Self-tuning BI Systems

Home University (UPC):

Alberto Abelló

and

Oscar Romero

Host University (TUD):

Maik Thiele

and

Wolfgang Lehner

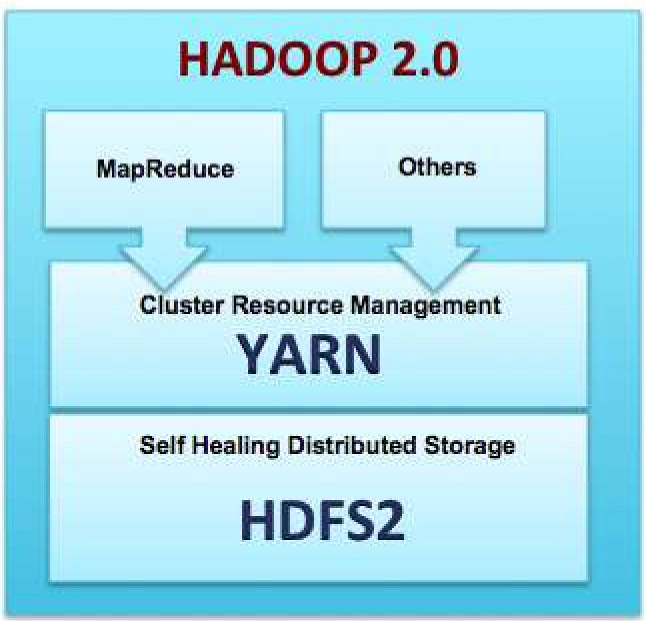
I am:

Rana Faisal Munir

Agenda

- Introduction
- Problem Statement
- Our Proposed Approaches
  - Rule-Based
    - Experimental Evaluation
    - Shortcomings
  - Cost-Based
    - Algorithm
    - Experimental Evaluation
- Current Work
- Future Directions
- Feedback and Questions

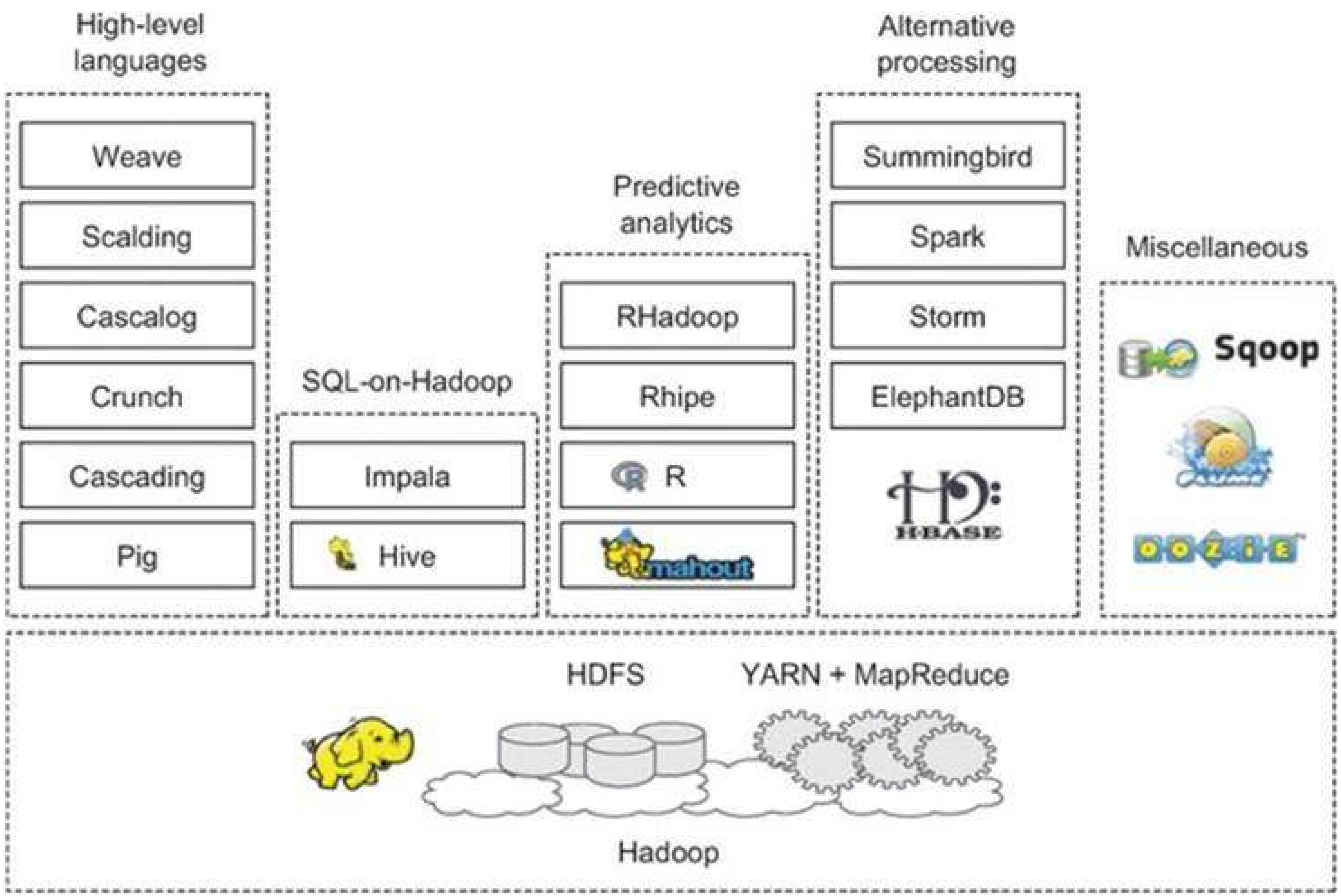
Introduction: Hadoop ecosystem



- Hadoop ecosystem is considered as a de-facto standard for processing large scale data
- It is a distributed system which provides
- Storage
  - Hadoop Distributed File System (HDFS)
- Processing framework
  - MapReduce
- Challenges
  - **Difficult to develop an application using core MapReduce programming model**
    - High-level languages require to facilitate developers and data analysts

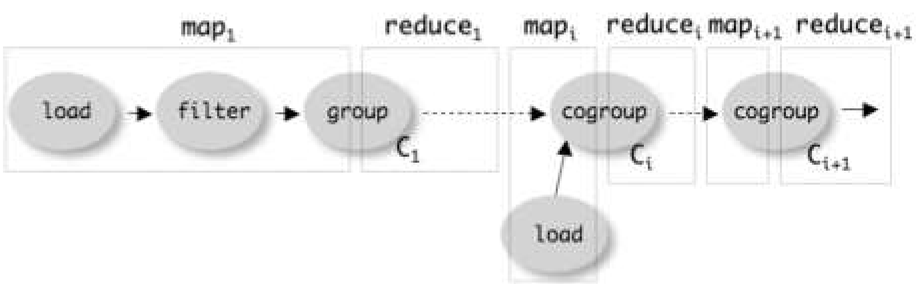
Source: https://www.packtpub.com/sites/default/files/Article-Images/B03750_02.png

Introduction: High-level Languages



Source: https://www.linkedin.com/pulse/hadoop-ecosystem-farshad-vahidpour

Introduction: High-Level Languages



- High-Level Languages (Apache Pig, Apache Hive, etc.)
  - Generate Directed Acyclic Graph (DAG) of MapReduce jobs
  - Each MapReduce job takes an input and produces an output for the next job
- Optimizations
  - Rule based
    - Reordering of operators
  - **Can we do more for optimization?**

Source: https://blog.linkedin.com/2010/07/01/linkedin-apache-pig

Introduction: Computational Redundancies

- **Avoid computational redundancies**
  - Filter out bad records, Spam e-mail
  - Data representation transformation
- **Microsoft found 30%-60% similarity in queries submitted for execution**
- **A Berkeley MapReduce workload characterization study shows 80% results are reused [1]**

[1] Y. Chen, S. Alspaugh, and R. Katz. Interactive Analytical Processing in Big Data Systems: A Cross-Industry Study of MapReduce Workloads. In VLDB, 2012

Introduction: Existing Materialized Solutions

- **ReStore: Reusing Results of MapReduce Jobs [VLDB 2012]**
- **m2r2: A Framework for Results Materialization and Reuse in High-Level Dataflow Systems for Big Data [BDSE 2013]**
- **Where they store?**
  - HDFS
  - I/O cost involves

Introduction: Existing Materialized Solutions

- **ReStore: Reusing Results of MapReduce Jobs**
  - Based on heuristics
    - Materialize an output of an operator and keep it in the repository
    - If it is **used within a window** time then keep it otherwise discards it
    - Future jobs can reuse
      - **Whole** materialized results
        - Join, Group By, Sort, etc.
      - **Partial** / some parts of materialized results
        - Projection and Selection
  - Can we further improve?

Introduction: Storage Layouts

- **Text Formats**
  - CSV, XML, JSON, etc.
- **Binary Formats**
  - Horizontal
    - SequenceFile, Avro, etc.
  - Vertical
    - Zebra, etc.
  - Hybrid
    - RCFile, ORC, Parquet, etc.

SequenceFile	[key,{cols with a delimiter}] [r1_a1,{r1_a2|r1_a3|r1_a4}] [r2_a1,{r2_a2|r2_a3|r2_a4}]
Avro	[store row-wise] [r1_a1,r1_a2,r1_a3,r1_a4] [r2_a1,r2_a2,r2_a3,r2_a4]
Zebra	[column groups] [{r1_a1,r1_a2},{r1_a3,r1_a4}] [{r2_a1,r2_a2},{r2_a3,r2_a4}]
Parquet	Horizontal and vertical partition [{r1_a1,r2_a1},{r1_a2,r2_a2}] [{r3_a1,r4_a1},{r3_a2,r4_a2}]

Introduction: Storage Layouts

- **None of them is the universal best choice;** different workloads require different storage layouts to achieve optimal performance [1]
- Certainly, there is **no single optimal layout** as it significantly influenced by the query workload. As the workload changes, the data layout may also need to be changed accordingly [2]
- However, existing materialized solutions are not considering them
  - They only use a fixed format for materialization
  - *Can we further improve?*

[1] I. Alagiannis, S. Idreos, and A. Ailamaki. H2O: A Hands-free Adaptive Store. In SIGMOD, 2014

[2] M. Y. Eltabakh. Data Organization and Curation in Big Data. In Handbook of Big Data Technologies, 2017

Problem Statement

- **Input**
  - For a given workload and a set of materialized nodes
- **Output**
  - An optimal storage format for every materialized node
    - Based on the subsequent access patterns (Future read operations)
- **Goal**
  - Helps in reading less data from the disk and overall, it reduces the execution time
- **Scope**
  - Focusing only on OLAP , not considering OLTP

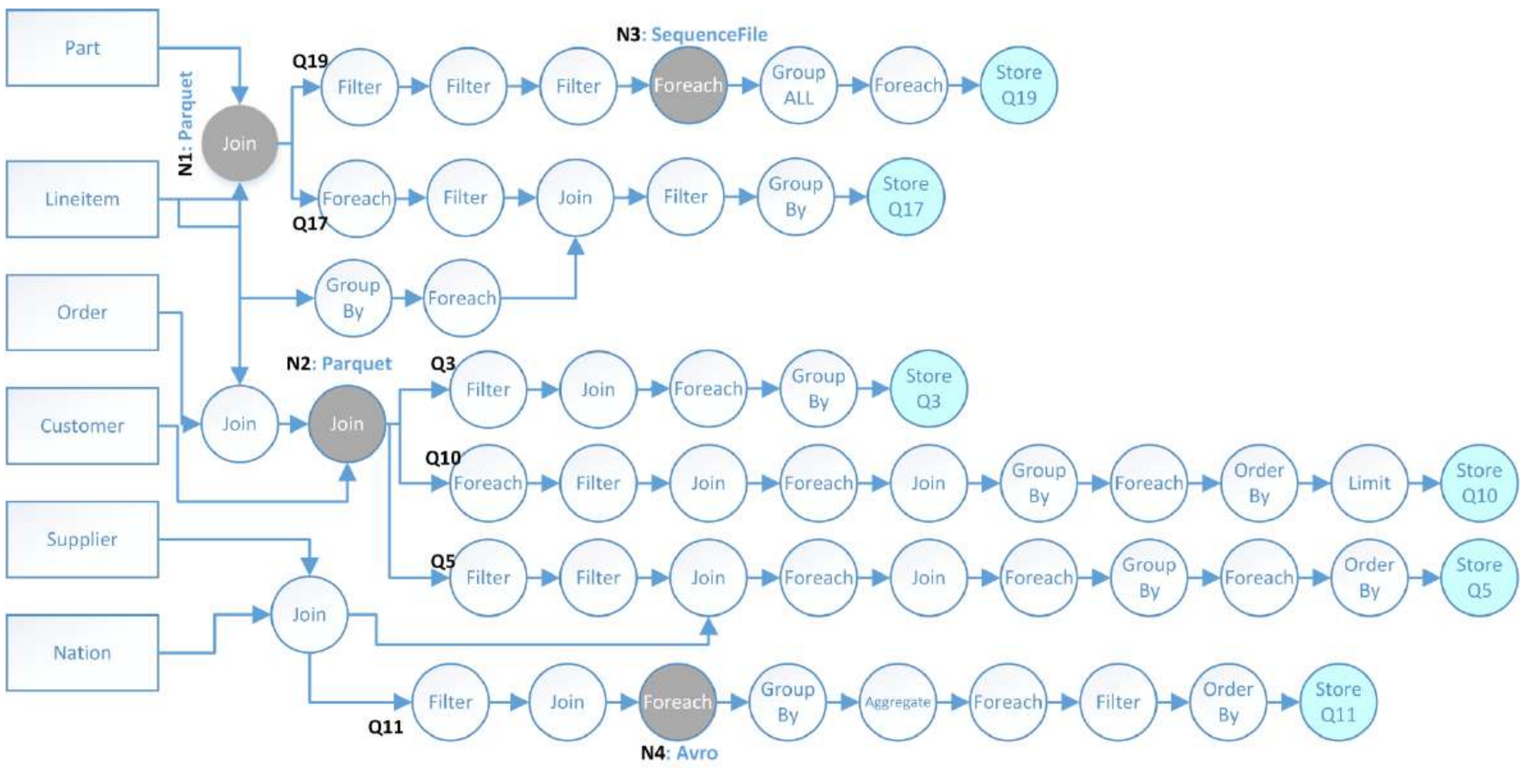
Our Proposed Approaches: Rule-based

- **Started with existing heuristics**
  - Horizontal layouts are good for full scan
  - Vertical layouts are good for projection
  - Hybrid layouts are good for both projection and selection
- **We have considered**
  - Two layouts
    - Horizontal
      - SequenceFile and Avro
    - Hybrid
      - Parquet
    - Nowadays, vertical layouts are subsumed in hybrid and we also could not find updated version of zebra for testing. Hence, we ignore it for our study

Our Proposed Approaches: Rule-based

- **Existing Materialized Solution**
  - ReStore
    - Has implementation in Pig and hence, we use Pig as a high-level language
- **Pig Operators**
  - FOREACH, FILTER, AGGREGATIONS
    - Use Parquet
  - JOIN, COGROUP, CROSS, etc.
    - Use Avro
  - Use SequenceFile if materialized node has two columns
- **No block-level compression and dictionary encoding for fairer comparison**

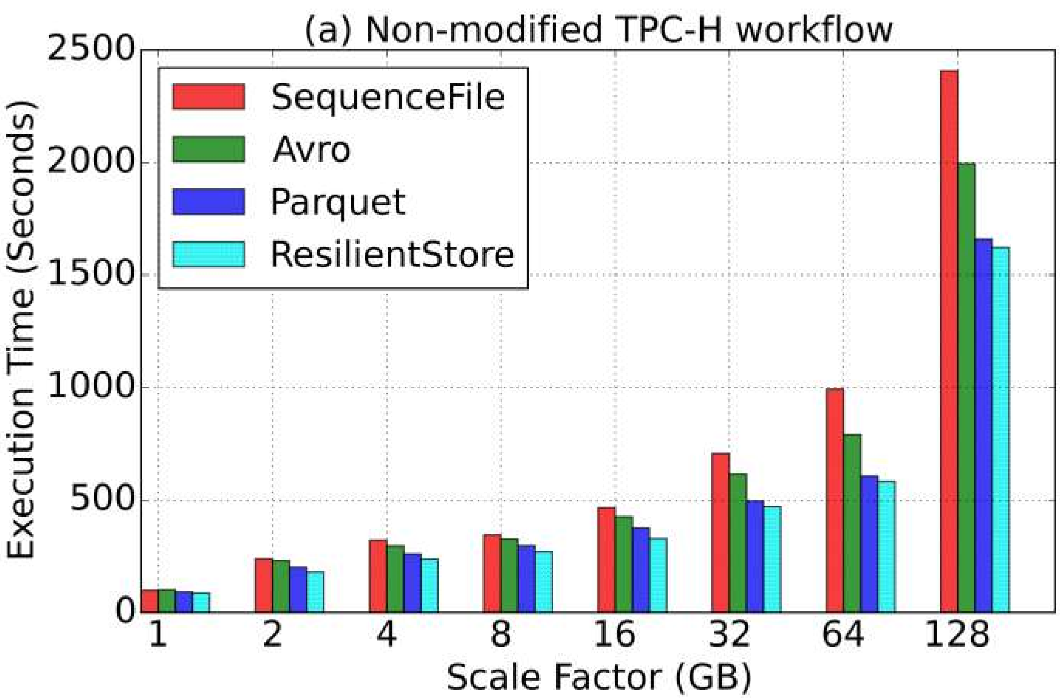
Rule-based Approach: Evaluation



Publication: Rana Faisal Munir et. al., ResilientStore: A Heuristic-Based Data Format Selector for Intermediate Results, MEDI 2016

TECHNISCHE UNIVERSITÄT DRESDEN

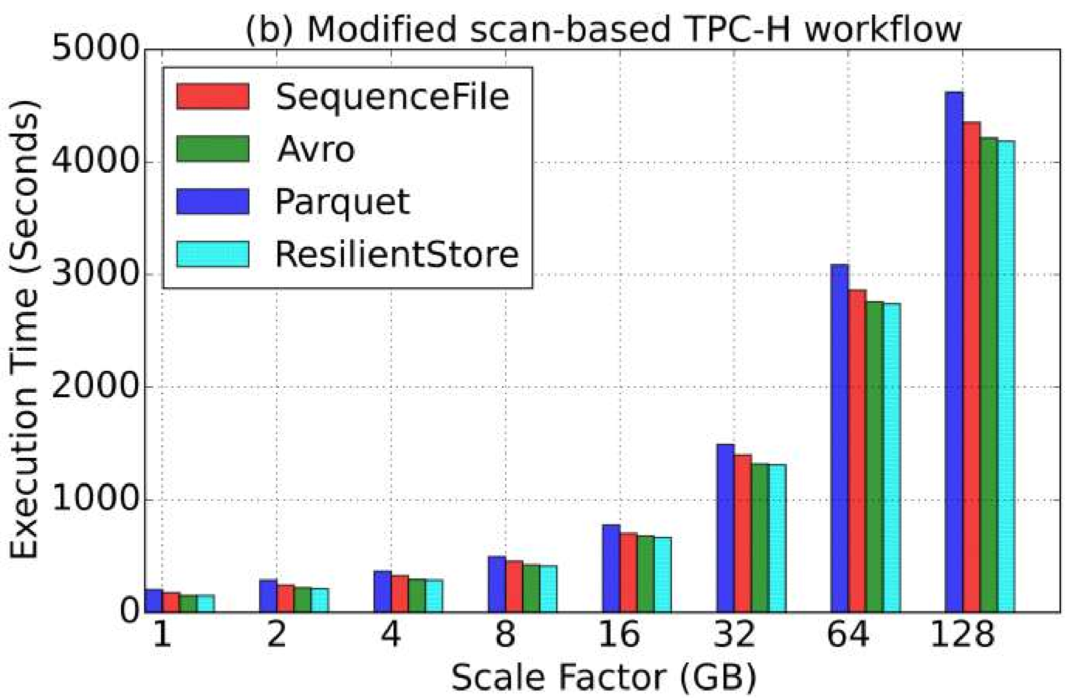
Rule-based Approach: Evaluation



- Our approach on average provides
  - 32% speedup over fixed SequenceFile
  - 19% speedup over fixed Avro
  - 4% speedup over fixed Parquet
  - Overall, it provides 18% speedup

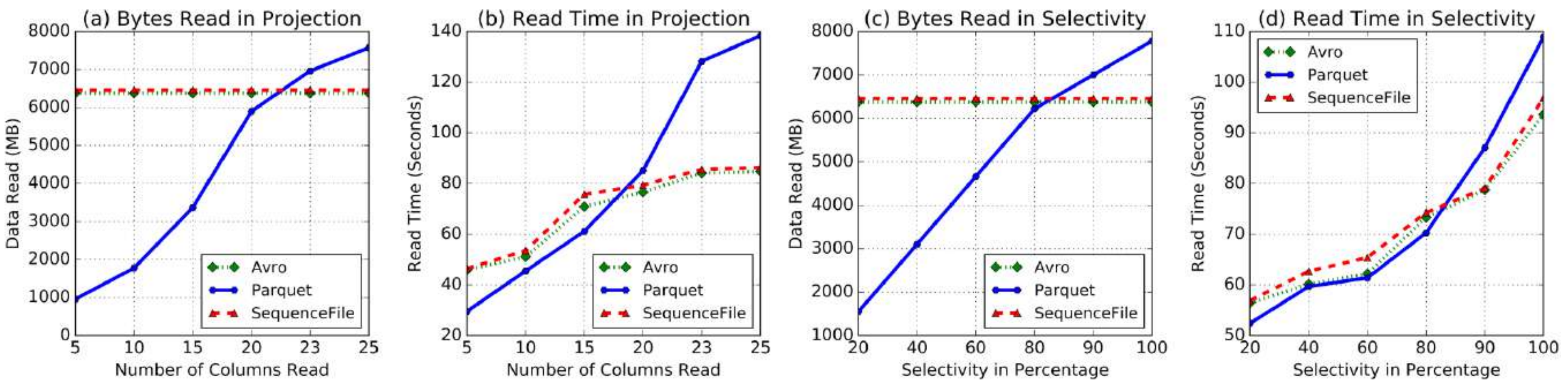
Rule-based Approach: Evaluation

- TPC-H queries have very low selectivity factor
  - It benefits Parquet
- To test performance of our solution in scan-based workload
  - TPC-H queries are modified



- Our approach provides
  - 9% speedup over fixed SequenceFile
  - 1.5% speedup over fixed Avro
  - 21% speedup over fixed Parquet
  - Overall, it provides 10% speedup

Rule-based Approach: Shortcomings



- We have observed
  - At certain point, Parquet cannot pushdown operators to the storage layer
  - When we have high selectivity or when we are reading large number of columns
  - In our rules
    - We are not considering these factors (selectivity and referred columns).
    - Whenever, there is a projection or selection operation
      - We assume that it is going to read less data
      - Which is not true always
    - To consider these factors, we need a cost model

Our Proposed Approaches: Cost-based

- **There are two costs**
  - Write Cost
  - Read Cost
    - Scan
    - Projection
    - Selection
- **Best Layout**
  - Which has overall less cost (Sum of both write and read costs)

Our Proposed Approaches: Cost-based

Variable	Description
|IR|	Number of rows in IR
Size (Row)	Average Row Size
Size (Col)	Average Column Size
Cols (IR)	Total columns of IR
RefCols (IR)	Number of columns used in a query
SF	Selectivity factor of a query

- **Intermediate Result (IR)**
  - These are the information which we have to record for every materialized node to use in our cost-based approach

Our Proposed Approaches: Cost-based

- **Two Cost Models**
  - Generic
    - Horizontal
    - Vertical
    - Hybrid
  - Instantiation
    - Horizontal
      - SequenceFile
      - Avro
    - Hybrid
      - Parquet

Generic: Write Cost Model

$$
\begin{aligned}
Size(Layout) &= Size(Header_{layout}) \\
&\quad + Size(Body_{layout}) \\
&\quad + Size(Footer_{layout})
\end{aligned}
\tag{1}
$$

$$
Used_{chunks}(Layout) = \frac{Size(Layout)}{Size(Chunk)} \tag{2}
$$

$$
Seeks(Layout) = \lceil Used_{chunks}(Layout) \rceil \tag{3}
$$

$$
\begin{aligned}
Size(Body_{horizontal}) = {} & (Size(Row) + Size(Meta_{HRow})) * |IR| \\
& + Size(Meta_{HBody})
\end{aligned}
\tag{6}
$$

Horizontal Layouts

$$
Size(OneColWithMeta) = Size(Col) * |IR| + Size(Meta_{VBody}) \tag{7}
$$

$$
Size(Body_{vertical}) = Size(OneColWithMeta) * Cols(IR) \tag{8}
$$

Vertical Layouts

$$
Used_{RG}(Hybrid) = \frac{(Size(Col) * |IR| + Size(Meta_{YCol})) * Cols(IR)}{Size(RowGroup)} \tag{9}
$$

$$
\begin{aligned}
Size(Body_{hybrid}) = {} & Used_{RG}(Hybrid) * Size(RowGroup) \\
& + \lceil Used_{RG}(Hybrid) \rceil * Size(Meta_{YRowGroup})
\end{aligned}
\tag{10}
$$

Hybrid Layouts

Generic: Read Cost Model

Scan: Amount of data read is the same to the size of write

Projection:

Vertical Layouts

$$Size(Projection_{vertical}) = Size(Header_{vertical}) + Size(Footer_{vertical}) + Size(OneColWithMeta) * RefCols(IR)$$

Hybrid Layouts

$$Used_{rows}(RowGroup) = \frac{|IR|}{Used_{RG}(Hybrid)} \quad (15)$$

$$Size(RefCols) = (Size(Col) * Used_{rows}(RowGroup) + Size(Meta_{YCol})) * RefCols(IR) \quad (16)$$

$$Size(Projection_{hybrid}) = Size(Header_{hybrid}) + Size(Footer_{hybrid}) + (Size(RefCols) + Size(Meta_{YRG})) * Used_{RG}(Hybrid) \quad (17)$$

Selection:

Hybrid Layouts

$$P(RGselected) = 1 - (1 - SF)^{Used_{rows}(RowGroup)} \quad (19)$$

$$Size(RowsSelected) = (Size(Col) * SF * |IR| + Size(Meta_{YCol})) * Cols(IR) \quad (20)$$

$$Used_{RG}(Selection_{hybrid}) = \begin{cases} Used_{RG}(Hybrid) * P(RGSelected) & \text{if Unsorted} \\ \left\lceil \frac{Size(RowsSelected)}{Size(RowGroup)} \right\rceil & \text{if Sorted} \end{cases} \quad (21)$$

$$Size(Selection_{hybrid}) = Size(Header_{hybrid}) + Size(Footer_{hybrid}) + (Used_{RG}(Selection_{hybrid}) * Size(RowGroup)) \quad (22)$$

Instantiation: SequenceFile

$$Size(Body_{horizontal}) = (Size(Row) + Size(Meta_{HRow})) * |IR| + Size(Meta_{HBody}) \quad (6)$$

$$Size(Meta_{SRow}) = ((Cols(IR) - 1) - 1) \quad (24)$$

$$Size(Row_{SeqFile}) = Size(Record_{length}) + Size(Key) + (Size(Col) * (Cols(IR) - 1)) \quad (25)$$

$$Size(KVPairs) = (Size(Row_{SeqFile}) + Size(Meta_{SRow}) * |IR| \quad (26)$$

$$Size(Meta_{SBody}) = \left\lceil \frac{Size(KVPairs)}{2000} \right\rceil * Size(SyncMarker) \quad (27)$$

$$Size(Body_{SeqFile}) = Size(KVPairs) + Size(Meta_{SBody}) \quad (28)$$

Instantiation: Avro

$$
\begin{aligned}
Size(Body_{horizontal}) = (Size(Row) + Size(Meta_{HRow})) * |IR| \\
+ Size(Meta_{HBody})
\end{aligned}
\tag{6}
$$

$$
\begin{aligned}
Size(Header_{avro}) = Size(Version) + Size(Schema) \\
+ Size(Codec) + Size(SyncMarker)
\end{aligned}
\tag{30}
$$

$$Size(TotalRows) = (Size(Row) + Size(Meta_{ARow})) * |IR| \tag{31}$$

$$Size(Meta_{ABody}) = (8 + Size(SyncMarker)) * \left\lceil \frac{Size(TotalRows)}{Size(B_{avro})} \right\rceil \tag{32}$$

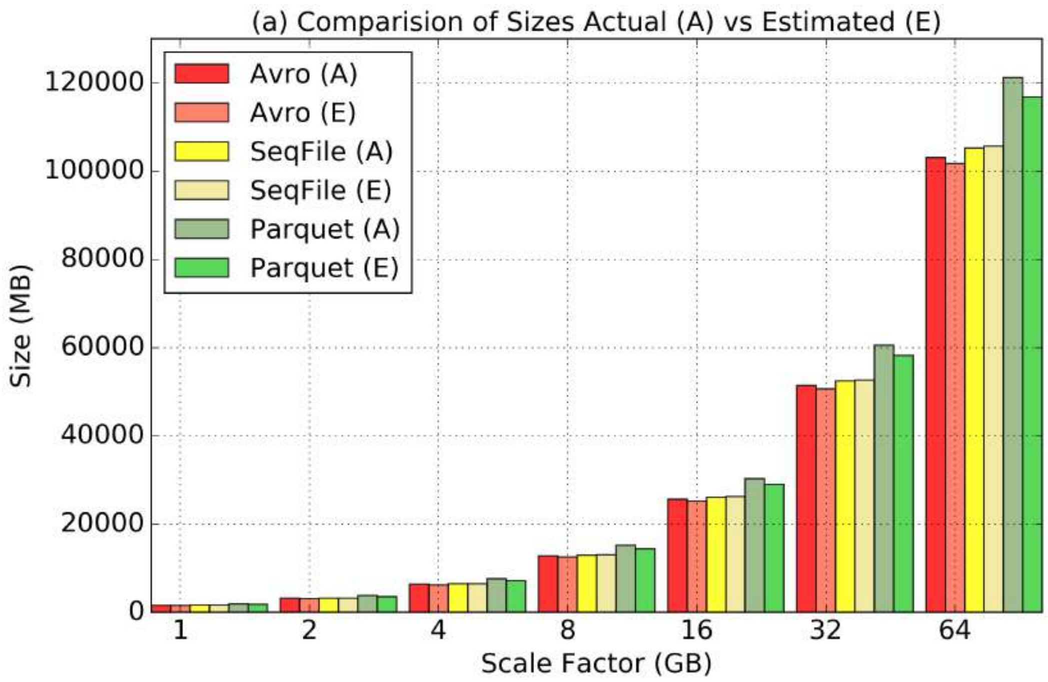
$$Size(Body_{avro}) = Size(TotalRows) + Size(Meta_{ABody}) \tag{33}$$

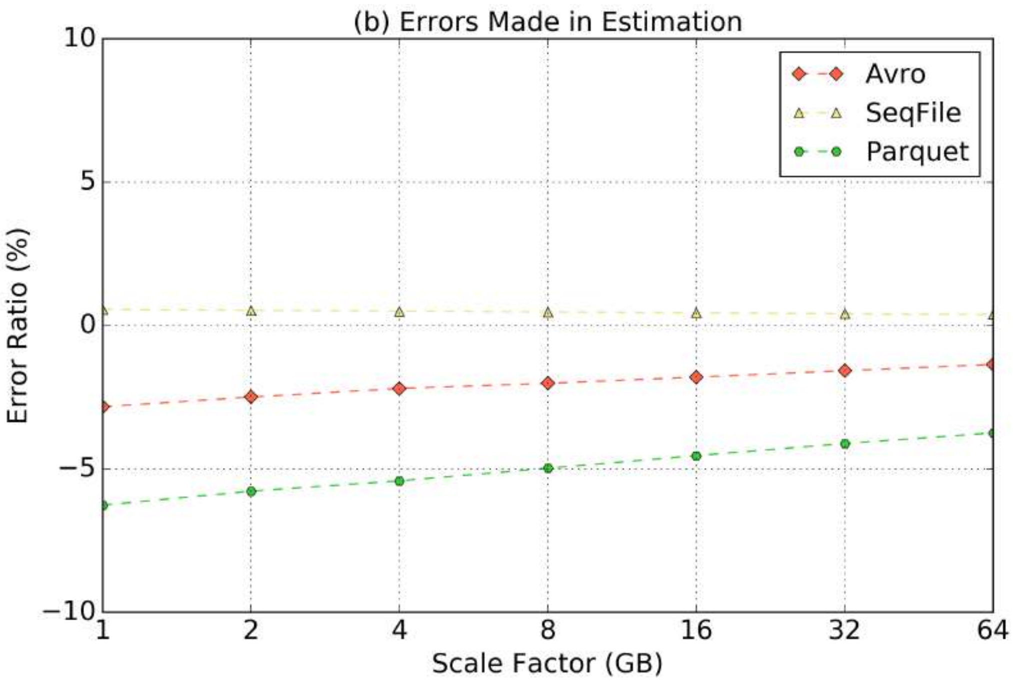
Version (5 Bytes)	Schema (Dynamic)	Compression Codec (4 Bytes)	Sync Marker (16 Bytes)	# of Rows in each Block (8 Bytes)	Row Size (8 Bytes)	Actual Rows (Dynamic)	....	Sync Marker after each block (16 Bytes)

Cost-based Approach: Two Phase Algorithm

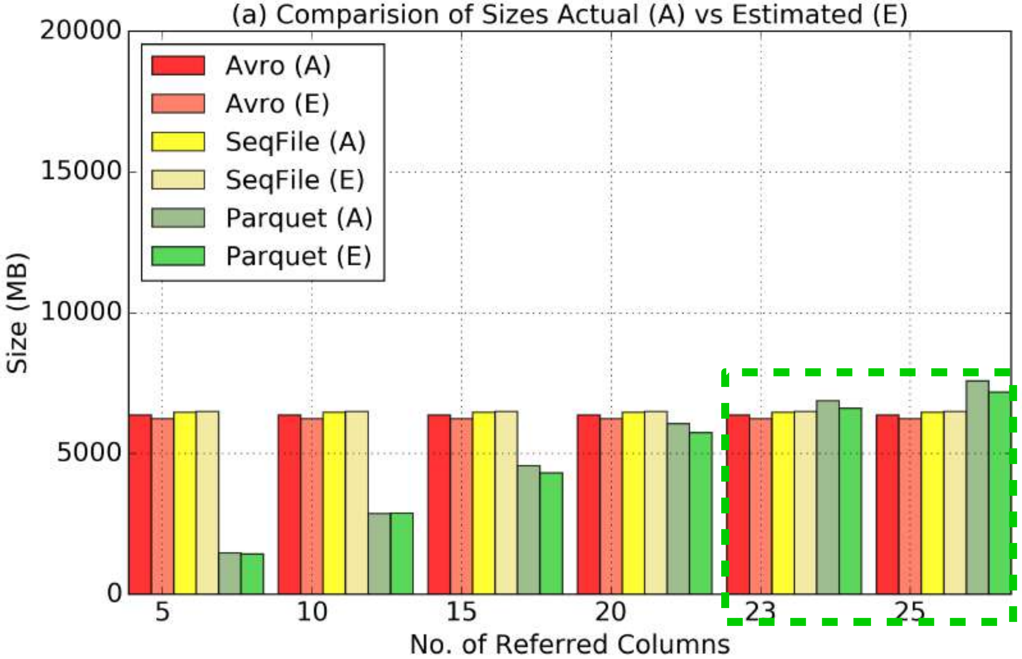
- **First phase**
  - No statistical information available
  - Choose storage formats using rule-based approach
  - Execute workflows and also record statistical information for every materialized node
    - Use job history files of Hadoop to extract information
- **Second phase**
  - Use cost based approach using recorded statistical information
  - Compare newly chosen storage formats with old storage formats chosen using rule-based approach
  - If it is the same than keep the same else discards the old one and use the new one to materialize

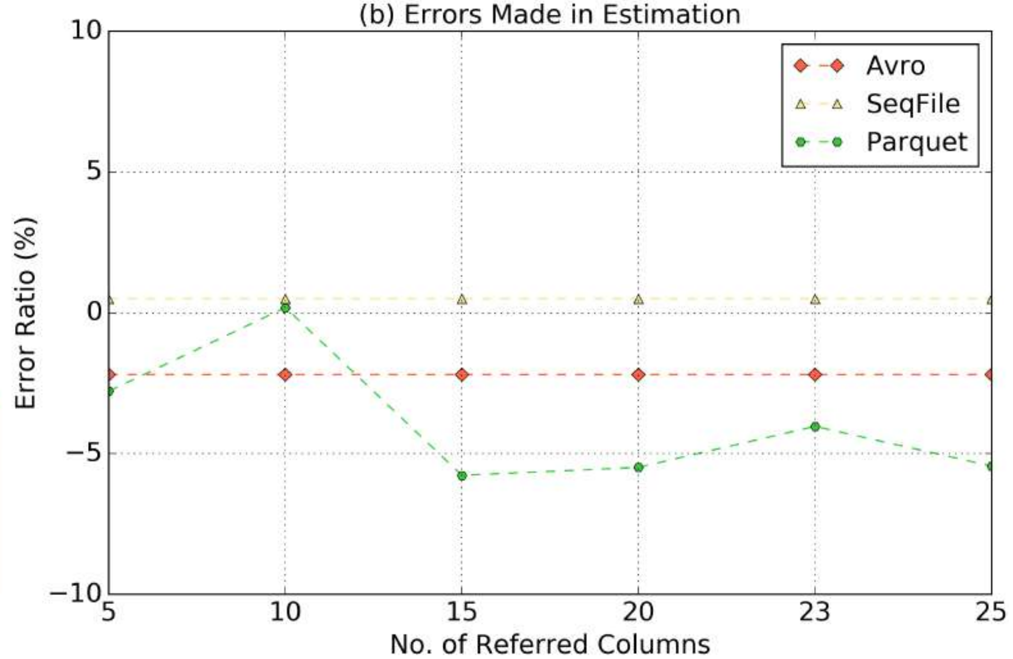
Validation of the Cost Model (Write and Scan)



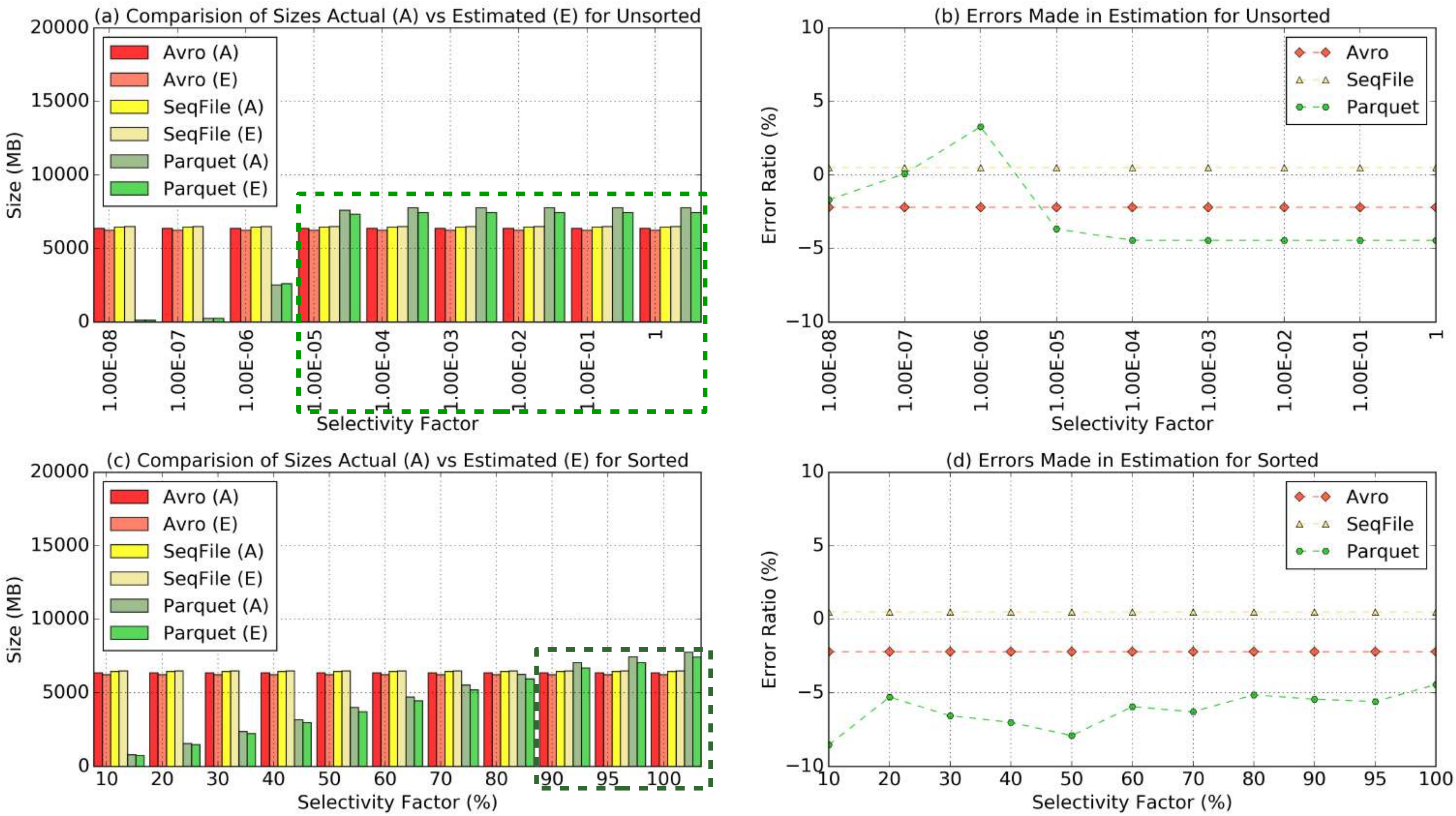


Cost-Based Approach: Validation (Projection)

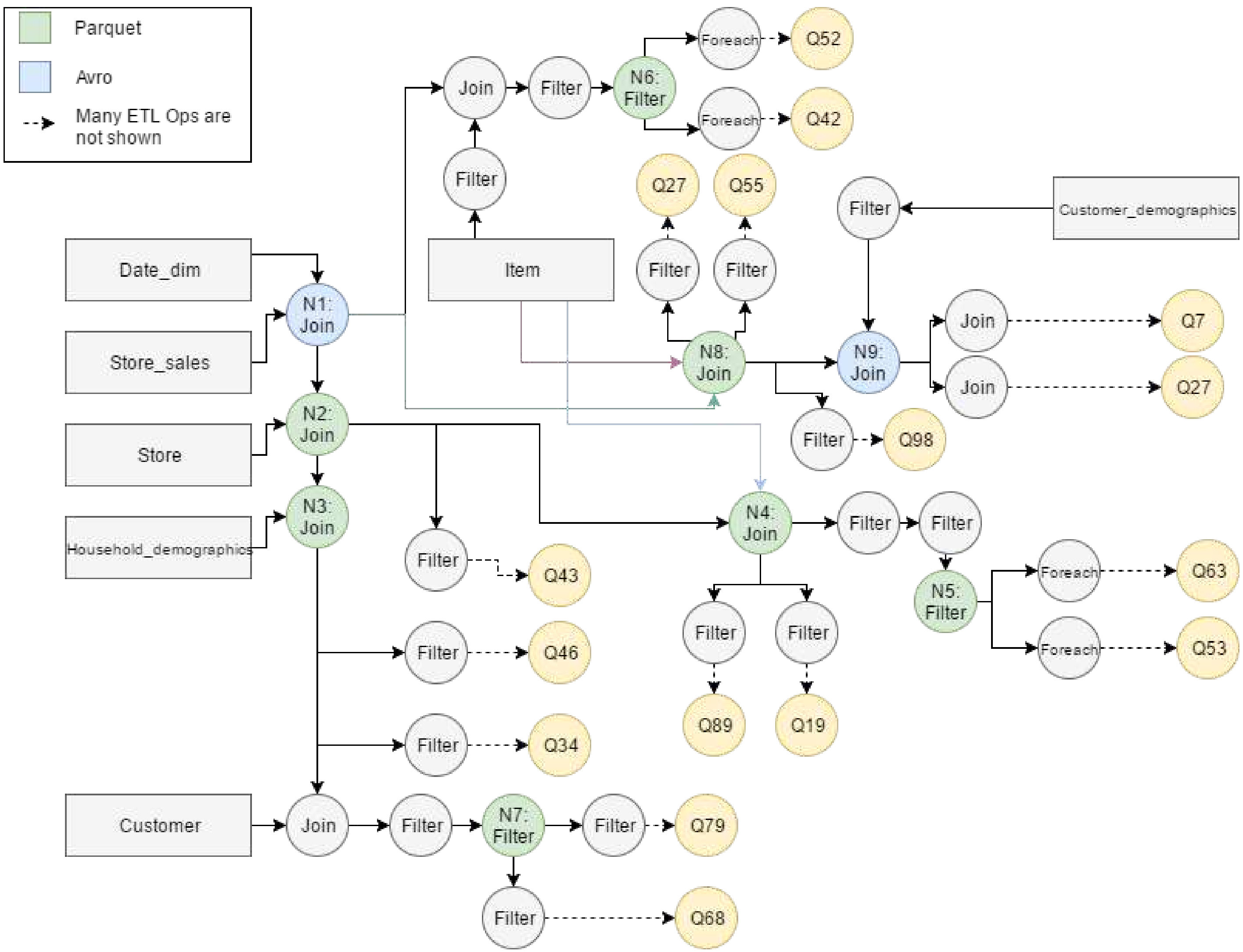




Cost-Based Approach: Validation (Selection)



Cost-based Approach: Experimental Evaluation



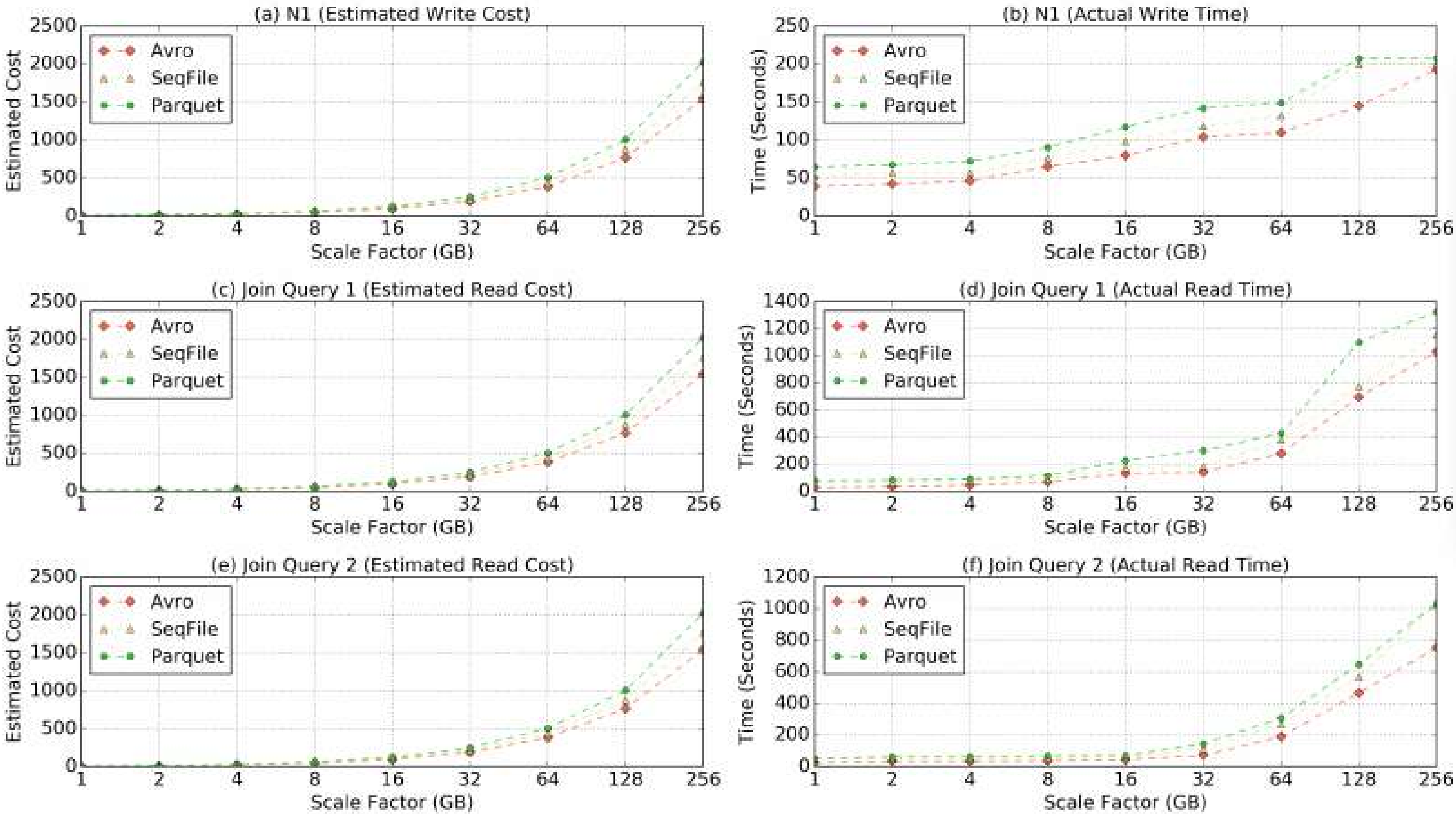
Cost-based Approach: Experimental Evaluation

Node	Subsequent Access Patterns	Statistics	First Phase (Rule-based)
N1	Join, Join	Nil	Avro
N2	Join, Join, Filter	Nil	Parquet
N3	Join, Filter, Filter	Nil	Parquet
N4	Filter, Filter, Filter	Nil	Parquet
N5	Foreach, Foreach	Nil	Parquet
N6	Foreach, Foreach	Nil	Parquet
N7	Filter, Filter	Nil	Parquet
N8	Join, Filter, Filter, Filter	Nil	Parquet
N9	Join, Join	Nil	Avro

Node	Subsequent Access Patterns	Statistics	First Phase (Rule-based)	Second Phase (Cost-based)
N1	Join, Join	-	Avro	Avro
N2	Join, Join, Filter	SF: 0.19	Parquet	Avro
N3	Join, Filter, Filter	SF: (0.59,0.01)	Parquet	Avro
N4	Filter, Filter, Filter	SF: (0.03,0.2,0.19)	Parquet	Avro
N5	Foreach, Foreach	Ref Cols: (3,3)	Parquet	Parquet
N6	Foreach, Foreach	Ref Cols: (4,4)	Parquet	Parquet
N7	Filter, Filter	SF: (0.13,0.92)	Parquet	Avro
N8	Join, Filter, Filter, Filter	SF: (0.19,0.03,0.01)	Parquet	Avro
N9	Join, Join	-	Avro	Avro

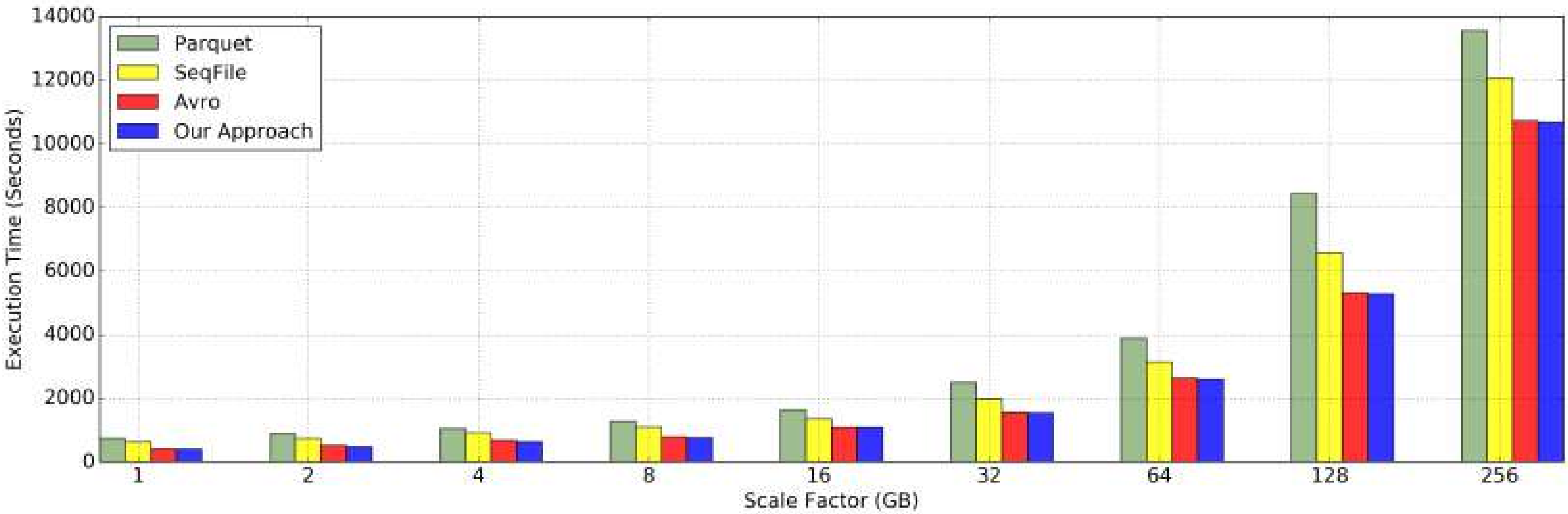
Execute and Record the Statistics for the Second Phase

Cost-based Approach: Experimental Evaluation



Note that we are comparing trend, not the actual numbers

Cost-based Approach: Experimental Evaluation



- Our approach on average provides
  - 60% speed up over fixed Parquet
  - 34% speedup over fixed SequenceFile
  - 3% speedup over fixed Avro
  - and overall it provides 33% speedup.

Current Work: Compression

- **Compression**
  - Block-level compression
    - Calculate compression and decompression cost
    - Add compression cost in write cost and decompression cost in read
    - Use random sampling to measure compress and decompress rate
  - Lightweight compression (Encoding)
    - It is data type dependent
    - Horizontal layouts don’t have its support
      - Currently, we are not considering it to do fairer comparison with horizontal layouts

Current Work: Tuning a Hybrid Layout

- **Motivation**
  - Data Lakes are popular
    - Dump all data from different sources
    - Many ETL jobs to process raw data and output a clean data in some binary format (i.e., Parquet)
    - Run analytical queries directly on the clean data using any distributed processing framework
  - Hybrid layouts are popular for their useful features
    - Schema
    - Projection and Selection pushdown

Current Work: Tuning a Hybrid Layout

- **Configurable Parameters**
  - Row Group Size
  - Data Page Size
  - Block-level Compression
  - Dictionary Encoding
- **Problem Statement**
  - Input
    - Workload Characteristics
    - Data Characteristics
  - Output
    - Optimal Sizes for
      - Row Group
      - Data Page
    - Enable or Disable
      - Compression
      - Dictionary Encoding

- **Why Workload Characteristics?**
  - Fine-grained partitioning for aggressive data skipping (SIGMOD 2014)
    - Filter commonality
      - 10% of the filters are used by 90% of the queries
    - Filter stability
      - Designing a data layout based on past query filters can also benefit future queries
  - Wide Table Layout Optimization based on Column Ordering and Duplication (SIGMOD 2017)
    - Query Update Frequency
      - Less than 5% queries change within a month

Current Work: Tuning a Hybrid Layout

- **Workload Characteristics**
  - Referred Columns
    - Min
    - 1st Quartile
    - Median
    - Mean
    - 3rd Quartile
    - Max
  - Selectivity
    - Min
    - 1st Quartile
    - Median
    - Mean
    - 3rd Quartile
    - Max
  - Attribute Affinity Matrix
  - Operator Affinity Matrix

- **Data Characteristics**
  - Each Column
    - Is Sorted or Not?
    - # of Repetitive Values
    - Distribution
      - Evenly Distribution
      - Skewed

Current Work: Tuning a Hybrid Layout

- **Heuristics**
  - Row Group and Data Page Sizes
    - Low Selectivity
      - Small to have less rows
    - High Selectivity
      - Large to have more rows to help in scan
    - Selectivity between low and high
      - Need to find an optimal row group size
      - We can utilize our cost model to estimate row groups

Future Directions

- **Include New Storage Format**
  - Carbondata
    - Index-based columnar format
      - Inverted indexes
      - Multi-level indexes
    - Column Groups
    - Sorting data
    - Rowid to map column value of each record
    - Include it in the cost model

Plan for ECTS

Activity	Place/Organised by	ECTS	General/Project course	Status
Mining Data Streams	UPC	1.0	Project	Passed
Academic Writing and Communication	UPC	1.0	General	Passed
PhD Retreat Event	TUD	2.0	Project - informal activity	Passed
IT4BI Summer School 2015	IT4BI	2.0	Project	Passed
Research Group Seminar	UPC	2.0	Project - informal activity	Passed
Text Mining and Sentiment Analysis	UPC	1.0	General	Passed
Python for Data Science	UPC	1.0	General	Passed
Time-Series Management	UPC	1.0	General	Passed
Introduction to Big Data with Apache Spark	EDX	1.0	Project	Passed
Scalable Machine Learning	EDX	1.0	Project	Passed
Conference Attendance	MEDI	2.0	Project - informal activity	Passed
IT4BI Summer School 2016	IT4BI	2.0	Project	Passed
Research Group Seminar	TUD	2.0	Project - informal activity	Passed
Big Data Architectures	UPC	1.0	Project	Passed
Apache Pig 101	BigData University	1.0	Project	Passed
Accessing Hadoop Data using Hive	BigData University	1.0	Project	Passed
Complex Event Processing	UPC	1.0	Project	Passed
Natural Language Processing and Sentiment Analysis	UPC	1.0	Project	Passed
IT4BI Doctoral Colloquium	IT4BI-DC	3.0	Project	Planned
Spanish Language Course I	UPC	2.5	General	Planned

Plan for Publications

1 ResilientStore: A Heuristic based Data Format Selector for Intermediate Results

- Authors: Rana Faisal Munir, Oscar Romero, Alberto Abelló, Besim Bilalli, Maik Thiele, and Wolfgang Lehner
- Description: This conference paper proposes a rule based approach to choose a layout for the intermediate results. This approach is implemented in Hadoop ecosystem to test its performance in real world environment. [Published: MEDI 2016]

2 Intermediate Results Materialization Selection and Format for Data-Intensive Flows

- Authors: Rana Faisal Munir, Sergi Nadal, Oscar Romero, Alberto Abelló, Petar Jovanovic, Maik Thiele and Wolfgang Lehner
- Description: Extension of our conference paper. [Submitted: Fundamenta Informaticae]

3 A Cost-based Storage Format Selector for Intermediate Results

- Authors: Rana Faisal Munir, Alberto Abelló, Oscar Romero, Maik Thiele, and Wolfgang Lehner
- Description: In this journal paper, we extended the rule based paper and replaces these rules with a cost model. This paper also focuses on choosing the proper layout for intermediate results. We will submit it in TODS.

4 Automatic Tuning of Hybrid Layouts Based on the Workload

- Authors: Rana Faisal Munir, Alberto Abelló, Oscar Romero, Wolfgang Lehner, and Maik Thiele
- Description: In this conference paper, we will focus on tuning hybrid layouts based on the workload. We will submit it in DASFAA.

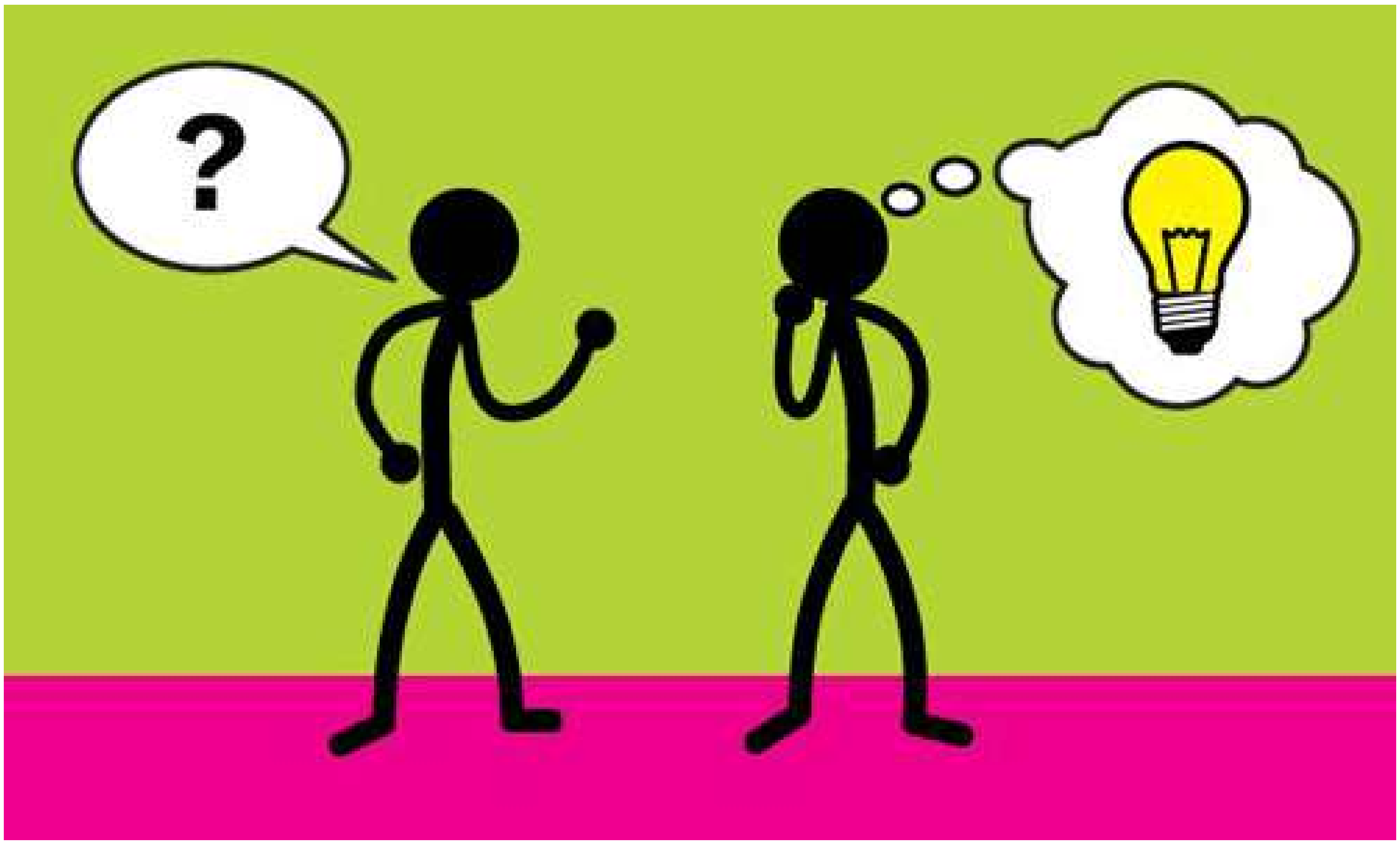
5 When is It Beneficial to Use Index-based Hybrid Layout?

- Authors: Rana Faisal Munir, Alberto Abelló, Oscar Romero, Wolfgang Lehner, and Maik Thiele
- Description: Recently, an index-based hybrid layout emerges as a new storage format. This layout promises a lot of features and the evaluation results show its benefits over simple hybrid layout. In this paper, we validate its performance in different workloads and we also propose a cost model for it. We will submit it to IEEE BigData.

Overall Status

March 2015 - July 2015	Identify the topic by reading state of the art and define clearly all the objective of the thesis.	**Milestone:** Delivery of Doctoral Project Plan (DPP)	DONE
September 2015 - March 2016	We completed our first milestone by proposing a rule-based approach and did a detailed evaluation to show its benefits. We shared our findings with the research community by writing a paper titled as "ResilientStore: A Heuristic based Storage Format Selector for Intermediate Results"[Appendix A].	**Milestone:** The rule-based research paper.	DONE
April 2016 - July 2016	We started working on a cost model and also prepared Thesis Proposal Report (TPR) by updating DPP with more concrete problems.	**Milestone:** Delivery of the TPR	DONE
September 2016 - March 2017	We completed our cost model and validated it through a detailed experiments by using standard benchmark suits. Our cost model showed better performance than the rule-based approach. We wrote a journal paper titled as "A Cost-based Storage Format Selector for Intermediate Results"[Appendix C].	**Milestone:** The Journal paper based on the cost model.	DONE
April 2017 - July 2017	We got invitation from our rule-based paper to submit as a journal in a special issue[Appendix B]. We updated our conference paper by collaborating with one of our colleague. We are also working on automatic tuning of hybrid layouts and planning to write a conference paper.	**Milestone:** Delivery of Research Progress Report (RPR) and submission of our cost-based journal paper.	Partially DONE
September 2017 - March 2018	We will submit automatic tuning paper and in the meanwhile, we will be working on designing a cost model for index-based hybrid layouts and publishing it as a conference paper. We will also be focusing on thesis writing.	**Milestone:** Submission of two conference papers and delivery of the Doctoral Thesis.	PLANNED

Feedback and Questions



Related Work

2008

Column-Stores
vs.
Row-Stores:
How Different
Are They Really? (SIGMOD)

2009

1) SequenceFile
2) Zebra
3) Avro

2011

1) Trojan Layouts (SOCC)
2) RCFile (ICDE)
3) Column-Oriented Storage Techniques for MapReduce (VLDB)

2013

1) ORC
2) Parquet
3) A Comparison of Knives for Bread Slicing (VLDB)

2014

1) H2O: A Hands-free Adaptive Store (SIGMOD)
2) Fine-grained partitioning for aggressive data skipping (SIGMOD)

2016

1) KCGS-Store: A Columnar Storage Based On Group Sorting Of Key Columns (Cloud)
2) Carbondata
3) Skipping-oriented partitioning for columnar layouts (VLDB)

2017

1) Wide Table Layout Optimization based on Column Ordering and Duplication (SIGMOD)

Current Work: Compression

- Use random sampling to record
  - $Compress_{Ratio}$ (Percentage)
  - $Rate_{Compress}$ (Bytes / Second)
  - $Rate_{Decompress}$ (Bytes / Second)
- Cost Model
  - Write
    - $Size_{compress}(X) = Size(X) * Compress_{Ratio}$
    - $Cost_{compress}(X) = Size(x)/Rate_{compress}$
    - $Cost_{Writecompressed}(X) = Cost_{Write}(Size_{compress}(X)) + Cost_{compress}(X)$
  - Read
    - $Cost_{decompress}(X) = Size_{operation}(X) / Rate_{decompress}(X)$
    - $Cost_{readcompress}(X) = Cost_{operation}(X) + Cost_{decompress}(X)$