Development of Replica Free Repositories using Particle Swarm Optimization Algorithm

Jeby K Luthiya
PG Student in Information Technology
Vivekanandha College of Engineering for Women
Elayampalayam, Tamilnadu

C. Umamaheswari
Assistant Professor in Information Technology Department
Vivekanandha College of Engineering for Women
Elayampalayam, Tamilnadu

ABSTRACT

The increasing volume of information available in digital media becomes a challenging problem for data administrators. Usually built on data gathered from different sources, data repositories such as those used by digital libraries and e-commerce brokers present records with disparate schemata and structures. The increased volume even created redundant data also in the database. So a system or method is become immense to control the redundancy and duplication. In the proposed approach, a method that makes use of PSO (Particle Swarm Optimization) algorithm for generating the optimal similarity measure to decide whether the data is duplicate or not. PSO algorithm is used to generate the optimal similarity measure for the training datasets. Once the optimal similarity measure obtained, the deduplication of remaining datasets is done with the help of optimal similarity measure generated from the PSO algorithm.

Keywords

PSO Algorithm, Genetic Algorithm, Database administration, Evolutionary computing, Database integration.

1. INTRODUCTION

The increasing volume of information available in digital media has become a challenging problem for data administrators. Usually built on data gathered from different sources, data repositories such as those used by digital libraries and e-commerce brokers present records with disparate schemata and structures. Also problems regarding to low response time, availability, security and quality assurance become more difficult to handle as the amount of data becomes larger. It is possible to say that the quality of the data that an organization uses in its systems is proportional to its capacity for providing useful services to their users. In this environment, the decision of keeping repositories with "dirty" data (i.e., with replicas, identification errors, disparate patterns, etc.) goes far beyond technical questions such as the overall speed or performance of data management systems. The solutions available for addressing this situation requires more than technical efforts, they need management and cultural changes as well.

To identifying and handling replicas is important to guarantee the quality of the information made available by intensive system such as digital libraries and e-commerce brokers. These systems may depend on consistent data to offer high quality services, and may be affected by the existence of duplicates in their repositories. A genetic programming (GP) approach was used to record deduplication [1]. The problem of detecting and removing duplicate entries in a repository is known as record deduplication [10]. This approach combines several different pieces of evidence extracted from the data content to produce a deduplication function that is able to identify whether two or more entries in a repository are replicas or not. Since record deduplication is a time consuming task even for small repositories, our aim is to foster a method that finds a proper combination of the best pieces of evidence, thus yielding a deduplication function that maximizes performance using a small representative portion of the corresponding data for training purposes. Then, this function can be used on the remaining data or even applied to other repositories with similar characteristics. Moreover, new additional data can be treated similarly by the suggested function, as long as there are no abrupt changes in the data patterns, something that is very importable in large data repositories. It is worth noticing that this (arithmetic) function, which can be thought as a combination of several effective deduplication rules, is easy and fast to compute, allowing its efficient application to the deduplication of large repositories. By record deduplication using Genetic Programming (GP) approach that generates gene value for each record using Genetic Operations. If that gene value matched with any other record that record was considered as a duplicate record. These operations are to enhance the attributes of given record. Genetic Operations are Reproduction, Mutation and Crossover [1].

Thus it shows how the selection of GP parameters can impact the performance of the record deduplication task. Experiment results show that different GP setups can cause significant difference over the effort required to obtain suitable solutions. The main contribution is a set of guidelines for setting the parameters of GP-based approach to record deduplication. Thus diminish the user effort on setting the GP parameters for this problem, since provide detailed explanations on the parameters and what is the impact of each one over the final results.

The experimentation of the proposed algorithms showed significant results. The proposed PSO algorithm has better results than the genetic algorithm based technique [5]. Evaluate the dataset on the basis of accuracy and time consumed for the deduplication purposes.

2. RELATED WORK

Finding duplicate records in one or linking records from several data sets are increasingly important tasks in the data preparation phase of many data mining projects, as often information from multiple sources needs to be integrated, combined or linked in order to allow more detailed data analysis or mining. The aim of such linkages is to match all records related to the same entity, such as a patient or customer. As common unique entity identifiers are rarely available in all data sets to be linked, the linkage process needs to be based on the existing common attributes. Record deduplication is a growing research topic in the database and related fields such as digital libraries and data integration. Today, this problem arises mainly when data is collected from disparate sources using different information description styles and metadata standards. Other common place for replicas is found in data repositories created from OCR documents. These situations can lead to inconsistencies that may affect many services such as searching and mining. To solve these inconsistencies it is necessary to design a similarity function that combines the information available in database records in order to tell whether a pair of records refers to the same real-world entity. In the realm of bibliographic citations, for instance, propose a number of algorithms for matching citations from different sources based on edit-distance, word matching, phrase matching, and subfield extraction.

As more strategies for extracting pieces of evidence become available, many works have proposed new different approaches to combine and use them. The authors [14] classify these approaches into the following two categories:(1)Training–based Approaches this category includes all approaches that depend on some sort of training – supervised or semi–supervised – in order to identify the replicas. Probabilistic and machine learning approaches fall in this category; (2) Ad–Hoc Approaches– This category includes approaches that usually depend on specific domain knowledge or specific string distance metrics. Techniques that make use of declarative languages can also be classified in this category.

2.1 Data Linkage, Deduplication and Artificial Data

The basic idea is to link records by comparing common attributes, which include person identifiers and demographic information [16]. In recent years, researchers started to explore machine learning and data mining techniques to improve the linkage process. Clustering, active learning [18], decision trees, graphical models, and learnable approximate string distances are some of the techniques used. This variety makes it difficult to validate the presented results and to compare new deduplication and linkage algorithms with each other. Tuning of parameters can result in high accuracy and good performance for a certain algorithm on a specific data set, but the same parameter values might be less successful on other data or in deferent application areas. There is clearly a lack of publicly available real world data sets for deduplication and data linkage, which can be used as standard test beds for developing and comparing algorithms, similar to data collections used in information retrieval or machine learning. However, because many real world data sets contain personal information, privacy and confidentiality issues make it unlikely that they can be made publicly available. Using identified data, where e.g. names and addresses are encrypted or re- moved is not feasible either, as many linkage algorithms specifically work on name and address strings [17].

2.2 A Probabilistic Data Set Generator

Developed a data set generator based on ideas from and improved in several ways. Our generator can create data sets containing names and addresses, dates, and telephone and identifier. It is implemented as part of the [16] data linkage system, and freely available under an open source software license. A user can easily modify and improve the generator according to her or his needs. A user specified number of original records are generated in the first step, and in the second step duplicate records are created based on these original records by randomly introducing errors. Each record is given a unique identifier; this allows the evaluation of error rates.

2.3 Record matching or linking

Record matching or linking is the process of identifying records, in a data store, that refer to the same real world entity or object. There are two types of record matching. The first one is called exact or deterministic and it is primarily used when there are unique identifiers for each record. The other type of record matching is called approximate.

Once the basic techniques for quantifying the degree of approximate match for a pair (or subsets) of attributes have been identified, the next challenging operation is to embed them into an approximate join framework between two data sets[10]. This is a non-trivial task due to the large (quadratic in the size of the input) number of pairs involved. There are set of algorithmic techniques for this task. A common feature of all such algorithms is the ability to keep the total number of pairs (and subsequent decisions) low utilizing various pruning mechanisms. These algorithms can be classified into two main categories.

1) Algorithms inspired by relational duplicate elimination and join techniques including sort-merge, band join and indexed nested loops. In this context, we shall review techniques like Merge/Purge (based on the concept of sorted neighborhoods), BigMatch (based on indexed nested loops joins) and Dimension Hierarchies (based on the concept of hierarchically clustered neighborhoods).
2)Algorithms inspired by information retrieval that treat each tuple as a *set* of tokens, and return those set pairs whose (weighted) overlap exceeds a specified threshold. This context review a variety of set joins algorithms.

2.4 Machine Learning Approaches

More related to this work are those proposals that apply machine learning techniques for deriving record-level similarity functions that combine field-level similarity functions [13], a machine learning technique is used to improve both the similarity functions used to compare record fields and the way these pieces of evidence are combined. In that work, the extracted evidence is encoded as feature vectors that are used to train a support vector machine (SVM) classifier to better combine them in order to identify replicas. The main idea behind this approach is that, given a set of record pairs, the similarity between two attributes (e.g., two text strings) is the probability of finding the best alignment between them, so the higher the probability, the bigger the similarity between these attributes. Using examples for training a learning algorithm to evaluate the similarity between two given names, i.e., strings representing identifiers. This approach is applied to both clustering and pair-wise matching, achieving satisfactory results.

In GP-based approach [1], it can be used to improve the Fellegi and Sunter's method results. There, GP is used to balance the weight vectors produced by that statistical method, in order to generate a better evidence combination than the simple summation used by it.GP-based approach to find the best evidence combination in a generic framework that is independent of any other technique. GP to combine attributes and similarity functions in order to create evidence combinations, and compared with those provided by the traditional human fixed chosen evidence design.

The identification and removal of replicas from data repositories is a very expensive task, provide some guidelines to help the user set up the most suitable values for the GP parameters. Right parameter setup choices can lead to faster and more efficient solutions.

2.5 Short-term Operator Success and Population Variation

It was discovered that this was due to the fact that the variation in the base population was destroyed too quickly. To investigate this an experiment was run where a population of 200 randomly generated operators of depth 5, acted on a population of 10,000 base genes of depth 7 using nodes AND, OR and NOT and terminals input-1, input-4. The operators were applied to 50 pairs of input genes each, which were chosen with a probability proportional to their initial fitness. The fitness of the base population was evaluated on the EVEN PARITY 3 problem before and after the application of the operators. The effect of the operators was evaluated by the average proportionate change in fitness of the operand genes before and after, by comparing the average of the input genes' fitness to the fitness of the resulting genes.

3. GENETIC PROGRAMMING

In GP [1] (or even some other evolutionary technique) to solve a problem, there are some basic requirements that must be fulfilled, which are based on the data structure used to represent the solution. In this case, have chosen a tree-based GP representation for the deduplication function since it is a natural representation for this type of function. These requirements are the following:

1. All possible solutions to the problem must be represented by a tree, no matter its size.

2. The evolutionary operations applied over each individual tree must, at the end, result into a valid tree.

3. Each individual tree must be automatically evaluated.

Each piece of evidence (or simply "evidence") *E* is a pair <*attribute; similarity function*> that represents the use of a specific similarity function over the values of a specific attribute found in the data being analyzed. For example, if we want to deduplicate a database table with four attributes (e.g., forename, surname, address, and postal code) using a specific similarity function the following list of evidence: *E1<name; Jaro>*, *E2<surname; Jaro>*, *E3<address; Jaro>*, and *E4<postalcode; Jaro>*. For this example, a very simple function would be a linear combination such as $F_S(E_1,E_2,E_3,E_4) = E_1,E_2,E_3,E_4$ and a more complex one would be $F_C(E_1,E_2,E_3,E_4) = E_1*(E_2{}^{E3}/E_4)$

To model such functions as a GP tree, each evidence is represented by a leaf in the tree. Each leaf (the similarity between two attributes) generates a normalized real number value (between 0.0 and 1.0). A leaf can also be a random number between 1.0 and 9.0, which is chosen at the moment that each tree is generated. Such leaves (random numbers) are used to allow the evolutionary process to find the most adequate weights for each evidence when necessary. The internal nodes represent operations that are applied to the leaves. In our model, they are simple mathematical functions (e.g., +,-,*, /,*exp*) that manipulate the leaf values.

According to all trees generated during a GP evolutionary process are tested against preevaluated data repositories where the replicas have been previously identified. This makes feasible to perform the whole process automatically, since it is possible to evaluate how the trees perform in the task of recognizing record pairs that are true replicas.

The fitness function is the GP component that is responsible for evaluating the generated individuals along the evolutionary process. If the fitness function is badly chosen or designed, it will surely fail in finding a good individual. In the experiments presented in this paper, we have used the F1 metric as our fitness function. The F1metric harmonically combines the traditional precision (P)and recall (R) metrics commonly used for evaluating information retrieval systems, as defined below:

$$P = \frac{\text{Number of correctly Identified Duplicated Pairs}}{\text{Number of Identified Duplicated Pairs}}$$

$$R=\frac{\text{Number Of correctly Identified Duplicated Pairs}}{\text{Number Of True Duplicated Pairs}}$$

$$F1=\frac{2*p*R}{P+R} \text{---------------------------} (1)$$

Here, this metric is used to express, as a single value, how well a specific individual performs in the task of identifying replicas. In summary, our GP-based approach tries to maximize these fitness values by looking for individuals that can make more correct decisions with fewer errors. A genetic programming approach to record deduplication that combines several different pieces of evidence extracted from the data content to find a deduplication function that is able to identify whether two entries in a repository are replicas or not.

4. PROPOSED PSO

Particle Swarm Optimization (PSO) is an optimization technique which provides an evolutionary based search. This search algorithm was introduced by Dr Russ Eberhart and Dr James Kennedy in 1995. PSO is a computational method that optimizes a problem by iteratively trying to improve a candidate solution by a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions. The outline of PSO is stated as follows

1. Create a 'population' of agents (called particles) uniformly distributed over X.

2. Evaluate each particle's position according to the objective function.

3. If a particle's current position is better than its previous best position, update it.

4. Determine the best particle according to the particle's previous best positions.

5. Update particles' velocities according to

$$Velocity_{ex} = Velocity^{0} + \Phi\ (pbest – pos^{0}) + \varphi\ (gbest – pos^{0})$$

Where, $Velocity^{0}$ = Current velocity

pbest = Current best position

pos^{0}= Current position

gbest= global best position

Φ, φ= random values in range [0, 1]

6. Move particles to their new positions according to

$pos= pos^{0} + Velocity_{ex}$

7. Go to step 2 until stopping criteria are satisfied.

In addition to traditional gradient-based optimization algorithms, there are many other heuristic techniques that compete with PSO such as genetic algorithm, simulated annealing, evolutionary programming, and most recently ant colony optimization
The advantages of mentioned algorithms over PSO are the following

• The availability of commercial versions of some algorithms like Matlab (genetic algorithm) and Excel premium solver (evolutionary programming).

• The extensive collection of books and research literatures, especially in the case of genetic algorithm and evolutionary programming, which cover these competing methods. Despite the simplicity of the PSO concept and implementation, its superiority is proven when compared with other techniques in many different application areas.

However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. The detailed information will be given in following sections. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied.

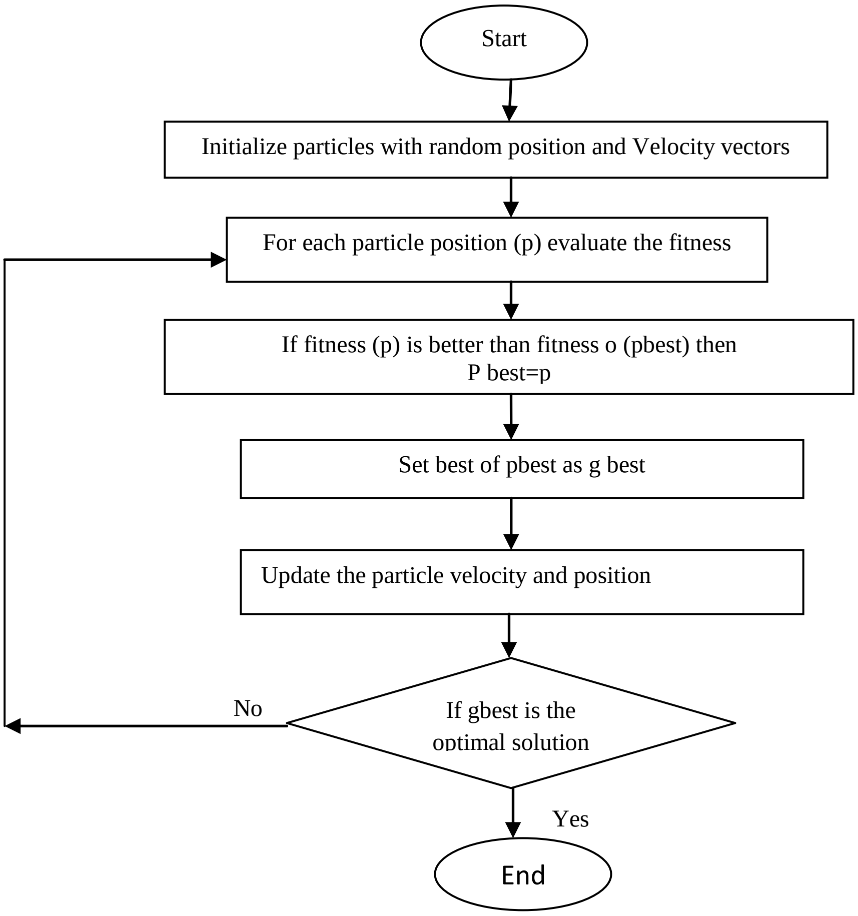
Start
Initialize particles with random position and Velocity vectors
For each particle position (p) evaluate the fitness
If fitness (p) is better than fitness o (pbest) then
P best=p
Set best of pbest as g best
Update the particle velocity and position
If gbest is the
optimal solution
No
Yes
End

Fig 1: Flow chart

5. Deduplication using PSO optimization algorithm

The proposed approach has two phases such as training phase and duplicate detection phase. The two measures are computed for all attributes of record pairs because different similarity operations have varying significance in different domains.

i. Levenshtein distance

The "Levenshtein distance" is computed by calculating the minimum number of operations that has to be made to transform one string to the other, usually these operations are: replace, insert or deletion of a character.

ii. Cosine similarity

The cosine similarity between the two records name field "Record 1" and "Record 2" are calculated as follows: First, the dimension of both strings are obtained by taking the union of two string elements in the record 1 and "record 2" as (word1, word2,….word N) and then the frequency of occurrence vectors of the two elements are calculated i.e. "record 1" = (<vector value1>, <vector value2>,……<>) and "record (<vector value1>, <vector value2>,……<>) . Finally we obtain the dot Product and magnitude of both strings.

5.1 Feature Vectors

Feature vectors represent the set of elements that is required for the detection of duplicate elements from the data repository. The vectors can be obtained from the processing of the two similarity measure values.

5.2 Algorithm: PSO-Based Deduplication

The PSO algorithms are some of the most used optimization algorithm in the field of data mining. The PSO algorithm is characterized by optimizing a number of solutions from a swarm of solutions. The typical mathematical methods used in the PSO algorithm give extra hand for the PSO to differ from other optimization algorithms.

5.2.1 Population

Populations are selected based on the cosine similarity calculated using Levenshtein distance function.

$$\mathbf{(a+b)^{2} + (c-d)^{2}}$$

$$\mathbf{((a+b)) * (c-d)^{2}}$$

$$\mathbf{(a^2+b^2) * (c^2-d^2)}$$

$$\mathbf{c\,(a+b) - d(a-b)}$$

5.2.2 Fitness function

By selecting expression to find duplicates, is called fitness function is selected based on the populations.

Accuracy= *<u>true positives true negatives</u>*
True positive false positive true negative false Negatives

5.2.3 New Population Generation

If selected fitness function does not reach fitness value have to change populations these populations are selected based on their position and velocity of expression.

5.2.4 Optimization

After full execution fitness function comes with number of solutions, the solutions are filtered out and best expression with highest fitness value is selected as solution.

5.2.5 Termination

Termination criteria set by user itself. The termination criteria will be number of iterations, if termination criteria is reached it will provide best solutions.

6. EXPERIMENTAL RESULTS

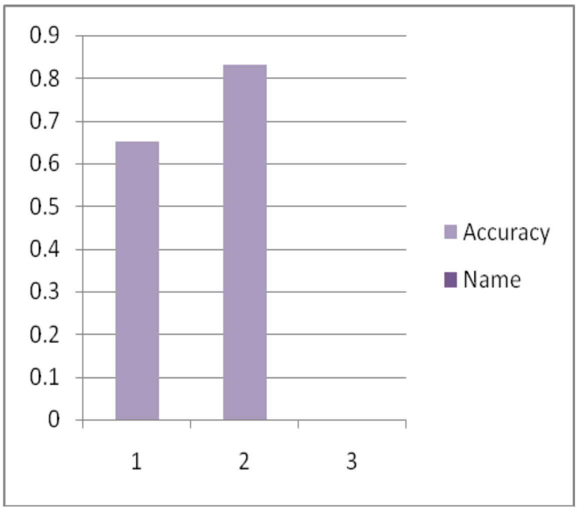

Fig 2: Accuracy

This graph includes optimization based algorithm such as GA, PSO. In accuracy contain Genetic 65% and PSO 83%. All algorithms are implemented in JAVA and executed on a PENTIUM IV, 250 GB, and 2GB RAM computer.

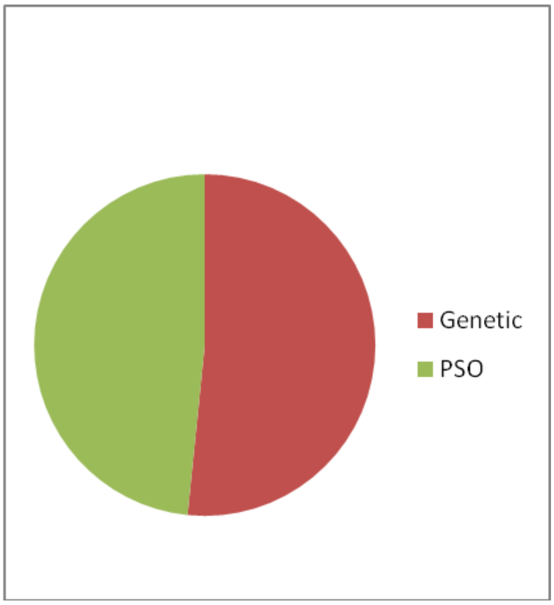

Fig 3: Memory

This graph is based on the memory. In this graph Genetic contain 80% and the PSO contain 75%.

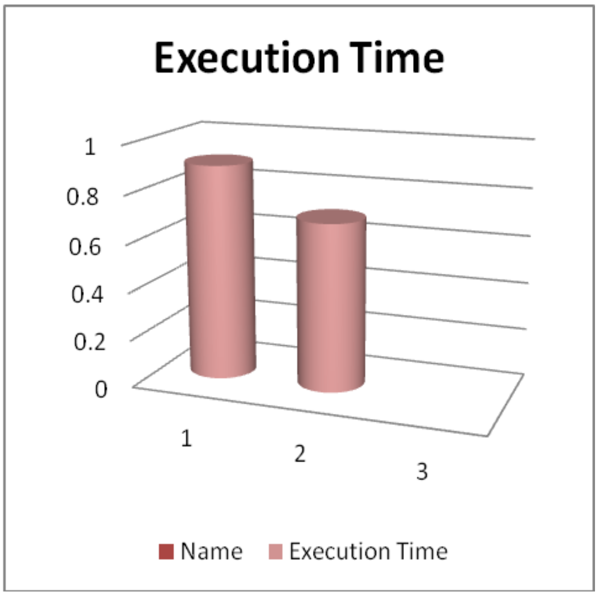

Fig 4: Execution Time

The last one is Execution Time. In this Execution Time the Genetic contain 90% and PSO contain 70%.

The experimentation starts from selecting the datasets as the input of the similarity computation by the similarity computation factors, listed in the above sections, such as Levenshtein distance method and cosine similarity method.

7. CONCLUSION

The deduplication has been one of the most emerging techniques for data redundancy and duplication. The duplication creates lots of problems in the information retrieval system. Thus Optimization algorithm is used to avoid the duplication. The technique proposed that, PSO algorithm can provide better performance and accuracy than the genetic algorithm based techniques. PSO algorithm is used to generate the optimal similarity measure for the training datasets. Once the optimal similarity measure obtained, the deduplication of remaining datasets is done with the help of optimal similarity measure generated from the PSO algorithm.

The experimentation of the proposed algorithms showed significant results. The proposed PSO algorithm has better results than the genetic algorithm based technique. Evaluate the dataset on the basis of accuracy and time consumed for the deduplication purposes.

7. REFERENCES

[1] Moises G. de Carvalho, Alberto H. F.Laender, Marcos Andre Goncalves, Altigran S. da Silva, “A Genetic Programming Approach to Record Deduplication”, *IEEE Transaction on Knowledge and Data Engineering*,pp 399-412, 2011.

[2] LuísLeitão and PávelCalado, "Duplicate detection through structure optimization", *ACM International conference on Information and knowledge management*, pp: 443-452, 2011.

[3] Ektefa, M, Sidi. F,Ibrahim. H, Jabar. M.A., Memar. S, Ramli. A, "A threshold-based similarity measure for duplicate detection ", *IEEE conference on Open systems*, pp: 37-41, 2011.

[4] Elhadi. M, Al-Tobi. A, “Duplicate Detection in Documents and WebPages Using Improved Longest Common Subsequence and Documents Syntactical Structures", *International Conference on Computer Sciences and Convergence Information Technology*,pp: 679-684,2009.

[5] Ye Qingwei, WuDongxing, Zhou Yu, Wang Xiaodong, " The duplicated of partial content detection based on PSO ", *IEEE FifthInternational Conference on Bio-Inspired Computing: Theories and Applications*, pp: 350 - 353, 2010.

[6] J Prasanna Kumar, and P Govindarajulu. "Duplicate and Near Duplicate Documents Detection: A Review". *European Journal of Scientific Research*, vol. 32, pp: 514-527, 2009.

[7] Dutch T. Meyer and William J. Bolosky, “A Study of Practical Deduplication", *Computer and Information Science*,pp: 1-13, 2011.

[8] Danny Harnik, Benny Pinkas, Alexandra Shulman-Peleg “Side channels in cloud services, the case of deduplication in cloud storage", vol. 8, no. 6, pp: 40-47, 2010.

[9] Yujuan Tan, Hong Jiang, Dan Feng, Lei Tian, Zhichao Yan, Guohui Zhou, " SAM: A Semantic-AwareMulti-Tiered Source De-duplication Framework for Cloud Backup”, *International Conference on Parallel Processing (ICPP)*, pp: 614-623, 2010.

[10] N. Koudas, S. Sarawagi, and D. Srivastava, “Record linkage: similarity measures and algorithms,” in Proceedings of the2006 *ACM SIGMOD International Conference on Management of Data*, pp. 802–803, 2006.

[11] C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki.Hydrastor: a scalable secondary storage. In Proc.7th USENIX Conference on File and Storage Technologies, 2009.

[12] C. Ungureanu, B. Atkin, A. Aranya, S. Gokhale, S. Rago, G. Cakowski, C. Dubnicki, and A. Bohra. Hydrafs: A high-throughputfile system for the Hydrastor content-addressable storage system. In Proc.*8th USENIX Conference on File and Storage Technologies*, 2010.

[13] W. Bolosky, S. Corbin, D. Goebel and J. Douceur.Single instance storage in Windows 2000.In Proc. *4th USENIX WindowsSystems Symposium*, 2000.

[14] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios, “Duplicate Record Detection: A Survey,” IEEE Trans. Knowledge and Data Eng., vol. 19, no. 1, pp. 1-16, Jan. 2007.

[15] S. Dorward and S. Quinlan.Venti: A new approach to archival data storage. In Proc.*1st USENIX Conference on File andStorage Technologies*, 2002.

[16] P. Christen, “Probabilistic Data Generation for Deduplication and Data Linkage,” Intelligent Data Eng. and Automated Learning, pp. 109-116, Springer, 2005.

[17] Fellegi, I. and Sunter, A.: A theory for record linkage. Journal of the American Statistical Society, December 1969.

[18] Sarawagi, S. and Bhamidipaty, A.: Interactive deduplication using active learning. Proceedings of the 8th ACM SIGKDD conference, Edmonton, July 2002.