A Comprehensive Study on Meta Data Indexing Methods for Big Data and Multi Dimensional Database

E.K.Girisan
Assistant Professor, Department of Computer Science, Sree Narayana Guru College, K.K Chavadi, Coimbatore, Tamil Nadu, India.

Reena Cherian
M.Phil Scholar, Department of Computer Science, Sree Narayana Guru College, K.K Chavadi, Coimbatore, Tamil Nadu, India

Abstract – Big data is the solution to handle large and complex data, which captures data, stores, analyze, search, querying, visualization and updating. Big data use predictive analytics, user behavior analytics and other advanced data analytics methods, which are extracted from large data source. The big data analysis is challenging due to its large data infrastructure. The data retrieval should have high storage performance and applications I/O. the integration of these two in an effective way may provide successful data analytics and query processing. So, it's important to analyze the earlier works on indexing techniques along with its merits and demerits to define an appropriate method. This survey gives the comparative analysis of distributed indexing technique such as Meta data indexing, multi dimensional data indexing methods for big data infrastructure.

Index Terms – Big Data, Indexing, Metadata Indexing, Search And Information Retrieval.

1. INTRODUCTION

Big Data concerns large-volume, complex, growing data sets with multiple, autonomous sources. With the fast development of internet and cloud data storage, Big Data is now rapidly expanding in all application domains [1]. It handles large and complex data's. This paper describes about big data, indexing methods and recent techniques used in the data indexing. This article presents a comparative analysis, which finally concludes the problems and issues of data handling and indexing methods. This data-driven model involves demand-driven aggregation of information sources, mining and analysis, user interest modeling, and security and privacy considerations. The paper helps to analyze the challenging issues in the data driven technique and also in the Big Data development. Data Indexing in big data is a challenging job which changes the personal data with non-personal data. The tremendous amount of electronic data floating around us such as operational data, customer data, web data, social data, marketing data, computer data, supply chain data, transaction data, behavioral data etc. The two troublesome patterns at present forcing noteworthy effects on IT industry and research groups are Cloud computing and Big Data. The paper provides the basic structure and introduction of big data and comparison of various indexing methods.

1.1 BIG DATA

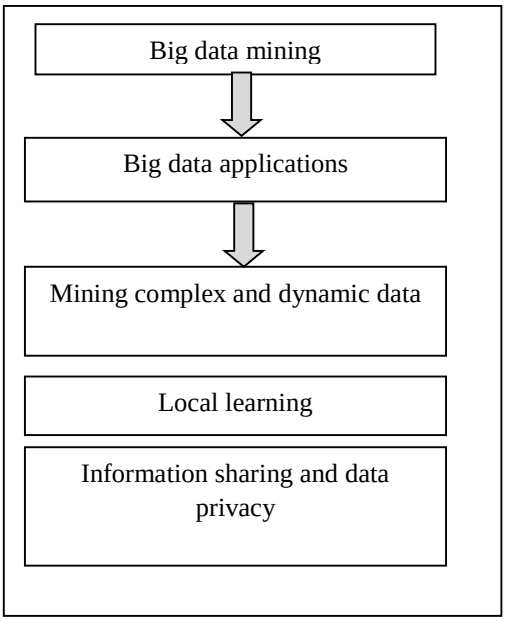

Fig 1.0 Big Data Process

Big data is a broad term for data sets so large or complex that traditional data processing applications are inadequate. Challenges include analysis, capture, pre-process, and search, sharing, storage, transfer, visualization, and information privacy. The term often refers simply to the use of predictive analytics or other certain advanced methods to extract value from data, and seldom to a particular size of data set. According to analysis of data sets can find new correlations, to spot business trends, health information extraction and so on.

Several domains regularly meet difficulties with large data sets in areas including Internet search, finance and business informatics.

Fig 1.0 shows the process involved in the big data analysis. This includes the data collection from various big data applications, mining uncertain and dynamic dataset from the applications. The collected information's are learned by local learning process and finally shares the resources or keeps them with security and privacy.

1.2 Indexing

Indexing is involved with the specific data extraction process from the data outsourced [2]. In modern system, the data size is huge and complex, when the application uses big data. So indexing is the effective method to retrieve data quickly. The index process involved with the attribute and attribute vectors. This includes two types of process hypothesis and verification. The hypothesis retrieval search and returns the data through index and it sometimes returns the data with false alarms. The next step verification eliminates the false alarm from the first step.

2. LITERATURE REVIEW

2.1 Indexing For Multi-Dimensional Database:

In paper [3], authors suggested multidimensional indexing system at physical level for OLAP databases in order to obtain multidimensional view of data at conceptual level. B-trees, which are perfect indexing data structures in relational database management system, were extended to multiple attributes to form multidimensional indexing system.

In the paper [4], authors proposed star schema as conceptual design of multidimensional databases. At physical level these multidimensional databases have B-tree based indexing on primary key of each dimension. Internal nodes of the B-tree contain the primary keys of the dimension tables. The leaf nodes of the B-tree contain the pointer to the records, in the fact table at the center of star schema.

In paper [5], authors worked on the traditional bitmap indexes and extended them to index, multidimensional data warehouse or OLAP engines. Bitmap indexing technique was optimized for both space and time under a given disk space constraints. The bitmap indexes are in use for indexing multidimensional database but failed in case of large data sets.

In paper [6], authors proposed an encoded bit map indexing for multidimensional data which improved the performance of known bitmap indexing in case of large cardinality domains. These encoded bitmap indexes are compared with related techniques such as bit slicing, projection index, dynamic bitmaps and range-based indexing. The problem of sparsity is solved, as a result of which performance of bitmap based indexing enhanced particularly for large multidimensional databases

In paper [7], authors proposed a heuristic approach to estimate, workload and storage space requirement for view materializations and indexing. Optimal trade-off between the space devoted to view materialization and that devoted to indexing is the result of estimation technique used.

In paper [8], authors provides the idea of an auto-index selection tool capable of analyzing large amounts of data and suggested a good set of indexes. The auto-indexing technique is based on clustering. Clustering technique like K-mean is applied for index selection for large databases. Performance of this clustering technique outperformed the Microsoft SQL server index selection tool.

In paper [9, 10], authors gave automatic materialized view and index selection technique for OLAP databases to dynamically determine which materialized views to be maintained. These materialized views helps in reducing query answering time. Design advisor tool was introduced for materialization of view and indexes in IBM DB. This tool automatically captures the workload, database and system information and provides optimal candidate attributes for materialized views

Dynamic time wrapping techniques were used for computing the input string against the model sequences and arithmetic averaging was used for updating the models. In this technique both symbol of string and length are considered for computations [11].

2.2 Meta Data Indexing Methods in Big Data

The metadata connected with documents which are uploaded can be indexed and used as criteria to search for and retrieve documents from distributed storage servers or big data servers. Not all metadata may be indexed and made available for searching. This indexing is based on the application and its requirement. The purpose of metadata indexing is to make the searching more effectively from huge and complex data sources. A text from a document gets indexed when the document is uploaded to server, which enable the user to search for words or phrases from within the document. Like metadata, searches for the full text of documents are done using a field on the search form when the search is created.

In paper [12], a bitmap based indexing scheme which is named as BIDS is developed to handle large amount of data in the distributed systems. The BIDS index is storage efficient and easy to maintain, which makes it more scalable. It is built on top of the underlying DFS and cached in the distributed memory. BIDS adopts WAH encoding, bit-sliced encoding and pre-sorting to ensure compactness. To further reduce the index size, the index is dynamically tuned based on the query patterns. And the BIDS also introduced query processing based on the indexing scheme. The query operators are transformed

into a set of bit-wise AND/OR operators, which can be handled more efficiently. The paper affected by computation cost, which incurs cost for storage and index build cost.

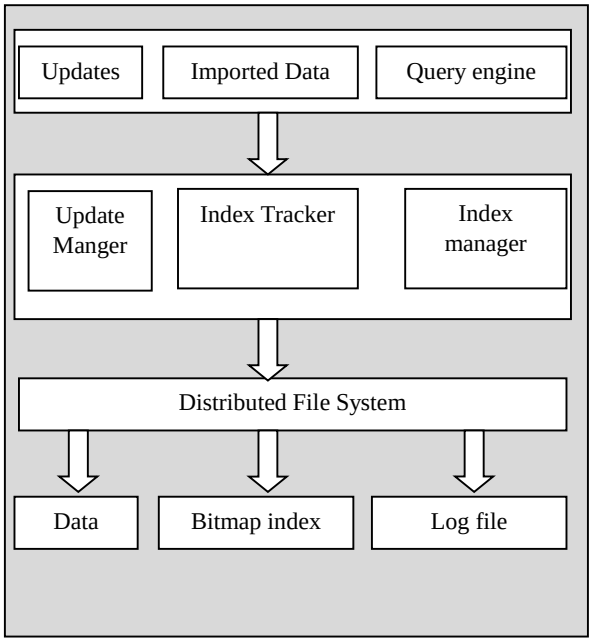

Fig 2.0 BIDS overview

BIDS indexing technique adopts a novel and query efficient partial indexing technique to reduce the size of indexes.

In paper [13], authors have presented the design and implementation of a scalable and high-throughput indexing scheme for cloud software as a service (SAAS). This assumes a local B+-tree is built for the dataset stored in each compute node. And to enhance the throughput of the system, It organizes compute nodes as a structured overlay and build a Cloud Global index, called the CG-index, for the system. Only a portion of local B+-tree nodes are published and indexed in the CG-index. Based on the overlay's routing protocol, the CG-index is disseminated to compute nodes. To save maintenance cost, it propose an adaptive indexing scheme to selectively expand local B+-trees for indexing. The scheme has been implemented and evaluated in Amazon's EC2, a real-world Cloud infrastructure.

Many data's are infrequently accessed in data servers. Cloud storage service providers commonly store such infrequent data and their Metadata in low-cost commodity hardware for cost effective storage. While, there are several kinds of storage services which need to ensure the high-performance access and retrieval to infrequent data. Since some of infrequent data have not been accessed for a long time, traditional metadata are not useful for searching them. In order to solve these problems, authors in [14] proposed an efficient and effective searchable metadata indexing based on data provenance, which is called P-index. P-index partitions correlative files into logical groups via provenance relationships of files. This method quickly cuts off the sub-trees which do not contain the query results to improve the efficiency of metadata search. Moreover, P-index adds the metadata extracted from data provenance into index structure to improve the effectiveness of metadata search. The authors evaluated the performance of P-index via two complex queries, range and k-nearest-neighbor (KNN) queries. Compared with state-of-the-art metadata index methods, P-index improves the efficiency and effectiveness of metadata search. However, the P-Index generates computational overhead.

In the paper [15], authors have presented a distributed metadata indexing technique called Dindex that achieves resiliency, flexibility, and efficiency in metadata queries. New data structures and algorithms are devised to characterize the distributed and hierarchical nature of Dindex along with an in-depth theoretical analysis of its feasibility. Dindex has been implemented with a lightweight distributed key-value store and integrated into a fully-fledged distributed file system with promising results: It delivers up to 60% faster file queries with negligible overhead. One important factor affecting Dindex's performance at runtime is access frequency (along with others such as access distribution that implies data skewness and load balance). Although the proposed data structure and algorithm does not directly take access frequency into account, from systems perspective Dindex can cache certain hot spot for frequent accesses. The Dindex techniques are not supported for a distributed data index. That is, a unified indexing system will be available for queries on both data and metadata. The Dindex's design principles are not integrated with databases and informational retrieval.

Table 1.0 Recent big data indexing methods comparison table

Indexing method	**Advantages**	**Disadvantages**
BIDS	- Supports dynamic query patterns - cached in the distributed memory	Size of index creates more complex
CGIndex	Global index scheme provides scalability. Effective for cloud	Real world configurations have dynamic performance issues

Dindex	Improves the performance runtime	not supported for a distributed data index

Existing indexing techniques shown in table 1.0 are discussed for big data and multi dimensional databases. The recent researches on indexing techniques imply the approach to improve and customize the indexing methods for big data. The performance of indexing accuracy is not widely studied. The development of new indexing technique for big data with accuracy and time efficiency is more important.

3. CONCLUSION

This paper presented a review of literature for multidimensional indexing and Meta indexing system for big data analytics system. Data indexing is the effective method for fast data retrieval. In the big data, the indexing for normal data is more difficult. So Meta data indexing method has been proposed. However, the indexing for either Meta data or normal data is performed. There is no technique available to support both data and Meta data. There is a strong motivation for enhancement, which can combine the big data and Meta data indexing. The efficient indexing systems are needed to be developed which with attributes of speed in data accessing and fault tolerance.

REFERENCES

[1] Srinivasa, S., & Bhatnagar, V. (2012). Big data analytics. In *Proceedings of the First International Conference on Big Data Analytics BDA* (pp. 24-26).

[2] Bertino, E., Ooi, B. C., Sacks-Davis, R., Tan, K. L., Zobel, J., Shidlovsky, B., & Andronico, D. (2012). *Indexing techniques for advanced database systems*(Vol. 8). Springer Science & Business Media.

[3] Codd E. F.(1993), Providing OLAP to User-Analysts: An IT Mandate byEFCodd, S B Codd and C T Salley, ComputerWorld, July 26, 1993.

[4] Gray J., Bosworth A., Layman A., and Pirahesh H., (1996). Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total, In Proceedings of the Twelfth International Conference on Data Engineering (ICDE '96), IEEE Computer Society, Washington, DC, USA, 152-159

[5] Chan Chee-Yong and Yannis E. Ioannidis.(1998), Bitmap index design and evaluation, In Proceedings of the 1998 ACM SIGMOD International conference on Management of data (SIGMOD'98),NY, USA, 355-366, DOI=10.1145/276304.276336 http://doi.acm.org/10.1145/276304.276336.

[6] Hing- Chuan Wu, and Alejando P. Buchmann(1998), Encoded Bitmap Indexing for Data Warehouses, In Proc. of the 14th Intl. Conf. On Data Engineering(ICDE 98), IEEE Computer Society Washington, DC, USA, 220-230

[7] Rizzi S.,and Saltarelli E. (2003), View Materializing vs. Indexing Balancing space constraints in data warehouse design., proc. 15th Int. conf. on Advance Information systems Engineering, (CAiSE '03), pages 502-519, Springer 2003

[8] Zaman Mujiba, Surabattula Jyotsna, and Gruenwald Le. (2004), An Auto-Indexing Technique for Databases Based on Clustering, In Proceedings of the Database and Expert Systems Applications, 15th International Workshop (DEXA '04), IEEE Computer Society, Washington, DC, USA, 776-780. DOI=10.1109/DEXA.2004.32 http://dx.doi.org/10.1109/DEXA.2004.32

[9] Agrawal S., Chaudhuri S., and Narasayya V.(2000), Automated selection of materialized views and indexes for SQL databases, Proc. 26th VLDB'00 Morgan kaufmann, 496-505.

[10] Agrawal, S., Narasayya V., and Yang, B(2004), Integrating vertical and horizontal partitioning into automated physical database design, In Proc. ACM SIGMOD international Conference on Management of Data, (SIGMOD '04), ACM, New York, NY, 359-370

[11] Sharma Mayank , Rajpal Navin , Reddy B.V. R., and Purwar Ravindra Kumar. (2013), Normalised LCS-based method for indexing multidimensional data cube, Int. J. Intell. Inf. Database Syst. 7, 2 , 180-204. DOI=10.1504/IJIIDS.2013.053550 http://dx.doi.org/10.1504/IJIIDS.2013.053550

[12] Lu, P., Wu, S., Shou, L., & Tan, K. L. (2013, April). An efficient and compact indexing scheme for large-scale data store. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on* (pp. 326-337). IEEE.

[13] Wu, S., Jiang, D., Ooi, B. C., & Wu, K. L. (2010). Efficient B-tree based indexing for cloud data processing. *Proceedings of the VLDB Endowment*, *3*(1-2), 1207-1218.

[14] Liu, J., Feng, D., Hua, Y., Peng, B., Zuo, P., & Sun, Y. (2015, November). P-index: An Efficient Searchable Metadata Indexing Scheme Based on Data Provenance in Cold Storage. In *International Conference on Algorithms and Architectures for Parallel Processing* (pp. 597-611). Springer International Publishing.

[15] Zhao, D., Qiao, K., Zhou, Z., Li, T., Lu, Z., & Xu, X. (2017). Toward Efficient and Flexible Metadata Indexing of Big Data Systems. *IEEE Transactions on Big Data*, *3*(1), 107-117.