Fast Spark Access To Your Data - Avro, JSON, ORC, and Parquet

Owen O'Malley
owen@hortonworks.com
@owen_omalley

June 2018

HORTONWORKS®
POWERING THE FUTURE OF DATA™

Who Am I?

- Worked on Hadoop since Jan 2006
- MapReduce, Security, Hive, and ORC
- Worked on different file formats
  - Sequence File, RCFile, ORC File, T-File, and Avro requirements

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

HORTONWORKS

Goal

- Benchmark for Spark SQL
- Seeking to discover unknowns
  - How do the different formats perform?
  - What could they do better?
- Use real & diverse data sets
  - Over-reliance on artificial datasets leads to weakness
- Open & reviewed benchmarks

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

The File Formats

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Avro

- Cross-language file format for Hadoop
- Schema evolution was primary goal
- Schema segregated from data
  - Unlike Protobuf and Thrift
- Row major format

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

JSON

- Serialization format for HTTP & Javascript
- Text-format with MANY parsers
- Schema completely integrated with data
- Row major format
- Compression applied on top

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

ORC

- Originally part of Hive to replace RCFile
  - Now top-level project
- Schema segregated into footer
- Column major format with stripes
- Rich type model, stored top-down
- Integrated compression, indexes, & stats

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Parquet

- Design based on Google's Dremel paper
- Schema segregated into footer
- Column major format with stripes
- Simpler type-model with logical types
- All data pushed to leaves of the tree

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Data Sets

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

NYC Taxi Data

- Every taxi cab ride in NYC from 2009
  - Publically available
  - http://tinyurl.com/nyc-taxi-analysis
- 18 columns with no null values
  - Doubles, integers, decimals, & strings
- 2 months of data – 22.7 million rows

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Sales

- Generated data
  - Real schema from a production Hive deployment
  - Random data based on the data statistics
- 55 columns with lots of nulls
  - A little structure
  - Timestamps, strings, longs, booleans, list, & struct
- 25 million rows

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Github Logs

- All actions on Github public repositories
  - Publically available
  - https://www.githubarchive.org/
- 704 columns with a lot of structure & nulls
  - Pretty much the kitchen sink
- 1/2 month of data – 10.5 million rows

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Finding the Github Schema

- The data is all in JSON.
- No schema for the data is published.
- We wrote a JSON schema discoverer.
  - Scans the document and figures out the types
- Available in ORC tool jar.
- Schema is huge (12k)

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

HORTONWORKS

Software

 © Hortonworks Inc. 2011 – 2016. All Rights Reserved

Software Versions

- All of these projects are evolving rapidly
  - Spark 2.3.1
  - Avro 1.8.2
  - ORC 1.5.1
  - Parquet 1.8.2
  - Spark-Avro 4.0.0
- Dependency hell 😈

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Configuration

- Spark Configuration
  - **spark.sql.orc.filterPushdown = true**
  - **spark.sql.orc.impl = native**
- Hadoop Configuration
  - session.sparkContext().hadoopConfiguration()
  - **avro.mapred.ignore.inputs.without.extension = false**

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Spark-Avro

- Benchmark uses Spark SQL's FileFormat
  - JSON, ORC, and Parquet all in Spark
  - Avro is provided by Databricks via spark-avro
- It doesn't support all of the Spark types
  - Timestamp as int96
  - Decimal as bytes

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

HORTONWORKS

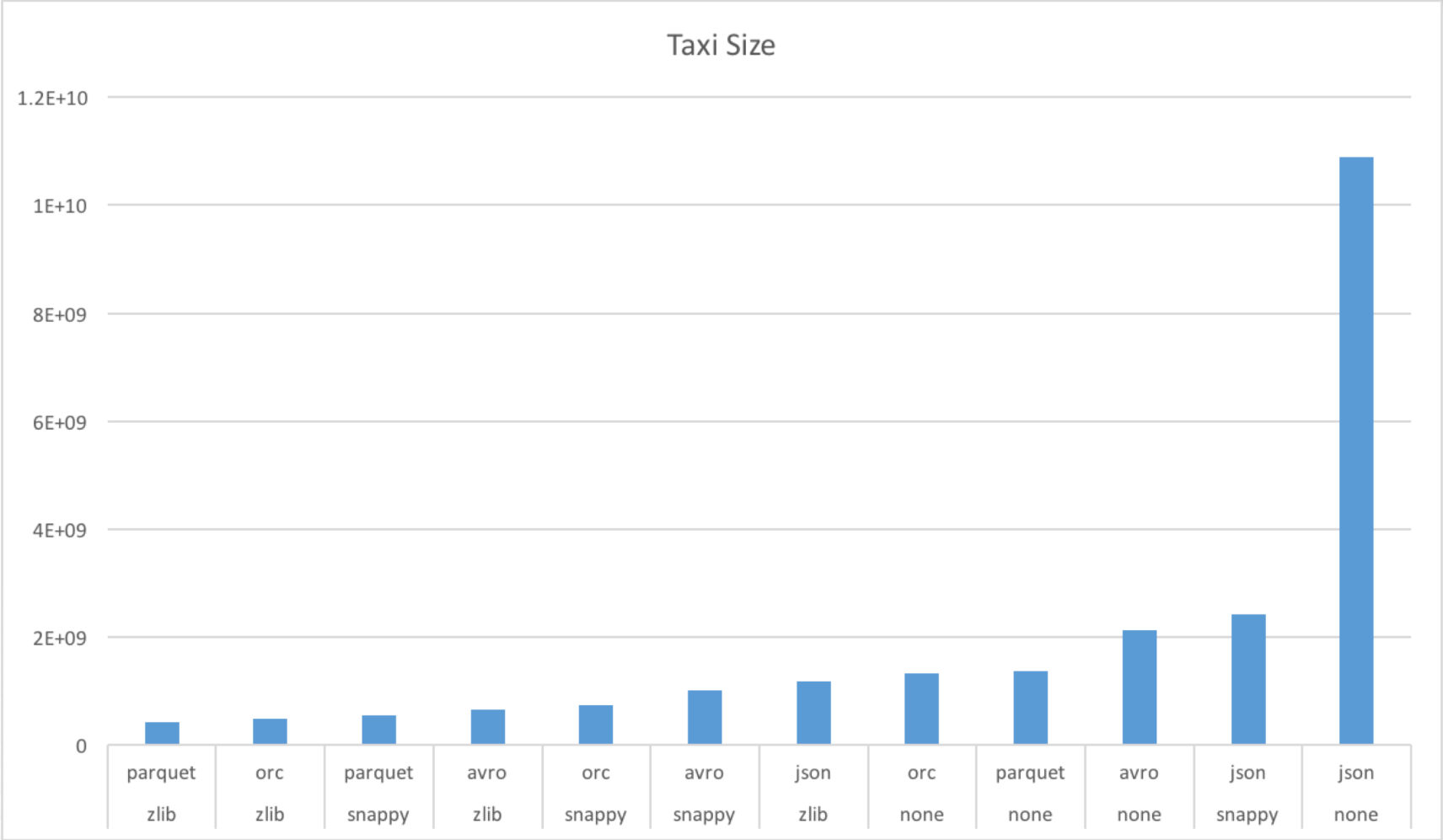
Storage costs

 © Hortonworks Inc. 2011 – 2016. All Rights Reserved

Compression

- Data size matters!
  - Hadoop stores all your data, but requires hardware
  - Is one factor in read speed (HDFS ~15mb/sec)
- ORC and Parquet use RLE & Dictionaries
- All the formats have general compression
  - ZLIB (GZip) – tight compression, slower
  - Snappy – some compression, faster

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

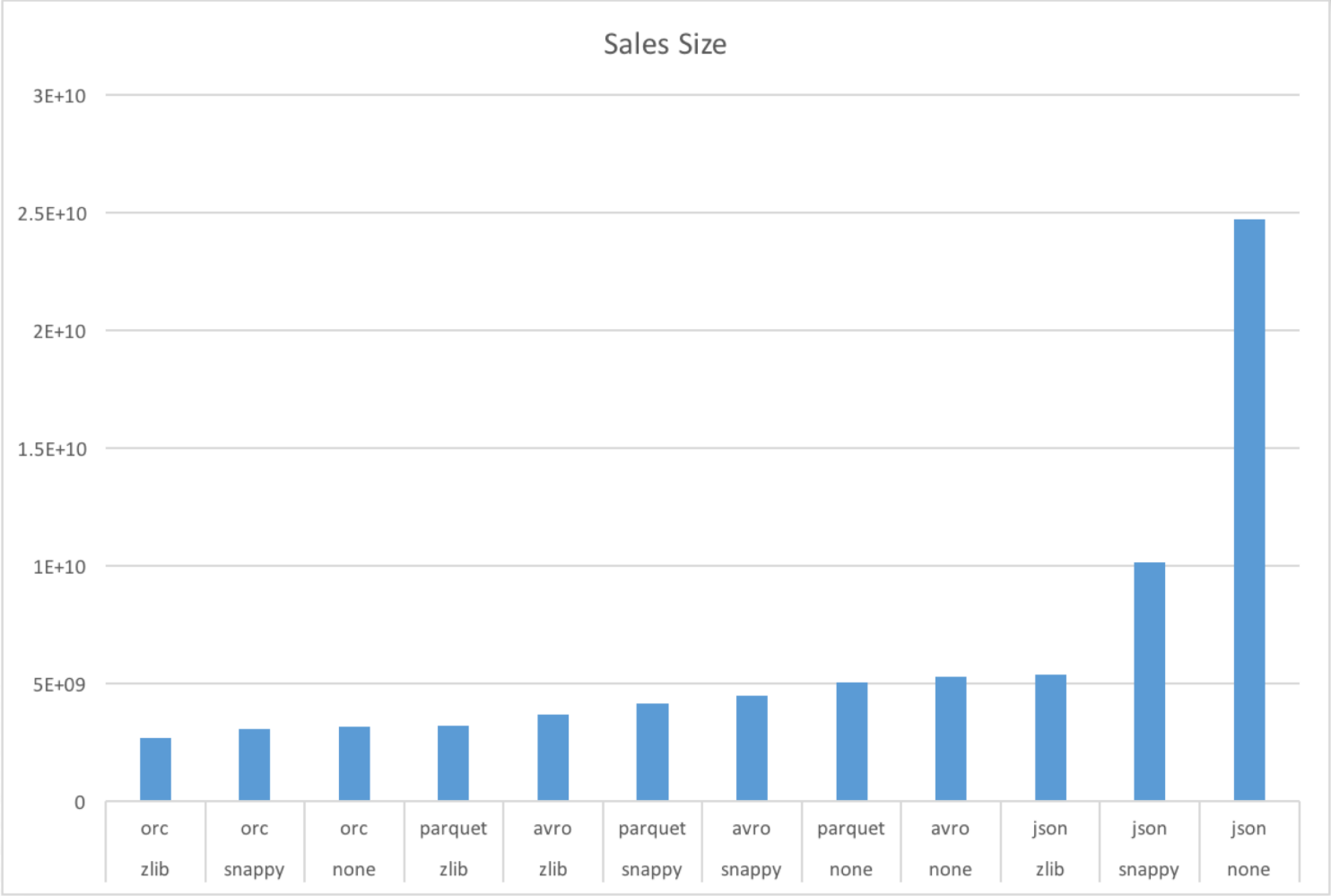


© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Taxi Size Analysis

- Don’t use JSON
- Use either Snappy or Zlib compression
- Avro’s small compression window hurts
- Parquet Zlib is smaller than ORC

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

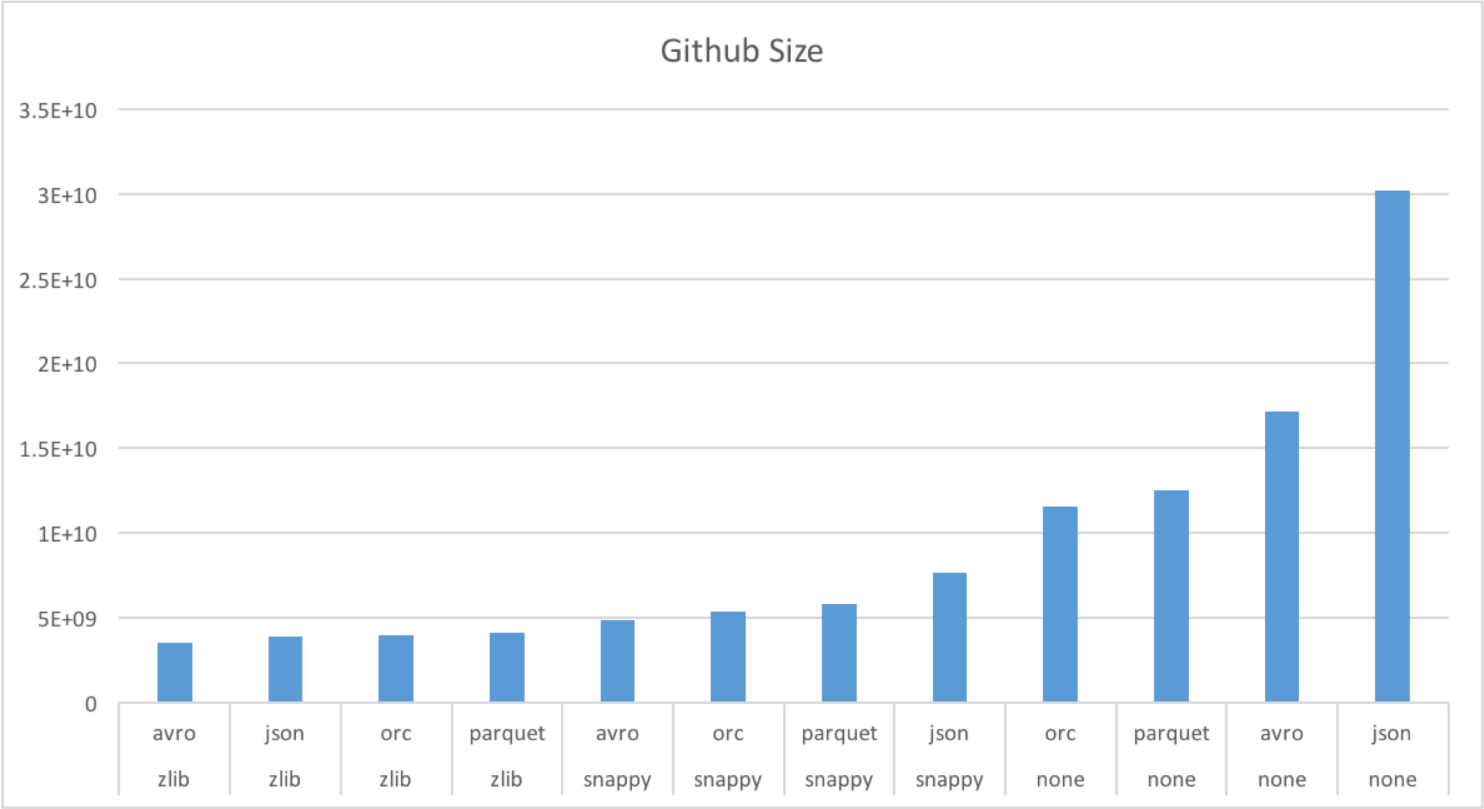


© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Sales Size Analysis

- ORC did better than expected
  - String columns have small cardinality
  - Lots of timestamp columns
  - No doubles ☺

© Hortonworks Inc. 2011 – 2016. All Rights Reserved



© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Github Size Analysis

- Surprising win for JSON and Avro
  - Worst when uncompressed
  - Best with zlib
- Many partially shared strings
  - ORC and Parquet don't compress across columns
- Need to investigate Zstd with dictionary

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

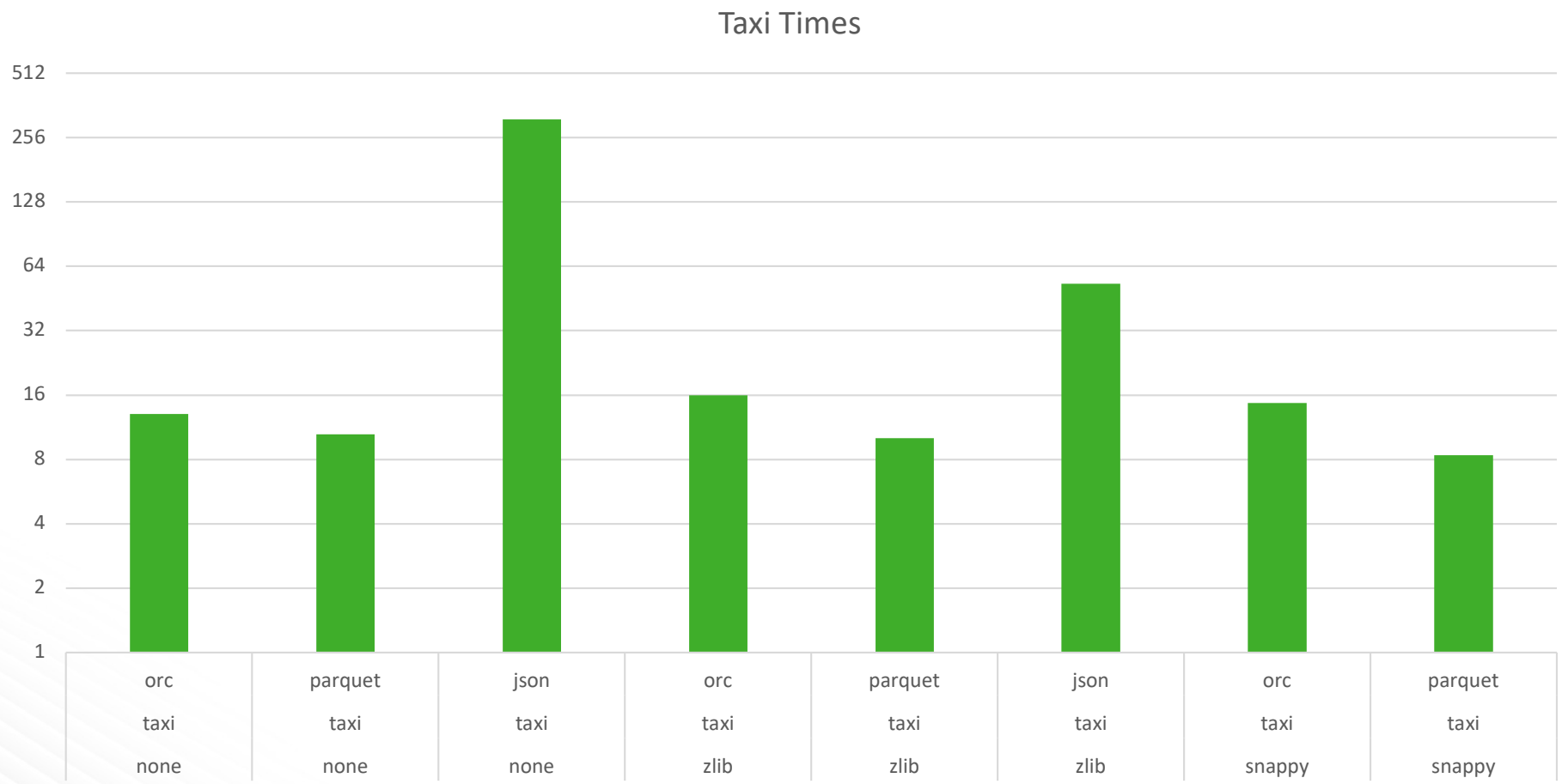
Use Cases

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Full Table Scans

- Read all columns & rows
- All formats except JSON are splittable
  - Different workers do different parts of file
- Taxi schema supports ColumnarBatch
  - All primitive types

© Hortonworks Inc. 2011 – 2016. All Rights Reserved



© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Taxi Read Performance Analysis

- JSON is very slow to read
  - Large storage size for this data set
  - Needs to do a **LOT** of string parsing
- Parquet is faster
  - ORC is going through an extra layer
  - VectorizedRowBatch -> OrcStruct -> ColumnarBatch

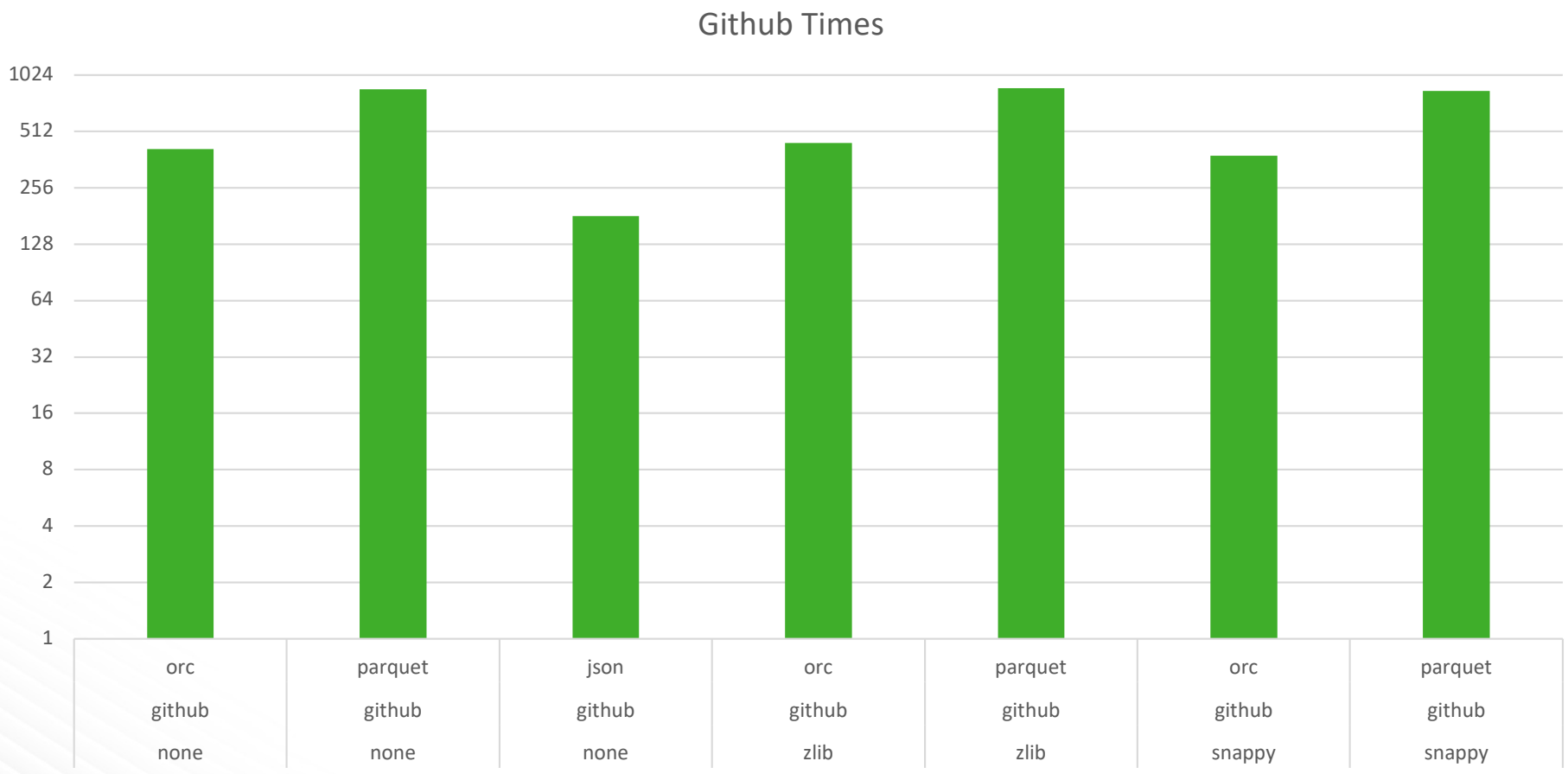
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Sales Read Performance Analysis

- Read performance is dominated by format
  - Compression matters less for this data set
  - Straight ordering: ORC, Parquet, & JSON
- Uses Row instead of ColumnarBatch

© Hortonworks Inc. 2011 – 2016. All Rights Reserved



© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Github Read Performance Analysis

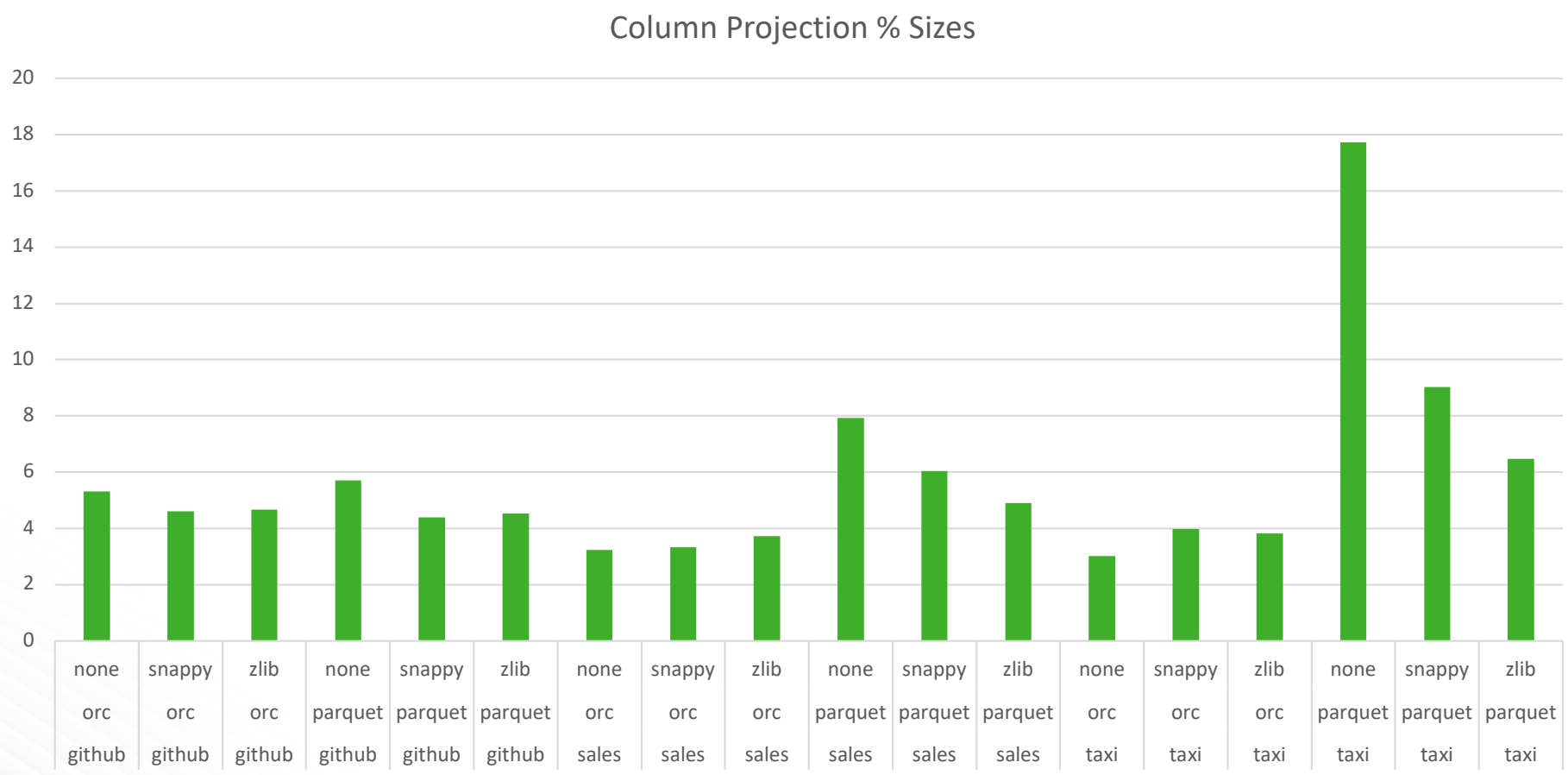
- JSON did really well
- A lot of columns needs more space
  - We need bigger stripes (add min rows in ORC-190)
  - Rows/stripe - ORC: 18.6k, Parquet: 88.1k
- Parquet struggles
  - Twitter recommends against Parquet for this case

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Column Projection

- Often just need a few columns
  - Only ORC & Parquet are columnar
  - Only read, decompress, & deserialize some columns
- Spark FileFormat passes in desired schema
  - Drop columns that aren't needed
  - JSON and Avro read first and then drop columns

© Hortonworks Inc. 2011 – 2016. All Rights Reserved



© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Predicate Pushdown

- Query:
  - select first_name, last_name from employees where hire_date between ‘01/01/2017’ and ‘01/03/2017’
- Predicate:
  - hire_date between ‘01/01/2017’ and ‘01/03/2017’
- Given to FileFormat via filters
- For benchmark, filter on a sorted column

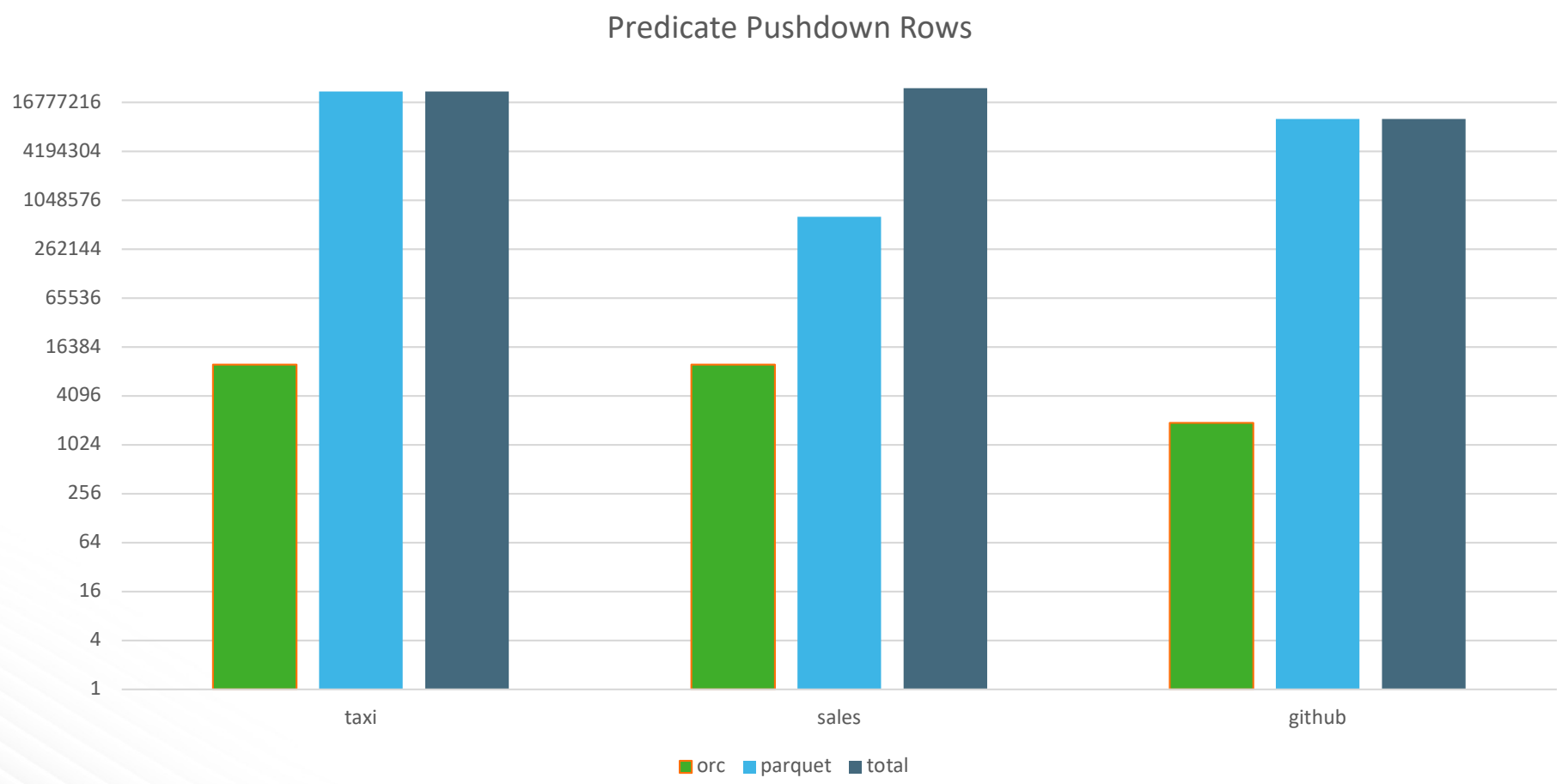
© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Predicate Pushdown

- ORC & Parquet indexes with min & max
  - Sorted data is critical!
- ORC has optional bloom filters
- Reader filters out sections of file
  - Entire file
  - Stripe
  - Row group (only ORC, default 10k rows)
- Engine needs to apply row level filter

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

HORTONWORKS



© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Predicate Pushdown

- Parquet doesn't pushdown timestamp filters
  - Taxi and Github filters were on timestamps.
- Spark defaults ORC predicate pushdown off.
- Small ORC stripes for Github lead to sub-10k row read.
- Because predicate pushdown is an optimization, it isn't clear when it isn't used.

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Metadata Access

- ORC & Parquet store metadata
  - Stored in file footer
  - File schema
  - Number of records
  - Min, max, count of each column
- Provides O(1) Access

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Conclusions

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Recommendations

- Disclaimer – **Everything** changes!
  - Both these benchmarks and the formats will change.
- Evaluate needs
  - Column projection and predicate pushdown are only in ORC & Parquet
  - Determine how to sort data
  - Are bloom filters useful?

© Hortonworks Inc. 2011 – 2016. All Rights Reserved

Thank you!

Twitter: @owen_omalley
Email: owen@hortonworks.com

 © Hortonworks Inc. 2011 – 2016. All Rights Reserved

HORTONWORKS®
POWERING THE FUTURE OF DATA™