International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

Query Algorithm to Reduce the Time Overhead of Similarity Detection

P. Dileep Kumar Reddy & G. Sai Prasanna
Lecturer, Department of CSE JNTUA College of Engineering, Anantapur Andhra Pradesh, India.

Department of CSE JNTUA College of Engineering, Anantapur Andhra Pradesh, India.

dileepreddy503@gmail.com & saiprasannagaddam@gmail.com

Abstract:

Data storage is a big issue based on retrieval of bulk amount of data from different sources. Cloud broker layer is added as an additional layer and it plays a major role in between cloud service providers. Previous approaches do not provides the features like completeness e.t.c. of the service selection recommendations to the cloud clients. MMB cloud tree technique provides a solution to overcome this problem. This approach checks the accuracy of the cloud brokers throughout the service selection method. The file similarity cannot be recognized by this approach. To overcome this problem, we introduce Enhanced position aware sampling algorithm. EPAS is used to spot the similar files in the cloud. This approach demonstrates a query algorithm which decreases the time overhead of similarity detection. Therefore, it is a best approach to recognize the similarity in the cloud.

Keywords: *Query Algorithm, Deduplication, Traditional Sampling Algorithm, Position Aware Sampling Algorithm, Enhanced Position Aware Sampling Algorithm.*

1. INTRODUCTION

Cloud service provides a various advantages like high storage space and high computational capabilities e.t.c. With the increasing of data, the risk and cost of data management are significantly increasing. In order to address this problem, data owners outsource their data to the cloud and access the data via internet. This leads to have large volume of redundant data [1] in the cloud. Therefore, Data deduplication is required to overcome these problems.

Data deduplication [2], [3], [4], [5], [6], [7] technique computes a single fingerprint for every information block by using algorithms such as MD5 and SHA-1. If the calculated fingerprint [8], [9] is already within the database, the information block doesn't store another copy, a pointer can be inserted to the first instance in place of the duplicated block. By doing so, data deduplication [10], [11] removes redundant data and stores a single copy. Disk bottleneck [12] is a major issue arises in the process of fingerprint search. This is because huge number of requests going to disk drives and generates a large volume of random disk [13] access which significantly decreases the throughput. Similarity based data deduplication techniques are used eliminate the redundant data. Similar characteristic values are only stored instead of storing all fingerprint indexes in memory.

The rest of the paper is organized as follows: Section 2 presents the Related work. Section 3 discusses the proposed approach. Section 4 presents the Performance Evaluation of the proposed approach. Finally Section 5 concludes the paper.

2. RELATED WORK

Our related work involves MMB cloud tree technique and similarity detection algortihtms to avoid redundant data.

2.1. MMB ^{CLOUD}-TREE: MMB cloud tree [14] technique allows clients to check the accuracy of cloud brokers answers. It is specifically tailored for cloud service selections. MMB cloud-tree technique is a best approach to select the best service and it is used to reduce the burden on the client side. Cloud Service Selection verification(CSSV) [15] scheme employs a idea of "partitioning of duties" which provides more security. Database construction, service selection, and results verification are the three phases involves in this approach. Collector works efficiently in cloud brokerage system [16]. The collector is an important layer and it does not communicate with other layers in cloud environment. It is only responsible for gathering the information from various services providers and filters the suspected data to implement an authenticated database of CSP's profiles. Duplicate files cannot be deleted in this approach

2.2. SHINGLE: Andrei et.al. used a mathematical method to convert the similarity detection problem into a set operation problem. This approach is usually called Shingle algorithm. The Shingle algorithm is useful to recognize similar web pages. The time overhead of Shingle algorithm extends with the expansion of file size . Therefore, it is not really favourable for large files.

2.3. SIMHASH: Charikar proposed a Simhash algorithm. Simhash algorithm works efficiently in recognizing the similar web based documents. It is completely different to traditional hash function techniques whose signature values are discrete.. On the contrary, Simhash has the property that the fingerprints of similar files only differ in a small number of bit positions. It can map a file into f-bit fingerprints. After transferring the source into m-bit by using Simhash algorithm, we can detect the similarity through calculating the hamming distance between two fingerprints. These approaches takes more time to evaluate the detection of similar files.

3. PROPOSED APPROACH

We design sampling based similarity approaches to detect data similarity and to perform upload and deletion of files.

SAMPLING BASED SIMILARITY IDENTIFICATION: In order to improve the detection of file similarity with low overhead and high accuracy , we design EPAS. Following are the prescribed algorithms.

3.1. TRADITIONAL SAMPLING ALGORITHM (TSA): TSA does not read whole files, but samples required information blocks to compute the fingerprints as similarity characteristic values. TSA is simple and it has a fixed overhead. TSA is explained in Algorithm 1. We sample N information blocks from file A, inject each information block sizing Lenc to a hash function. We then obtain N fingerprint values that are collected as a fingerprint set $Sig_A(N,Lenc)$. By analogy, we will have a fingerprint set $Sig_B(N,Lenc)$. Then the degree of similarity can be calculated between file A and file B. It is very sensitive to file modifications. A small modification would cause the sampling positions shifted, thus resulting a failure. It provides less overhead in compared to Shingle, Simhash and Traits algorithms.

Algorithm 1: TSA Algorithm

```
Data: fd, N, Lenc, T
1 begin
2     LenR = (FileSize - Lenc*N)/(N - 1)
3     for i← 1 to N do
4         offset = (i - 1)*(Lenc + LenR)
5         lseek(fd, offset, SEEK_SET)
6         read(fd, buf, Lenc)
7         Md5(buf, Lenc, Md5Val)
8         put(Md5Val, SigA)
```

Algorithm 1-TSA Algorithm

3.2. Position-Aware Similarity Algorithm (PAS): TSA is sample and effective. One bit modification would end in a failure of similarity

detection. PAS is introduced to provide a solution for this problem. PAS catches more real sampling positions than that of TSA, if we compare the sampling positions achieved by PAS against the positions offset by the optimal algorithm. It can avoid the shifting of sampling positions generated from slight file modifications in the middle and the end of the source files. In addition, if it is applied to data depulication system.

Algorithm 2: PAS Algorithm.

```
Data: fd, N, Lenc, T
1  begin
2      FileSize = (FileSize/T)*T
3      LenR = (FileSize - Lenc*N)/(N - 1)
4      LenR = LenR > 0 ? LenR : 0
5      for i← 1 to N - 1 do
6          offset = (i - 1)*(Lenc + LenR)
7          lseek(fd, offset, SEEK_SET)
8          read(fd, buf, Lenc)
9          Md5(buf, Lenc, Md5Val)
10         put(Md5Val, SigA)
11     lseek(fd, -Lenc, SEEK_END)
12     read(fd, buf, Lenc)
13     Md5(buf, Lenc, Md5Val)
14     put(Md5Val, SigA)
```

Algorithm 2- PAS Algorithm

3.3. Enhanced Position-Aware Similarity Algorithm (EPAS): EPAS maintains the advantages of both PAS and TSA. Therefore, EPAS is planned to alleviate this drawback by enhancing PAS. EPAS maintains the benefits of each PAS and government agency. EPAS respectively samples N/2 information blocks from the head and the tail of the modulated file. It then maps these data blocks into fingerprints by utilizing hash functions and obtains the similarity characteristic value. N can be taken as a fixed value and it is lesser than file size. Since EPAS only samples and calculates N information block fingerprints, the tie and computation complexity of EPAS are O(1).

Algorithm 3: EPAS Algorithm.

```
Data: fd, N, Lenc, T
1  begin
2      FileSize = (FileSize/T)*T
3      LenR = (FileSize - Lenc*N)/(N - 1)
4      LenR = LenR > 0 ? LenR : 0
5      for i← 1 to N/2 do // Sampling from file
       head
6          offset = (i - 1)*(Lenc + LenR)
7          lseek(fd, offset, SEEK_SET)
8          read(fd, buf, Lenc)
9          Md5(buf, Lenc, Md5Val)
10         put(Md5Val, SigA)
11     for i← 1 to N - N/2 do // Sampling from file
       tail
12         offset = (i - 1)*(Lenc + LenR)
13         lseek(fd, -(offset + Lenc), SEEK_END)
14         read(fd, buf, Lenc)
15         Md5(buf, Lenc, Md5Val)
16         put(Md5Val, SigA)
```

Algorithm 3-EPAS Algorithm

4. PERFORMANCE EVALUATION

Efficiency of EPAS algorithm can be verified by comparing with existing algorithms like Shingle, Simhash and Traits.

EPAS ALGORITHM EVALUATION: In this section, we have a tendency to value the time overhead, query time, memory and C.P.U utilization, preciseness and recall of EPAS against the well-known similarity detection algorithms like Shingle, Simhash and Traits. The Lenc, N, and a couple of are set as 32 B, 4, and 0.5, respectively. According to the work, hamming distance is chosen as three, and therefore the variety of hold on tables is decided as four. All the measurements during section are performed with information set D1and D2 [17] . In order to decrease the storage consumption, EPAS algorithmic rule uses 8 bits to store a fingerprint. Therefore, it takes thirty two bits for every file. However, the redundant tables of Simhash would like 256 bits to store the fingerprints of every file. Our

experimental results describes that the EPAS considerably performs the present documented algorithms in terms of time overhead, central processing unit and memory occupation.

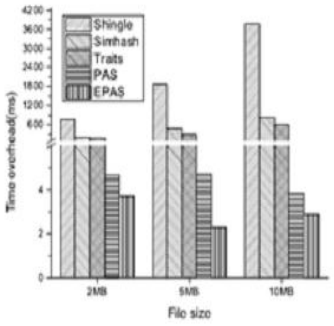

Fig. 1: The time overhead of EPAS, Shingle, Simhash, Traits and PAS algorithm with different file size 2MB, 5MB and 10MB

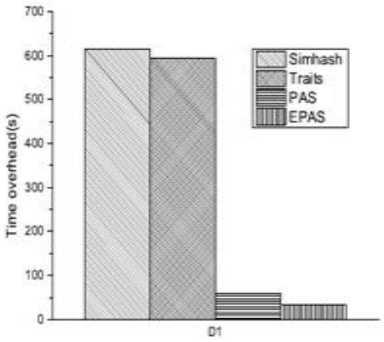

Fig. 2: The time overhead of Simhash, Traits, PAS and EPAS algorithm with data set D1.

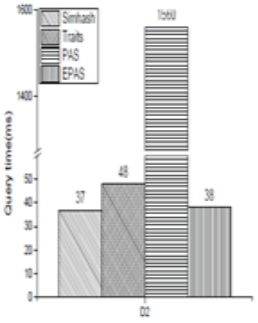

Fig. 3: The query time of Simhash, Traits, PAS and EPAS algorithm with data set D2.

The time overhead is calculated with three different file size including 2MB, 5MB, and 10MB. Fig 1 shows that EPAS takes less overhead compared to Shingle, Simhash, Traits and PAS. Fig. 2 investigates the time overhead with a real data set D1. It shows the same trend as Fig. 1. Fig. 3 demonstrates the query time of simhash, Traits, PAS and EPAS. EPAS only stores single copy of similarity values in memory. This method reduces the time overhead of querying and comparing. Therefore, the performance of EPAS is enhanced.

5. CONCLUSION

Enhanced Position-Aware Sampling algorithm (EPAS) can be proposed to spot the file similarity in the cloud environment. Various experiments are performed to evaluate the performance of EPAS. Corresponding analysis and discussion of the parameter selection are introduced in our approach. The evaluation of precision and recall demonstrates that EPAS is very effective in detecting the file similarity in contrast to Shingle, Simhash, Traits and PAS. The experimental results also suggest that the time overhead, C.P.U and memory occupation of EPAS are much less than that of those algorithms. Therefore, we believe that EPAS may be applied to the cloud environment to decrease the latency and achieve both the efficiency and accuracy.

6. REFERENCES

[i] J. Gantz and D. Reinsel, "The digital universe decade-are you ready," *IDC iView*, 2010.

[ii] H. Biggar, "Experiencing data de-duplication: Improving efficiency and reducing capacity requirements," The Enterprise Strategy Group, 2007.

[iii] D. Bhagwat, K. Eshghi, D. D. Long, and M. Lillibridge, "Extreme binning: Scalable, parallel deduplication for chunk-based file backup," in Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on. IEEE, 2009, pp. 1–9.

[iv] D. R. Bobbarjung, S. Jagannathan, and C. Dubnicki, "Improving duplicate elimination in storage systems,"

ACM Transactions on Storage (TOS), vol. 2, no. 4, pp. 424–448, 2006.

[v] D. T. Meyer and W. J. Bolosky, "A study of practical deduplication," ACM Transactions on Storage (TOS), vol. 7, no. 4, p. 14, 2012.

[vi] B. S. Baker, "On finding duplication and near-duplication in large software systems," in Reverse Engineering, 1995., Proceedings of 2nd Working Conference on. IEEE, 1995, pp. 86–95.

[vii] W. Xia, H. Jiang, D. Feng, and Y. Hua, "Silo: a similarity-locality based near-exact deduplication scheme with low ram overhead and high throughput," in Proceedings of the 2011 USENIX conference on USENIX annual technical conference. USENIX Association, 2011, pp. 26–28.

[viii] Y. Zhou, Y. Deng, and J. Xie, "Leverage similarity and locality to enhance fingerprint prefetching of data deduplication," in Proceedings of The 20th IEEE International Conference on Parallel and Distributed Systems. Springer, 2014.

[ix] L. Song, Y. Deng, and J. Xie, "Exploiting fingerprint prefetching to improve the performance of data deduplication," in Proceedings of the 15th IEEE International Conference on High Performance Computing and Communications. IEEE, 2013.

[x] G. S. Manku, A. Jain, and A. Das Sarma, "Detecting nearduplicates for web crawling," in Proceedings of the 16th international conference on World Wide Web. ACM, 2007, pp. 141–150.

[xi] Y. Hua, X. Liu, and D. Feng, "Data similarity-aware computation infrastructure for the cloud," IEEE Transactions on Computers, p. 1, 2013.

[xii] B. Zhu, K. Li, and R. H. Patterson, "Avoiding the disk bottleneck in the data domain deduplication file system." in *Fast*, vol. 8, 2008, pp. 1–14.

[xiii] Y. Deng, "What is the future of disk drives, death or rebirth?" *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 23, 2011.

[xiv] Jingwei Li, Anna squicciarini, Dan Lin, Smitha sundareswaran, and chunfu jia "MMB*cloud*-tree: Authenticated Index for Verifiable Cloud Service Selection, IEEE transactions on Dependable and secure computing, vol. 14,no. 2,pp. 185-198,March 2017.

[xv] Z. ur Rehman, O. K. Hussain, S. Parvin, and F. K. Hussain, "A framework for user feedback based cloud service monitoring,"in 2012 Sixth International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), 2012, pp. 257–262.

[xvi] S. Sundareswaran, A. Squicciarini, and D. Lin, "A brokerage-based approach for cloud service selection," in 2012 IEEE 5th International Conference on Cloud Computing (CLOUD). IEEE, Aug. 2012, pp. 558–565.

[xvii] G. Linden, "Make information helpful," http://home.blarg.net/_glinden/StanfordDataMining.2006-11-29, ppt, 2006.

.