I.J. Information Technology and Computer Science, **2017, 7, 69-76**
Published Online July 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2017.07.08

An Efficient String Matching Technique for Desktop Search to Detect Duplicate Files

Dr. S. Vijayarani
Assistant Professor, Department of Computer Science, Bharathiar University, Coimbatore, Tamilnadu, India
E-mail: vijimohan_2000@yahoo.com

Ms. M.Muthulakshmi
M.Phil. Research Scholar, Department of Computer Science Bharathiar University, Coimbatore, Tamilnadu, India
E-mail: abarajitha.uma@gmail.com

***Abstract*—**Information retrieval is used to identify the relevant documents in a document collection, which is matching a user's query. It also refers to the automatic retrieval of documents from the large document corpus. The most important application of information retrieval system is search engine like Google, which identify those documents on the World Wide Web that are relevant to user queries. In most situations, users may download the files that are already downloaded and stored in their computer. Then, there is a chance of multiple copies of the files that are already stored in different drives and folders on the system, which in turn reduces the performance of the system and these files occupy a lot of memory space. Analyzing the contents of the file and finding their similarity is one of the major problems in text mining and information retrieval. The main objective of this research work is to analyze the file contents and deletes the duplicate files in the system. In order to perform this task, this research work proposes a new tool named Duplicate File Detector Tool i.e. DFDT. DFDT helps the user to search and delete duplicate files in the system at a minimum time. It also helps to delete the duplicate files not only with the same file category, but also with different file categories. Boyer Moore Horspool and Knuth Morris Pratt string searching algorithms are existing algorithms and these algorithms are used to compare the file contents for finding their similarity. This work also proposes a new string matching algorithm named as W2COM (Word to Word COMparison). From the experimental results it is observed that the newly proposed W2COM string matching algorithm performance is better than Boyer Moore Horspool and Knuth Morris Pratt algorithms.

***Index Terms*—**Content Analysis, File similarity, String matching, Boyer Moore Horspool, Knuth Morris Pratt, W2COM.

I. Introduction

Information retrieval (IR) is discovered the documents of an unstructured nature that satisfies information need from within a large collection of documents. This system normally searches in collections of unstructured or semi-structured documents [23]. The need for an information retrieval system occurs when a collection reaches a size where customary cataloguing techniques can no longer survive. The general applications of information retrieval systems are digital libraries, media search, search engine like desktop search, mobile search, and web search, etc., [24]. This work mainly focused on the desktop search, which detects the duplicate files in the computer.

The main motivation behind this proposed work is that there is a tremendous growth in the internet and the sophisticated developments in the hardware technology provide the users to download and store a lot of information [10]. In most situations, users may download the files, which are already downloaded and stored in the computers. There is a possibility of duplicate files, which are stored in different drives and folders on the system, which reduces the system performance and occupies extra memory space [6]. There are a number of tools available to delete the duplicate files in the system. The main disadvantage of these tools is that they only help to delete the files with similar categories (doc to doc, pdf to pdf, txt to txt, xls to xls).In file comparison, string searching algorithms are used and it tries to find a position where one or more than a few strings (also called patterns) are found within a string or text [2] [16].

II. Related Works

From the literature, we come to know that many algorithms are used for detecting patterns and string matching. BRSS [Berry-Ravindran and Skip Search] is a hybrid algorithm, proposed by Abdulwahab Ali et al. [3] which performs character comparison effectively, hence it is used for DNA searching, Protein sequence searching and English text searching. Connection is a file system search tool [Craig A.N, 8] which combines traditional content-based search and context information gathered from user activity. By tracing file system calls, the connection can identify temporal relationships between files and use them to expand and reorder traditional content search results. This tool has improved both average recall and average precision over a state-of-the-art content-only search system. String searching algorithms plays a major role to detect patterns in the text.

Ababneh Mohammad et al. [1] has proposed occurrences algorithm and this algorithm finds all the occurrences of the pattern in the text. Three important steps of this algorithm is, pattern preprocessing, text preprocessing and searching. Depending on the results of the preprocessing, the searching process is performed.

Another string matching algorithm, named ACM proposed by JormaTarhio et al. [15]. This algorithm required minimum memory requirement for performing string matching process. Bo hong et al. [6] proposed a new method DDE for identifying and coalescing identical data blocks in Storage Tank, a SAN file system. This design employs a combination of content hashing, copy-on-write and lazy updates to achieve its functional and performance goals.

DDE executes primarily as a background process. Gregory et al. [9] they have reported on the results of extracting useful information from text notes captured within a Customer Relationship Management (CRM) system to segment and thus target groups of customers likely to respond to cross-selling campaigns. These notes often contain text that is indicative of customer intentions. The results indicate that the notes are meaningful in classifying customers who are likely to respond to purchase multiple communication devices. A Naïve Bayes classifier outperformed a Support Vector Machine classifier for this task. When combined with structured information, the classifier performed only marginally better.

Anthony Scimeet al. [4] in data analysis, when data are unattainable, it is common to select a closely related attribute as a proxy. But sometimes a substitution of one attribute for another is not sufficient to satisfy the needs of the analysis. In these cases, a classification model based on one dataset can be investigated as a possible proxy for another closely related domain's dataset. If the model's structure is sufficient to classify data from the related domain, the model can be used as a proxy tree. Such a proxy tree also provides an alternative characterization of the related domain. They present a methodology for evaluating datasets, as proxies along with three cases that demonstrate the methodology and the three types of results.

The remaining section of this work is organized as follows; Section 3 illustrates the review of literature. Section 4 describes the objective of the problem and contribution. Experimental results are discussed in section 5 and conclusions are given in section 6.

III. Problem Objective and Contribution

The main objective of this research work is to analyze the file contents and deletes the duplicate files in the system by finding the similarity between files. In order to find the duplicate files, files can be compared using string searching algorithms. Boyer Moore Horspool and Knuth Morris Pratt algorithms are used in this research work. The new algorithm W2COM is proposed for comparing files and finds duplicate files. The efficiency of these algorithms is verified by three performance factors; execution time, memory requirement and relevancy accuracy.

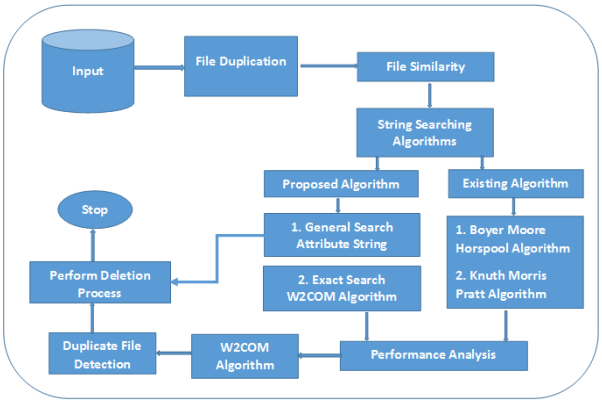

Fig.1. System Architecture

Data set

In order to find the memory utilization, the real dataset is taken from the system using FileList Tool. FileList is a command line utility providing a list of files of the selected path in the CSV format. This dataset consists of 25393 instances and 4 attributes, namely file name, file size, extension and path of the file. The different types of files used in this research are pdf, doc, docx, xls and jpg and these files are used for file comparison.

File comparison

In this phase, two types of file searching are used. First one is general search and the second one is an exact search. On general search, files are compared based on its file properties. In the exact search method, word by word comparison is done based on file contents. In this search, three algorithms are used. Boyer Moore Horspool and Knuth Pratt algorithms are existing algorithms and W2COM is the newly proposed algorithm. These algorithms are used to find and delete duplicate files and also discover the relevance between files.

A. General search

Attribute based search algorithm

Attribute based search algorithm is used to search the duplicate files very fast. This search technique is used to search the duplicate files based on their properties. For file comparison, the file name, file size, page count, number of lines, number of words and keywords are property attributes. In image comparison, the image name, image size, height of the image, the width of the image and the number of pixels are considered as property attributes [19]. For comparing excel files, the file name and file size are considered as file properties. Two performance measures are used in this general search method; they are memory and execution time. It is one of the quick search techniques used for finding and removing the duplicate files. Relevancy can based on the similarity of the file properties such as file name, file size, line count, and so on.

Input: Classified System Files
Output: Duplicate files in the system
Method:

1. Collect the classified system files as input.
2. For File Comparison //(either document, pdf or txt)
   - 2.1 Check the file names
   - 2.2If (filenames does not match) then Goto Step 5 else
   - 2.3 Compare the file size
   - 2.4If (file size does not match) then Goto Step 5 else
   - 2.5 Verify the page count, number of words and line count.
   - 2.6 Identify the duplicate files and delete the file.
   - 2.7 Go to step 5
3. For Image Comparison
   - 3.1 Check the name of the image
   - 3.2 If there is no match, then goto5 step else
   - 3.3 Check if the image size, height, width, number of pixels of the image are same then
   - 3.4 Identify the duplicate images and delete the image
   - 3.5 Else go to step 5
4. For excel files
   - 4.1 check if the file name is same than
   - 4.2 verify its file size than
   - 4.3 Consider, it is a duplicate and delete
   - 4.4 Else
5. Move to the next file

Algorithm 1 - Attribute based search algorithm

B. Exact Search

Boyer Moore Horspool Algorithm

The Boyer Moore Horspool algorithm or Horspool's algorithm is an algorithm for searching substring in large strings. This algorithm was published by Nigel Horspool in 1980. It is a generalization of the Boyer–Moore algorithm which is associated with Knuth–Morris–Pratt algorithm [15]. The algorithm deals space of time in order to attain an average-case complexity of O(N) on random text and O(MN) in the worst case, where the pattern length is M and the search string length is N [21].

In the Boyer-Moore- Horspool algorithm, it compares the text character t_i with the last character p_m of the pattern. If they match, then it compares the preceding characters of the text with the corresponding characters in the pattern sequentially right to left, until to detect either an occurrence of the pattern or a mismatch on a text character. Suppose, irrespective of the match is occurring, it slides the pattern according to the next occurrence of the character t_i in the pattern. [7] [17]. The number of positions to be moved is determined by the value of skip (t_i).

Computation of the skip table in the Boyer-Moore Horspool algorithm has a subtle difference with the original skip table definition proposed in the Boyer-Moore algorithm. In the Boyer- Moore algorithm, the value of skip (p_m) is always 0. In the Horspool version, skip (p_m) = m if p_m is unique within the pattern (i.e., the character p_m does not appear in any other location in the pattern); otherwise skip (p_m) = m-k, where p_{m-k} is the penultimate (rightmost) appearance of the character p_m in the pattern [17][23].

Boyer Moore Horspool Algorithm

1. Initialize pattern length m←|p|;
2. Initialize the text length n ←|t|;
3. Compute skip table GENERATE-SKIP-TABLE(∑,p);
   a. Set pattern length m←|p|;
   b. Initialize skip table skip (σ) = m for all symbols a σ ∑;
   c. Initialize pattern index j ← 1;
   d. For jth character P_j in the pattern, set skip (p_j) ←m-j;
   e. Increment pattern index j←j+1;
   f. If j<m-1 then go to step 4;
   g. Stop.
4. Initialize text pointer i←0;
5. Initialize pattern pointer j←m;
6. While j>0 and t_{i+j}= p_j
   Do move pattern pointer to left j←j-1;
7. If j=0 then
   Print "pattern occurs at text index" i+1;
8. Shift the text pointer i←i + skip(t_{i+m});
9. If i ≤ n – m then
   Go to step 5 to continue matching process.
10. Terminate

Algorithm 2 - Boyer Moore Horspool

Knuth Morris Pratt Algorithm

The Knuth–Morris–Pratt proposed a linear time string searching algorithm (or KMP algorithm) by analysis of the naïve algorithm. The algorithm was perceived in 1974 by Donald Knuth and Vaughan Pratt, and independently by James H. Morris and they published it jointly in 1977.The implementation of Knuth-Morris-Pratt algorithm is well-organized because it reduces the total number of comparisons of the pattern against the input string.

A matching time of O(n) is accomplished by avoiding comparisons with elements of 'S' that have formerly been involved in the comparison with some element of the pattern 'p' to be matched. i.e., backtracking on the string 'S' certainly not occurs.

At a high level, the KMP algorithm is related to the naive algorithm; it considers shifts so as from 1 to n−m, and it defines if the pattern matches at that shift. [20] The difference is that the KMP algorithm uses information gathered from partial matches of the pattern and text to permit shifts that are guaranteed not to result in a match.

Components of KMP algorithm

1. The prefix function, Π
   The prefix function, Π for a pattern summarizes knowledge concerning, however the pattern matches against the shifts of itself. This information may be accustomed avoid useless shifts of the pattern "p". It also indicates how much of the last comparison can be reused if it fails. In other words, this qualifies avoiding backtracking on the string "S".
2. The KMP Matcher With string "S", the pattern "p" and prefix function "Π" as inputs, the prevalence of "p" in "S" is found and the algorithm returns the variety of shifts of "p" after which the existence is found.
3. Running - time analysis: The period of time for computing the prefix function is Θ (m) and period of time of matching function is Θ (n). The total of O(n + m) run time [7].

Knuth-Morris-Pratt *(p. t, Next)*
1. Initialize the pattern length j←1;
2. Initialize the text length k←1;
3. Set length of the pattern m←|p|;
4. Set length of the text n ←|t|;
5. **While** j>0 and $p_j \neq t_i$ **do**
   Shift pattern pointer (j←Next (j));
6. Advance text pointer i←i+1
7. Advance pattern pointer j←j+1;
8. **If** j>m **then**
   Print "pattern occurs at text index" i-m
   Else shift pattern pointer j←Next (j);
9. **If** i ≤ n and j ≤ m **then**
   Gotostep 5 to continue pattern matching
   Else stop

Algorithm 3 - Knuth Morris Pratt

The computational complexity of Knuth Morris-Pratt algorithm is O (n) in both the worst and average cases for the pattern matching phase. By analyzing the matching algorithm, it can be shown that the assignment j ← Next (j) in step 5 never exceeds the total execution of the increment operation i←i+1in step 6. The pattern is therefore shifted to the right for a total of almost n times, and hence the computation complexity of the matching phase is O(n). Similarly, it shows that the processing time for initialization of the Next table is of the same order O(m). [13] As a result, the worst case is overall computational complexity of the algorithm is O(m+n).

C. Proposed Algorithm

W2COM Algorithm

This search technique is used to search the duplicate files based on file content. The algorithm used for this search is W2COM. It not only compares the content with same extension but also with different extension. In the existing algorithms, Boyer Moore Horspool algorithm works with small alphabet & large patterns and the Knuth-Morris-Pratt algorithm works only with the small alphabet & pattern. In the proposed technique, the algorithm works with the large alphabets & large patterns.

Input: Classified System files CSF
Output: 1) Search Successful or Unsuccessful 2) Similarity Measures
Method:
1. Input the Classified system files as a token, CSF=<T1, T2…Tn>;
2. Store the file content as a digests, TFe ← File content;
3. Generate TFe by using sliding window algorithm. Set Window length W=I where i= 1 to n;
4. Identify the sentence separator (, .) and store the sentence as S.
5. Compute the length of the sentence S l=S.Length ();
6. Compare the length between files if l1≠l2 then go to Step 1
7. Remove the blank space from S;
8. Compute fingerprint for the words using MD5 algorithm;
   8.1. Append Padded Bits
   8.2. Append Length
   8.3. Initialize Message Digest Buffer.
   8.4. Process message in 16- word blocks.
   8.5. Output
9. Set range for fingerprint as (0, 2^{k-1})
10. Calculate the similarity between tokens;
    | $TFe_{(W,S)}$ (Token A)| ∩ $TFe_{(W,S)}$ (Token B)| / | $TFe_{(W,S)}$(Token A)| Ụ $TFe_{(W,S)}$ (Token B)|
11. Process Terminated

Algorithm 4 - W2COM Algorithm

Generate Next Table
1. Initialize the pattern pointer j ←1;
2. Initialize overlap length of the pattern k ←0;
3. Initialize Next table, Next (1) ←0
4. While (k>0 and $p_j \neq p_k$) do k ← Next(k);
5. Increment pattern pointer j ←j+1;
6. Increment overlap length k ←k+1;
7. If($p_j = p_k$) then Next(j) ←Next(k) else Next(j) ←k;
8. If(j<m) then go to step 4;
9. Stop.

Algorithm 5 – Next Table Generation

First, the files are collected from the system and the original dataset are created. After that, system files can be initialized as a token T_1, T_2 … T_n. Using the sliding window algorithm, generate TFe (Transformed Feature element) and set window length w=i where i = 1 to n. Here the whole content cannot be considered; instead it identifies the sentence separator (,).Then find out the length of the sentence, if the length of the sentence is equal, then continue the comparison, otherwise another file is considered. The fingerprint of a token in a file is a set of digests that describes the file contents. The set of digests is referred to as the Transformed Feature element (TFe) of a file. The individual digests are called the Feature Element (FEs). The Transformed Feature Element of a file is TFe (Pe). Fingerprint represents the hashing value of the string. At each step, the algorithm computes a fingerprint using MD5 of W consecutive tokens (A token could be either a single word or character and use character based token that fall within the window). Each fingerprint is in the range (0, 2k-1) where k is a configuration parameter.

Deleting the Duplicate Files

The details of the duplicate files, which are found in different drives and folders, are displayed based on the relevant accuracy of the files. DFD tool will display a selection list showing the "duplicate found" message. The similarity must be considered separately for each file. Based on the user decision, the DFD tool deletes the duplicate files.

IV. Performance Evaluation

In order to perform the analysis, there are three performance measures are used; execution time, memory utilization and the accuracy of the algorithm. For file comparison, the existing and proposed algorithms are implemented in JAVA and the system configuration is Intel Core i3 processor running at 2.4GHz, 4 GB RAM, 64 bit Window 7 Ultimate.

Search Time

Search time refers the amount of time taken to

searching the duplicate files in the system. It is estimated by counting the number of elementary operations performed by the particular algorithm, where an elementary operation takes a fixed amount of time to perform.

Memory Utilization

Memory utilization measures the total amount of memory space occupied by the system. Space complexity refers the total memory space taken by the algorithm with respect to the size of the input. It includes both auxiliary space and space used by input.

A. Memory Utilization of Searching Algorithms

The following table shows the memory utilization of searching algorithms namely Knuth Morris Pratt, Boyer Moore Horspool and w2com algorithms. From the experimental results, it is observed that W2COM algorithm gives the best results than other two algorithms.

Table 1. Memory Utilization

Algorithm	Memory (In Kb)
Knuth Morris Pratt	154378
Boyer Moore Horspool	239876
W2com	71903

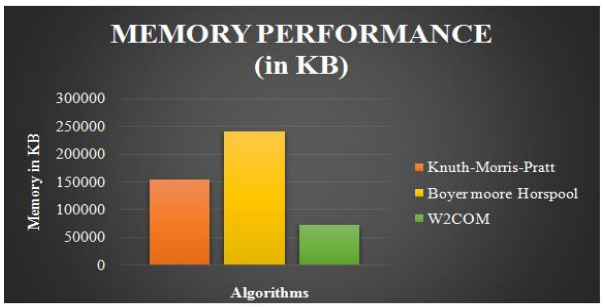

Fig.2. Memory Utilization

From the above graph (Fig.2), it is analyzed that the W2COM algorithm attains lower memory space when compared to other string searching algorithms.

B. Execution Time Performance of Searching Algorithms

The following table shows the execution time performance for single file of searching algorithms namely Knuth Morris Pratt, Boyer Moore Horspool and W2COM algorithms. From the experimental results, it is observed that W2COM algorithm performs well than other two algorithms.

Table 2. Execution Time

Algorithm	Time (in ms)
Knuth-Morris-Pratt Algorithm	1220573
Boyer Moore Horspool Algorithm	1054122
W2COM	879158

Fig.3. Execution Time

Figure 3 shows the search time required for searching single file by Boyer Moore Horspool, Knuth Morris Pratt and W2COM techniques. From the results, it is observed that the W2COM technique has required minimum search time than other two techniques.

Table 3. Accuracy Measures for String Searching Algorithms (using sample files)

Description	File Name	Boyer Moore Horspool	Knuth Morris Pratt	W2COM
		Relevancy (%)	Relevancy (%)	Relevancy (%)
Same Content with Same Extension	data.doc	100	100	100
	mining.doc	100	100	100
	text.doc	100	100	100
Same Content with different extension	data.pdf	98	99	100
	file1.txt	97	98	100
	textmining.docx	98	99	100
Different Content with different extension	tm.pdf	85	89	92
	ijircce.doc	74	85	90
	comparison.txt	90	91	94

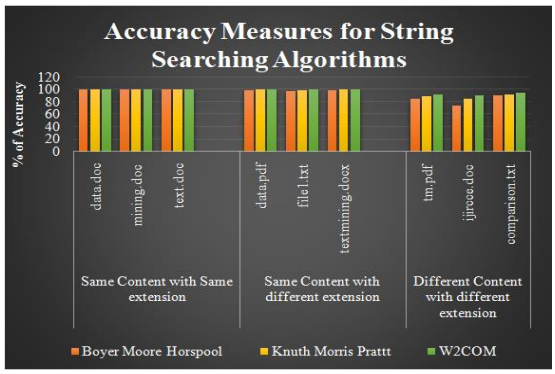

Fig.4. Relevancy Accuracy

Duplicate File Detector Tool – Snapshots

Here the Duplicate File Detector Tool snapshots are given. The quick search results are given fromFigure 5 to Figure 12. In Exact search, the same file with same extension results are given from Figure 13 to Figure 16 and the same file with different extension results are given from Figure 17 to Figure 20.

General Search

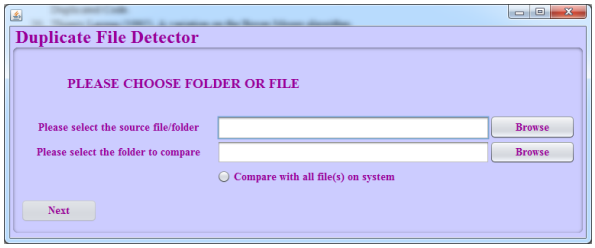

Fig.5. General Search-Home Page

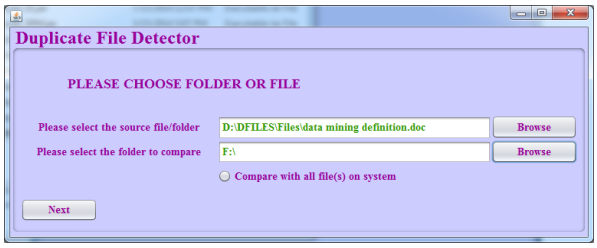

Fig.6. General Search-Folder or File Selection

Fig.7. General Search with same extension

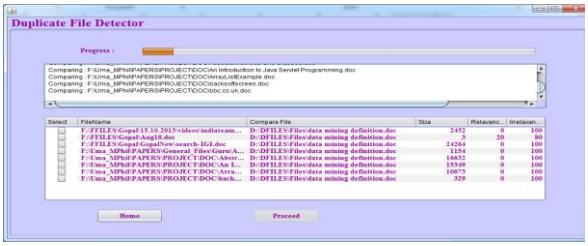

Fig.8. General Search- File Comparison

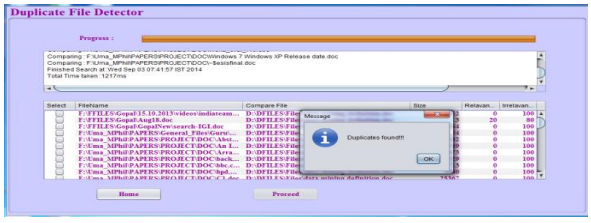

Fig.9. General Search – Finding Duplicate Files

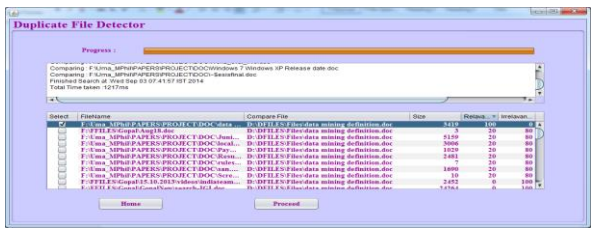

Fig.10. General Search

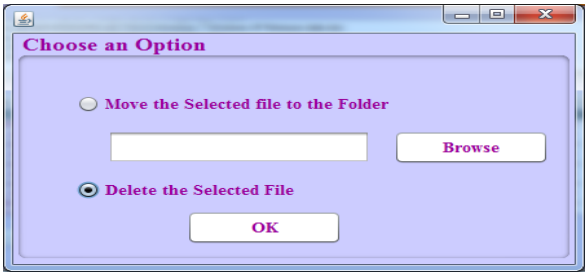

Fig.11. General Search- Deleting Duplicate Files

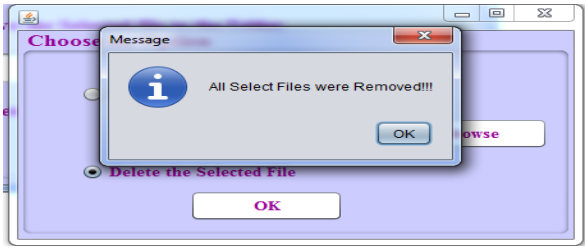

Fig.12. General Search – Final Result

Exact Search - Same File with Same Extension

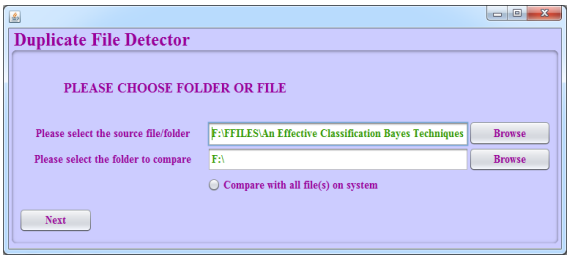

Fig.13. Exact Search- Same File with Same Extension

Fig.14. Exact Search- Same File with Same Extension

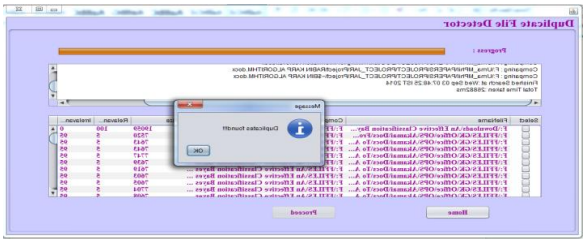

Fig.15. Exact Search- Same File with Same Extension

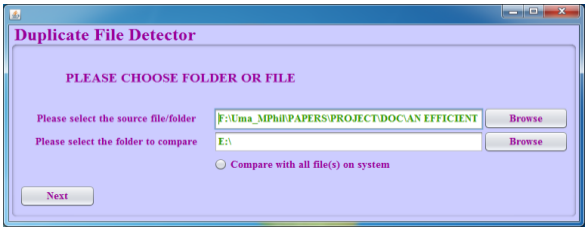

Fig.16. Exact Search-Same File with Different Extension

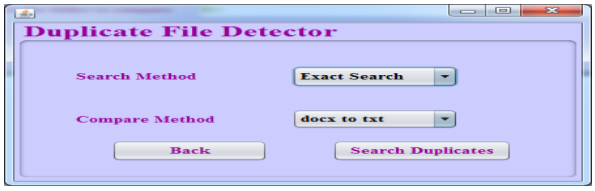

Fig.17. Exact Search-Same File with Different Extension

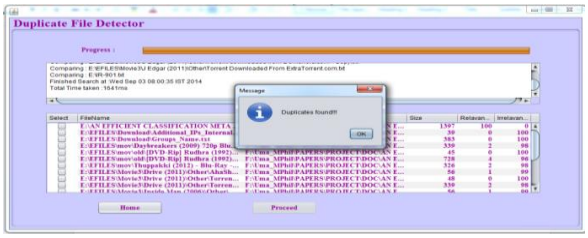

Fig.18. Exact Search- Same File with Different Extension

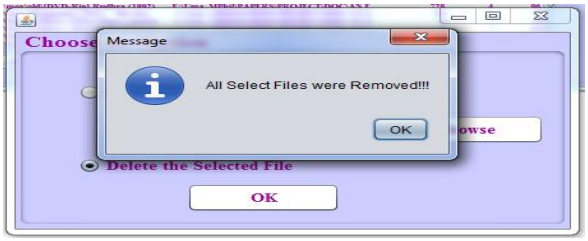

Fig.19. Exact Search- Same File with Different Extension

V. Conclusion and Future Work

Information retrieval (IR) is the activity of obtaining the information resources, which is relevant to a user's need from a collection of information resources. This search can be based on full text or other content based indexing. In this research work, two different string searching algorithms namely Boyer Moore Horspool algorithm and Knuth Morris Pratt algorithm have been discussed and their performance measures shows that, Knuth Morris Pratt algorithm performed better than other algorithm. New string searching algorithm namely W2COM has been proposed which efficiently performed the task. By analyzing the experimental results, it is clear that the W2COM technique needs minimum search time for searching the duplicate files. In terms of memory utilization, it takes less amount of memory space when compared to other algorithms.

Recently, various number of duplicate file finder tools are available but the main disadvantage of these tools are, they check the file content with same categories. DFD tool helps the user to search and delete duplicate files in the system at minimum time. It also helps to delete the duplicate files not only with the same categories but also with different categories. In future, indexing technique can be applied to the DFD tool for searching the duplicate files faster.

Reference

[1] Ababneh Mohammad, OqeiliSaleh and Rawan A Abdeen, Occurrences Algorithm for String Searching Based on Brute-Force Algorithm, Journal of Computer Science, 2(1): 82-85, 2006.

[2] Ankur Singh Bist, Pattern Matching Algorithms for Computer Virus Detection, International Journal of Engineering Sciences & Research Technology, Singh 2(1), P.No.28-29, 2013.

[3] Abdulwahab Ali Al-Mazroi and Nur'aini Abdul Rashid, A Fast Hybrid Algorithm for the Exact String Matching Problem, American Journal of Engineering and Applied Sciences 4 (1): 102-107, 2011.

[4] Anthony Scime, NilaySaiya, Gregg R. Murray and Steven J. Jurek, "Classification Trees as Proxies", International Journal of Business Analytics (IJBAN), volume 2, issue 2.

[5] Bin Wang, Zhiwei Li, Mingjing Li and Wei-Ying Ma, Large-Scale Duplicate Detection for Web Image Search, Multimedia and Expo, IEEE International Conference, 353-356, 2006

[6] Bo Hong and DemynPlantenberg, Duplicate Data Elimination in a SAN File System, In Proceedings of the 21st IEEE / 12th NASA Goddard Conference on Mass Storage Systems and Technologies, 2004.

[7] Christian Charras, Thierry Lecroq and Joseph Daniel, A Very fast string searching algorithm for small alphabets and long patterns, Combinational Pattern Matching, 9th Annual Symposium, CPM 98 Piscataway, New Jersey, USA, 2005.

[8] Craig A. N. Soules, Gregory R. Ganger, Connections: Using Context to Enhance File Search, ACM SIGOPS Operating Systems Review - SOSP '05,Volume 39, Issue 5, 2005.

[9] Gregory Ramsey and Sanjay BapnaText mining to identify customers likely to respond to cross-selling campaigns: reading notes from your customers, International Journal of Business Analytics (IJBAN), volume 3, issue 2

[10] George Forman, KaveEshghi andJaapSuermondt, Efficient Detection of Large-Scale Redundancy in Enterprise File Systems, ACM SIGOPS Operating Systems, Volume 43 Issue 1, 84-91 2009.

[11] Harish B S, S Manjunath and D S Guru, Text Document Classification: An Approach Based on Indexing,International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.2, No.1, 43-62, 2012.

[12] HemlataSahu, ShaliniShrma, SeemaGondhalakar, A Brief Overview on Data Mining Survey, International Journal of Computer Technology and Electronics Engineering, Volume 1, Issue 3, 114-121, 2000.

[13] Ian H. Witten and Eibe Frank, Data Mining Tools and Techniques practical Machine Learning, 2011 (Book).

[14] JormaTarhio and EskoUkkonen, Approximate Boyer-Moore String Matching, SIAM Journal on Computing, Volume 22 Issue 2, 243 – 260, 1993.

[15] MilošRadovanović, andMirjanaIvanović, Text Mining: Approaches and Applications, Volume. 38, No. 3, 227-234, 2008.

[16] Olivier Danvy, Henning Korsholm Rohde, On Obtaining the Boyer-Moore String-Matching Algorithm by Partial Evaluation, Journal of Information Processing Letters, Volume 99 Issue 4, 158-162, 2005.

[17] Robert S. Boyer and J. Strother Moore, A fast string Searching Algorithm, Communication of the ACM, Volume 20 Issue 10, 762-772, 1977.

[18] Simon Wahlström, Evaluation of String Searching Algorithms, 2004.

[19] SriharshaOddiraju, BOYER-MOORE, December 16,

2011.
[20] StephaneDucasse, Matthias Rieger& Serge Demeyer, A Language Independent Approach for Detecting Duplicated Code, Proceeding IEEE International Conference on Software Maintenance, 109 – 118, 1999.
[21] Thierry Lecroq, A variation on the Boyer-Moore algorithm, Journal of Theoretical Computer Science, Volume 92 Issue 1, 119-144, 1992.
[22] Vijayarani S, and Muthulakshmi M, Comparative Study on Classification Meta Algorithms, International Journal of Innovative Research in Computer and Communication Engineering, Vol. 1, Issue 8,1768-1772, 2013.
[23] Vishal Jain, Mayank Singh, Ontology Based Information Retrieval in Semantic Web: A Survey, International Journal of Information Technology and Computer Science(IJITCS), vol.5, no.10, pp.62-69, 2013. DOI: 10.5815/ijitcs.2013.10.06
[24] Divya K.S., Dr. R. Subha, Dr. S. Palaniswami, Similar Words Identification Using Naive and TF-IDF Method,International Journal of Information Technology and Computer Science(IJITCS), 2014, 11, 42-47, DOI: 10.5815/ijitcs.2014.11.06

Authors' Profiles

Dr. S.Vijayarani, MCA., M.Phil., Ph.D., working as Assistant Professor in the Department of Computer Science, School of Computer Science and Engineering, Bharathiar University, Coimbatore, Tamilnadu, India. Her fields of research interest are Privacy Preserving Data mining, Text Mining, Web Mining, Image Mining, DataStreams, Information Retrieval and Big Data. She has authored a book and published more than 80 papers in the international journals and conferences.

Ms. M. Muthulakshmi M.Sc has completed M.Phil in Computer Science in the Department of Computer Science, Bharathiar University, Coimbatore. Her fields of interest are Data Mining, Text Mining and Semantic web mining. She has published papers in International journals and conferences.

How to cite this paper: S. Vijayarani, Ms. M.Muthulakshmi,"An Efficient String Matching Technique for Desktop Search to Detect Duplicate Files", International Journal of Information Technology and Computer Science(IJITCS), Vol.9, No.7, pp.69-76, 2017. DOI: 10.5815/ijitcs.2017.07.08