ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 20, April 2016

An In-Memory approach to facilitate Big Data Processing for Web Search Engine

RagavanN[1],Athinarayanan S[2]
M.Tech Student[1],Assistant Professor[2]
Department of Information Technology, PSN College of Engineering & Technology,
Tirunelveli, India.[1,2]

Abstract **- Crawling, Indexing and Searching are the important process of Search Engine. Search Engine crawled data are so large or complex and it may be in structured or unstructured format. This large data set becomes Big Data and it is one of the emerging technologies that have changed the world. Growing main memory capacity has fueled the development of in-memory big data management and processing. By eliminating disk I/O bottleneck, it is now possible to support interactive data analytics.This paper discusses about Search Engines and its performance optimization by storing the data in flat binary files and in In-memory database.Data gathered from crawling resources are spilt into Text and Numeric data and link need to be provided based on Offset Indexing. Frequent access data need to be loaded into Main memory using cache technologies such as web-cache in Apache or IIS. Bulk data are stored in In-memory databases like REDIS. Big Numeric data can be stored in Flat File Binary Format such that they can be loaded into RAM in a single read operation. As run time disc access in flat file binary files are faster than conventional files, big data are stored in terms of Binary format which reduces the storage space than other file formats. Some issues such as fault-tolerance and consistency are also more challenging to handle in in-memory environment. This paperalso analyzes the challenging issues in the data-driven model in the Search Engine Big Data.**

Keywords **-Big Data Analytics, In-Memory Analytics, Flat Binary File, Bin File Processing**

1. INTRODUCTION

1.1. GYM (Google, Yahoo, Microsoft)

Normally search engine analyzes the contents of each page to determine how it should be indexed (for example, words can be extracted from the titles, page content, headings, or special fields called meta tags). The explosion of Big Data has prompted much research to develop systems to support ultra-low latency service.In 1995, when the number of"usefully searchable" Web pageswas a few tens of millions, it waswidely believed that "indexing thewhole of the Web" was alreadyimpractical or would soon become sodue to its exponential growth. The GYMsearch engines—Google, Yahoo!, andMicrosoft are indexing almost athousand times as much data andbetween them providing reliable sub-secondresponses to around a billionqueries a day in a plethora oflanguages.Now a day's many Search Engine providers (Google, Yahoo) are implementing their own custom file system in order to store the crawl big data efficiently. Google developed its own file system called GFS [11] (Google File System).

1.2. Disc based system

Database systems have been evolving over the last few decades, mainly driven by advances in hardware, availability of a large amount of data, emerging applications and so on. The data management systems are increasingly fragmented based on application domains. A major challenge for organizations is quickly accessing and moving huge quantities of data. SSDs[12] help with this because they have no moving parts and thus can access data faster than hard drives, which must rotate to a given position before being able to read information. "SSDs[12] will serve as the working memory for big-data analysis. SSDs[12] are 10 times more expensive per gigabyte of capacity than hard drives.

1.3 In-Memory Analytics

In-memory analytics is an approach to querying data when it resides in a computer's random access memory

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 20, April 2016

(RAM)[12], as opposed to querying data that is stored on physical disks. This results in vastly shortened query response times, allowing business intelligence (BI) and analytic applications to support faster business decisions. As the cost of RAM[12] declines, in-memory analytics is becoming feasible for many businesses. BI and analytic applications have long supported caching data in RAM, but older 32-bit operating systems provided only 4 GB of addressable memory. Newer 64-bit operating systems, with up to 1 terabyte (TB) addressable memory (and perhaps more in thefuture), have made it possible to cache large volumes of data potentially an entire data warehouse or data mart in a computer's RAM.In addition to providing incredibly fast query response times, in-memory analytics can reduce or eliminate the need for data indexing and storing pre-aggregated data in OLAP cubes or aggregate tables. This reduces IT costs and allows faster implementation of BI and analytic applications.

1.4 Big Data and Storage Challenge

Fisher et al. [1] pointed out that big data means that the data is unable to be handled and processed by most current information systems.Laney [2] presented a well-known definition (also called 3Vs) to explain what is the "big" data: volume, velocity, and variety. Later studies [3, 4] pointed out that the definition of 3Vs is insufficient to explain the big data we face now.

Traditional disk storage analytics may not be able to handle such large quantities of big data.New storage technologies are not in the near future. For that existing technologies led by the venerable hard drivewill have to step up.Now a day's many Search Engine providers (Google, Yahoo) are implementing their own custom file system in order to store the crawl big data efficiently.Google developed its own file system called GFS (Google File System). The report of IDC – International Data Corporation [5] indicates that the marketing of big data is about $16.1 billion in 2014. Another report of IDC [6] forecasts that it will grow up to $32.4 billion by 2017. The reports of [7] and [8] further pointed out that the marketing of big data will be $46.34 billion and $114 billion by 2018, respectively. So there is a great storage challenge of big data.

Sampling and compression are two representative data reduction methods for big data analytics because reducing the size of data makes the data analytics computationally less expensive, thus faster, especially for the data coming to the system rapidly. In addition to making the sampling data represent the original data effectively [9], how many instances need to be selected for data mining method is another research issue [10] because it will affect the performance of the sampling method in most cases.

1.5In-Memory Database - REDIS

Redis is an open source (BSD licensed), in-memory data structure store, used as database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, with range queries, bitmaps and geospatial indexes with radius queries. Redis has built-in replication scripting, and different levels of on-disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster. Redis is not a plain key-value store, actually it is a data structures server, supporting different kind of values. What this means is that, while in traditional key-value stores you associated string keys to string values, in Redis the value is not limited to a simple string, but can also hold more complex data structures. Redis hashes look exactly how one might expect a "hash" to look, with field-value pairs.

2.0. SYSTEM IMPLEMENTATION

Our proposed system discusses about Search Engines and its performance optimization by storing the data in flat binary files and in In-memory database. Data gathered from crawling resources are spilt into Text and Numeric data and link need to be provided based on Offset Indexing. Frequent access data need to be loaded into Main. Bulk data are stored in In-memory databases like REDIS. Big Numeric data can be stored in Flat File Binary Format such that they can be loaded into RAM in a single read operation.

2.1 Preparing In-Memory Dataset

Crawling and parsing are the main functionalities of search engine.Crawl data are parsed using Content Parsing algorithm and keywords (user search criteria) are generated and auto ids are assigned to each keyword. These keywords are stored in RDBMS. The Crawled Web page URLs are maintained in No SQL distributed database. The relationship between Web Page and its parsed keywords are stored as big data flat binary file format since the run time disc access in flat file binary files are faster than conventional files, also Binary format which reduces the storage space up to 35% than other file formats.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 20, April 2016

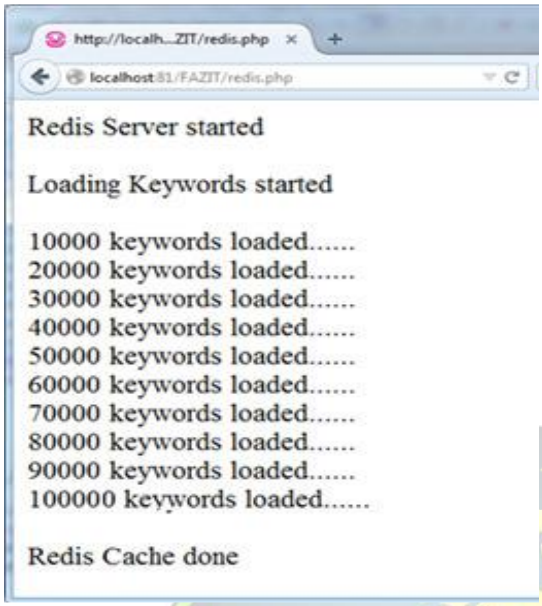

Fig.2.1.a.REDIS Client developed in PHP

Newly generated keywords in RDBMS are and loaded into In-Memory database with key-value hash structure in a distributed architecture fashion periodically. In our proposed system MYSQ is used as RDBMS which holds the Keyword details. As billons of single keywords and millions of complex keywords are needed to be stored, we have maintained 26 keywords table as per alphabetical order.

For example, if the newly arrived single keyword is "PSN", and complex-keyword is "PSN College of Engineering", then these keywords will go to the P-keyword table. A REDIS client service is then to be implemented in any language (PHP) as shown in Fig.2.1.a. The REDIS server accepts request from REDIS client by listening at a specified port 6379. The REDIS client will look up into MYSQL keywords table and loaded into the In-Memory database (REDIS) in specified frequent interval of time.

2.2 Preparing In-Memory FileSystem

In our proposed system, the search engine crawled data are separated into numeric and text. Mostly the text data are served as master data (ex: resultant page urls) and the numeric data are served as detail data such as resultant page_ids after searching from user side. It seems, detail data are redundant in nature. So numeric data are redundant which occupy more space than master data. In Web Search Engine system, text data are maintained in RDBMS and numeric data are stored in terms of binary file system. The crawled pages' master level entries are then saved into database and the remaining numeric type page details such as page_id, keyword_ids are stored in binary files. Here the binary files contain headers, blocks of metadata used by a computer program (usually in c++) to interpret the data in the file. The header often contains a signature or magic number which can identify the format. The header file which contains metadata is stored in memory, so that master operations are fast. Furthermore, it is easy and efficient for the master to periodically scan through its entire state in the background. This periodic scanning is used to implement dynamic loading, free-up the files from memory. The master file size is small and it often contains the detail file offset details. Detail files on the other hand grow up to any limit. It can reach up to Terra size. Since detail files are lookup by means of hop method during runtime by search routine, it is not fully loaded into the memory. This requires offset data (gathered from master file) that are dynamically retrieved in a single read.

Two types of binary files are created in proposed system namely PageMaster.bin and PageDetail.bin

A. PageMaster.bin:

There are 5 columnar fields and these data will be stored in a single column format. That is KeywordCount column data will be stored first. After that KeywordIds values will be stored and so on. Fig.2.2.a. shows the PageMaster.bin prototype.(which contains 100 keywords from 500 crawled pages).

PageMaster.bin
100
K1
K2
. . . .
K100
K1_offset
K2_offset
. . . .
K100_offset
500
P1
P2
. . .
P500

Fig.2.2.a. PageMaster.bin prototype

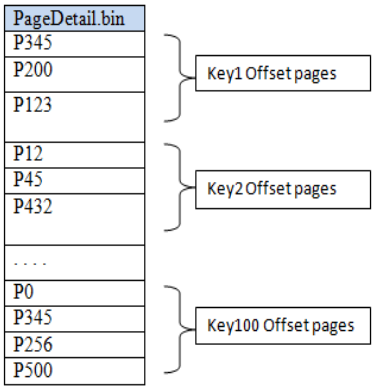

Fig.2.2.b.PageDetail.bin prototype

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 20, April 2016

(i) KeywordCount:- Total Unique Keywords count. This values is stored as an unsigned integer 32 bit.

(ii) KeywordIds:- Available KeywordIds that are generated from the crawled data.

(iii) KeywordPageOffset:- Each keyword may belong to so many crawled web-pages. Each keyword Page Index are stored in detail bin file. The page index offset in PageDetail.bin per each keyword is called KeywordPageOffset

(iv) PagesCount:- Total Unique Keywords count. This values is stored as an unsigned integer 32 bit.

(v) PageIds:- Available Unique PageIds from the Unique Keywords. Since in big data crawling, the crawled web page is increasing day by day. So it is needed to store/assign page Id as an unsigned long data type (64 bit).

B. PageDetails.bin

This binary file maintains the Page-Indexes found for each unique keyword from Master file. The size of the file will be directly proportional to the number of keywords found in the crawled web pages. So in case of web page article having more text content (especially Wikipedia), then the detail file size will grow in TB range. In such cases, detail files can be split into reasonable equal size junk files. Based up the KeywordPageOffset in master file, the corresponding detail file will be fetched. The reason for using page index instead of page id is, some keywords belong to more than one pages, so that duplicate page ids are required to maintain in detail file. As the data type of page id is specified as unsigned long type (64 bit), storing duplicate entries will greatly increasethe bin file size. So page index (32 bit) is used here. This is shown in Fig.2.2.b

III. RESULTS AND DISCUSSION

In our proposed system, Search Engine big data are stored in terms of Binary format which reduces the storage space up to 35% than other file formatsas shown in Fig.3.1. REDIS client is developed in PHP. Redis client can be launched either into the browse or execute the Redis client in the server terminal and Redis Server should be started at port no: 6379.Keywords areloaded into In-memory Redis. Flat master binary files created from search engine crawled data are loaded into memory using search routine and it is ready to accept request from user side. Once request arrived, the keyword id is fetched from in-memory redis.

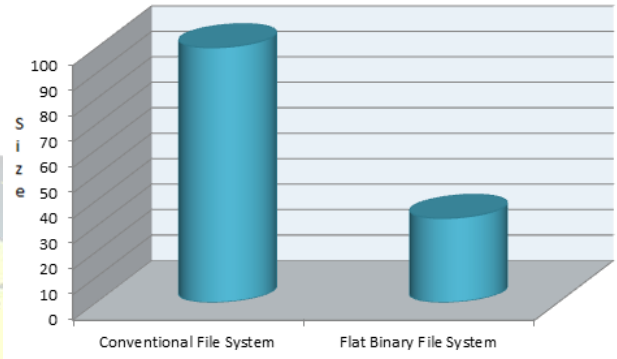

Fig.3.1. Big Data File size between Conventional and Flat binary file system.

Conventional DB lookup timing for 1 million records is around 20 milliseconds (average) In Redis In-Memory database, the memory lookup time: 3 to 7 millisecond. For mass amount of data, DB look is increasing geometrically (without indexing), whereas for Redis In-memory database it is constant independent of number of data. Thus DB logging bottleneck is eliminated in memory database systems. Speed efficiency increased to = 285 %. Fig 3.2.shows the DB lookup performance of conventional vs in-memory method.

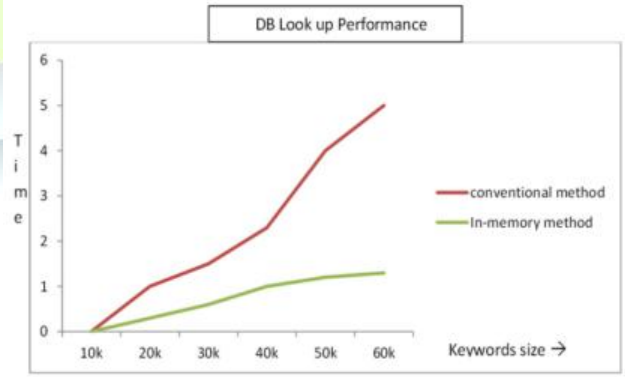

Fig 3.2.DB lookup performance of conventional vs in-memory method.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 20, April 2016

IV. CONCLUSION

The important feature of the big data files is that can be loaded into RAM in a single read operation. This increases the loading performance. Once the page master level entries are loaded into the memory structure, its relevant detail entries are fetched in hop method from the detail file. Its relevant offset positions maintained the master files are pre-loaded into the memory data structure. So the run time disc accesses in flat file binary files are faster than conventional files. Also big data are stored in terms of Binary format which reduces the storage space up to 35% than other file formats.Since the frequently accessed data such as user searchable keywords are placed in In-Memory database, whenever user searches any keywords, the keyword ids are fetched from REDIS database in less than 1 milli second. Then by using the keyword id, required master and detail binary files are selected. The resultant page-ids are fetched from detail binary file by using the key-offset from master bin file. As master binary files are already loaded into RAM during the startup of search routine, this idea increases the searching performance of search engine.

ACKNOWLEDGEMENT

The researchers duly acknowledge the support provided by the Management and Principal of PSN College of Engineering and Technology -Tirunelveli by means of providing all the research facilities.

REFERENCES

[1] Fisher D, DeLine R, Czerwinski M, Drucker S. Interactions with big data analytics. Interactions. 2012;19(3):50–9.

[2] Laney D. 3D data management: controlling data volume, velocity, and variety, META Group, Tech. Rep. 2001. [Online]. Available: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

[3] van Rijmenam M. Why the 3v's are not sufficient to describe big data, BigData Startups, Tech. Rep. 2013. [Online]. Available: http://www.bigdata-startups.com/3vs-sufficient-describe-big-data/.

[4] Borne K. Top 10 big data challenges a serious look at 10 big data v's, Tech. Rep. 2014. [Online]. Available: https://www.mapr.com/blog/top-10-big-data-challenges-look-10-big-data-v.

[5] Press G. $16.1 billion big data market: 2014 predictions from IDC and IIA, Forbes, Tech. Rep. 2013. [Online]. Available: http://www.forbes.com/sites/gilpress/2013/12/12/16-1-billion-big-data-market-2014-predictions-from-idc-and-iia/.

[6] Big data and analytics—an IDC four pillar research area, IDC, Tech. Rep. 2013. [Online]. Available: http://www.idc.com/prodserv/FourPillars/bigData/index.jsp.

[7] Taft DK. Big data market to reach $46.34 billion by 2018, EWEEK, Tech. Rep. 2013. [Online]. Available: http://www.eweek.com/database/big-data-market-to-reach-46.34-billion-by-2018.html.

[8] Research A. Big data spending to reach $114 billion in 2018; look for machine learning to drive analytics, ABI Research, Tech. Rep. 2013. [Online]. Available: https://www.abiresearch.com/press/big-data-spending-to-reach-114-billion-in-2018-loo.

[9] Cormode G, Duffield N. Sampling for big data: a tutorial. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014. pp 1975–1975.

[10] Satyanarayana A. Intelligent sampling for big data using bootstrap sampling and chebyshev inequality. In: Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering, 2014. pp 1–6.

[11] S. Ghemawat, H. Gobioff, and S.-T.Leung, "The google file system," in Proc. 19th ACM Symp. Operating Syst. Principles, 2003,pp. 29–43.

[12] Hao Zhang, Gang Chen, "In-Memory Big Data Management and Processing: A Survey", IEEE transactions on knowledge and data engineering, VOL. 27, NO. 7, JULY 2015

[13] S. Sanfilippo and P. Noordhuis. (2009). Redis [Online]. Available: http://redis.io