International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 6, Issue 8, August 2017 **ISSN 2319 - 4847**

Data Deduplication – Overview and Implementation

Somefun Olawale Mufutau[1], Nwala Kenneth[2], Okonji Charles[3] , Omotosho Olawale Jacob[4]

[1]Computer Science Department Babcock University, Ilisan Remo Ogun State, Nigeria

[2]Second Author Affiliation with address

Abstract

It is noticeable globally that the rate at which information is stored and transferred has improved greatly. With computers and other multimedia in wide use and the explosion of the internet, the amount of data available exploded as well and there is also massive data transfer among individual and networks. Therefore there is need to think of how to save cost of buying storage devices and find a way of transferring information with less cost and reduce network bandwidth. Data deduplication is technique for eliminating duplicate copies of repeating data; it is more intelligent and specialized data compression. This technique is used to improve storage utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent. This paper depicts various kind of data deduplication available. This paper depicts various kind of data deduplication available. The objective is to present enough detail about data deduplication to enable application and implementation of file-level deduplication. The system designed is a file-level (or single instance storage), post process deduplication to identify files that have similar content in the storage, state their location or path. The user can select the storage device to search. It is designed on windows 8 using java programming language and netbeans 8.0 IDE. With this system deployed it makes it possible to easily identify and remove redundant in repository. Further work is to form GUI for chunk or block-level deduplication in order to be able to select the threshold value for the block or chunk.

Keywords: Deduplicatio,; Redundant, File-level deduplication, Block-level Deduplication, Hashing.

1.Introduction

Since information is growing geometrically, it advisable to employ deduplication in order to reduce the amount of storage needed, save cost and network bandwidth. Storing large amount of data efficiently in terms of both space and time is of paramount concern. Sometimes duplicate needs to be identified and removed or store with pointer or the differences. There are many techniques of deduplication, each with its own advantages and disadvantages. Deduplication can be performed at file level or the file can be broken down into chunks or blocks. There some algorithms to identify duplicated copies and replace them with pointers in the storage. There are some that indicate the level of similarity of one file with the other. The purpose or usage determines which algorithm to use.

2.Literature review

For storage management, many sorts of technologies and concepts emerges continuously, data compression, increment backup, delta encoding and data deduplication.

Data Compression [1] is a technology that reduces user storage space. Data compression can be loissy or lossless data compression. It transforms the expression of string into another kind of expression, which contains the same information but is as shorter as possible in size. By decreasing the file size, we can save more disk space to store more documents.

Types and methods of Deduplication

Deduplication can be categorized into file-level, and block or chunk level [2][3]. Location can also be used as classification; it is called source-based deduplication if data are deduplicated at the client and the particular data is sent without duplication if otherwise it known as target-based. When the client first harsh each data segment before upload and send the results to verify if such data already exist it is called source-based deduplication. Client-based deduplication is vulnerable to attack although it achieves bandwidth saving [4].

File Level Deduplication

File level deduplication is deduplication where the whole file is considered. It is also called Single Instance Storage (SIS) [5][2][6].The file-level deduplication can be performed most easily and the chances of collision are low. Although it achieve high lookup efficiency but when a byte changes, the whole file appears different..

Block or Chunk Level Deduplication

This is when file is divided into chunks or blocks and the individual chunks or block is considered for deduplication by comparing their fingerprints [7]. If the fingerprints which are hash values computed by hash function are found the same with the one already in the repository, pointers are used to indicate the data chunk already in existence. The match frequency is dependent on the chunk size and the threshold used; the amount of data that must be stored or transferred can be greatly reduced. Each chunk of data is assigned identification, calculated by the software, typically using cryptographic hash functions. Block deduplication requires more processing power than the file deduplication, since the number of identifiers that need to be processed increases greatly [8]. The unit of data being process is a "chunk". Chunk may be defined by physical layer constraint or by comparing the size of the entire file or by finding the pattern that occurs within internal file boundaries through sliding the reference window along the file stream. Chunk size is determined by suitable algorithm. Since each chunk is to be hashed individually, this type of deduplication requires more processing power and there is tendency of hash collision. Chunking can be fixed or variable [9][10][11]. This is common to disk-based backup technology [8].

Fixed-Length or Fixed Block

File is broken into segment, or block or chunk of fixed length. Reference window of fixed size may be used to look for the chunk. The block may be of the fixed size like 4KB, or 8KB, 32 etc [6][8].

Variable-Length or Variable Block

File is divided into variable length block or chunk. Variable length window is used for deduplication[6][8].

Content Defined Chunking (CDC) algorithm

In Content Defined Chunking (CDC) algorithm [12] the incoming data stream is broken into chunks when the content window before the breakpoint meet-up with condition stipulated. CDC algorithm is used to divide the data stream into chunks; fingerprint is computed by chunk fingerprinting (example, SHA-1 value) for each of the chunks. Fingerprint indexing and querying stores the fingerprints according to a certain data structure for indexing and queries the index to find chunks of identical fingerprints. Data storing and management stores new chunks and representative reference pointers for the duplicated chunks. The sliding-window-based CDC algorithm [13] have been the dominating CDC algorithms though is not optimal, especially in terms of deduplication performance, since it has to slide the content window byte by byte.

Two Threshold Two Divisor (TTTD)

Another most frequently used variable-length chunking algorithm [14]. The TTTD chunking method ascertains that chunks smaller than a particular size are not produced. But the chunks produced might escape duplicate detention as larger chunks are more likely to be related than smaller ones. The method ignores the chunk boundaries after a minimum size is reached. TTTD applies two techniques where two divisors (D, the regular divisor and D0, the backup divisor) are used. By applying these divisors, TTTD guarantees a minimum and maximum chunk size. A minimum and a maximum size limit is used for splitting a file into chunks for searching for duplicates.

Byte-level deduplication

Chunks are compared byte by byte and this makes it more accurate although it takes time [15][16].
Post-process deduplication

Target deduplication

Data is stored on the storage device and then process at any time deemed fit for deduplication[17]

In-line deduplication

Deduplication is done before data are stored in the storage device. Although it requires more computing power but it manages space. Hash calculations are created on the target device as the data enters the device in real time. [18].

This is when deduplication occurs near where the data is finally stored or transferred to.

Network data deduplication

This happen in the Network to reduce the number of bytes that must be transferred between nodes or terminals. This can reduce the bandwidth and the throughput [19].

Basic Sliding Window (BSW)

BSW is used for marks the boundary of file and stipulate break condition logic. Chunk boundary is computed based on the fingerprint algorithm. File boundary is marked based on break condition. The problem with this approach is the chunk size. The size of the chunk cannot be predicted with this approach. But it is possible to predict the probability of

getting a larger chunk or a smaller one. This probability is defined based on the probability of getting a particular fingerprint[13].

Hash based Deduplication

Hashing is the method of producing unique check sum value or mathematical representation of specific chunk or block of data. The value generated is then the fingerprint used to reference data in metadata when checking for similarity or duplication. There various algorithm with different reliability. Collision occurs when two or more chunks have the same fingerprint or signature [17]. Hash based data de-duplication methods use a hashing algorithm to identify "chunks" of data. Commonly used algorithms are Secure Hash Algorithm 1 (SHA-1) and Message-Digest Algorithm 5 (MD5) [20]. Using MD5, a fixed length output of 128 bits is produces when a variable length message is process. A cryptographic hash function is used for security of document [21].

Evaluation of similarity measure for Text Document

Degree of similarity between documents sentences needs to be considered and measure when discussing deduplication, document clustering and text summarization.

Text similarity is partitioned into three approaches; String-based, Corpus-based and Knowledge-based similarities and also categorized into lexical and semantic. Words with similar character sequence are lexically similar and is associated with String-Based algorithms which considers the arrangement and the sequence of character in the document [22].

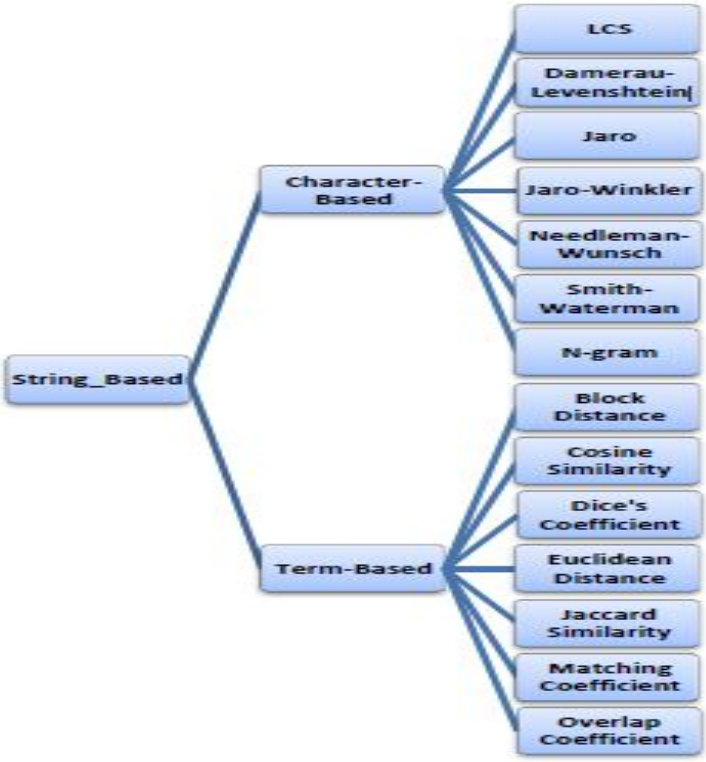

Figure 1 (Wael H.G, Aly A. F,)

String-Based Similarity

String similarity matching considers the composition, sequence and arrangement of the characters in document. A string metric is a measure used for this consideration.

Longest Common SubString (LCS): The length of similar sequence of characters is considered in two strings [22].

Jaro Distance takes into account the arrangement and order of characters between two strings. It also considers the difference in spelling [23][24].

Edit or Levenshtein distance convert from one form of string to another by involving edit operations. Edit operation involve inserting, deleting or replacing a simple character by another [25].

Smith-Waterman peruse the strings involves to find the best alignment [22].

Euclidean Distance: Square root of the difference in square of corresponding elements in the strings of the two vectors [22].

Cosine Distance: Each text is treated as a vector and the similarity between the vectors is determined [24].

Jaccard coefficient is also called Tanimoto coefficient, measures similarity as the intersection divided by the union of the objects [24][26]. Jaccard index is a statistic used for comparing the degree of similarity and diversity of samples sets. Jaccard index is also called as Jaccard similarity coefficient [3][7].

Pearson's Correction Coefficient is also a measure of the extent to which two vectors are related.
Clustering is a technique that organizes a large quantity of unordered text documents into a small number of meaningful and coherent clusters in order to aid easy browsing and thereby providing a basis for intuitive and informative navigation mechanisms. Clusters are formed using k-means clustering algorithm. Records within the clusters can be matched by divide and conquer method [27][28].

3.METHODOLOGY

It is observed that an attempt to download a particular file to the same folder or directory where such document already exist prompt the dialogue box in figure 2. This is in-line deduplication process. This dialogue box does not appear if the file has been copied to different folder or directory. Therefore it is essential to design a system or application by which we can find a duplicate or repeated file even if it post process. Figure 2 show the diagram of the in-line dialogue box.

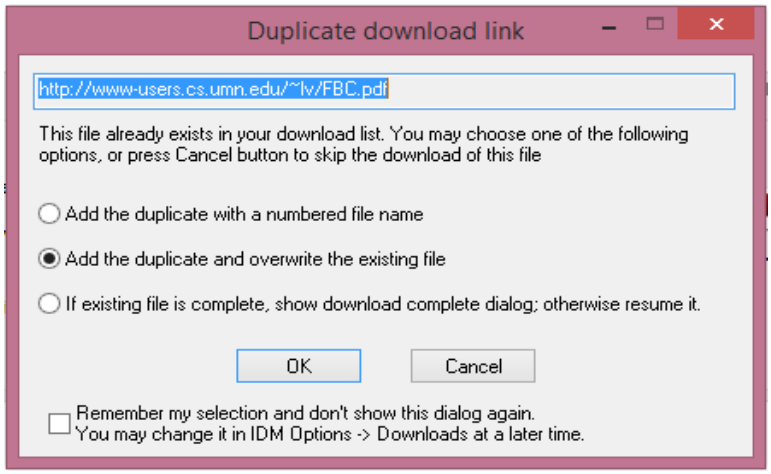

Figure 2 Dialogue box for in-line deduplication

With the designed system in figure 2, the user selects the directory to search for duplicate, click on open file to select the filename and then click start. The application searches the directory and compares the files with the content of the file stated. The application has five fields or columns; (1) Number of the file compared, (2) Filename of the files compared, (3) Status (4) Match (5) Path. It lists the filename of the list of the files compared, give their path, their status. If the file is not yet compare, it gives pending as the status, if searched it give check as the status.

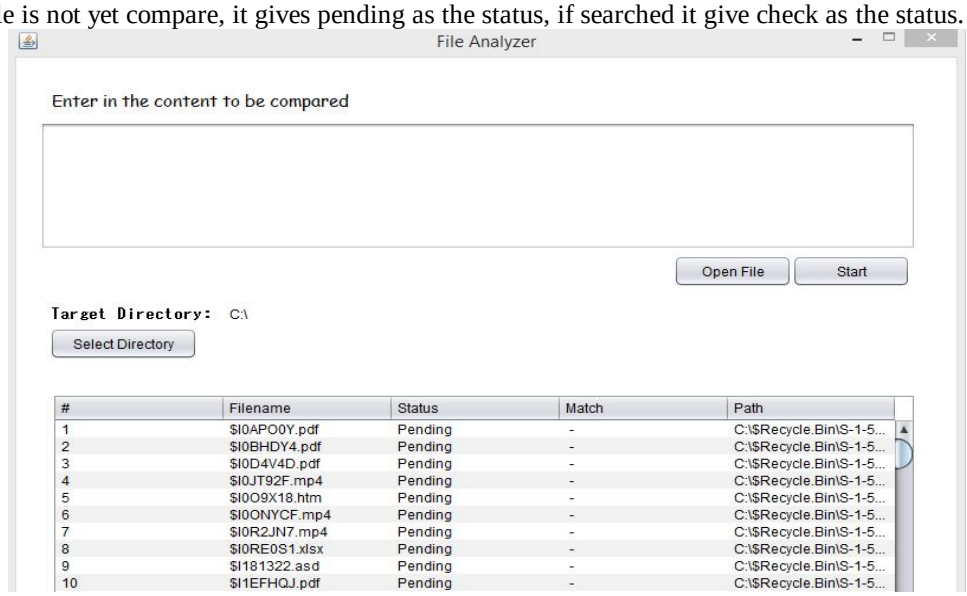

Figure 3 If the file matches the file searched for, it gives matched if not it gives false.

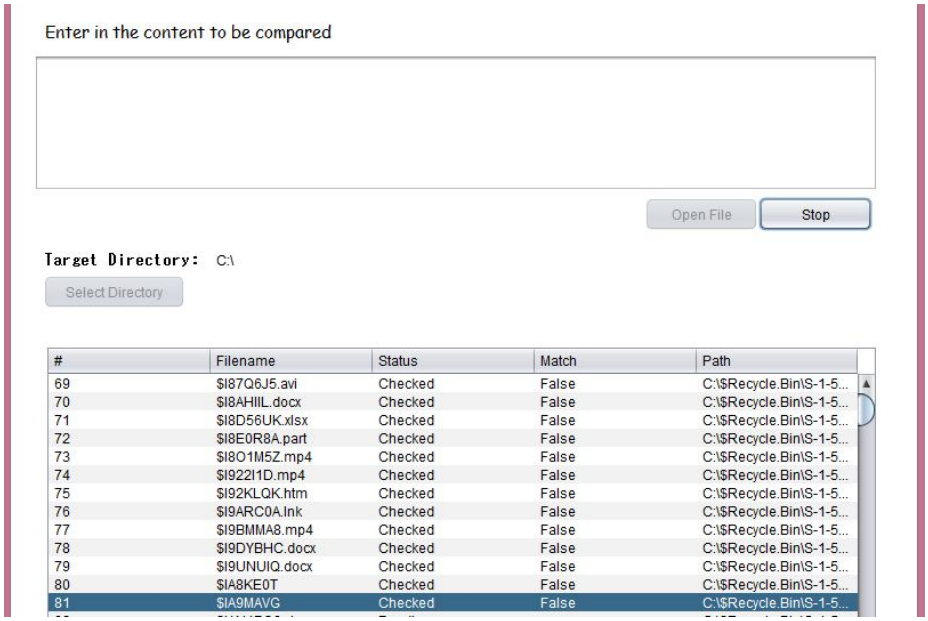

Figure 4(a)

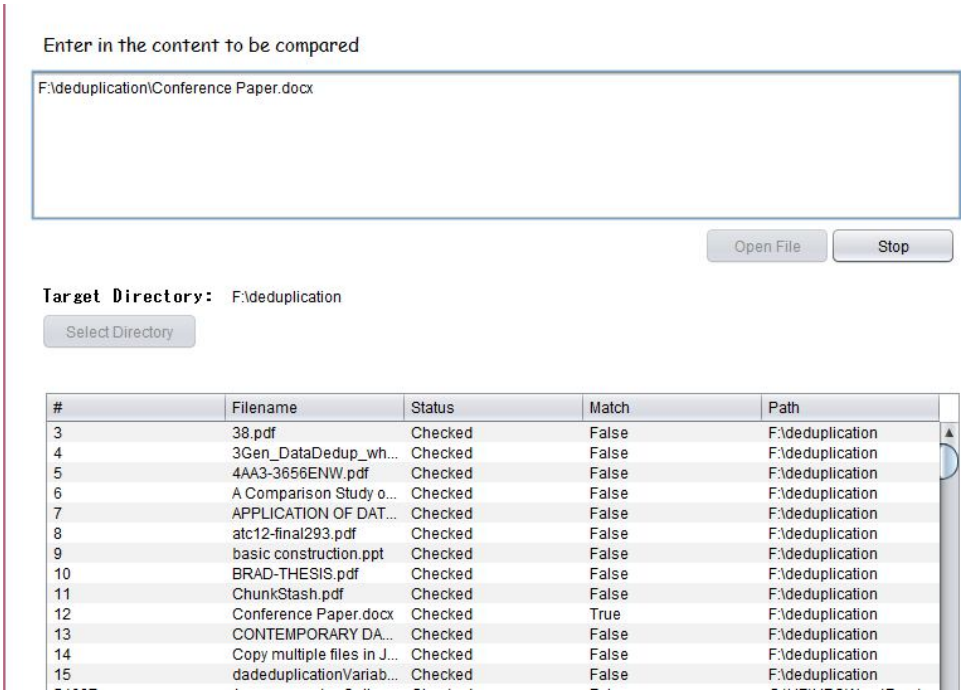

Figure 4(b) When the search is completed, the user press stop button. There will appear a message box that gives the number of matches found

Figure 4(c)

```
Start
Create GUI interface
Select task
Case "Compare file" {
   Enter content to the text box
   Click Open file to open file selection dialog box
   Navigate and select a file
   Attaché a file
   Click start
          filecontent = file.getText
   Search text = Textbox.getText
   Content length = filecontent.length
   for (I =0; I < file content.length; I ++) {
      if ((substr (filecontent, I, search content.length).length = searchtxt.length){
         report matching
         } else
      } report false
Case "compare folder"{
   Click select directory
   Navigate and select a folder
   Attaché the folder
   Click start
          for each file in the folder
          filecontent = file.getText
          fileposition = folder.position of (file)
   for (I = fileposition; I <file.length;) {
   filepositioncontent = file in position (i).getText
      if (filepositioncontent = = filecontent){
         report matching
         } else
      } report false
```

4.CONCLUSION AND RECOMMENDATION

The paper peruses the techniques of deduplication and similarity matching. With this system deployed it makes it possible to easily identify and remove redundant in the repository. The method is simple straight forward. The method efficiently identifies structurally similar documents as duplicate.

5.FUTURE WORK

The method can also be designed with other programming language like C and C++. GUI for chunk or block-level deduplication will be formed in order to be able to select the threshold value for the block or chunk.

REFERENCE

[1]. Somefun O.M, Adebayo A. O. "Evaluation of Dominant Text Data Compression Techniques," International Journal of Application or Innovation in Engineering & Management (IJAIEM) Volume 3, Issue 6, June 2014.
[2]. Wen Xia, Hong Jiag, Dan Feng, Yu Hua "Silo: A similiarity-Locality based Near Exact Deduplication Scheme with Low RAM overhead and High throughput"
[3]. PrajaktaNaxanePravin, Prof, Mangesh R. Wanjari Nagpur, Maharashtra "Cleaning Heterogenous Data & Removing Duplicates" International Journal of Advanced Research in Computer Science and Software Engineering Volume 6. Issue 5, May 2016 ISSN:2277 128X
[4]. Laura DuBois, Robert Armatruda "Backup and Recovery Accelerating Efficiency and Driving Down IT costs Using Data Deduplication" White paper (2010) Sponsor by EMC Corporation
[5]. Druva, Inc. www.druva.com "Data Deduplication methods" retrieved 19/02/2017.
[6]. MattewBrisse, Quantum Gideon Senderov "Data Deduplication: Methods for Achieving Data Efficiency" SNIA

[7]. ShengmeiLuo, Guangyan Zhang, Chegwen Wu, Samee U. Khan, KeginLi "Boafft: Distributed Deduplication for Big Data Storage in the Cloud" IEEE TRANSACTIONS ON CLOUD COMPUTING, VOL 4, NO. X, XXXXX 2016
[8]. Fang Yan, Yuan Tan: "A method of Object-based Deduplication". Journal of Network Vol 6, No 12, December 2011
[9]. Keren Jin, Ethan L Miller "The effectiveness of Deduplication on Virtual Machine Disk Images" Storage System Research Center University of California, Santa Cruz
[10]. Iuon-Chang Lin, Pro-Chin Chien "Data Deduplication Scheme for Cloud Storage' International Journal of Computers consumers and control (IJ3C), Vol.1 No2 (2012)
[11]. IderLkhagvasuren, Jung Min So Jeong Gun Lee, Jin Kim, Young WoongKo " Design and Implementation of storage system using Byte-index Chunking Sheme" International journal of software Engineering and Application Vol.8, No1(2014) pp 33-34
[12]. Chuanshuai Yu, Chenwei Zhang Mao, Fulu "Leap-based content Defined Chunking Theory and Implementation.
[13]. A. Muthitacharoen, B. Chen and D. Mazieres, "A low-bandwidth network file system", ACM SIGOPS Operating Systems Review, vol. 35, no 5, (2001), pp. 174-187
[14]. Eshghi K, Tang HK. A framework for analysing and improving content-based chunking algorithm. 2005
[15]. DeltaStor Data Deduplication: A Technical Review. White paper
[16]. S.Quinlan, S. Dorwnd "Venti: a new approach to archival storage" Proceeding of the FAST 2002 Conference on File and Storage Technologies, (2002)
[17]. Chang-Ching Yang, Chen-Mou Cheng, Sheng-De Wang " Two-phase Pattern Matching for Regular Expression in Intrusion Detection system" Journal of Information Science and Engineering 26, 1563-1582, 2010
[18]. KiranSrinivasan, Tim Bisson, arth Goodson, kaladharVoruganti "iDedupi Latency-aware, inline data deduplication for primary storage"
[19]. H.M Jung, S.Y. Park, J. G. Lee and Y. W. ko, "Efficient Data Deduplication System Considering File Modification Pattern", International Journal of Security and Its Applications, vol. 6, no.2, 2012
[20]. Zhe Sun, Jianming Young. "A novel approach to data deduplication over the engineering-Oriented cloud systems" Computer Aided Engineering 20(1), 45-50
[21]. R.Roshdy, M.Fouad, M. Aboul-Dahab "Design and Implementation a New security Hash Algorithm Based on MD5 and SHA 256" International Journal of Engineering Science & Emerging Technologies August 2013 Volume 6, Issue 1 pp: 29-36
[22]. Wael H. Gomaa, Aly A Fahmy. "A Survey of Text Similarity Approaches"
[23]. Jaro M.: "Probabilistic linkage of large public health data file, Statistics in Medicine (1995) pp 491-8
[24]. William W. Cohen, PradeepRavikumar, Stephen E. Frienberg " A comparison of string Distance Metrics for Name-Matching Tasks"
[25]. Gonzalo Navarro: "A Guided Tour to Approximate String Matching" Department of Computer Science, University of Chile, Santiago-Chile
[26]. SuphakitNiwattanakul, JatsadaSingthongchai, EkkachaiNaenudorn, SupachanumWanapi: "Using of Jaccard Coefficient for keywords Similarity". Proceedings of the International MultiConference of Engineers and Computer Scientists 2013 Vol I, 2013 Hong Kong.
[27]. L. Chitra Devi, S.M Hansa, G. N. K. Suresh Babu " A Genetic Programming Approach for Record Deduplication" International Journal of Innovative Researh in Computer and Communication Engineering Vol.1, Issue 4, 2013
[28]. 3Gen Data Deduplication Technical Discussion – White Paper (3Gen Data Systems)
[29]. EDSON Group "Thin Deduplication: A competitive Comparison" HP 3P AR