RESEARCH ARTICLE OPEN ACCESS

Review on Indexing Methodologies for Microblogs

A. Vegi Fernando*, Dr. K. Ramesh**

*(Computer Science and Engineering, SCAD College of Engineering and Technology, Cheranmahadevi
Email: a.vegifernando@gmail.com)

** (Department of Computer Applications, Regional Office, Anna University, Tirunelveli
Email: rameshk7n@yahoo.co.in)

Abstract:

Bigdata is the term used for datasets so large and complicated that it becomes difficult to process using traditional data management tools or processing applications. Microblogging is an area where data is produced in large volumes. Microblogging is a broadcast medium that exists in the form of blogging. Microblogs allows user to exchange small forms of contents such as short sentences, individual images or video links. Since microblogs are created at tremendous speed, indexing microblogs for real time search is challenging. Commonly used indexing techniques are append-only indexing, provenance-based indexing, spatio-temporal based indexing, partial indexing, log structured index and compact indexing. This survey aims at providing an insight about different indexing structures pertaining to microblogs and also summarizes the benefits and limitations of various indexing techniques.

Keywords — **Microblogs, append-only indexing, provenance-based indexing, spatio-temporal based indexing, partial indexing, log structured index and compact indexing.**

I. INTRODUCTION

Bigdata: A massive volume of both structured and unstructured data that is so large which is too difficult to process using traditional database and software techniques. Bigdata[8] has 3V properties: Volume, Variety and Velocity. Volume refers to the size of the data that is being worked with. With the advancement of technology and with the invention of social media, the amount of data is growing very rapidly. This data is spread across different places in different formats, in large volumes ranging from Gigabytes to Terabytes, Petabytes and even more. Velocity refers to the speed at which the data is being generated. In today's scenario, decision makers want the necessary information in the least amount of time as possible. Variety refers to the different formats in which data is being generated and stored. Apart from the traditional flat files, spread sheets and relational databases there are a lot of unstructured data stored in the form images, audio files, video files, web logs, sensor data and many others.

There are a variety of Bigdata origins. Sources of Bigdata can be broadly classified into six different categories[8]: (i) Enterprise data, (ii) Transactional data, (iii) Social media, (iv) Activity generated data, (v) Public data and (vi) Archives. Enterprise data: There are large volumes of data in Enterprises in different formats. Common formats include flat files, emails, word documents, spread sheets, presentations, HTML pages, pdf documents, XMLs and legacy formats. Transactional data: Every Enterprise has some kind of applications which involves performing different kinds of transactions like web applications, mobile applications, CRM systems and many more. To support the transactions in these applications there are usually one or more databases as a back-end infrastructure. Social Media: There is a large amount of data generated on social networks like Twitter, Facebook and many more. The social networks usually involve unstructured data formats such as text, images, audios and videos. Activity Generated data: There is a large amount of data being generated by machines which surpasses the data volume generated by humans. These include

data from medical devices, sensor data, surveillance videos, satellites, cell phone towers, industrial machinery and other data generated mostly by machines. Public Data: This data includes data that is publicly available data like data published by government, research data published by research institutes, data from weather and meteorological department, census data, Wikipedia and other data which are freely available to the public. Archives: Organizations archive a lot of data which is either not required anymore or is very rarely required. In today's world with hardware getting cheaper, no organization wants to discard any data instead they want to store as much data as possible. Other data that is archived includes scanned documents, scanned copies of agreements, records of completed projects and many more.

Microblogging[11] is a broadcast medium that exists in the form of blogging. A microblog differs from a traditional blog in that its content is typically smaller in both actual and aggregated file size. Microblogs allows users to exchange small contents such as short sentences, individual images and video links. Microblogs produces large volume of information. A recent study[11] shows that Twitter accepts 500,000,000 tweets per day and Sina Weibo has hundred million users that create over 10,000 microblogs per second. Microblogs are created at tremendous speed and Query requests from users keeps constantly changing. Hence many problems need to be resolved in this area like microblog ranking, microblog spatio-temporal queries and indexing structure for microblogs.

Index is typically used to restrict access to only the relevant data. Indexes are basically divided into primary index and secondary index. Primary index holds relevant data and it thus improves the evaluation of queries of all ranges. Secondary indexes are auxiliary structures that are used to store less frequently used data. However retrieving the relevant data from disk can be a costlier operation since it may be scattered over many pages. Several Indexing Structures have been used to efficiently store and retrieve microblogs. Techniques namely Append-only inverted index, Provenance-based indexing, Spatio-Temporal based indexing, Partial indexing, Log-structured based index and Compact indexing. Different techniques were used in diverse times with an improvement from the previous methodology. Still continuous research is on the pipeline searching for the efficient indexing methodology.

This paper focuses on different indexing methodologies in relation to that of microblogs and also summarizes its benefits and limitations.

II. CHARACTERISTICS OF INDEXING STRUCTURES

A. Digestion Rate

Microblogs arrive in high rates. Hence, the indexing structure should digest the incoming microblogs so that it does not miss any data. Acceptable digestion rates for microblogs real-time applications are thousand microblogs per second. This means the indexing structure should digest each microblog in a fraction of a milli-second.

B. Searchability latency

Searchability latency is the measure of how fast the microblogs will be available in search results. This measure is defined as the measure of the average latency between any microblog being available in the system till it becomes available in the search results. Most of the real-time applications that are built on the top of microblogs would accept searchability in no more than few seconds.

C. Memory hit

The index structure manages hundreds of millions of microblogs that have recently arrived in main-memory and receives many queries on this recent data. Memory hit is the measure of the number of queries that can retrieve their answers entirely from memory. The larger this number the overall performance is better. The optimal is to have 100% of memory contents are actually used.

III. SURVEY OF MICROBLOG INDEXING METHODOLOGIES

Microblogs produces large volume of information at a higher rate. They are created at tremendous speed and the user requests for queries vary constantly. It is always been a challenge to store such enormous amount of information and retrieve it efficiently. The following are the indexing methodologies widely used with related to microblogs used in different time periods and they are shown in Table I

TABLE I
INDEXING METHODOLOGIES

Sl.no	Types of Indexing methodologies
1.	Partial indexing approach
2.	Append-only inverted index
3.	Provenance based Indexing
4.	Spatio-Temporal based Indexing
5.	Log Structured Index
6.	Compact Indexing

A. *Partial Indexing Approach*

Partial Indexing[6] is an indexing approach where the whole database is not indexed. It just index the microblogs that are very likely to be the results of a certain query. Tweet Index(TI) is one of the indexing methodology that uses partial indexing approach

Tweet Index[4] differentiates frequent queries from infrequent queries. For any microblog, if it is relevant to alteast one frequent query, it is inverted in to the inverted index otherwise it is inserted into an append-only log. The contents of the append-only log is periodically pushed into the inverted index based upon the frequency ratio. The keyword query is processed using the inverted index without looking into the append-only log.

The partial indexing approach[6] improves the update efficiency as it reduces the number of microblogs that is to be processed in a real-time manner. It also has several deficiencies: (i) infrequent queries could account for 70% of the queries issued by the user. Hence if the infrequent queries are placed only in the append-only log, it creates a negative experience. (ii) The Distribution of real time queries between inverted indices and append-only log could change periodically. The infrequent query can become a frequent query within a short period. It again results in missing results for certain queries that were considered infrequent but actually became frequent over time. Hence partial indexing might end up in issuing incomplete results.

B. *Append-Only Inverted Index*

In the partial indexing approach, the microblogs that were considered to be accessed frequently are placed in the index while the other microblogs were inserted in the append-only log whereas in append-only inverted index, all the microblogs whether it may be a frequently or infrequently accessed, they all were placed in the append-only Earlybird servers.

Earlybird[1] is the core retrieval engine that supports Twitter's real time search service. Tweets from Twitter first enter the ingestion pipeline. In the ingestion pipeline they are tokenized and annotated with metadata(e.g. Fig. 1). Since the tweets are in huge volumes, the tweets are partitioned across multiple Earlybird severs. The Earlybird servers index the tweets. The component called as Updater pushes the dynamic resonance signals to the Earlybird servers. Dynamic resonance signals are updated over time(Example: Number of retweets a tweet receives). At query time, a Blender server parses the user query and passes it to the multiple Earlybird servers. These servers uses a ranking function that combines the dynamic resonance signals to compute a personalized relevance score for each tweet. The highest ranking most recent tweets are returned to the Blender. The Blender merges and re-ranks the results. Then the results are returned to the user. Earlybird has a 10 second indexing latency and 50ms query latency.

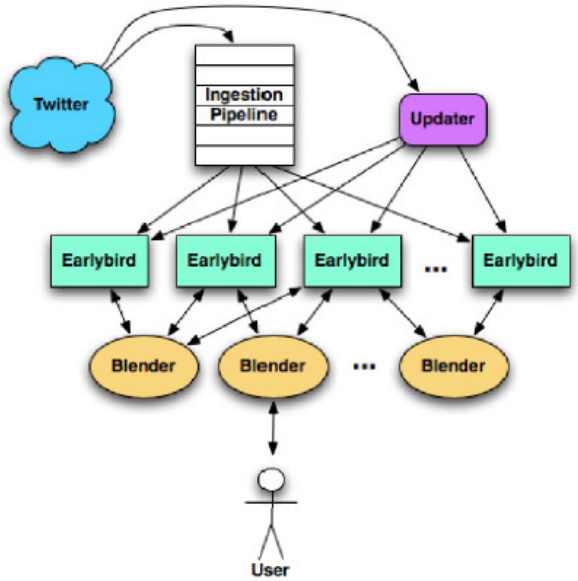

Fig. 1 Architecture of Earlybird-Twitter's real-time search engine

C. *Provenance based Indexing*

Provenance refers to the data origin identification and transformation logging, demonstrating of great value in recent database and workflow systems. The provenance model is used to achieve the connection between microblog messages using the provenance information.

The Provenance based indexing framework[3] is divided into two parts: (i) in-memory processing unit and on-disk storage back-end. The in-memory unit has a summary index to manage bundle collections and a bundle pool to maintain bundles. The back-end storage is used to keep the finished bundles that no longer receive updates. The bundle summary index retains the enough indicants to aid the new messages routing and efficiently placing them into appropriate bundles(e.g. Fig. 2). If found, the new messages are inserted into the most appropriate one. If no such bundle exists, new bundle will be created. Bundles are retained in memory for fast match and insertion. After some iterations of new message processing, some old bundles will be eliminated or flushed back into the disk.

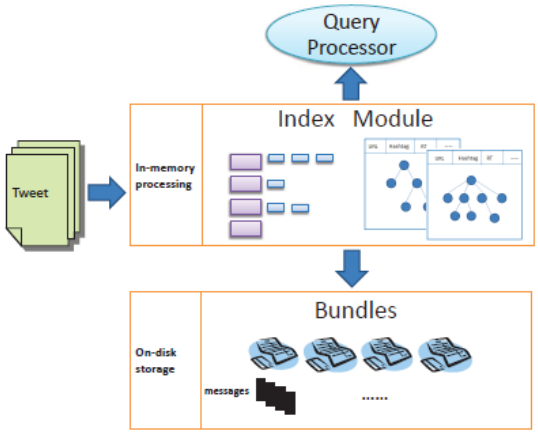

Fig. 2 Framework of Provenance based Indexing

D. *Spatio-Temporal based Indexing*

Spatio-Temporal indexing is a system for real-time support of top-k spatio-temporal queries on microblogs, where users are able to browse recent microblogs near their locations. Mercury[2] is a system for support of top-k spatio-temporal queries[12][13] on microblogs. Mercury bounds it's search space to include only those microblogs that have arrived within certain spatial and temporal boundaries. Here only top-k microblogs are returned in the search results. Mercury employs the following modules to achieve the spatio-temporal limit on microblogs.

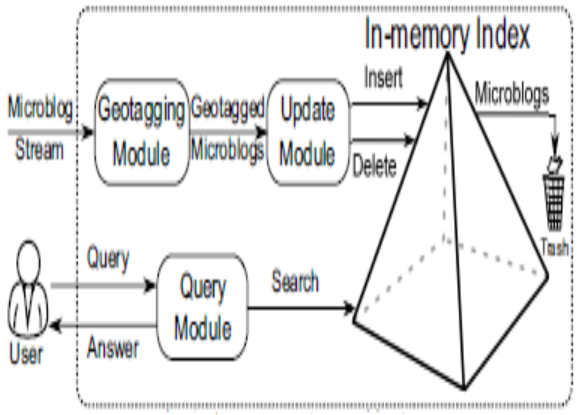

Fig. 3 Mercury System Architecture

Geotagging module: Geotagging module receives the incoming stream of microblogs, extracts the location of each microblog and forwards each micrioblog along with its extracted location to the update module. If the microblog arrives from more than one location, multiple versions of the location is given as output.

Update module: The update module ensures that all incoming queries can be answered accurately from the in-memory contents that are indexed. The newly coming microblogs are inserted into the in-memory index structure. Then the set of microblogs that have to be moved out from the in-memory are decided without sacrificing the query answer quality. Suppose if there is a very tight memory , a load shedding module is used to trade off between decrease in query accuracy and savings in memory consumption

Query Module: Given a query, the query module employs spatio-temporal pruning techniques to return the final answer.

E. *Log Structured Indexing*

The log structured indexing has a sequence of exponentially increasing sized inverted indices and the contents in those indices are merged periodically.

1) Log Structured Inverted Indices: The Log Structured Inverted Indices (LSII) [5] is a structure for exact real time search on microblogs. The core of LSII is a sequence

of Inverted Indices with exponentially increased sizes such that new microblogs are first inserted into smallest index and later moved into larger indices in batch manner. LSII consists of a set of inverted indices of exponentially increasing sizes(e.g Fig. 4). When a new microblog arrives, it is first inserted into the smallest inverted index I_o. When the number of microblogs in I_o exceeds a certain threshold, the entries of I_o to I_1 are merged and I_o is emptied. During query processing all the inverted indices are scanned simultaneously using the Threshold Algorithm to efficiently compute an answer set. The LSII considerably reduces the amortized update cost per microblog since each microblog can only be involved in a small number of index mergers. The size of I_o is very tiny and hence it incurs overhead. LSII suffers from enormous merging overhead that consumes available thread resources thereby greatly reducing the resources for query processes.

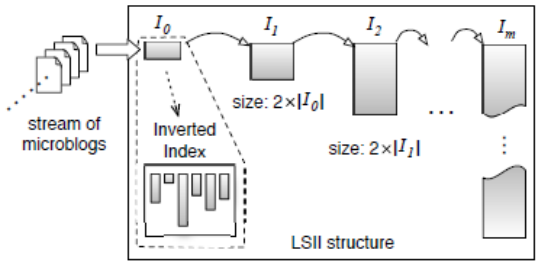

Fig. 4 Architecture of LSII

2) Log Structured Adaptive Merging Strategy(LS-AMS): The LS-AMS[11] Structure consists of an inverted index I_o and a sequence of index packages I_1, I_2..... I_m with exponentially increasing sizes. The I_o is the active index responsible for receiving new microblogs and the others are static indices for queries. The static indices are separated into two parts. I_1 to I_p have several indices and it adopts batch merging strategy. I_{p+1} to I_m has only one index and it adopts direct merging strategy. When an inverted index in the first part becomes full, all the inverted indices of this package are batch merged into a larger index of the next level. When an package in the second part becomes full, it is moved into a larger index of the second level. The P is the boundary between the two parts. The value of P is changeable depending upon the query requests. The LS-AMS is flexible and dynamic. It is capable of varying up to m different structures. When P decreases to 0, LS-AMS is a traditional log-structured index.

F. Compact Indexing

Compact indexing [7] design adopts the interval-based block partitioning scheme. The terms are split into two categories: rare- terms and common-terms based on the list length. Their respective inverted lists are rare-term list and common-term list respectively. Each keyword query can be categorized into three types of queries: (i) rare-term query which contains only the rare-terms, (2) common-term query which contains only the common-terms and (iii) mixed-term query which contains both rare-terms and common-terms. A compact storage mechanism is designed for storing the document id and the related scoring factors to be stored in the inverted list which significantly reduces the size of the index. For query processing in singular mode, cache-conscious data structures for efficient aggregation is used. For query processing in mixed mode(e.g. Fig 5), a novel idea is proposed by treating the rare term as the fancy list which normally consists of entries with large scores. It incurs no additional storage cost and search complexity. Compact Indexing achieves a good trade off between space overhead and search performance. It also reduces the size of inverted lists and save memory consumption. Compact Indexing significantly accelerates the search speed without causing extra space overhead or search complexity. Compact indexing considers efficiency as its primer concern. The experiments in compact indexing was done using 600 million tweets.

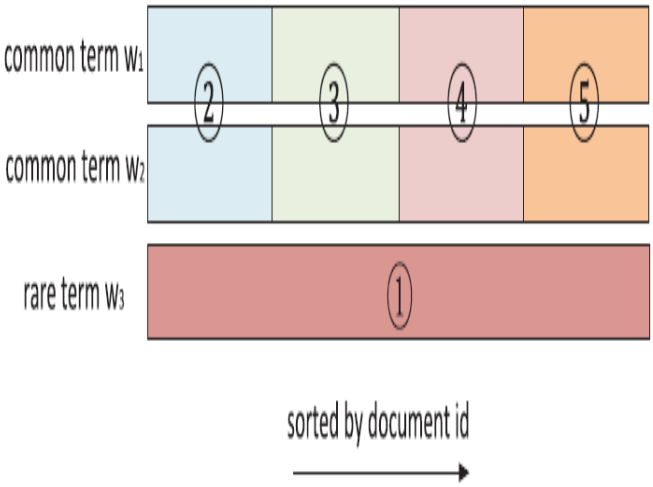

Fig. 5 Block evaluation order of mixed term query processing

IV. COMPARISON OF INDEXING STRUCTURES

Table II summarizes the various types of indexing structures. The methodology, advantages and challenges are also described along with the authors and paper titles that were published in that specific area.

TABLE II
INFORMATION ABOUT THE DIFFERENT INDEXING METHODOLOGIES

Author	Title	Methodology	Advantages	Limitations
Partial Indexing Approach				
C. Chen, F. Li, B. C. Ooi, and S. Wu[4]	Ti: An efficient indexing mechanism for real-time search on tweets	Partial Indexing approach	It improves the update efficiency	The distribution of queries between log and inverted indices could change frequently
Append-only Inverted Index				
M.Busch, *et al*.,[1]	Earlybird: Real-time search at twitter	Append-only Inverted Index	Microblogs are efficiently inserted	It can miss important results for a query if the results are not ranked high enough in terms of time
Provenance-based Indexing				
J. Yao, B. Cui, Z. Xue, and Q. Liu[3]	Provenance-based indexing support in micro-blog platforms	Provenance-based Indexing	Popular topics can be searched easily	Misses some important queries with regard to time and space
Spatio-Temporal based Index				
A.Magdy, *et al*.,[2]	Mercury: A memory-constrained spatio-temporal real-time search on microblogs	Spatio-Temporal based Index	Searching is efficient	Searching is bound to time and space
A.Magdy, *et al*.,[13]	Demonstration of Taghreed: A System for Querying, Analyzing, and Visualizing Geotagged Microblogs	Spatio-Temporal based Index	Searching is efficient	Searching is bound to time and space
A.Magdy, *et al*.,[12]	Towards a Microblogs Data Management System	Spatio-Temporal based Index	Searching is efficient	Searching is bound to time and space
Log-Structured Indices				
L. Wu,W, Lin, X. Xiao, and Y. Xu[5]	Lsii: An indexing structure for exact real-time search on microblogs	Log-Structured Indices with batch merging strategy	It reduces the amortized cost for merging	Merging overhead consumes the resources.
F. zhao, J.Liu, J.Zhou, H.Jin[11]	LS-AMS: An Adaptive Indexing Structurefor Realtime Search on Microblogs	Log-Structured Indices with batch and direct merging strategy(Adaptive)	It is flexible and dynamic	Comparatively search time is more
Compact Indexing				
D.Zhang, *et al*.,[7]	Compact Indexing and Judicious searching for Billion scale microblog retrieval	Compact Indexing	Smaller index, search time is faster	Too many processing to be done

V. FUTURE RESEARCH DIRECTIONS

Based upon the extensive survey conducted on the indexing structures for microblogs, we have come to the following inferences. Each indexing structure has its own limitations such as incomplete query results, higher processing time, consuming more resources, widespread storage usage and more searching time. The above said are some of the challenges faced in developing the indexing structure in microblogs which has to be addressed by the scientific community. A real-time indexing structure is required addressing the above said challenges.

VI. CONCLUSION

The usage of microblogs is emerging in a higher rate today. Managing the incoming data in microblogs is a challenging task since the microblogs are generated in an enormous speed and it continues to increase daily. In spite of availability of larger number of indexing techniques for microblogs, it is a challenging task since there are few pitfalls in each technique. The drawbacks arises from different arena such as the importance given to time and space, the importance given to the popularity of the topic, storage complexities, merging overhead and processing time. In the present survey, six important indexing techniques applied for microblogs are enlightened shortly and it gives better understanding about the indexing techniques. Due to the increasing rate in which the microblogs are generated, it is very hard to achieve a generic indexing method. This makes indexing methodology a demanding research area and an active area of focus to develop a novel approach.

REFERENCES

[1] M. Busch, K. Gade, B. Larson, P. Lok, S. Luckenbill, and J. Lin,"Earlybird: Real-time search at twitter," in Proc. 28th IEEE Int.Conf. Data Eng., 2012, pp. 1360–1369.

[2] A. Magdy, M. F. Mokbel, S. Elnikety, S. Nath, and Y. He, "Mercury: A memory-constrained spatio-temporal real-time search on microblogs," in Proc. 30th IEEE Int. Conf. Data Eng., 2014, pp. 172–183.

[3] J. Yao, B. Cui, Z. Xue, and Q. Liu, "Provenance-based indexing support in micro-blog platforms," in Proc. 28th IEEE Int. Conf. Data Eng., 2012, pp. 558–569.

[4] C. Chen, F. Li, B. C. Ooi, and S. Wu, "Ti: An efficient indexing mechanism for real-time search on tweets," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 649–660.

[5] L. Wu,W, Lin, X. Xiao, and Y. Xu, "Lsii: An indexing structure for exact real-time search on microblogs," in Proc. 29th IEEE Int. Conf. Data Eng., 2013, pp. 482–493.

[6] Yuan, Wang and Ding, " A Real-time search structure and classification algorithm of microblog based on partial indexing," in TELKOMNIKA Indonesian Journal of Electrical Engineering,2014, pp. 2271-2277

[7] D.Zhang, L. Nie, H. Luan, k.Tan, T. Chua and H. Shen, " Compact Indexing and Judicious searching for Billion scale microblog retrieval," in ACM Transactions on Information Systems, Vol 35, No.3, Article 27, 2017

[8] Y. Demchenko, Laat and P. Membray,"Defining architecture components of the bigdata ecosystem," in IEEE, pp. 104-112

[9] A.Magdy, R. Alghamdi and F. Mokbel, " On main-memory flushing in microblogs data management systems," in ICDE 2016 conference, pp. 445-456

[10] H. Huang, J.Li, R. Zhang, W.Yu and W. Ju, " LiveIndex: A distributed online index system for temporal microblog data," in IEEE 2015, pp. 884-887

[11] F. zhao, J.Liu, J.Zhou, H.Jin, "LS-AMS: An Adaptive Indexing Structurefor Realtime Search on Microblogs," in IEEE transactions on big data, vol. 1, No. 4, 2015, pp. 125-137

[12] A. Magdy, F. Mokbel, "Towards a Microblogs Data Management System," in 16th IEEE international conf. on mobile data management, 2015, pp. 271-278

[13] A. Magdy, F. Mobel, et al., "Demonstration of Taghreed: A System for Querying, Analyzing, and Visualizing Geotagged Microblogs," in ICDE conference, 2015, pp. 1416-1419.