Performance Modeling of Big Data

Krushnaraj Kamtekar, kamtekar327 (at) wustl.edu (A paper written under the guidance of Prof. Raj Jain)

Download

Abstract

The term Big Data is used to describe data sets which are complex and so large that the conventional relational database applications are inadequate. This data contains valuable patterns and knowledge which were difficult to extract previously. With the cloud architecture and open source software, extracting value from big data has become easy. Even small companies and startups can afford big data processing by renting server time in the cloud. Apache Hadoop, an open source data processing platform is a leader in the big data revolution. Some other tools include MapR. Recent advances in hardware, memory and networking have facilitated data processing on the fly. In this paper, we present a survey of the performance modeling of various big data tools and techniques used for analysis, storing, searching, sharing and transfer of data. This includes identifying the bottleneck devices or resources and finding out methods and techniques to improve the performance of the same.

Keywords

Big Data, RDBMS, Hadoop, MapReduce, Clusters, Analysis, Bottleneck, Performance, Concurrency

Table of Contents

- 1. Introduction
- 2. Traditional Database Systems and Big Data Tools
  - 2.1 RDBMS
  - 2.2 Overview of MapReduce
- 3. General Approach to Performance Modeling
  - 3.1 Feature Selection
  - 3.2 Identifying Bottleneck
- 4. Performance Modeling Approaches for Big Data Architecture
  - 4.1 Performance Modeling of MapReduce Task
  - 4.2 Performance Modeling of Data Transfers
  - 4.3 Performance Modeling of In-Memory Datasets
  - 4.4 Performance Modeling for Optimizing Data in the Cloud
  - 4.5 Performance Modeling for Interpretation of Big Data Cloud
- 5. Summmary
- 6. References

1. Introduction

Big data is an ever evolving term that describes large volumes of both structured and unstructured data that can be mined for knowledge discovery. It has revolutionized the way business is being done in the present age. It is playing an important role in decision making in all the sectors of life. Big Data has hidden patterns and knowledge which, if extracted, can help to identify trends in businesses, reduce production and maintenance costs, fight crime, predict the expectations of the consumers and help us live better.

[SAS] Big Data has been defined with the four V's
i) Volume : The volume of data determines the importance of data.
ii) Variety : The class or the family to which Big Data is closely associated to.
iii) Velocity : The speed of data generation and data processing.
iv) Value : The knowledge associated with it.

Knowledge extracted from such data is the key to success in today's competitive environment. However, extraction and interpretation of knowledge from such data is a difficult task. Big data analysis includes several tasks like data collection, data cleaning, analysis, pattern recognition and knowledge discovery. Apart from these basic tasks, there are other system specific complex issues like speed, storage, synchronization, concurrency and optimization.

The big data architecture is made up of many systems. Each such system has some processing task associated with it. Since data is generated at a rapid pace the processing should be done on the fly. All business practices depend and revolve around big data. Future business models and policies will be built on the knowledge extracted from big data. In this paper we try to find the problems associated with such systems and try to model a better solution. Our approach would be to identify bottlenecks in the system. Bottlenecks are those spots where the system is lacking efficiency or performance.

In this paper, Section 2 discusses about the traditional database systems, and the basic MapReduce task in distributed computing. Section 3 discusses a general approach to the performance modeling. Section 4 discusses the bottlenecks in the big data architecture and suggests ways in improving the bottleneck devices or resources. Section 5 gives a summary of the paper. Section 6 and 7 lists out the references used and the acronyms used.The model that is built is experimentally tested and compared against the present system under observed conditions and was found to be superior to the present system. We later summarize the importance of performance modeling in system design tasks.

2.Traditional Database System and Big Data Tools

Apache Hadoop is an open source software used for processing large datasets built for distributed applications. In this section we discuss traditional Database Systems, their limitations and the MapReduce task done by big data tools like Hadoop for distributed applications.

2.1 RDBMS

RDBMS stands for Relational Database Management Systems. It uses SQL as a query language to add, update, delete and perform sophisticated query operations on the underlying Database. The most popular ones are ORACLE, MS-SQL Server, MySql and Microsoft Access. A table is used to store data in a database. A column in the table will be set as the primary key which uniquely identifies the row. A foreign key will be a primary key in one table and will be a column in the other table. Thus, a foreign key acts as a link between two tables. RDBMS helped to overcome the limitations of the older file systems which were previously used as databases. But due to the volume of the data that is generated today, this traditional form of database system is falling short.

Some of the limitations of RDBMS are

1) Difficult to run RDBMS technology in distributed environment.
2) Making changes to the schema is a tedious task.
3) Difficult to extract knowledge from the database.

2.2 Overview of MapReduce

MapReduce is a programming model for processing and generating large data sets with a parallel algorithm on a cluster. The MapReduce results are obtained in two main steps : a) map step and b)reduce step

a) Map Step : Here the job is mapped into several parallel tasks and allocated to nodes that work on a subset of data and store the output in a temporary database.

b) Reduce Step : Here each node processes each group of data and key in parallel and produces a single output.

The Reduce Step turns large volumes of data into a compressed form of itself. All map() functions are performed in parallel as they are independent of each other. If all the output functions from the map() function are given to the same reducer then the set of reducer functions can perform the reduction at the same time. Figure 1. shows the basic working of the MapReduce task.

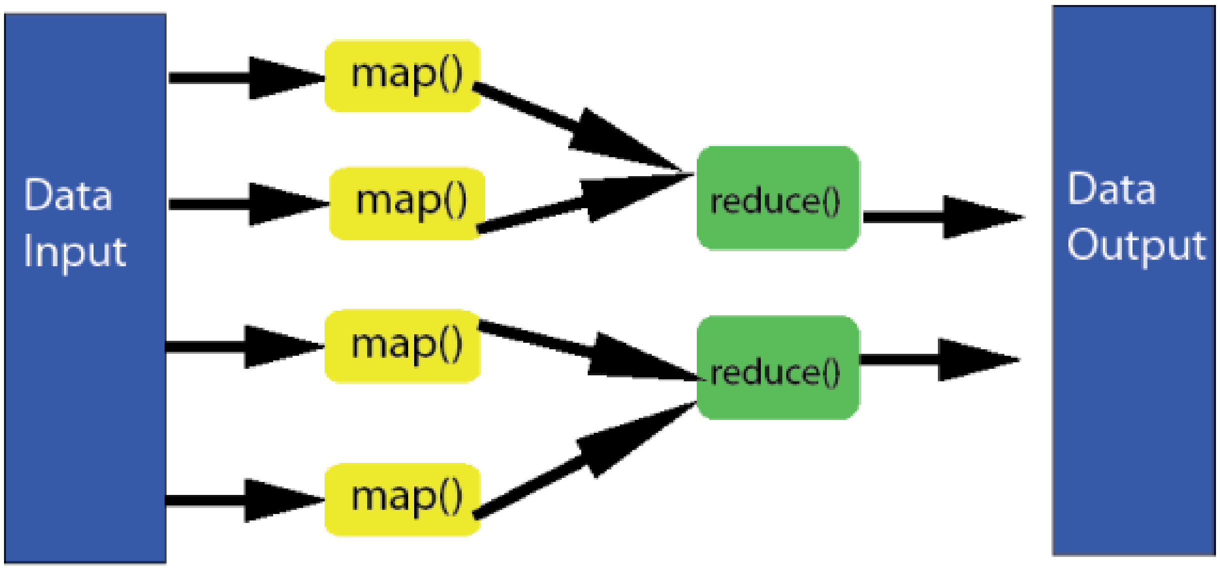

Figure 1. Graphical view of MapReduce Task

3. General Approach to Performance Modeling

In this section we discuss some general techniques which can be used in performance modeling. The performance of big data applications is significantly affected if the underlying architecture has defects. Performance modeling proposes a model which quantifies the behavior of the application under normal circumstances. This model can be used to strike a comparison with the compared system which helps in identifying problems and limitations of the compared system. It helps in establishing a comparison criterion which can be used in system analysis. [Jain91] For example the performance criteria when a user request a service in the cloud computing environment can be set as follows

i) perform the service correctly.
ii) perform the service incorrectly.
iii) fails to perform the service

User Request

Figure 2 . Graphical representation of a user request

3.1 Feature Selection

Once the comparison between the performance model and proposed/current system is done, a feature needs to be selected which best explains the differences or similarities.[Eduardo14] Feature selection should not be biased. After feature selection is done, we can perform some tests like observing the mean of the features. If the difference in means is large, then that feature can be used to distinguish between the two systems. If the difference in means is negligible, then we cannot get enough insight on the proposed/current system.

3.2 Identifying Bottleneck

In a cloud computing environment, there will always be some resources whose utilization would be a lot higher compared to the utilization of the other resources. This leads to performance and efficiency issues.[Jain91] Due to their higher utilization, these resources have the highest demand and are called bottleneck devices/resources. The most important goal of performance modeling is identifying the bottleneck. Improving this device would provide the highest payoff in terms of performance. Improving other resources would have little effect on system performance.[BandwidthPlace] For Example, if there are multiple devices connected to the internet at the same time it will create a slow internet speed. The total amount of bandwidth entering a home is always 100%. If one device is busy with a movie download and uses up 50% of the bandwidth, the other devices are left with only 50% of the bandwidth to work with. This is the bottleneck effect. If U is the utilization of the device/resource and D is the demand for the same then,

$U \propto D$

The device or resource with the highest demand will have the highest utilization.

4. Performance Modeling Approaches for Big Data Architecture

In this section we discuss the various components and techniques used in the big data architecture and discuss methods to improve their performance. Processing of big data should be done on the fly. The underlying architecture will have some components which are bottlenecks in the architecture. The idea is to find the bottleneck device or resource or task which requires the highest processing and execution time and propose relevant solutions to improve its performance.

4.1 Performance Modeling of MapReduce Tasks

Apache Hadoop has around 200 configuration parameters. Hadoop users face the uphill task of running applications in an economical way without compromising on performance. The default configuration parameters are not optimal for all types of applications. The most important configuration parameters in Hadoop are a) Number of mapping jobs, b) Number of mapping slots for the mapping jobs. Processing one job at a time would increase the response time for other jobs. Also there are chances that some jobs may be starved. [Zhang13] proposes a mathematical model for concurrency in the MapReduce task and uses the following two characteristics of the MapReduce task

a) Setup Time for the Map
b) Early Shuffle.

The idea is to bind the maximum number of concurrent map tasks to the total number of map slots. Thus every map slot will have a small portion of the task to perform and the overall execution time of the task would be reduced significantly. The concurrency factor can be calculated as follows

Concurrency = Map Tasks / Map Slots
The impact of the concurrent model for different tasks can be observed in Figure 3. This chart compares the execution time of the map task under default settings of Hadoop and the tuned settings of Hadoop for concurrency.

Hadoop setting vs Tuned Hadoop Settings

Figure 3. Comparison between default Hadoop setting vs Tuned Hadoop settings

4.2 Performance Modeling for Data Transfers

The two most common protocols used in data transfer are User Datagram Protocol (UDP) and Transmissions Control Protocol (TCP). UDP is a connectionless protocol and has less features for error detection and control. TCP is a connection oriented protocol which provides better error detection and control. However for big data transfers which are extremely time sensitive, these protocols are inadequate. In cases where data processing is done on the fly in a distributed setting, we need data to be transferred quickly and in a time sensitive manner. [Yu14] develops a GridFTP protocol model which not only provides faster data transfers but also takes care of the fairness factor so that other data transfers activities are not pre-empted or starved. GridFTP is a modified protocol implementation and has the Application and Transport Layer within its scope.
Experiments were was conducted to test the speed of data transfer using GridFTP protocol and having a background TCP flow in a congested network. The transfer rate for GridFTP was set initially to 10Gb/s and that of the background TCP flow was varied from 1Gb/s to 10Gb/s. It was found that GridFTP reaches an equilibrium state while competing with the background TCP flow and at the same time making sure that its utilization is the maximum.

Protocol Throughput

Figure 4. Protocol Throughput compared to background traffic

4.3 Performance Modeling of In-Memory Datasets

Having a working copy of in-memory datasets has its advantages. It helps to manage the datasets faster and efficiently. But the benefits only hold when the size of the in-memory datasets does not exceed that of the main memory and the buffer pool. If the size of the datasets is more than that of the memory and buffer pool, some part of the datasets get stored on the disk and then virtual memory policies like paging and segmentation are used to support the dataset. [Graefe14] develops a simulation of a buffer using swizzling pointers which removes the overhead of the traditional buffer pool. Swizzling eliminated the need to have a mapping between the object identifier and the address of the in-memory objects. It provides a direct reference to the image of the object in memory. It was found that buffer pool with swizzling pointers outperform the traditional buffer pool. Also the performance of this buffer pool was comparable to the in- memory database when the size of datasets was smaller than the size of main memory. Figure 5. shows that the performance of swizzling pointers is comparable to that of the main memory when the dataset is in memory.

Main memory vs Swizzling Pointer

Figure 5. Comparison between Main Memory Performance vs Swizzling Pointer Performance for an in memory dataset.
Most B-tree operations have root to leaf traversal, so the pointers of root-leaf linkage are good points to swizzle. When this happens the in memory page-image of the parent page gets updated, swapping the ID of the child page with the pointer to the buffer pool frame holding the in-memory image of the child image. For simplicity the pointers are swizzled one at a time.

4.4 Performance Modeling for Optimizing Data in the Cloud

Big data is growing at a rapid pace. The traditional database queries no longer provides enough insight about the underlying data. The rise of cloud computing has spawned the field of data analysis. However developing cloud programs is a difficult task. Apache Hadoop requires user to think and code from scratch. Secondly, deploying such programs in the clouds is a difficult task as well. Users have to know the hardware specifications, execution parameters and configuration details suited for their code. [Huang13] proposes a solution called *Cumulon* which simplifies the development and deployment of such programs using matrix-based statistical analysis.
During the development stage Cumulon will think and code using linear algebra. For the deployment of the application Cumulon will present the user a list of possible plans along with the rough estimate of completion time and the total cost incurred.
Cumulon first creates logical equivalent of the program using logical operators that relate to standard matrix operations like Add, Multiple, Inverse, Transpose. The logical representation is then reformatted to obtain improved versions of the representations. For each such representation Cumulon creates a physical page template which is a strategy for parallel execution of the program. The deployment plan is shown in a tuple (L, O, P, Q) where

L is a physical plan template
O is the parameters setting for all physical operators in L
P is provisioning settings for hardware
Q is settings for configuration.

Matrices are created and fetched using tiles which are submatrices of defined size. The Cumulon model makes sure that a tile can have many simultaneous reader or a single writer. The task in Cumulon will always read input matrices and write output matrices. These matrices are disjoint. In case of dependent tasks the execution is done serially. The hardware gets configured to slots and a task scheduler will assign the task to each such slot

4.5 Performance Modeling for Interpretation of Big Data Cloud

Cloud computing applications should be scalable and should be able to process enormous amount of data. The cloud runtime does the mapping of the logical flow with the underlying cluster. Although the cloud allows the user to develop and deploy the application, but due to the highly distributed nature of the application it becomes difficult to fine-tune these application. [Rabl13] implements a highly scalable and lightweight performance analyzer tool for Hadoop named *HiTune*. This analyzer makes it feasible for users to understand and interpret the behavior of the big data cloud and thus help in taking educated decisions regarding the efficiency of the application. It allows integration of the performance tasks to the data flow model which helps the user to better understand the application faults and hardware problems. The data flow based performance model of *HiTune* consists of three components tracker, aggregation engine and analysis engine. The tracker creates sampling records of every program and sends it to the aggregation engine which collects sampling information from various other trackers and stores it in a cluster. The analysis engine is responsible for generating the analysis reports. These reports can be used by user to fine tune their applications.

5. Summary

In this paper we discussed the importance of big data in the present age. Then we tried to identify the bottlenecks in the underlying architectures and proposed techniques which can improve the performance of the bottleneck devices. Identifying bottlenecks is a very important task in performance modeling of any system. Improving the performance of the bottleneck device/resource improves the system performance significantly. We conclude that performance modeling of big data would help in

i. Improving performance : Increases the efficiency and performance of the system.

ii. System design and tuning : Ensures that the changes made using the model work coherently with the other devices or resources.

iii. Application design and tuning : Ensures that the application meets end user requirements.

iv. Improving scalability : Enables building distributed applications for the service to be provided to a larger audience.

6.References

1. [Pal14] Amrit Pal, Kunal Jain, Pinki Agrawal, Sanjay Agrawal, "A Performance Analysis of MapReduce Task with Large Number of Files Dataset in Big Data Using Hadoop",CSNT '14 Proceedings of the 2014 Fourth International Conference on Communication Systems and Network Technologies, Pages 587-591.

2. [Yu14] Se-young Yu, Nevil Brownlee, Aniket Mahanti, "Performance and Fairness Issues in Big Data Transfers", Proceedings of the 2014 CoNEXT on Student Workshop, Pages 9-11

3. [Rabl13] Tilmann Rabl, Sergio Gomez-Villamor, Mohammad Sadoghi, Victor Muntes-Mulero, Hans-Arno Jacobsen, Serge Mankovskii, "Breaking the boundary for whole-system performance optimization of big data", ISLPED '13, Pages 126-131

4. [Steed12] Chad A. Steed, Daniel M. Ricciuto, Galen Shipman, Brian Smith, Peter E. Thornton, Dali Wang, Xiaoying Shi, Dean N. Williams, "Big data visual analytics for exploratory earth system simulation analysis", Computers & Geosciences, Pages 71-82

5. [Huang13] Botong Huang, Shivnath Babu, Jun Yang, "Cumulon: optimizing statistical data analysis in the cloud", SIGMOD '13 Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, Pages 1-12

6. [Wei11] Wei Wei, Nan Cao, Jon Atle Gulla, Huamin Qu, "ImpactWheel: Visual Analysis of the Impact of Online News", WI-IAT '11, Pages 465-474

7. [Xu12] Weijia Xu, Wei Luo, Nicholas Woodward, "Analysis and Optimization of Data Import with Hadoop", IPDPSW '12, Pages 1058-1066

8. [Mesiti14] Marco Mesiti, Stefano Valtolina, "Towards a User-Friendly Loading System for the Analysis of Big Data in the Internet of Things", COMPSACW '14, Pages 312-317

9. [Liu12] Jialin Liu, Yong Chen, "Improving Data Analysis Performance for High-Performance Computing with Integrating Statistical Metadata in Scientific Datasets", SCC '12 Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, Pages 1292-1295

10. [Redekopp13] Mark Redekopp, Yogesh Simmhan, Viktor K. Prasanna, "Optimizations and Analysis

of BSP Graph Processing Models on Public Clouds", IPDPS '13, Pages 203-214

11. [Graefe14] Goetz Graefe, Haris Volos, Hideaki Kimura, Harumi Kuno, Joseph Tucek, Mark Lillibridge, Alistair Veitch, "In-memory performance for big data", Proceedings of the VLDB Endowment, Pages 37-48

12. [Bian14] Zhaojuan Bian, Kebing Wang, Zhihong Wang, Gene Munce, Illia Cremer, Wei Zhou, Qian Chen, Gen Xu, "Simulating Big Data Clusters for System Planning, Evaluation, and Optimization", BRACIS '14, Pages 391-400

13. [Yang11] P. Yang, M. Chuah, "Performance evaluations of data-centric information retrieval schemes for DTN", Computer Networks: The International Journal of Computer and Telecommunications Networking, Pages 541-555

14. [Kwon11] Ohbyung Kwon, Jae Mun Sim, "Effects of data set features on the performances of classification algorithms", Expert Systems with Applications: An International Journal, Pages 1847-1857

15. [Dai11] Jinquan Dai, Jie Huang, Shengsheng Huang, Bo Huang, Yan Liu, "HiTune: dataflow-based performance analysis for big data cloud", USENIXATC'11, Pages 7-7

16. [Liu12] Jialin Liu, Yong Chen, "Improving Data Analysis Performance for High-Performance Computing with Integrating Statistical Metadata in Scientific Datasets", SCC '12 Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, Pages 1292-1295

17. [Zhang13] Fan Zhang, Malluhi Q.M, Elsyed T.M, "ConMR: Concurrent MapReduce Programming Model for Large Scale Shared-Data Applications", ICPP, Pages 671-679

18. [Tudoran14] Radu Tudoran, Olivier Nano, Ivo Santos, Alexandru Costan, Hakan Soncu, Luc Bouge, Gabriel Antoniu," JetStream: enabling high performance event streaming across cloud data-centers", DEBS 14 : 8th ACM International Conference on Distributed Event-Based Systems, Pages 23-34

19. [Jain91] Raj Jain, "The Art of Computer Systems Performance Analysis", Wiley Publishers, 1991 Edition, ISBN - 10 : 0471503363, Mobius

20. [Eduardo14] Luis Eduardo Bautista Villalpando, Alain April, Alain Abran, "Performance analysis model for Big Data applications in Cloud Compting" , Journal of Cloud Computing. Pages 1-20

21. [SAS] "SAS", http://www.sas.com/en_us/insights/big-data/what-is-big-data.html

22. [BandwidthPlace] "BandwidthPlace", http://www.bandwidthplace.com/slow-internet-speed-avoid-the-bottleneck-article/

7.List of Acronyms

- UDP : User Datagram Protocol
- TCP: Transmission Control Protocol
- RDBMS : Relational Database Management Systems
- SQL : Structured Query Language
- MS-SQL Server : Microsoft SQL Server

Last Modified: May 1, 2015

This and other papers on performance analysis of computer systems are available online at
http://www.cse.wustl.edu/~jain/cse567-15/index.html
Back to Raj Jain's Home Page