Baltic J. Modern Computing, Vol. 5(2024), No. 2, 233-250
http://dx.doi.org/10.22364/bjmc.2017.5.2.06

A Systematic Review of SQL-on-Hadoop by Using Compact Data Formats

Daiga PLASE

Faculty of Computing, University of Latvia, Riga, Latvia

Daiga.Plase@accenture.com

Abstract. There are huge volumes of raw data generated every day. The question is how to store these data in order to provide faster data access. The research direction in Big Data projects using Hadoop Technology, MapReduce kind of framework and compact data formats shows that two data formats (Avro and Parquet) support schema evolution and compression in order to utilize less storage space. In this paper, a systematic review of SQL-on-Hadoop by using Avro and Parquet has been performed over the past seven years (2010–2016) using publications of conference proceedings and journals of IEEEXplore, ACM Digital Library, ScienceDirect. With the help of search strategy followed, 152 research papers have been identified out of which 27 (from year 2013-2016) have been analyzed deeply as relevant papers. At the end, the conclusion has been made that direct comparison by compactness and fastness between Avro and Parquet do not exist in analyzed scientific articles.

Keywords: Big Data, Hadoop, HDFS, Avro, Parquet.

1. Introduction

The amount of data captured by social media, the Internet of Things, enterprises and different types of applications is growing exponentially. There are huge volumes of raw data every day, but these data do not yield much information until processed. As a result of processing, raw data sometimes ends up in a database, which enables the data to become accessible for further processing and analysis in a number of different ways.

Towards distributed and real-time processing of large data sets – the so-called Big Data – the traditional computing techniques are becoming insufficient (Chandra et al., 2012). Hadoop is one of the most common open source Big Data frameworks in the industry today, capable of carrying out common Big Data related tasks. There is growing business demand for Hadoop technology usage in Big Data analysis such as storage, biological data, road, traffic, travel and tourism, telecommunication, enterprise data, citizens' info (Polato et al., 2014). In addition, Hadoop technology is becoming popular in such areas as cloud computing, internet data management (storage, load balancing), implementing MapReduce algorithms for providing solutions to various problems of handling large amount of data, in proposing new models by using HDFS (Sharma et al., 2014).

Often raw data are stored in specific text formats, for instance: JSON, CSV, XML, etc. These formats allow data to be structured and available for humans to read and edit them in most convenient manner. However, storing raw data in a plain text has a significant drawback – there is a disk space needed to store such files. But for Big Data cluster powered by Hadoop it is even a bigger problem because of the high replication factor of each data block within Hadoop File System – HDFS. For instance, recommended HDFS replication factor is 3. That means each raw data block will be replicated 3 times across data nodes. Thus it is crucial to select appropriate data format that enables HDFS storage space utilization in a more efficient manner according to the task defined. Secondly, data storage format may impact the speed of data processing with Hadoop tools, like Hive. Several binary data storage formats exist. Some of them are RCFile, ORC, Avro, Parquet. These formats are designed for systems that use MapReduce kinds of framework. It is a structure that is a systematic combination of multiple components including data storage format, data compression, and optimization techniques for data reading.

This article investigates the research direction in Big Data projects using Hadoop Technology, MapReduce kind of framework and compact data formats such as Avro and Parquet and answers the research questions what are known about the differences in performance (query execution time) between compact data formats Avro and Parquet and which data format (Avro or Parquet) is more compact? The aim of this article is to find scientific articles (indexed in scientific libraries) as scientific evidence that answers the question in which format (Avro or Parquet) to store huge volumes of raw data better: in order to reduce storage space but at the same time – to speed up data processing. There are various researches available in internet that are devoted to this question although these articles are mostly commercially supported, not scientific, not based on TPC or other kind benchmark, not reliable enough to support the decision about better data storage format from compactness and performance point of view. The research task is coming basically from business demand. Why Avro and Parquet? The answer on this question is given in Section 2. Briefly, both are compact, fast, binary data formats that rely on schema. Avro is row-based. Thus there is an assumption that it can perform faster with scan queries. But Parquet is column-based data format. Thus there is an assumption that it can perform faster with aggregation queries. The task of the literature review is to find scientific articles that confirm these assumptions.

It is performed as a small-scale literature review. However, it can be considered as a complete systematic literature review within the scope of this article, for instance, the chosen search strategy and the selected time period.

The systematic review is carried out by identification of research, selection of studies by various authors, deciding upon the inclusion and exclusion criteria and analyzing the amount of publications done in this domain during the time period of year 2010 to 2016. This paper limits its scope to publications done in IEEE Digital Library (IEEE Xplore), ACM Digital Library and ScienceDirect.

2. Background

The Hadoop Technology is commonly being used to manage Big Data projects. Hadoop is now the de facto standard for storing and processing big data, not only for unstructured data but also for some structured data (Chen et al., 2014). The Hadoop Distributed File System (HDFS) is designed to reliably store very large data sets, and to stream those data sets at high bandwidth to user applications (Shvachko et al., 2010). As a result, providing SQL analysis functionality to the big data resided in HDFS becomes more and more important. Hive is a pioneer system that supports SQL-like analysis to the data in HDFS (Chen et al., 2014). This review focuses not only on Hive. Other SQL-on-Hadoop systems such as HortonWorks Stinger or Cloudera Impala are acceptable too, if tests and comparisons of the performance are based on queries selected or derived from world-renowned benchmarks like TPC-H or TPC-DS.

There is another sphere of binary data storage format utilization on direct data sources. For instance, service data gathering from mobile phones to get specific insights of people's behavior or in order to create other kind of location intelligence reports. Assuming that a GPS data packet (timestamp, longitude and latitude) is 100 bytes on average and that the smartphone generates it every 8 seconds, quick math calculations result in 0.043 MB/h, 1.03 MB/day and 376 MB/year. In 2014 over 1.2 billion smartphones were sold (WEB, a). If 1 billion devices produce a GPS data packet every 8 seconds, it results in 1 PB/day. This means that we need ~1000 disk drives with size 1TB in order to store these data. The volume of data is enormous. The question is where and how to store these data in order to provide database for faster execution of data queries. This is the main rationale for this review.

The data storage formats mentioned in Introduction section (Text/CSV (WEB, b), JSON (WEB, c), Avro (WEB, d), SequenceFile (WEB, e), RCFile (He et al., 2011), ORC file (WEB, f), Parquet (WEB, g)) have some advantages and disadvantages. From storage space economy point of view only Avro, SequenceFile, RCFile, ORC file and Parquet support compression. In addition, Avro and Parquet data format support the schema evolution also. This is the main reason why Avro and Parquet has been chosen to find scientific articles that investigates the advantages or disadvantages of these compact data formats.

Avro (WEB, d) is a row-based storage format, also described as a data serialization system similar to Java Serialization. Avro provides rich data structures, a compact, fast, binary data format, a container file to store persistent data, remote procedure call (RPC) features. There is not required code generation to read or write data files, or to use or implement RPC protocols. Alternative systems include Java Serialization, Thrift (WEB, h) and Protocol Buffers (WEB, i) that only work with compile time code generation. Furthermore, Avro can provide more optimized runtime performance (Palmer et al., 2011).

Avro relies on schemas. A schema defines the structure of the data and is used in data reading and writing process. The data schema is defined with JSON and stored into Avro file during data writing process. When Avro data are read, the schema used when writing are always present. This allows data to be written with no per-value over-heads.

Avro is used to save many small files in a single Avro file in HDFS to reduce the namenode memory usage because of user-defined patterns and specific data encoded into binary sequence and stored into a large containing file (Zhang et al., 2014).

Parquet (WEB, g) is a column-based storage format, optimized for work with multi column datasets. Parquet use cases typically involve working with a subset of those columns rather than entire records. One of the most-often cited advantages of columnar data organizations is data compression (Stonebraker et al., 2005) and reduced disk I/O (Abadi et al., 2009) that improves performance of analytical queries (Floratou et al., 2014). Data compression algorithms perform better on data with low information entropy (high data value locality). Thus the system achieves the I/O performance benefits of compression without paying the CPU cost of decompression (Abadi et al., 2009). The layout of Parquet data files is optimized for queries that process large volumes of data.

Information stated ahead is known part of compact data formats, such as Avro and Parquet. Thus, Avro and Parquet choice for the deeper investigation is based on the necessity to investigate these formats by using various queries (scan, aggregation and join) from a world-renowned benchmark like TPC-H and prove the assumption that Avro supported row-oriented data access should provide better performance on scan queries, e.g., when all columns are of interest for the processing, but Parquet format as a counterpart should provide better performance on column-oriented queries, e.g. when only a specific set of those is selected.

Considering that short background information (the rationale for the survey) is given in this and introduction section, the next section of the paper includes literature review. Section III discusses the research methodology used to extract the relevant data for systematic review. Section IV comprises the result set, followed by the conclusion in Section V.

3. Research methodology

This study has been undertaken as a systematic literature review (SLR) based upon guidelines established for the Software Engineering domain (Kitchenham and Charters, 2007). In this section, the protocol used in the SLR has been provided, the research question and its components have been specified, and the requirements regarding the source and primary study selection, the evidence collection and the method of synthesis of such evidence have been established. The results regarding each step are provided alongside the protocol, except summary, which is addressed in Section 4.

In accordance with guidelines (Kitchenham and Charters, 2007), the following steps have been performed in order to conduct this research:

- Defining the objective and the research question that the review is intended to answer.
- Defining the search strategy to be used to do primary studies including looking for terms and resources to be searched.
- Selection of primary studies: Individual studies contributing to a systematic review are called primary studies. The goal of this step is to find out numberless primary studies related to the selected domain.
- Piloting the selection of criteria on a subset of primary studies in order to determine which studies are relevant or which should be excluded from a systematic review. There are several inclusion/exclusion criteria to be considered:
    - o relevance of the topic;
    - o relevance of the subjects;
    - o context;
    - o publication venue.

In addition, it is important to develop a quality checklist in order to assess the individual studies.

- Assessment of quality. In order to evaluate the quality of the collected data, it is necessary to determine the strength of each individual research paper and give more detailed inclusion/exclusion criteria. Previously developed and filled in quality checklist can help to assess the quality of each individual research paper.
- Extraction of relevant data. It is important to define how the information required from each primary study could be obtained.

- Data synthesis. It is necessary to extract data from the primary studies in order to answer the research question, tabulate data in a consistent manner and determine whether the formulated results from the extracted data are consistent with each other or not.

3.1. Objective and research questions

In context of the information given in Introduction and Background section of this article, it is crucial to select an appropriate data format that reduces HDFS storage space and improves the speed of data processing with Hadoop tools, like Hive. The objective of this work is to perform systematic literature review in order to answer the research questions:

RQ.1: What are the differences in performance (query execution time) between compact data formats Avro and Parquet?

RQ.2: Which data format (Avro or Parquet) is more compact?

3.2. Search strategy

To answer the research question, the search strategy has been defined and an extensive search for research papers has been conducted. During data retrieval, the boundaries of the systematic review have been set. The search strategy includes definition of the search scope by research keywords, search strings and sources.

Research keywords have been chosen based on the research question. The synonyms to the keywords have not been considered because the term like "Hadoop" is unique general term that can only be supplemented with related terms such as "Big data", "HDFS", "MapReduce", "Hive" or specific data formats, like "RCFile", "SequenceFile", "ORC", "Avro", "Parquet" etc. The term "Hadoop" has been predefined based on the names defined by the Hadoop developers in the Apache Hadoop website. The final search string has been based on the experience from the pilot searches starting from a broadest search by the term "Hadoop" when IEEEXplore Digital Library's Full Text & Metadata search results in 6,348 articles, and ending by a narrow search by the term "RCFile" when only 4 results have been returned. Sometimes search strings have to be adapted according to the specific needs of digital libraries, but it is not necessary in this case. The search string used to obtain the initial results of this review consists of a Boolean expression:

```
((Hadoop OR HDFS) AND (Avro OR Parquet))
```

The operator OR has been used in Boolean expression in order to extend the list of results and retrieve more articles where Avro or Parquet data format is mentioned independently from each other. The same approach has been applied for terms Hadoop and HDFS, because in the context of data compactness and storage some authors, for instance (Zhang et al., 2014), are using the term HDFS instead of Hadoop.

The criteria used to select sources of studies have been defined as follows:

- Must have web search mechanism;
- Search mechanisms must allow customized searches by title and abstract (preferable – full text);
- Must support the search using Boolean expressions;

- The abstract of paper should be available for free. The abstract unavailability is the main reason why Springer Link has not been chosen as acceptable source during the first step (selection of primary studies). Therefore this review can be extended in direction to cover studies from Springer Link.
- Full articles must be available for download using available contracts between University of Latvia and the digital library. Google Scholar can also be acceptable;
- The digital libraries should index papers on the Big Data topic written in English. Thus, the search strategy limits the search to the papers written in English.

With the search string defined, the following digital libraries have been chosen as sources:

- IEEEXplore Digital Library
- ACM Digital Library
- ScienceDirect

There are two additional search criteria: items and the publication period. The searched items are limited to Journal articles and conference papers, but the publication period is set from year 2010 till 2016, covering seven year period when the most active time of the Hadoop development was (Polato et al., 2014).

3.3. Selection of primary studies

The search query presented in Section 3.2. has been used to retrieve the candidate articles from the digital library systems in the time period of 2010-2016. As shown in Fig. 2, the first initial step is based on it.

Search in digital libraries results in total 152 candidate papers: 133 from IEEE, only 1 from ACM, and 18 from ScienceDirect. As shown in Table 1 and Fig. 1, Hadoop Technology has drawn interest of researchers in the past seven years.

Table 1. Search results

Year	**Digital Library**			**Total**
	IEEE	**ACM**	**ScienceDirect**	
2010	1		1	**2**
2011	1		1	**2**
2012	7			**7**
2013	10		2	**12**
2014	21	1	5	**27**
2015	38		6	**44**
2016	55		3	**58**
Total	**133**	**1**	**18**	**152**
Including deeply analyzed	*22*	*1*	*4*	*27*

The graph shown in Fig. 1 is used in order to better visualize the trend of research articles about Hadoop Technology and compact data formats in the past seven years. As shown in Fig. 1 the number of publications of research papers in conference proceedings, journals and magazines has significantly increased from the year 2010 to 2016.

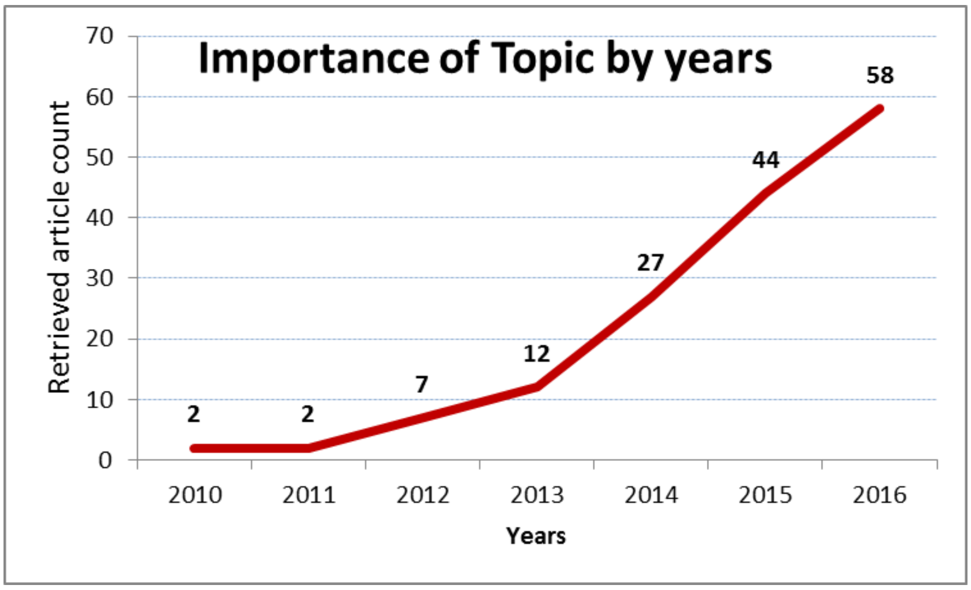

Fig. 1. Importance of topic by years

In the first step, the search results from all digital libraries have been gathered by using citations download or export function. Thus, search results have been obtained in CSV or other delimiter separated format and imported in Excel. Subsequently, the results from all 3 digital libraries have to be summarized in one format sheet where common data fields have been defined. All the relevant studies used for this review are presented in Excel, available at Dropbox (WEB, j). Several parameters have been defined for future analysis and documented for each retrieved article in the summary sheet:

- Document Title
- Authors
- Year
- Abstract
- URL
- Link to PDF
- Keywords
- Article Citation Count
- Patent Citation Count
- Reference Count
- Source (IEEE, ACM or ScienceDirect)
- Number of pages
- Journal Name if Journal article
- Country

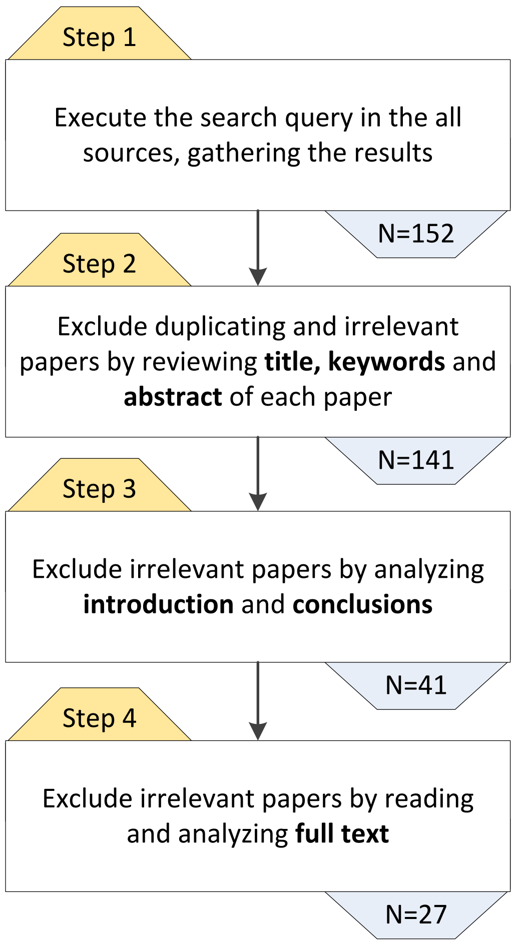
Step 1
Execute the search query in the all sources, gathering the results
N=152
Step 2
Exclude duplicating and irrelevant papers by reviewing **title, keywords** and **abstract** of each paper
N=141
Step 3
Exclude irrelevant papers by analyzing **introduction** and **conclusions**
N=41
Step 4
Exclude irrelevant papers by reading and analyzing **full text**
N=27

Fig. 2. Article selection process

In the second step, Excel macro functionality has been developed in order to perform faster screening and abstract text zooming from delimiter separated metadata. Then, the title, keywords and abstract of all papers have been reviewed. As a result, only 141 papers have been left as relevant. Two from 11 skipped papers have been recognized as irrelevant because the term “Parquet” has been used in the context of wood, but the search by term “HDFS” has not been performed precisely, e.g., it has been applied to HDF surface and letter “S” has been ignored in ScienceDirect search. Although presented in different conferences, other two papers (Tsai et al., 2014a) and (Tsai et al., 2014b) have been recognized as very similar and devoted to NoSQL topic.

In the third step, all remaining 141 papers have to be analyzed individually to confirm the relevance in the context of the review. To select or discard papers, inclusion and exclusion criteria have been defined as follows. The abstract, introduction and conclusions of the paper should have something about such topics as storage space utilization, HIVE, SQL, HDFS, data formats (Avro or Parquet), compactness measurements, performance measurement, queries. The checklist regarding these selection criteria has been developed by supplementing previously created Excel. Information about the publication venue (country) has been analyzed in this step as well.

In the fourth step, a full text reading has been performed for the remaining 41 article. The quality checklist has been created and filled in this step. The full text of articles has been found by using Google Scholar.

3.4. The assessment of relevance and data synthesis

Table 1 and Fig. 1 depict data synthesis from the year 2010 to 2016 respectively. It can be clearly seen that the number of selected studies has increased in past seven years. There are 2 studies in 2010, 2 in 2011, 7 in 2012, 12 in 2013, 27 in 2014, 44 in 2015 and 58 in 2016. After applying the selection criteria, only 27 papers have been selected for data extraction and analysis: 1 study in 2013, 3 in 2014, 13 in 2015 and 10 in 2016.

Table 2. Relevance checklist for selected primary studies

Identi-fier	Reference	Avro ana-lyzed	Par-quet ana-lyzed	Com pact-ness mea-sured	SQL que-ries exe-cuted	TPC bench-mark used	Amount of cita-tions
[PS1]	(Biookaghazadeh et al., 2015)		x		x		1
[PS2]	(Cejka et al., 2015)	x		x	x		4
[PS3]	(Yan and Yuan 2015)		x		x	x	0
[PS4]	(Choi et al., 2015)		x		x	x	0
[PS5]	(Luckow et al., 2015)		x		x	x	8
[PS6]	(Zhang et al., 2014)	x		x			7
[PS7]	(Mammo et al., 2015)	x			x		2
[PS8]	(Grover et al., 2015)		x		x		4
[PS9]	(Dong et al., 2015)	x		x			2
[PS10]	(Zhang et al., 2013)	x			x		0
[PS11]	(Zhou et al., 2015)		x		x	x	0
[PS12]	(Haynes et al., 2015)		x		x		1
[PS13]	(Pirzadeh et al., 2015)		x		x	x	1
[PS14]	(Floratou et al., 2014)	partly	x		x	x	62
[PS15]	(Son et al., 2015)	x			x	x	1
[PS16]	(Tapiador et al., 2014)	partly	x	x	x		11
[PS17]	(Kilias et al., 2015)		x	x	x		2
[PS18]	(Wullink et al., 2016)		x	x	x		9
[PS19]	(Li et al., 2016)		x	x	x		0
[PS20]	(Xu et al., 2016)		x		x		0
[PS21]	(Begoli et al., 2016)		x	x	x		4
[PS22]	(Liu Yue et al., 2016)		x		x	x	0
[PS23]	(Liu Jun et al., 2016)		x		x	x	0
[PS24]	(Wullink et al., 2016)		x		x		2
[PS25]	(Giannakouris et al., 2016)		x		x	x	0
[PS26]	(Mehmood et al., 2016)		x	x	x	x	0
[PS27]	(Tummalapalli et al., 2016)	x			x		1

The selection criteria are based on the assessment of the quality performed with the help of quality checklist. As shown in Table 2, the relevance criteria are based on the research questions, e.g., is the article about Avro, Parquet or both formats, provide comparison by compactness or the performance based on queries selected or derived from the world-renowned benchmarks like TPC-H or TPC-DS (as indirect quality criteria). As shown in Table 2, only 11 studies are based on world-renowned benchmark like TPC-H or TPC-DS. Most of all selected primary studies (10) are originated in the USA. The second place is taken by China (7) but the remaining articles (10) are originated in the Germany, Greece, India, Ireland, Korea, Netherlands, Pakistan and Spain.

3.5. Analytics and conclusions

Why have the selected articles been considered as relevant or excluded from future analysis and final conclusions? In order to answer this question, the short insight of each article is useful.

Biookaghazadeh et al. (2015) introduces a self-describing data format NetCDF that is not supported by existing big-data systems. In this article, four type queries are defined and executed on the raw storage format CSV and NetCDF. The experiment results obtained from typical queries on a geoscience dataset show that the introduced approach substantially outperforms the traditional CSV-based approach. The authors mention only Parquet format in context of the need to improve scientific data formats such as NetCDF and HDF for big-data systems.

Cejka et al. (2015) from Siemens AG company compares file size of four different formats: Java, Protocol Buffers, Thrift and Avro. Avro’s results show that it is much slower in write speed, however much faster in read speed than Protocol Buffers and Thrift. The file compression in Apache Avro is best. In order to evaluate the time of the retrieval of entries, the author’s defined benchmark is used to retrieve data from such databases as Storacle, H2, MongoDB. Parquet format is not analyzed in this paper.

Yan and Yuan (2015) build another TPC-DS benchmark by removing columnar optimization, they name it TPC-DS2, optimize the resource utilization, and maintain fairness among different types of queries. The authors present a price-based algorithm which achieves optimization objective by implementing algorithm in the open source Impala system and conducting a set of experiments in a clustering environment using the TPCDS workload. Experimental results show that coordinated resource management solution can increase the aggregate utility by at least 15.4% compared with simple fair resource share mechanism, and 63.5% compared with the FIFO resource management mechanism. This work demonstrates significant advantage of Parquet format. Avro format is not mentioned in this paper.

Choi et al. (2015) compares the CSV file format and Parquet file format via MicroBricks and x86 clusters. The authors carry out the TPC-H benchmark by means of an open source distributed SQL engine in Hadoop in both architectures. The experimental results are promising for the MicroBricks computing, and the results show that the query response times of the MicroBricks computing architecture outperforms those of commodity cluster without hurting the innate advantages of the MicroBricks cluster architecture. Avro format is not analyzed in this paper.

Luckow et al. (2016) compares different queries derived from TPC-DS and TPC-HS benchmarks and executed on Hive/Text, Hive/ORC, Hive/Parquet, Spark/ORC, Spark/Parquet. Hive/Parquet shows better execution time than Spark/Parquet. Select, aggregate and join queries are executed on a comparable infrastructure Hive/Spark versus RDBMS. Generally, the RDBMS can outperform Hive and Spark – however, both deliver a solid performance at a lower cost. But Avro format is not analyzed here.

Zhang Shuo et al. (2014) compares raw data storage formats versus Avro and propose an original solution to store, read and write different small files on HDFS. However, there is no direct comparison of different data formats and Parquet is not presented there. It is worth mentioning that authors select Avro as a target binary data format and demonstrate its efficiency in both read and write operations.

Mammo and Srividya (2015) propose a Presto-based architecture, Presto-RDF that can be used to store and process big RDF data and SPARQL to SQL compiler. The comparative analysis of the performance of Presto (distributed SQL query engine) in processing big RDF data against Apache Hive has been done. However, Parquet format is not mentioned in this work and Avro is only mentioned in the context of future work, because the RDF data is stored as a text file, which is not optimal. This work can be extended to test using RCFILE, ORC, AVRO formats, which are better optimized than the text file.

Grover et al. (2015) focuses on benchmarking multiple SQL-like big data technologies over Hadoop based distributed file system (HDFS) for Study Data Tabulation Model (SDTM) used in clinical trial databases for improving the efficiency of research in clinical trials. The benchmark proposed in this paper provides an overview of the capabilities of SQL-on-Hadoop platforms such as Hive, Presto, Drill and Spark. The authors mention format Avro and Parquet, but they do not analyze these formats in any kind of comparison. Only Parquet format is mentioned in the future work section as lightweight and fast with a columnar layout, hence it can significantly boost IO performance.

Dong et al. (2015) introduces the Record-aware Compression (RaC) scheme that makes the compressed contents splittable, uses a lightweight Hadoop Record Reader and preserves the parallelism and data locality properties as much as possible. In general, RaC can be used with other analytic platforms such as Spark and higher level abstractions of MapReduce such as Hive. In the evaluation, the authors show that using RaC can greatly reduce data loading time and the required system memory. More importantly, the authors observe that the time spent on decompressing data in memory is trivial compared to the time required for loading data from persistent storage to memory. The experimental results lead the authors to believe that content-aware and data-specific compression is very promising in big data processing and analysis. However, there is no direct comparison of Avro and Parquet data format in the SQL point of view.

Zhang Zhen'an et al. (2013) introduce Alovera, a fast stream processing system for large-scale data. Alovera can easily serialize the records to HDFS by using Avro. The authors prove that the record-oriented data need nearly half of the time to be uncompressed while Avro is used to serialize the data stored in columnar format, and it is efficient to de-serialize the data. Parquet format is not analyzed in this paper.

Zhou et al. (2015) explore a Workload Aware Column Order solution, WACO, to boost the scan operator in a wide table. Although this article does not investigate Avro, the authors implement WACO solution on Parquet data format on top of Hadoop 2.0. The authors conduct extensive experiments of the real-world TPC-H benchmark and SDSS dataset for simulating a wide table to demonstrate the superiority of our solution. The experiment results show that this approach is 2x faster than the state-of-the-art.

Haynes et al. (2015) introduces Terra Populus that acts as the bridge between big data sources and researchers. Researchers are provided with convenient web applications that allow them to access, analyze, and tabulate different datasets under a common platform. Terra Populus' Tabulator application employs Parquet on Spark to build dynamic queries for analyzing large population survey data. The authors recognize that

Parquet allows greater compression per data type. Parquet usage gives a high compression ratio while still allowing for fast data fetching. However Avro format is not analyzed here.

Pirzadeh et al. (2015) reports on an evaluation of four representative Big Data systems (such as MongoDB, Hive, AsterixDB, and a commercial parallel shared-nothing relational database system) using a micro-benchmark called BigFUN. Parquet is used in benchmarking while Avro is not mentioned at all.

Floratou et al. (2014) compares three analytical job execution environments available in Hadoop ecosystem. Hive on MapReduce, Hive on Tez and Impala have been analyzed here by using a world-renowned benchmark like TPC-H. As a result, the authors confirm that Impala has better performance versus Hive (both versions). The authors conduct a detailed analysis of Impala and Hive using TPC-H data, stored as text, Parquet and ORC file formats. This is the most cited article from all deeply analyzed studies. Although, the authors compare ORC versus Parquet, the ORC format cannot be considered as completely row-based format like Avro. This can be considered as good article for ORC and Parquet format comparison, but not for Avro versus Parquet.

Son et al. (2015) proposes a novel column-store method called SSFile for Hadoop-based distributed systems. SSFile increases the actual amount of data processed per task and supports representative columnar execution techniques for efficient query processing. Through experiments authors show that SSFile significantly improves the performance of distributed processing. Avro schema is used in SSFile creating and benchmarking while Parquet format is not mentioned at all. Furthermore, the authors use only a few queries from the TPC-H benchmarking and do not argument this choice.

Tapiador et al. (2014) compares the data set size for different compression and format approaches like CSV(Row), Plain(Row), Snappy(Row), GBIN(Row), Snappy(Column), GBIN(Column). Google Snappy codec gives a much better result as the decompression is faster than Deflate (GBIN). It takes half of the time to process the histograms (50%) and the extra size occupied on disk is only around 23%. This confirms the suitability of Snappy codec for data to be stored in HDFS and later on analyzed by Hadoop MapReduce work flows. Although this article gives an answer to the question about compactness between different compression and format approaches like CSV(Row), Plain(Row), Snappy(Row), GBIN(Row), Snappy(Column), GBIN(Column), it does not compare data format Avro versus Parquet from compactness point of view or in another kind of comparison, for instance, SQL query execution time. The data storage model approaching performance comparison does not give a transparent view of how it is obtained.

Kilias et al. (2015) proposes INDREX, a system that provides a single and comprehensive view of the whole process combining both relation extraction and later exploitation with SQL. The authors use Parquet data format to store data in HDFS and observe a compression ratio of a nearly 10x factor: The data size in the Parquet.io file format is reduced from 107GB to roughly 10GB. The authors do not analyze Parquet format in the SQL kind of comparison. Avro format is not analyzed here at all.

Wullink et al. (2016) presents the design of ENTRADA, a high-performance data streaming warehouse that enables researchers and operators to analyze vast amounts of network traffic and measurement data within interactive response times. ENTRADA uses off-the shelf Impala query engine and Parquet file format based on Google's Dremel to achieve high performance. There are highlighted the advantages of the Parquet as an efficient columnar storage format for data storage, which differs from traditional row-based storage in the way the data elements are ordered. However, the

Parquet format is not analyzed in the SQL kind of comparison or compared with row-based Avro format.

Li et al. (2016) gives an overview of Impala's architecture, performance optimization possibilities and conducts a set of experiments in big data analysis system based on Impala to verify the effect of optimization. The performed data compression and queries show that Parquet with snappy compression consistently outperforms by up to 3.7x the text format. Avro format is not analyzed here.

Xu et al. (2016) compares the use of HIVE in conjunction with the MapReduce and Spark programming frameworks, analyzes its performance using different data storage formats, and exemplifies potential use cases to facilitate data analysis of connected vehicles. The authors recognize that the Parquet reduce the space required to store information and introduce limitations on how the data can be split and the number of mappers and reducers that may be used. Parquet is used in benchmarking together with ORC while Avro is not used at all.

Begoli et al. (2016) presents a service platform for schema-less exploration of data and discusses the key features and software architecture of the platform, the underlying core components (Apache Parquet, Drill, the web services server), the runtime profiles and performance characteristics of the platform. The Parquet is used in performance evolution while Avro is not used there.

Liu Yue et al. (2016) compares the data filtering performance of the indexes such as Compact Index, Aggregate Index, Bitmap Index, DGFIndex and the index in ORC file with different columnar storage formats (ORC, RCFile and Parquet) by conducting comprehensive experiments using uniform and skew TPC-H data sets and various multi-dimensional queries. The authors suggest the best practices how to improve multi-dimensional queries in Hive under different conditions. The row-based storage format like Avro is not analyzed here.

Liu Jun et al. (2016) proposes a novel Impala simulation framework to help IT professionals to understand its performance behavior. The Impala simulator has been validated against various software and hardware configurations, using the well-known TPC-DS benchmark. The Parquet is used for performance validation as default format used by Impala while Avro is not used there.

Giannakouris et al. (2016) presents MuSQLE, a system for SQL-based analytics over multi-engine environments. The detailed experimental evaluation, integrating PostgreSQL, MemSQL and SparkSQL under MuSQLE, demonstrates system's ability to accurately decide on the most suitable execution engine. MuSQLE can provide speedups of up to 1 order of magnitude for TPCH queries, leveraging different engines for the execution of individual query parts. However, the authors do not analyze compact data formats in any kind of comparison.

Mehmood et al. (2016) analyzes the performance of three SQL-on-Hadoop systems (Hive, Impala and Tajo) by applying TPC-H benchmarks. The experimentation is done with two major and largely used file formats for columnar databases (ORC and Parquet). The results show that Impala outperforms Hive and Tajo by 5X to 10X when the workload dataset fits in its memory. The row-based storage format like Avro is not analyzed here.

Tummalapalli et al. (2016) presents the processing time of Impala, Hive, Pig and MySQL Cluster on a simple data model with simple queries while the data is growing. Evaluation results indicates that Impala achieves acceptable performance for some data analysis and processing tasks even compared with Hive and Pig and MySql cluster. The authors do not analyze Avro and Parquet storage file formats in any kind of comparison.

4. Results

This section presents the results of analysis of the relevant studies and answer to the research question. Although the field is in its earliest stages, there is clear evidence of the increasing interest focused on big data studies, Hadoop Technology, HDFS and compact, fast, binary data formats. The aim to give a synthesized overview on the trend of the research publications of Hadoop technology and answer the research question is reached by detailed analysis of the relevant studies.

The analysis of extracted data and initially retrieved studies show that Hadoop Technology and compact data formats have drawn interest of various researchers in the past seven years. As shown in Fig. 1, the number of publications of research papers in conference proceedings, journals and magazines has significantly increased from the year 2010 to 2016.

How about an answer to the research questions?

RQ.1: What are the differences in performance (query execution time) between compact data formats Avro and Parquet?

RQ.2: Which data format (Avro or Parquet) is more compact?

As shown in Table 2, only 2 papers ([PS14] and [PS16]) satisfy requirements partly - have a focus on row-based and column-based formats, although the row-based format is not exactly Avro, but 11 papers use TPC benchmark (as indirect quality criteria) for one of the formats, mostly Parquet. This might be because of row-based (Avro) and column-based (Parquet) data format specifics limiting comparison.

There is a good comparison of compactness and performance of two formats (Hortonworks' and Microsoft's proposed ORC (Optimized Row Columnar) file format, which is the next generation of RCFile, and Impala's promoted Parquet file format, which is a columnar format inspired by Dremel's storage format) in the first paper ([PS14]). The 1TB data compression results show that *snappy* compression provides slightly better query performance than *zlib* and *gzip*, for the TPCH workload, on ORC and Parquet file formats. For this reason, the authors only report results with *snappy* compression in both Hive and Impala. If the file format Avro to be replaced with ORC in RQ.2, the answer on this question could be that Impala Parquet format is 1,29x compact than Hive-MR ORC format. By applying snappy compression Impala Parquet-Snappy format is 1,07x compact than Hive-MR ORC-Snappy format. By using Hive-Tez system instead of Hive-MR the difference between Parquet and ORC format is even smaller. As for performance, this paper ([PS14]) provides the ratio between Hive-Tez's response time over Impala's response time for the snappy compressed ORC file and the snappy compressed Parquet format also. The results show that Impala Parquet-Snappy format is 4,2x faster than Hive-MR ORC-Snappy format in average by analyzing 22 TPC-H query execution time. However, this paper ([PS14]) does not provide answer on both research questions about the differences in performance and compactness by analyzing Parquet versus Avro.

The second paper ([PS16]) does not compare file formats from performance (query execution time) point of view but it contains a valuable answer to the second research question about compactness as there are compared different compression and format approaches like CSV(Row), Plain(Row), Snappy(Row), GBIN(Row), Snappy(Column), GBIN(Column). The results show that Snappy (Column) is 1,03x compact than Snappy (Row). As for GBIN compression approach, the GBIN (Column) is 1.16x compact than GBIN(Row), whereas GBIN (Column) versus Snappy (Column) is 1,38x compact.

However, Google Snappy codec gives a much better result as the decompression is faster than Deflate (GBIN). In the light of these observations, it can be concluded that Snappy (Column) format is better, although the more direct comparison including Snappy (Avro) could be advisable for new research.

In summary, the first paper [PS14] provides indirect answer on RQ.1 about performance: if Impala Parquet-Snappy format is 4,2x faster than Hive-MR ORC-Snappy format in average then there is a high probability that Parquet format is better than Avro. If Impala Parquet format is 1,29x compact than Hive-MR ORC format then the answer on RQ.2 about compactness could be that Parquet is more compact than Avro. However, these are only an assumptions, because these papers does not provide direct comparison with Avro and thereby – precise answer on question how much times Parquet is faster and compact than Avro.

Other papers do not provide clear and reliable answer to the research questions at all. Most of 27 papers are addressed only to one format (Avro or Parquet), but not to both.

Thereby, in order to completely and precisely answer the both research questions there is a need to perform an experiment similar to the first paper ([PS14]) by adding Avro and classifying queries at least in two categories (scan and aggregation queries) in order to highlight row-based and column-based binary file format advantages respectively.

5. Final conclusions

In this section, far from an exhaustive overview, some of the final conclusions are given to prevent recurrence.

There is a gap and need for additional experiments and studies in order to answer the research question about Parquet and Avro format more precisely. All 27 studies are not containing direct focus on comparing two binary data storage formats – Parquet and Avro because of both design specifics. Parquet as stated in the official documentation (WEB, g) is a column-oriented data storage format. Thus, it should provide better performance on column-oriented queries, e.g., when only a specific set of those is selected. As a counterpart, Avro format is designed for row-oriented data access, e.g., when all columns are the interest of processing. Most of all deeply analyzed studies have a focus on one of the formats, mostly Parquet. This might be because of row-based (Avro) and column-based (Parquet) data format specifics limiting comparison. However, it does not satisfy business demand for knowledge about both data format comparison. This is the gap for the future research.

In this review, 27 papers have been studied in order to evidence Hadoop Technology popularity and fast, compact, binary data format development necessity. A high diversity of Hadoop Technologies and used data formats has been noticed. It was a time consuming process to classify all studies, extract relevant information, assess validity and reliability, develop checklists and make conclusions at the end. There are several scientific articles that are devoted to performance analyze of SQL-on-Hadoop systems like Hive, Impala, Tajo, Spark, even by applying TPC-H benchmarks. However, these papers do not provide clear and reliable answer to the research question about differences in performance (query execution time) between compact data formats Avro and Parquet. Most of 27 papers are addressed only to one format (Avro or Parquet), but not to both. In summary, there is no evidence in the analyzed scientific articles that could strictly confirm an assumption that Avro can perform faster with scan queries but

Parquet - with aggregation queries. The executed queries have not been classified as scan or aggregation in the deeply analyzed scientific articles where the TPC benchmark is used. It would be useful to introduce such a classification in the future research with file formats and SQL-on-Hadoop systems.

References

Abadi, D., Boncz, A. P., Harizopoulos, S. (2009). Column-oriented database systems, *Proceedings of the VLDB Endowment 2.2, 2009*, 1664-1666.

Begoli, E., et al. (2016). Real-Time Discovery Services over Large, Heterogeneous and Complex Healthcare Datasets Using Schema-Less, Column-Oriented Methods. *Big Data Computing Service and Applications (BigDataService), IEEE Second International Conference on. IEEE*.

Biookaghazadeh, S., et al. (2015). Enabling scientific data storage and processing on big-data systems. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Cejka, S., Ralf, M., Alfred, E. (2015). Java embedded storage for time series and meta data in Smart Grids. *IEEE International Conference on Smart Grid Communications (SmartGridComm).*

Chandra, D.G., et al. (2012). A study on cloud database. *Computational Intelligence and Communication Networks (CICN), Fourth International Conference on. IEEE*.

Chen, Y., et al. (2014). A study of sql-on-hadoop systems. *Big Data Benchmarks, Performance Optimization, and Emerging Hardware. Springer International Publishing*, 154-166.

Choi, H., et al. (2015). An evaluation of alternative shared-nothing architecture for analytical processing systems. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Dong, D., and John, H. (2015). Record-aware compression for big textual data analysis acceleration. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Floratou, A., Umar, F. M., and Fatma, Ö. (2014). Sql-on-hadoop: Full circle back to shared-nothing database architectures. *Proceedings of the VLDB Endowment 7.12*, 1295-1306.

Giannakouris, V., et al. (2016). MuSQLE: Distributed SQL query execution over multiple engine environments. *Big Data IEEE International Conference on. IEEE*.

Grover, A., et al. (2015). SQL-like big data environments: Case study in clinical trial analytics. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Haynes, D., et al. (2015). High performance analysis of big spatial data. *Big Data (Big Data), IEEE International Conference on. IEEE*.

He, Yongqiang, et al. (2011). RCFile: A fast and space-efficient data placement structure in MapReduce-based warehouse systems. *Data Engineering (ICDE), IEEE 27th International Conference on. IEEE*.

Kilias, T., Alexander, L., and Periklis A. (2015). INDREX: In-database relation extraction. *Information Systems 53*, 124-144.

Kitchenham, B., Charters, S. (2007). Guidelines for Performing Systematic Literature Reviews in Software Engineering, *EBSE Technical Report, EBSE-2007-01*.

Li, K., et al. (2016). The research of performance optimization methods based on Impala cluster. *Communications and Information Technologies (ISCIT), 16th International Symposium on IEEE.*

Liu, Y., et al. (2016). Performance Evaluation and Optimization of Multi-dimensional Indexes in Hive. *IEEE Transactions on Services Computing*.

Liu, J., et al. (2016). Planning Your SQL-on-Hadoop Deployment Using a Low-Cost Simulation-Based Approach. *Computer Architecture and High Performance Computing (SBAC-PAD), 28th International Symposium on. IEEE*.

Luckow, A., et al. (2015). Automotive big data: Applications, workloads and infrastructures. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Mammo, M., and Srividya K. Bansal. (2015). Distributed SPARQL over Big RDF Data: A Comparative Analysis Using Presto and MapReduce. *Big Data (BigData Congress), IEEE International Congress on. IEEE*.

Mehmood, A., et al. (2016). Performance analysis of shared-nothing SQL-on-Hadoop frameworks based on columnar database systems. *Innovative Computing Technology (INTECH), Sixth International Conference on. IEEE*.

Palmer, N., et al. (2011). Towards collaborative editing of structured data on mobile devices. *Mobile Data Management (MDM), 12th IEEE International Conference on. Vol. 1. IEEE*.

Pirzadeh, P., Michael, J. Carey, and Till, W. (2015). BigFUN: A performance study of big data management system functionality. *Big Data (Big Data), IEEE International Conference on. IEEE*.

Polato, I., et al. (2014). A comprehensive view of Hadoop research – A systematic literature review. *Journal of Network and Computer Applications 46: 1-25*.

Sharma, M., et al. (2014). Investigating the inclinations of research and practices in hadoop: A systematic review. *Confluence The Next Generation Information Technology Summit (Confluence), 5th International Conference - IEEE.*

Shvachko, K., et al. (2010). The hadoop distributed file system. *Mass Storage Systems and Technologies (MSST), IEEE 26th Symposium on. IEEE*.

Son, J., et al. (2015). SSFile: A novel column-store for efficient data analysis in Hadoop-based distributed systems. *Information Sciences 316*, 68-86.

Stonebraker, M., et al. (2005). C-store: a column-oriented DBMS. *Proceedings of the 31st international conference on Very large data bases. VLDB Endowment*.

Tapiador, D., et al. (2014). A framework for building hypercubes using MapReduce. *Computer Physics Communications 185.5*, 1429-1438.

Tummalapalli, S., et al. (2016). Managing Mysql Cluster Data Using Cloudera Impala. *Procedia Computer Science 85*, 463-474.

Tsai, C.-P., and Hung-Chang, H. (2014a). Streaming in NoSQL. *Parallel and Distributed Systems (ICPADS), 20th IEEE International Conference on. IEEE*.

Tsai, C.-P., et al. (2014b). Publish/Subscribe in NoSQL. *Computational Science and Engineering (CSE), IEEE 17th International Conference on. IEEE*.

Wonjin, L., et al. (2014). A big data management system for energy consumption prediction models. *Digital Information Management (ICDIM), Ninth International Conference on. IEEE*.

Wullink, M., et al. (2016). ENTRADA: A high-performance network traffic data streaming warehouse. *Network Operations and Management Symposium (NOMS), IEEE/IFIP*.

Wullink, M., et al. (2016). ENTRADA: enabling DNS big data applications. *Electronic Crime Research (eCrime), APWG Symposium on. IEEE*.

Xu, W., et al. (2016). Supporting large scale connected vehicle data analysis using HIVE. *Big Data International Conference on. IEEE*.

Yan, W., and Yuan, X. (2015). Coordinated Resource Management for Large Scale Interactive Data Query Systems. *Cluster, Cloud and Grid Computing (CCGrid), 15th IEEE/ACM International Symposium on. IEEE*.

Zhang, S., et al. (2014). A strategy to deal with mass small files in HDFS. *Intelligent Human-Machine Systems and Cybernetics (IHMSC), Sixth International Conference on. Vol. 1. IEEE*.

Zhang, Z., et al. (2013). Alovera: A Fast Stream Processing System for Large-Scale Data. *ChinaGrid Annual Conference (ChinaGrid), 8th. IEEE*.

Zhou, N., Xiao, Z., Shan, W. (2015). WACO: Workload Aware Column Order for Scan Operator in Wide Table. *IEEE Conference on Collaboration and Internet Computing (CIC)*.

WEB (a). *Gartner Says Smartphone Sales Surpassed One Billion Units in 2014.* `http://www.gartner.com/newsroom/id/2996817` [Accessed: 19-Apr-2017].

WEB (b). *CSV files.* [Accessed: 19-Apr-2017]. `https://tools.ietf.org/html/rfc4180`

WEB (c). *JSON specification.* [Accessed: 19-Apr-2017].
`https://tools.ietf.org/html/rfc7159`

WEB (d). *Apache Avro specification*. [Accessed: 19-Apr-2017].
`http://avro.apache.org/docs/current/spec.html`

WEB (e). *SequenceFile*. [Accessed: 19-Apr-2017].
`https://wiki.apache.org/hadoop/SequenceFile`

WEB (f). *LanguageManual ORC*. [Accessed: 19-Apr-2017].
`https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ORC`

WEB (g). *Parquet official documentation*. [Accessed: 19-Apr-2017].
`https://parquet.apache.org/documentation/latest/`

WEB (h). *Apache Thrift*.
`http://thrift.apache.org` [Accessed: 19-Apr-2017].

WEB (i). *Protocol Buffers*.
`https://github.com/google/protobuf` [Accessed: 19-Apr-2017].

WEB (i). *Protocol Buffers*.
`https://github.com/google/protobuf` [Accessed: 19-Apr-2017].

WEB (j). *Summary (Excel) of all relevant studies used for this review, available at Dropbox*.
`https://www.dropbox.com/sh/6o4q8kadqfogusm/AACn43bb2QFTLbBfsFThMFd1a?dl=0` [Accessed: 19-Apr-2017].

Received October 5, 2016, revised May 9, 2017, accepted June 28, 2017