Storage Size Estimation for Schemaless Big Data Applications: A JSON-based Overview

Devang Swami and Bibhudatta Sahoo

National Institute of Technology Rourkela, Rourkela, Odisha, India 769008

swamx.mi@gmail.com, bdsahu@nitrkl.ac.in

Abstract. Numerous technologies have been proposed for storing big data on the Cloud platform. However, choice of these technologies is always application specific. Determining a strong model is a perplexing task, which makes it necessary for the architects and designers to review the requirements and choose a solution. This paper presents 14 data models available in the market. Above all, there are more than 45 database solutions available in the market, which can be categorized into one of the data models each of which is applicable to its own set of use cases (However, there are few products, which could not be categorized into any of these 14 data models). Contributors have figured out that while storing schema-less information, the size of data stored in the database is higher than the original size. Metadata information and physical schema are the two responsible factors for such a high amount of storage requirement. Mathematical models and experimental evaluations conducted show that MongoDB requires storage space many times more than the original size of data. A storage space estimation equation for JSON based solutions has been suggested, which can compare the storage requirement size using space required by CSV as a base. This may be used to decide an approximate amount of storage space required by the application, before buying a storage space on the Cloud environment.

Keywords: Big Data, Schemaless Data, Cloud, Storage

1 Introduction

Big data is a buzz word which usually represents enormous data which cannot be processed by a single system due to its bulky size, large variety, and high-speed of generation. Advancement in IT technologies is primary reason for generation of big data. At any given time period only a fraction of big data is useful for most application domains. Hence, many experts and researchers have recommended use of cloud for big data to optimally manage and reduce the overall cost of operating such systems. Cloud computing is a model which carters three services of its users, namely dynamisms, abstraction and resource sharing. Generally a storage structure is defined in the physical data model. A physical data model is a representation of data on secondary storage device and it also includes other data structures like indexes and others. It also defines the constrains of the database systems, like the data types available to store a data, number of secondary indexes allowed, and others. As shown in Figure 1, a physical data model comprises of Message Format, File structure, Physical schema, and other entities. There are two ways in which data in a table may be stored either in row-order or column-order considering options provided by physical schema. [32].

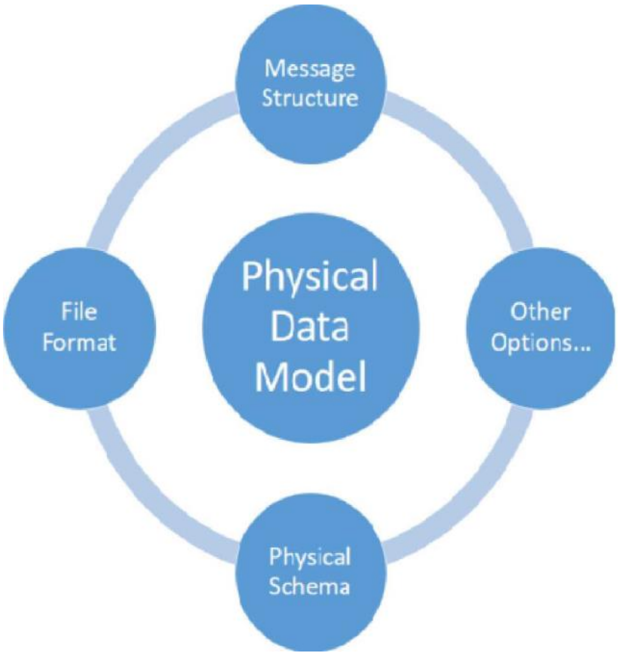


Fig. 1. Physical Data Model [32]

The physical schema defines the storage space required to organize the data on secondary storage devices. Also it defines the number of indexes and limit the data structures which can be used to create index. A mathematical model can be used to estimate the size of storage space required to store data. We found that storing 1.5GB blogging data with three secondary indexes (including a Text search index) was stored by MongoDB in 2.63GB which was 1.7 times the original size. It is very critical to know storage space requirement because it will impact the decision process of buying a storage space. Also, most cloud service providers limit access to storage by limiting number of IOPS performed by an application. Hence, it is in the best interest of application developers & designers to have a detailed knowledge of physical schema of a data model or database before deciding to host the data on the Cloud. In section 2, relevant works on physical schema, data models and past attemps to estimate storage size for different physical schema are discusses. Successively, a mathematical model of storage space requirement for JSON-based databases is proposed. In section 4, a simulation of the derived model would be discussed and the results would be experimental verified. Finally, contributors would conclude the work.

2 Literature Review

A true benchmark in the field of large-scale database management systems was achieved by information retrieval model by E Codd [8]. Only few works discuss and suggests new models for evaluating the pros and cons of big data systems. In Table 1 a list important trends relating to evaluation of data models is revealed for the period starting from early 1970's to present.

Table 1. Findings and Open Problems

Research	Findings and Open Problems	Year
[8]	The provisions for data description tables in recently developed information systems represents a major advantage towards the goal of data independence	1970
[18]	New metadata information types, such as QoS of service for storage, and algorithms to exploit them, may be needed to meet emerging trends.	1996
[30]	Schema-last is a probably a niche market.	2005
[26]	The high increase of disk usage compared to raw data is due to additional schema as well as version of information that is stored each key-value pair.	2012
[14]	Integration of structured and unstructured data and information from distributed, heterogonous virtual	2013
[7]	Data storage and search schemas (or Indexes) are responsible for high latency & overhead	2014
[19]	Applications often drive the design of the underlying	2014

Table 2. Data Models for Big Data Applications

List of Data Models for Big Data Solutions				
Content Stores	Graph	Native	RDBMS	Time Series
Document	Key-Value Stores	Navigatio	RDF Stores	Wide-Column
Event Stores	Multi-value Stores	Object Oriented	Search Engines	

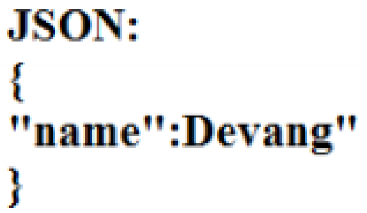
Table 3. Database Solutions for Big Data Applications

List of Databases for Big Data Solutions				
Adabas	Db4o	Hypertable	MySQL	Solr
Algebraix	DynamoDB	IDMS	Neo4j	Sphinx
Amaxon Search	Elasticsearch a	IMS	NEventStore	Titan
Azure Docu-mentDB	Event Store	Jack Rabbit	ObjectStore	TC-TT
BaseX	Flare	Jena	Oracle BigData SQL	UniData, uni-Verse
Cache	Google Cloud Bigtable	MarkLogic	Oracle SQL	Versant Object Database
Cassandra	Google Cloud Datastore	Microsoft Azure	Redis	Voldemort
Couchbase	Google Search	Microsoft SQL	Scalaris	VoltDB
CouchDB	GraphDB	ModeShape	Sedna	...
D3	HBase	MongoDB	Sesame (or	

Three of every four companies have found the necessity of using or shifting to Big Data solutions in the next two years [17]. These industries would be facing a great challenge of researching and choosing a big data technology as they have a large variety of solutions to choose from. With 10+ Data Models (listed in Table 2) and 45+ DBMS systems (listed in Table 3) are available for various applications. However, a single solution does not fit all purpose of the industry, hence it becomes eventually necessary to combine one or more solutions into a single conglomerated system that solves all the business problems. For instance, Oracle Big Data System, provides both NoSQL and/or Hadoop cluster options to its customer with SQL. A major problem for choosing such technologies is that very few models such as Relational, Object-oriented, and Object-Relational have been built on strong mathematical model. Now, modelling of storage is a non-trivial challenge and in many cases demands evaluation of designs. If resource requirement cannot be justified, it would become increasingly difficult to monitor the growth of the system data and could adversely affect performance considering that scalability issue is not tackled in the right way.

Many prominent tools and technologies have been proposed in past to estimate the size of storage space required. MySQL also provides a perl script to estimate the size of storage space required for storing a database on the cluster based storage engine named NDB based on size of storage space used by InnoDB storage engine to store the data [3]. InnoDB storage engine uses Barracuda file organization. Neo4j, a graph based database also provides a calculator to estimate storage space, main memory and processing power required at a node to store & process the data [1]. Neo4j calculator takes number of nodes, size of a single node, number of edges and storage size of each edge as input to approximate the storage space required [1].

3 Storage Estimation Model for JSON-based databases

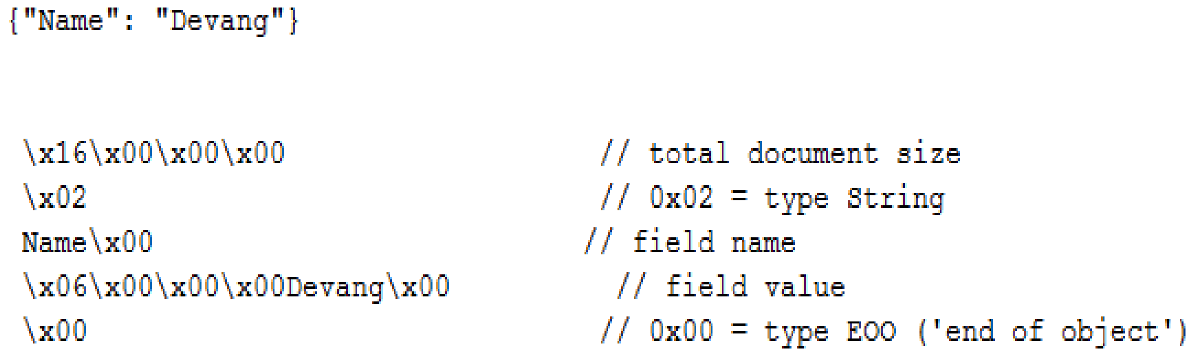


```
JSON:
{
"name":Devang"
}
```

Fig. 2. A simple JSON document

JSON has been one of the most influential format in the movement of migration from RDBMS to NoSQL [25]. JSON has found its place among many application domains with semi-structured and unstructured data [16] [10] [4]

[6]. Many databases and solutions have extended JSON to suit their needs like BSON. BSON is a communication and storage protocol used by MongoDB, which is derived from JSON.



```
{"Name": "Devang"}


 \x16\x00\x00\x00                    // total document size
 \x02                                // 0x02 = type String
 Name\x00                           // field name
 \x06\x00\x00\x00Devang\x00           // field value
 \x00                                // 0x00 = type EOO ('end of object')
```

Fig. 3. Physical Schema of MongoDB (BSON)

Figure 2 depicts a json document with a single field, "name" and its value "Devang". Figure 3 describes the storage schema of BSON which is a communication and storage protocol used by MongoDB. BSON is a storage structure which is derived from JSON. From the figures, it is also evident that BSON will consume much large storage size than JSON, owing to extra information it keeps for recording the data. Although, this extra information does help in increasing throughput by informing about type and size of data, helping I/O processor make smart decisions (if relevant technologies are available and programmed to use). Above all, this extra information also helps the I/O processor decide how much bits to skip so as to find next document making read task faster. Nevertheless, one cannot ignore the increment in amount of storage space they require. We propose to derive a model that can help us to estimate the factor by which storage size of JSON increases in comparison with storage size required by CSV. Although, the model is derived for JSON, it is applicable across all databases and solutions that use JSON or its derivatives (e.g. BSON, MessagePack [1], etc [5]).

The storage estimation model is explained by considering the physical schema of CSV and JSON storage schema's. For the purpose of modelling storage space requirement we proposed comparing storage with flat file databases like CSV as the raw storage size because of all available formats CSV has been more commonly used by many literatures as a physical schema of choice due to its simplicity and high level of human readability that it offers [28] [12] [29] [11].

Consider a source S, which emits data at regular intervals. This data may be

[1] MessagePack is a JSON-like but comparatively smaller in size [2].

stored in table T with following properties:

- A Table T consists of N columns and R rows.
- Each column of the table has on average b(k) bytes of data for kth column.
- Total number of bytes for each row of the table on average is B = $\sum_{k=0}^{n} C(k)$
- Each column header is of size c(k) bytes of data for kth column.

For simplicity we assume that the source releases data at regular intervals. It can be considered that source follows some distribution for generating data. Thus it can be said that the number of rows for the given table T can be approximated using the prior attained distribution. Also, generating data is a characteristic of the Source. Hence, the maximum number of bytes required to store data in a file can be estimated. Thus, we can get the value of bi from the source itself. By getting N, which is the number of data items required to be stored in the table, by using distribution, which predicts when the given source will produce the data. Thus by knowing, bi, R and N, we can compute B. Finally, the size of column header ci can be measured since the developer or DBA decides the column name.

CSV organizes the data in row-order format so that columns are mentioned in the first line and all successive lines store the data. Now amortized size2 of column stored in CSV file would be $\sum_{k=0}^{n} C(k)$ and since B bytes is the average size of a row, data would take B x R. Hence, it can be concluded that for CSV store the size of data would be CSV Size = (B x R) + $\sum_{k=0}^{n} C(k)$ bytes. In JSON-based stores, each row is in the format { column1 name: value, column2 name: value, ...} as shown in Figure 3. Hence, the size of each row in such a physical schema3 would be (B + $\sum_{k=0}^{n} C(k)$) bytes. For R number of rows in the table, the size of database would be MC_Size = R x (B + $\sum_{k=0}^{n} C(k)$) bytes. Thus, the ratio of storage size for JSON-based store to CSV would be ((B + $\sum_{k=0}^{n} C(k)$)) / ((B x R) + $\sum_{k=0}^{n} C(k)$).

[2] We use the term amortize because we donot consider the size of putting other characters like comma, carriage return, space for null values and other special characters.

[3] We are not including comma, other special characters and null values since we only are after a rough estimate.

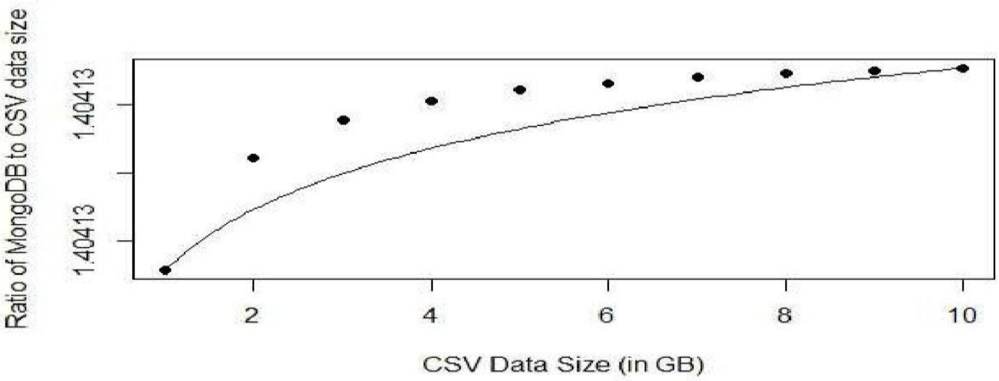


Fig. 4. Simulation: Ratio of MongoDB to CSV Data Size

4 Experiment

Experimental evaluation has been conducted with a simulation for total column field storage size of 136 byes and Row size of 474 byes for varying number of Row for NYC Taxi cab database [9] that is used for traffic patterns analysis of Taxi cabs to reduce pollution was utilized. To obtain size of column on an average we created a dummy document with all the values NULL or not set. We used this as a reference since we are only offer amortized comparison of the storage size requirement. Figure 4 is a CDF and thus its corresponding PDF is "Exponential". Which suggests that exponential increase in mongodb storage size could be noticed when the size of raw data increases linearly. And the results obtained from simulation are produced in Figure 4.

Table 4. Ratio of MongoDB to CSV Data Size

Year-Month of Data	No. of Recor	CSV Size (Cumula	Mongo DB	Ratio (MongoDB size / CSV
2016-	10	1.6	2.3	1.412
2016-	11	4.86	7.0	1.45
2016-	12	9.9	14.	1.49
2016-	11	16.68	25.	1.44
2016-	11	24.83	38.	1.57
2016-	11	34.6	50.	1.45

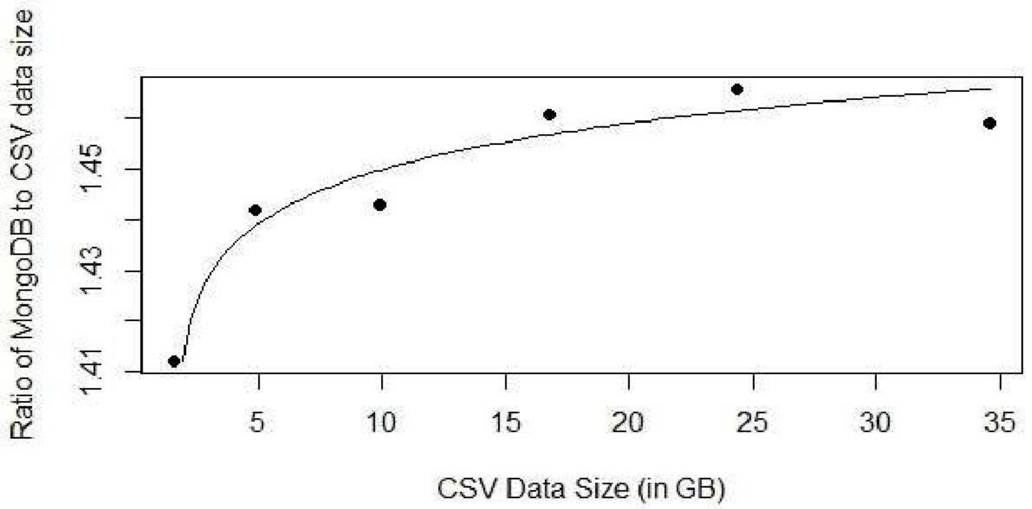


Fig. 5. Experiment: Ratio of MongoDB to CSV Data Size

Results of the simulation were verified by inserting the data of NYC Yellow Taxi dataset in the big data solution, MongoDB (a JSON-based store) using WiredTiger storage engine. MongoDB was used for experiment as it is an open source solution, it uses JSON-like physical schema named BSON and is an extremely popular NoSQL data store [13]. On storing the data in MongoDB the size of stored data increased by 1.4 times the size of storage space used by CSV as shown in Table 4. The results of the experiments are shown in Figure 5 which confirms the trend suggested by the model. Thus, using the model and simple math's we can devise a storage factor for estimating the size of storage space required by JSON and its derivatives.

Table 5. MongoDB Throughput (Wall Clock Time)

Fi le	Import Start Time	Import End Time
20	10:33:49	10:46:59
20	10:52:35	11:08:28
20	11:12:48	11:21:39
20	16:13:35	16:25:49
20	16:27:45	16:39:03
20	16:45:50	16:57:00

Above all, from the experiment it is discovered that MongoDB takes on an average 10-13 minutes to import a csv file of size 1.6 GB on a standard noncommercial grade hard drive with 5400 RPM disk speed on a machine with 8GB RAM and Intel core-i5 ***6th*** generation processor.

5 Conclusion

This paper has listed 14 data models and 45+ databases that provides a glimpse of wide range of solutions available in the market for different big data applications. Researchers in the given work had also proposed a model that proved the of storage size determination by using physical schema for JSON-based stores. It had also been proved that the increment in disk utilization is due to the requirement of storing schema and version information into the table so as to allow storing semi-structured or unstructured data. This increased disk usage with respect to raw size shows exponential increment as the size of data increases. In near future, a comprehensive research for uniting structured, semi-structured & unstructured data from different data inception points needs to be carried out. This research should be from the perspective of storage and QoS achievement using minimum resources so that it assists decision makers to make an optimal choice for their application. Finally, the WiredTiger Storage Engine of MongoDB takes 1.4 times more space than CSV file for NYC Taxi Cab Dataset including a primary index. Also, the simulation of proposed model varied from the experimental values by 5% to 11%.

References

1) Hardware sizing calculator. https://neo4j.com/hardware-sizing/, accessed: 2016-09-30
2) Messagepack. http://msgpack.org/index.html, accessed: 2016-09-26
3) Ndbcluster size requirement estimator. https://dev.mysql.com/doc/refman/5.7/en/mysql-cluster-programs-ndb-size-pl.html, accessed: 2016-09-30
4) Aho, A.V., Sethi, R., Ullman, J.D.: Compilers, Principles, Techniques. Addison wesley Boston (1986)
5) del Alba, L.: Data serialization comparison: Json, yaml, bson, messagepack. https://www.sitepoint.com/data-serialization-comparison-json-yaml-bson-messagepack/, accessed: 2016-09-26
6) Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a

collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247-1250. ACM (2008)

7) Chen, C.P., Zhang, C.Y.: Data-intensive applications, challenges, techniques and technologies: A survey on big data. Information Sciences 275, 314-347 (2014)

8) Codd, E.F.: A relational model of data for large shared data banks. Communications of the ACM 13(6), 377-387 (1970)

9) Commission, N.T..L.: Tlc yellow taxi trip record data. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml, accessed: 2016-09-30

10) Consortium, W.W.W., et al.: Json-ld 1.0: a json-based serialization for linked data (2014)

11) Cook, K.B., Kazan, H., Zuberi, K., Morris, Q., Hughes, T.R.: Rbpdb: a database of rna-binding specificities. Nucleic acids research 39(suppl 1), D301-D308 (2011)

12) Cranford, K.: How to excel with sas. In: Proceedings of the 28 th Annual SCSUG Conference, Austin, Texas, September (2007)

13) DB-engines.com: Dbms rankings 2017 (2016)

14) Demirkan, H., Delen, D.: Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud. Decision Support Systems 55(1), 412-421 (2013)

15) ENDPOINT.com: Benchmarking top nosql databases

16) Finn, R.D., Mistry, J., Tate, J., Coggill, P., Heger, A., Pollington, J.E., Gavin, O.L., Gunasekaran, P., Ceric, G., Forslund, K., et al.: The pfam protein families database. Nucleic acids research p. gkp985 (2009)

17) Gartner.com: Gartner report

18) Gibson, G.A., Vitter, J.S., Wilkes, J.: Strategic directions in storage i/o issues in large-scale computing. ACM Computing Surveys (CSUR) 28(4), 779-793 (1996)

19) Kambatla, K., Kollias, G., Kumar, V., Grama, A.: Trends in big data analytics. Journal of Parallel and Distributed Computing 74(7), 2561-2573 (2014)

20) Kant, K.: Data center evolution: A tutorial on state of the art, issues, and challenges. Computer Networks 53(17), 2939-2965 (2009)

21) Katal, A., Wazid, M., Goudar, R.: Big data: issues, challenges, tools and good practices. In: Contemporary Computing (IC3), 2013 Sixth International Conference on. pp. 404-409. IEEE (2013)

22) Khalifa, S., Elshater, Y., Sundaravarathan, K., Bhat, A., Martin, P., Imam, F., Rope,

D., Mcroberts, M., Statchuk, C.: The six pillars for building big data analytics ecosystems. ACM Computing Surveys (CSUR) 49(2), 33 (2016)

23) Montoya, J.A., Velez-Gallego, M.C., Villegas, J.G.: Capacitated facility location problem with general operating and building costs (2012)
24) NetApp, I.: Netapp all flash fas storage arrays Padhy, R.P., Patra, M.R., Satapathy, S.C.: Rdbms to nosql: reviewing some next-
25) generation non-relational databases. International Journal of Advanced Engineering Science and Technologies 11(1), 15-30 (2011)
26) Rabl, T., Gomez-Villamor, S., Sadoghi, M., Muntes-Mulero, V., Jacobsen, H.A., Mankovskii, S.: Solving big data challenges for enterprise application performance management. Proceedings of the VLDB Endowment 5(12), 1724-1735 (2012)
27) Sanders, P.: Algorithm engineering for big data. In: GI-Jahrestagung. p. 57 (2014)
28) Shafranovich, Y.: Common format and mime type for comma-separated values (csv) files (2005)
29) Sharma, T.C., Jain, M.: Weka approach for comparative study of classification algorithm. International Journal of Advanced Research in Computer and Communication Engineering 2(4), 1925-1931 (2013)
30) Stonebraker, M., Hellerstein, J.: What goes around comes around. Readings in Database Systems 4 (2005)
31) Strohbach, M., Daubert, J., Ravkin, H., Lischka, M.: Big data storage. In: New Horizons for a Data-Driven Economy, pp. 119-141. Springer (2016)
32) Whitehouse, O.: Fea consolidated reference model document (2005)