Volume 4, Issue 9, September 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Hash Function Approach Supporting Search Functionality in the Context of Big Data within Social Network

Sweata Mishra[*]
School of computing science
Kaziranga University,
Assam, India

Bhardwaj Choudhury
School of computing science
Kaziranga University,
Assam, India

Sandeep Rakshit
School of computing science
Kaziranga University,
Assam, India

Abstract— Big data consists of huge amount of unstructured data, which is difficult to analyse. To manage this data, we present a simple and efficient hashing scheme that we have applied in social network data, which is a major source of big data. We have used a novel and theoretically easy to understand scheme of chaining to remove collision, which may be the outcome while storing these giant data. Our main contribution is that we have created dimensions of all the data stored to control format of data display. We have also described the design and implementation of an efficient algorithm to conduct search operation on these data.

Keywords— Hashing, Chaining, Big data, Social Network, Searching

I. INTRODUCTION

Algorithms for handling big data have successfully produced numerous ways for managing and sequencing large amount of data. Primary objectives of these algorithms are to effectively handle large-scale data, extract actionable patterns, and gain insightful knowledge. These algorithms involves various methods such as genetic algorithm, support vector machines, self organising maps, decision tree, neural network and cluster analysis, to disclose the hidden patterns inside the large data set.

Large amount of unstructured data are difficult to collect, manage, store, analyze, predict, visualize and model. The described method can expand researchers' capability of understanding new phenomena due to the use of social media and another source of data and improve decision making and business intelligence to provide better services and develop innovative opportunities [1]. Data was analyzed using software Hive and Hadoop. For the analysis of data with different formats, cloud structure was laid. [2]

Then there arrived the prime need to analyze the unstructured data, so the Hadoop framework was being laid for the analysis of the unstructured large data sets.

In this paper, we have focused on managing social data, which is one of the major source of big data. Social media is defined as the group is of Internet-based applications that build on the ideological and technological foundations of Web 2.0 and that allow the creation and exchanges of user-generated content [4]. Social media is a collection of different types of social sites such as Facebook, Twitter, Newspaper, radio etc. According to the site *thenextweb.com*, Indian citizens spend one in four minutes online using social networking sites, more than any other Internet activity [5].

These data are likely to be expanded with more content in the future. Hence it will be difficult for users and service provider to manage and handle these massive user-generated data and chances of collision also increases, hence we have used hash function to remove collision at some extent. Following are some steps which can be used to manage and handle big data.[6]

1. Create dimensions of all the data stored.
2. All the dimensions should have keys using hash algorithm ensuring uniqueness.
3. All structured and unstructured data should be integrated.
4. Different technologies should be used to deal with different format of data.

II. RELATED WORK

In paper [7], authors proposed hashing to facilitate efficient kernel and give deviation bounds from the exact kernel matrix. This has applications to estimation on strings and graphs. Paper[8] present an efficient EPH algorithm for very large key sets. It also demonstrate the scalability of EPH algorithm by constructing minimum perfect hash functions for a set of 1.024 billion URLs . In paper[9], authors proposes a novel algorithm for constructing minimal perfect hash function for huge set of keys. The algorithm is based on classical divide and conquer technique. n paper [10] the author discusses about the traditional databases and the databases required with Big data concluding that the databases don’t solve all aspects of the Big data problem and the machine learning algorithms need to be more robust and easier for unsophisticated users to apply

III. DATA MODEL

The relatives' data will be stored in a sparse, distributed, persistent, multi dimensional sorted map. The map consist of a row and 'n' numbers of column. The URL of site will be the row key and different aspects of the page will be the column names, which stores the content of the web page

ROWS:

Each of the rows consists of a row key i.e. the URL of the page itself. Every read and write operation in a single row key is automatic, regardless of the number of columns. Whenever a client opens a social site, he will be entering a particular row where the key is URL of the page. Through that row, client will be able to access each of the columns.

COLUMNS:

All the data stored comes under column family. There can be any number of columns in a web page, as the information in a site is not limited. Each of the columns has a unique key .i.e. column key. A table can have unbounded number of column, our intent is to that the column families be small, to understand it in better way.

We have considered a social networking site ytalk.com, which is the row key for the table. Here, messages, profile picture tags, etc are the column key that will store data given by user.

Each of the cells contains multiple versions of same data. A hashing function H(X) transforms key value to an address, and since this transformation may produce same key for different address space, hashing function may lead to collision in address space[7][8]. There will be collision if for example, for same email, two messages arrives at same time. Applications that need to avoid collision must generate unique timestamp. We have used chaining technique throughout for collision removal.

www.ytalk.com	**Message**	**Profile picture**	**Tags**	**………**
www.abc.com	……..	………	……..	…………

Fig1: Data model in dimension showing row key n column key

IV. COLLISION REMOVAL USING HASHING:

Collisions are obvious in this type of table, it is also unavoidable. To handle these collisions table implementation has some collision resolve strategies.

All these methods require that keys be stored in the table, together with associated values [9]. Some of the methods of collision removal are [10]:

a) Separate chaining.
b) Open addressing.
c) Coalesced hashing
d) Cuckoo hashing.
e) Robin Hood hashing.
f) 2-choice hashing.
g) Hopscotch hashing.

We have confined our study in separate chaining, as it require less time to search the content of table.. Separate chaining involves a design in which the table is an array of references to linearly linked lists.

If some new data record hashes to a value in the table that is already occupied, the new record is inserted into the linked list pointed by the table location .Chaining hash table with linked list requires basic data structure and some algorithm

We have assumed social network data for hashing. If we consider messages coming to a particular id, which results in collision. For handling such type of situation we have used separate chaining method as shown in Figure2. For separate chaining, there will be an additional memory outside the table to hold the colliding data.

In fig2, we have described how incoming messages are arranged in proper place. Whenever two messages arrive at the same time, the collision is resolved using separate chaining method, where the next message is stored into an additional storage space using link to that location.

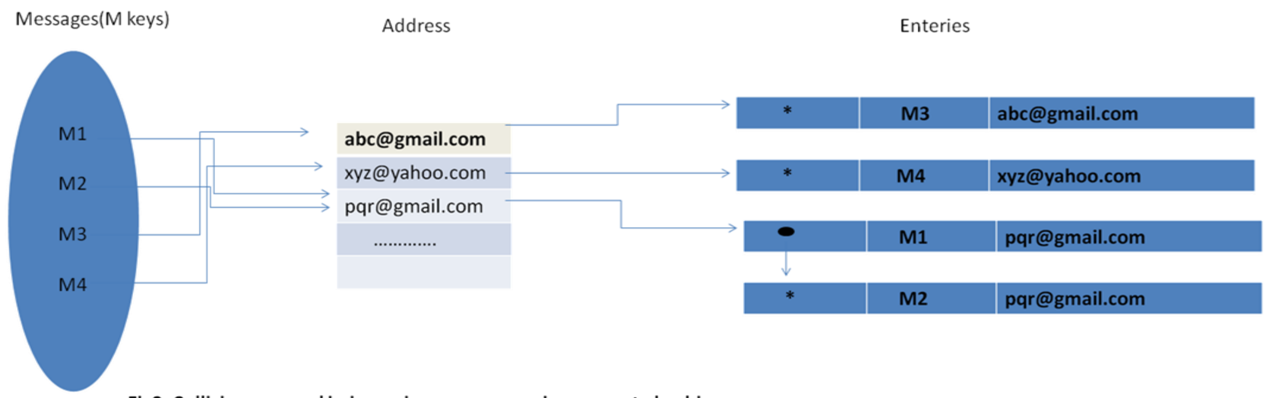

Fig2: Collision removal in incoming messages using separate hashing.

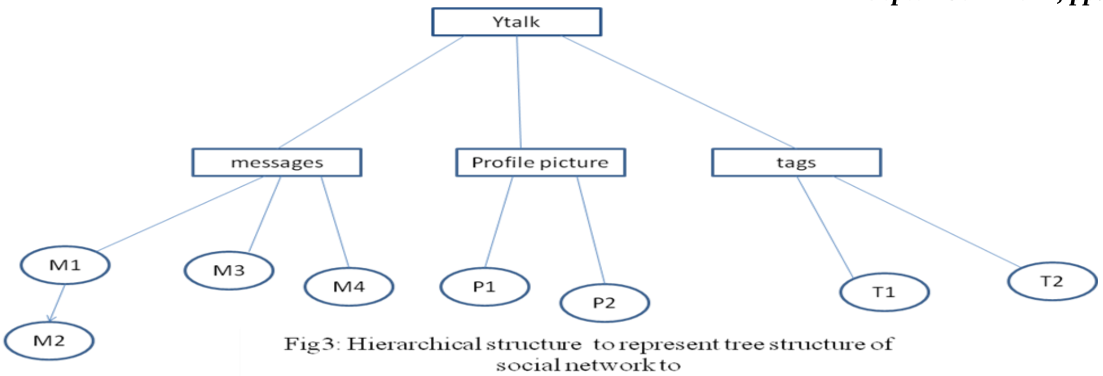

Fig3: Hierarchical structure to represent tree structure of social network to

Accordingly, we can get an hierarchical structure as shown in figure3. We have created a tree for the above example of social network data. The configuration will be as follows:

```
{type:"char", value:"ytalk"},
{type:"branch", list:                              [
  {type:char,value:" message"},
  {type:"value",key:"m1",data:{pqr{type:key.pqr@gmail.com,key=m1}}},
  {type:"value",key:"m2",data:{pqr{type:key.pqr@gmail.com,key=m2}}},
  {type:"value",key:"m3",data:{abc{type:key.abc@gmail.com,key=m3}}},
  {type:"value",key:"m4",data:{xyz{type:key.xyz@yahoo.com,key=m4}}},
  ],                                             [
  {type:char,value:"profile"}
  {type:"value". key:"p1",data:{pqr{type:"key.pqr@gm
  ail.com",key=p1}}},
  {type:"value",key:"p2",data:{abc{type:"key.abc@gmail.com",key=p2}}}
   ],
   [
   {type:char,value:"tag"}
  {type:"value",key:"t1",data:{abc{type:"key.abc@gmail.com",key=t1}}}
  {type:"value",key:"t2",data:{pqr{type:"key.pqr@gmail.com",key=t2}}}
  ]
   ]}
```

Fig4: Configuration of hierarchical structure shown in Fig3

Here, we have created a tree with root ytalk(url), below which we have branches like messages, profile tag etc. In this way, data can be grouped. The first level represent the data in groups and in second level, these data are split according to their arrival.

V. SEARCHING IN TREE

Social network data can be easily represented using graph[12]. Such a graph can be analysed using tree like structure. For searching any content of social data, we can search tree using BFS(Breadth First Search)[13]. BFS explores nodes nearest to the root first, before any other node. For the above example of social network data, the nodes will be visited in following order:

a) The root will be visited first, i.e. site will be opened first.
b) Then the next level nodes will be visited, .i.e., site will be beaked into messages, profiles, tags etc.
c) After searching second level, third level nodes will be visited, i.e. each group of message is broken to simplify the

search.

The tracing path will be as follows:

In this way, following the tracing path, we can search the data. The nodes can be traced, until we get the data is open.

```
1.OPEN=[ytalk] ;CLOSED=[ ]
2.OPEN=[message, profile, tag] ; CLOSED=[ytalk]
3.OPEN=[profile,tag,m1,m3,m4]; CLOSED=[message, ytalk]
4.OPEN=[tag,m1,m3,m4,p1,p2]; CLOSED=[profile,message,ytalk]5
5.OPEN=[m1,m3,m4,p1,p2,t1,t2];
CLOSED=[tag, profile,message,ytalk]
6.OPEN=[m3,m4,p1,p2,t1,t2,m2]; CLOSED=[m1,tag,profile,message,ytalk]
```

Fig5: Tracing path for hierarchical structure shown in Fig3.

VI. CONCLUSIONS

This paper presented a novel technique to handle large data and collision free storage and retrieval of data. we have described a distributed structure for storing big amount of data of social network. There are many advantages of using this system. The most efficient advantage is that, it has reduces the chance of collision, which may cause data loss. Using the above structure will also be efficient to retrieve any object.

A simple method of separate chaining is used for collision removal. we have also created a hierarchical structure and described how to implement that structure as well as the tracing path.

we are also planning to add several additional features in the structure, such as handling cross data for multiple masters as our future work.

REFERENCES

[1] Gundecha and Liu" Mining Social Media"- A Brief Introduction Tutorials in Operations Research, c 2012 INFORMS.

[2] Edmon Begoli, James Horey, “Design Principles for Effective Knowledge Discovery from Big Data” Joint Working Conference onSoftware Architecture & 6th European Conference on Software Architecture, 2012

[3] Kapil Bakshi, “Considerations for Big Data: Architecture and Approach”, IEEE, 2012

[4] A. M. Kaplan and M. Haenlein. Users of the world, unite! The challenges and opportunities of social media. Business Horizons 53(1):59-68, 2010.

[5] Josh Ong (August 2012), article retrieved from http://thenextweb.com/in/2012/08/20/social-networkingites-now-occupy-25-online-time-india/

[6] Avita katal, Mohammad Wazid, R H Goudar,"Issues, challenges, tools And god practices" 978-1-4799-0192-0/13/$31.00 ©2013 IEEE.

[7] Q.S.J. Petterson, G.Dror, J.L.A Samola, S.V.N Vishwanathan,"Hash Kernel For Structured Data" Journal of machine learning research 11(2009), published 06-09.

[8] F.C.Botelho, N.Ziviani,"External perfect hashing for very large key sets"CIKM’07, November 6–8, 2007, Lisboa, Portugal.

[9] F.C.Botelhlo, Y.Kohayakawa, N.Zivani,"An Approach for Minimal Perfect Hash Functions for Very Large Databases" web document http://cmph.sourceforge.net.

[10] Sam Madden, “ From Databases to Big Data”, IEEE, Internet Computing, May-June 2012.

[11] Donald E. Knuth, The art of computer programming, volume 3: (2nd ed.) sorting and searching, Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 1998

[12] W. D. Maurer , T. G. Lewis, Hash Table Methods, ACM Computing Surveys (CSUR), v.7 n.1, p.5-19, March 1975 [doi>10.1145/356643.356645]

[13] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2009). Introduction to Algorithms (3rd ed.). Massachusetts Institute of Technology. pp. 253–280. ISBN 978-0-262-03384-8.

[14] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). "Introduction to Algorithms (2nd ed.)". MIT Press and McGraw-Hill. 221–252. ISBN 978-0-262-53196-2.

[15] Timothy M.Dietrich, Nicholas J.De lillo," Implementing Hashing Using Separate chaining Using Java" school of computer science and information system, Pace university, number 185, 2003.

[16] S. Mishra, R. borboruah, B. Choudhury, S. Rakshit,"Modelling of Social Network Using Graph Theoretical Approach" Micro-2014 Vol.1 pp:137-14 ISBN:81-85824-46-0

[17] A.Kaur, P.Sharma, A.Purva,"A appraisal paper on Breadth first search, Depth-first search and Red black tree" International journal of scientific and research publication, vol4, march 2014 ISBN:2250-3153.