beachday

Beach day this weekend? Check out the #HamptonBeach sand sculptures
As a Dad, this is my favorite Sand Sculpture at the 2016 Hampton Beach Sand Sculpture Contest.

#HappyFathersDay to all the wonderful dads we know. Look at what's on #HamptonBeach! Enjoy this gorgeous day!
Informatics becomes the bottleneck as cost of gathering data goes down (mobile, social, IoT, surveillance, machine logs, …)

The Promise
Abundance of historical context to discover insights via analytics

The Dilemma
How to build capacity to absorb all information even if available?

Storage tends to be the long pole in analytics, bypassed as far as possible!

Instead, … could storage leverage characteristics of the data to address this dilemma?
Example: VM utilization data characteristics (Cloud Optimizer)
Historical utilization based forecasting (capacity planning, placement, migration)

**UTILIZATION DATA FOR A INSTANCE FOR 4 SNAPSHOTS**

Snapshots similar 98% of the time

1500 VM's, 90 days data
Snapshot = 3 days
Storage systems can accumulate much data with similar content. Especially true of data that is stored primarily for the purpose of analytics.

Similar patterns arise naturally.

Observations that show consistent repetitive variation
– Sensor, surveillance, geo-satellite data

Data generated by similar sources in a similar context
– System utilization data for similar workloads
– Human generated info on same viewpoint

Results of similar computation on similar data

Often semantically or statistically similar but not duplicates ...

\[ \text{Jaccard Similarity} = \frac{|X \cap Y|}{|X \cup Y|} = 0.5 \]

... unlike typical data/compute de-dup
Should similarity detection be a fundamental storage primitive? Recognizing and exploiting this data property @ storage has several benefits

Advantages

- Accelerate analytics by skipping similar data sets
- Optimize storage by assigning differential value to data sets
- Prevent generation of nearly same data across workflows
- Analytical applications and frameworks operate without modification
- Reduce data movement, unlike similarity check at a higher layer

Ex: Similarity based storage optimization
Provided we can get it right

Challenging Goals

– Similarity detection should not be heavy weight
  – Rule out dissimilarity with minimal cost
  – State maintained with minimal storage overhead

– Save resources IO, CPU, .. and time to get analytics results while preserving accuracy
  – Similarity detection should reflect application relevant features

– Seamlessly exploit similarity at the storage level
  – Without application (or even framework) involvement
Therefore …. 

SEeSAW – Similarity Exploiting Storage
The Idea

Identify similar data sets NOT JUST at the raw input but at intermediate transformation outputs (TO) to accelerate analytics workflows.

Similarity at later stages reflects semantics of the model.
How? Intercept transformation outputs to detect similarity

Similar data seen?

Seamlessly bypass remaining stages and reuse results

Example: Cloud Optimizer VM resource usage forecasting

Mark data at all previous stages as semantically similar

Plugin leverages lineage mechanism in Spark/Alluxio
Identify the Optimal stage and thresholds during first few runs

Pick optimal stage to test for similarity

What is the time/space saved vs Time to compute similarity

1. Identify the Optimal stage and thresholds during first few runs
2. What is the time/space saved vs Time to compute similarity

Diagram:

- Data Sources:
  - T1: Raw data
  - T2: Minutes Alignment
  - T3: Hourly Rollup
  - T4: Cycle Detection
  - T5: Seasonality
  - T6: Trend detection
  - T7: Forecasting

- SPARK/HADOOP
- Storage
  - Size: 2
  - Size: 5
  - Size: 12

Hewlett Packard Enterprise
Subsequent runs only test and exploit similarity at optimal stage

Similar data at optimal stage?

Opportunity to save significant % of execution time?

Opportunity to save significant % of storage space?

Example: Cloud Optimizer VM resource usage forecasting
Early Results
How much similarity is observed?

Similarity at the optimal stage across snapshots taken 3 days

<table>
<thead>
<tr>
<th>Similarity Threshold (Manhattan distance)</th>
<th>&lt;=2</th>
<th>&lt;=5</th>
<th>&lt;=8</th>
<th>&lt;=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of Instances similar</td>
<td>88.0952381</td>
<td>97.5</td>
<td>98.0952381</td>
<td>99.28571429</td>
</tr>
</tbody>
</table>

Snapshot 1 and 4

<table>
<thead>
<tr>
<th>Similarity Threshold</th>
<th>&lt;=5</th>
<th>&lt;=8</th>
<th>&lt;=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage Data set</td>
<td>97.5%</td>
<td>98%</td>
<td>99.28</td>
</tr>
</tbody>
</table>
Early Results
Do similar inputs at the optimal stage lead to similar analytics outputs?

Output similarity follows input similarity

Snapshot 1-4

Variation < 0.5 %
Early Results
What was the performance benefit observed?

Time Saving using SEE-SAW with Forecasting Engine

- Speed-up 97%

Execution Time Savings

- Cycle + Seasonality + Trend Detection + Forecasting Engine
- OneDHaar + coefficient comparison
- Connecting to similarity service in Alluxio
- Data ingestion from DB + minutes alignment + hourly rollup

Using SEE-SAW

Without SEE-SAW
Open challenges
Storage systems that exploit such data characteristics – worthwhile or too radical?

How should storage ensure ..
– Appropriate features and algorithms for measuring similarity?
– Similarity based decisions that preserve analytics accuracy?
– Enough history of prior executions is maintained for similarity detection and optimal stage analysis?

Are there other use cases for storage based similarity detection?

Are there other data characteristics storage systems could leverage?
Thank you

suparna.bhattacharya@hpe.com