Volume 1, No. 3, Sept-Oct 2010
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

Improved Suffix Tree Clustering for Efficient Document Clustering

Sonia*
YMCA University of Science and Technology
Faridabad, India
soniabansal2@yahoo.com

Niranjan Kumar
ONGC, Mumbai
India
Jmi_bansal24@yahoo.com

Abstract: Document clustering is a technology that puts pages into groups and is useful for categorizing, organizing, and refining search results. When clustering using only documents, Suffix Tree Clustering (STC) outperforms other clustering algorithms by making use of phrases and allowing clusters to overlap. STC is a linear time clustering which is based on identifying phrases that are common to groups of documents. STC treats a document as a string, making use of proximity information between words, at the same time, it is incremental. Suffix Tree Clustering has been proved to be a good approach for documents clustering. This paper introduces the suffix tree based document clustering with cluster ranking function and a new ranked list of clusters after applying in the algorithm is introduced to overcome the problems with overlapping clusters. Using this method, we can get a better clustering result and effective number of clusters.

Keywords: Suffix Tree; Document Clustering; STC; Web Document; Ranked cluster.

I. INTRODUCTION

Document clustering has been studied intensively recently because of its wide applicability in areas such as web mining, search engines, information retrieval, and topological analysis. Clustering documents into groups can organize large bodies of text for efficient browsing and searching [5,6]. A lot of different texts clustering algorithms have been proposed in the literature, including Agglomerative Hierarchical Clustering (AHC) [4,11], Scatter/Gather [7,8] and K-Means [5]. Any clustering technique replies on four concepts: data representation model, similarity measure, clustering model and clustering algorithm that generates the clusters using the data model and the similarity measure. The Vector Space Document (VSD) model is a very widely used data representation model for document classification and clustering today [19]. The common framework of this data model starts with a representation of any document as a feature vector of the words that appear in documents of the data set. The term-weights of the words are also contained in each feature vector. The similarity between two documents is computed with one of several similarity measures based on two corresponding feature vectors, e.g. cosine measure, Jaccard measure, and Euclidean distance measure. Inverted indices assume that the text can be seen as a sequence of words. This restricts somewhat the kinds of queries that cab be answered. Other queries such as phrases are expensive to solve. Moreover, the concept of word does not exist in some applications such as genetic databases. Further there are many key requirements for Web document clustering like relevance, browsable summaries, overlapping, snippet tolerance, speed and incrementally. An incremental and linear time algorithm named Suffix Tree Clustering (STC) has been defined to fulfill these requirements. Next section describes the related work on suffix tree clustering. In the section III Suffix Tree Document Clustering model with algorithm has been defined. The web document clustering using suffix tree is introduced under section IV. Section V concludes and gives detail of future work.

II. RELATED WORK

Document clustering has been traditionally investigated as a means of improving the performance of search engines by pre-clustering the entire corpus [2,25], and a postretrieval document browsing technique as well [1, 3, 9]. The methods used for document clustering covers several research areas, such as database, information retrieval, and artificial intelligent including machine learning and natural language processing. STC is a linear time clustering algorithm, which is based on identifying phrases that are common to groups of documents [9,10]. A phrase is an ordered sequence of one or more words. There are two distinct characteristics attracting to study suffix tree document model and STC algorithm. Firstly suffix tree document model proposed a new flexible n-grams approach to identify all overlap nodes (phrases) among the documents as Longest Common Prefixes (LCPs). Secondly, one or several phrases are naturally selected to generate a topic summary to label the corresponding cluster during building the clusters[18]. STC algorithm clusters standard documents as well as document snippets. In suffix tree document model, a document is considered to be a set of suffix substrings, the common prefixes of the suffix substrings are selected as phrases to label the edges of a suffix tree [12,22]. STC algorithm is developed based on this model and works well in clustering Web document snippets returned from several search engines. The suffix tree model and STC have been defined [9]. When using only textual information for clustering has shown that STC outperforms other algorithms but suffix tree based method suffers from large memory requirements and poor locality characteristics. SHOC [14] uses suffix array for phrase extraction and organizes the snippets in a hierarchy via an SVD (Singular Value Decomposition) approach. Lingo [16] uses SVD on a term-document matrix to find meaningful long labels, generates flat clustering result. Zeng [15] re-formalizes the clustering

problem as a salient phrase ranking problem. It uses phrases rather than words and that it allows clusters to overlap. A co-occurrence based hierarchical clustering method is used to group search results into hierarchical and overlapping clusters. CoHC outperforms all other clustering algorithms [23].

III. SUFFIX TREE DOCUMENT CLUSTERING MODEL

A suffix tree is a data structure that admits efficient string matching and querying. Suffix trees have been studied and used extensively, and have been applied to fundamental string problems such as finding the longest repeated substring, strings comparing and text compression [13,17] Zamir and Etzioni presented a Suffix Tree Clustering (STC) algorithm on document clustering [9,10]. STC is a linear time clustering algorithm that is based on a suffix tree which efficiently identifies sets of documents that share common phrases. STC treats a document as a string, making use of proximity information between words, at the same time, it is incremental and has an *O(n)* time complexity, where *n* is the total number of words in all combined document snippets.

A suffix tree document is a concept that describes how a set of meaningful features is extracted from a document. Suffix tree document model considers a document $d=w_1w_2.....w_m$ as a string consisting of words w_i, not characters (i=1,2,…,m). A suffix tree of document *d* is a compact trie containing all suffixes of document *d*. The nodes of the suffix tree are drawn in circles. In Fig. 1 Each internal node has at least two children. Each edge is labeled with a non-empty substring of a document called a phrase, and its suffix node is labeled by the phrase too. Then each leaf node in the suffix tree designates a suffix of a document, each internal node represents an overlap phrase shared by at least two suffixes. The more internal nodes shared by two documents, the more similar the documents tend to be. Each internal nodes is attached with a box respectively. The numbers in the box designate the documents which have traversed the corresponding node. Each upper number designates a document identifier, the number below designates the traversed times of the document.

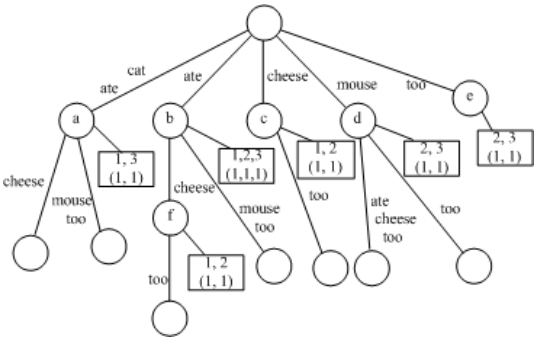

Fig. 1: The suffix tree of tree documents ``cat ate cheese", ``mouse ate cheese too" and ``cat ate mouse too"

STC is a linear time clustering algorithm that is based on identifying the phrases that are common to groups of documents. STC approach first identifies sets of documents that share phrases and extracts these phrases as candidate cluster labels. Base clusters are created according to these cluster labels. Candidate cluster labels are ranked and some of them are selected as the final cluster labels. A phrase in our context is an ordered sequence of one or more words. We define a base cluster to be a set of documents that share a common phrase. The STC has four logical steps: (1) document cleaning (2) identifying base clusters using a suffix tree, and (3) combining these base clusters into clusters (4) produced the scored cluster in the form of ranked document clusters to the user as shown in Fig 2**.** A Suffix Tree Clustering (STC) algorithm does not treat a document as a set of words but rather as a string, making use of proximity information between words [26]. STC relies on a suffix tree to efficiently identify sets of documents that share common phrases and use this information to create clusters and succinctly summarize their contents for users.

STC is a linear time clustering algorithm that is based on a suffix tree which efficiently identifies sets of documents that share common phrases. The algorithm is defined in next section has the important characteristic that the outputted clusters can have overlapping documents.

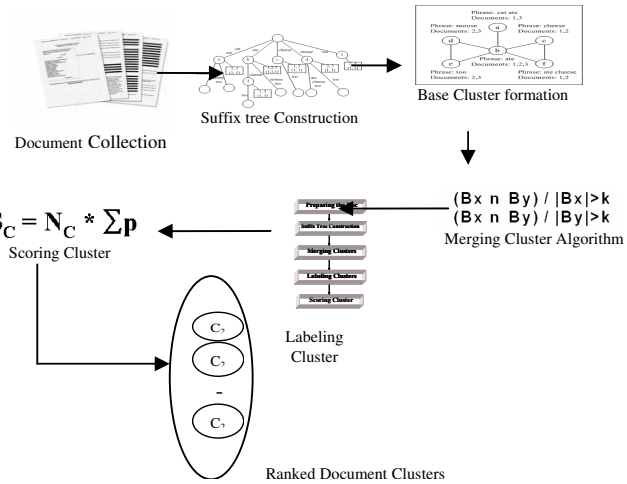

Fig. 2. Framework of Suffix Tree Document Clustering Model

A. STC Algorithm for Document Clustering

The original STC algorithm is developed based on the suffix tree document model and defined in three steps as follows:

Step 1. The common suffix tree generating A suffix tree S for all suffixes of each document in $D=\{d_1,d_2,.....,d_N\}$ is constructed. Each internal node containing at least two different documents is selected to be a base cluster, which is composed of the documents designated by the box, and labelled by the phrase of the node.

Step 2. Base cluster selecting Each base cluster B is assigned a score s *(B),*

$$s(B)=|B|.\, f(|P|) \qquad (1)$$

where $|B|$ is the number of documents in B, and $|P|$ is the number of words in P. Then all base clusters are sorted by the scores, and the top k base clusters are selected for cluster merging in Step 3.

Step 3. Cluster merging A similarity graph consisting of the k base clusters is generated. An edge is added to connect two base clusters B_i and B_j if the Jaccord coefficient of B_i and B_j is larger than 0.5, say, when.

$$\frac{|B_i \cap B_j|}{} > 0.5 \qquad (2)$$

$|B_i \cap B_j|$

The connected components in this graph form the final clusters. For example, the nodes a, b, c, d, e, and f are selected to be the base clusters in the suffix tree of Figure 3. Finally the 6 base clusters form a final cluster after cluster merging is shown in Fig. 3.

Step 4. Produce the scored cluster in the form of ranked document clusters to the user. Hence each cluster is sorted by its score and presents the clusters to the user ranked by highest score.

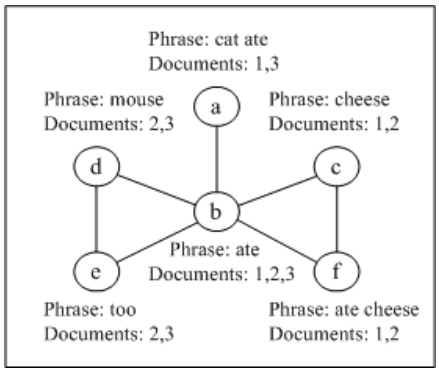

Fig. 3: The base cluster graph

It would be better to attempt to rank the clusters by some measure that attempts to present clusters that do not overlap much with previous clusters.

Step 5. Clusters using this algorithm have overlapping documents and common for one highly ranked document to contain similar documents to another highly ranked document. So the clusters are produced to user in the form of ranked list.

IV. WEB DOCUMENT CLUSTERING USING SUFFIX TREE ALGORITHM

A Suffix Tree Clustering (STC) algorithm does not treat a document as a set of words but rather as a string, making use of proximity information between words [9]. STC relies on a suffix tree to efficiently identify sets of documents that share common phrases and use this information to create clusters and succinctly summarize their contents for users. STC is a linear time clustering algorithm that is based on a suffix tree which efficiently identifies sets of documents that share common phrases. It ensures that a large number of substantial clusters can be generated on the web, each of which can be labeled accurately [19,20]. The Web document clustering algorithm has many steps as shown in Figure 4.

A. Preparing the Documents

Retrieve each document snippet from any search engine and parse, filter, and stem the results. Results are received from search engine in a multi-threaded fashion, and the actual construction of suffix tree can occur concurrently as documents are retrieved over the network.

B. Suffix Tree Construction

A suffix tree is a data structure that essentially has one node for every possible phrase in a collection of documents. A suffix tree has nice feature that any string can be inserted onto the tree in linear time by beginning at the root of the tree, and using logic to decide whether or not to add a new branch of the tree to traverse or split into multiple branches.

C. Merging Clusters

Once the suffix tree has been constructed, each node on the suffix tree that contains multiple documents can feasibly be considered a cluster, i.e. it is a string that is contained in multiple documents. To reduce the number of clusters, under go cluster merge phase. By using the merging criteria one can decide to merge or not to merge the 2 given node. Merge the clusters x and y if $(N_x \cap N_y) / |N_x > k$ and $(N_x \cap N_y) / |N_y > k$. N_x is the set of unique documents in cluster N_x, and k is a constant between 0 and 1. Effectively, this criterion requires that each cluster must have at least the fraction k of its documents in common with the other cluster. Hence maintain the data structure to do this efficiently and to determine the documents and labels corresponding to each merged cluster updated.

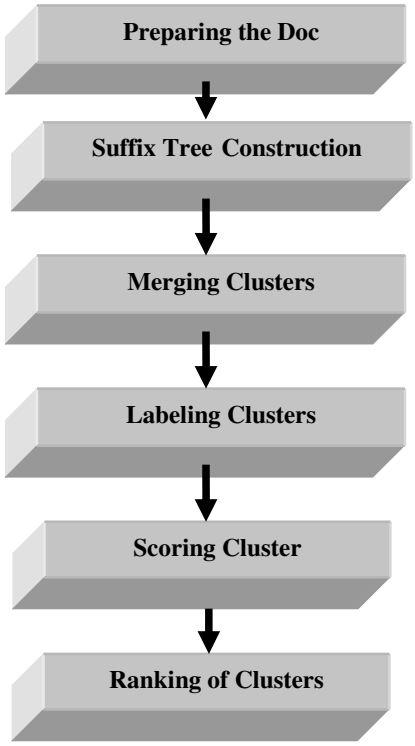

Figure 4. STC in Web document clustering

D. Labeling Clusters

After merging suffix tree nodes[21], a list of clusters has been generated, each of which corresponds to one or more labels in the original suffix tree and contains a list of documents that contain any of those labels

E. Scoring Cluster

Now score all resulting sorted clusters. A cluster with more documents should have the higher score than an

equivalent cluster with fewer documents. The factor that label contributes to score should be proportional to the expected probability of that label. Labels that are subset of one another are likely to be same merged cluster so not counted them double.

Score S_C for Cluster C is

$$S_C = N_C * \sum_{l_0}^{l_n} p(l_i)$$

Where N_C is the number of documents in cluster C. l_o and l_n are the labels in C and are not subsets of any other label in C. For each label l, the probability p(l) is define as

$$p(l) = \sum p(w)$$

The probability of string l is simply the product of the probabilities of each word w in l means independence of words in string.

Now the probability of each word w,

$$p(w) = \log(1/f_w) \text{ if } f_w > 0 \text{ and}$$
$$p(w) = \log(1/.5) \text{ if } f_w = = 0$$

where f_w is the frequency of w on the entire web.

F. Ranking of Clusters

A group of scored clusters are obtained after applying the document clustering algorithm each of which contains more or less relevant documents. By ranking the clusters we expect to determine reliable clusters and adjust the relevance score of documents in each ranked list such that relevant scores become more reasonable [24]. In order to find the ranked cluster of the query three measures are applied: Normalised Match Ratio (NMR), Normalised Order Ratio (NOR) and Log Odds Ratio (LOR).

NMR shows the number of terms related to the topic. It calculates the number of terms with high relevance with a given query and uses this to decide which cluster will be used. After which it shows them according to their rank. The NMR measures can be defined as

$$NMR = \frac{|Matches|}{|\text{Re } ferenceTermList|}$$

NOR shows the number of terms related to the topic within a selected cluster. It shows cluster according to the level of relevance within the selected clusters and shows the term list. It is defined as $NOR = \frac{|Matches|}{|\text{Re } ferenceClusters|}$

LOR is a statistical measure of how relevant a term is to a cluster. It is the probability value that shows the similarity among the terms and how much they are relevant to the topic. Its value can be expressed as a probability unlike the two prior estimated values.

LOR = (Odds of an event in one Group) / (Odds of it Occurring in another Group)

By applying each of three estimated measures the cluster that matches with the topic can be shown. These measures can be used intuitively. The ranking algorithm is defined as Ranking Algorithm = NMR ∩ NOR ∩ LOR.

This technique analyses the contents of each cluster to determine the top terms and associate them with probability values.

Hence each cluster is ranked by its score and presents the clusters to the user ranked by highest rank. It would be better to attempt to rank the clusters by some measure that attempts to present clusters that do not overlap much with previous clusters. This is important because clusters using this algorithm have overlapping documents and common for one highly ranked document to contain similar documents to another highly ranked document.

V. CONCLUSION

With suffix tree clustering (STC), search results can be clustered fast, automatically, and each cluster is labeled with a common phrase. Due to the large memory requirement of suffix tree, some other approaches have been proposed, with lower memory requirement. But unlike other algorithms, STC is an incremental algorithm and a promising approach to work on a long list of snippets returned by search engines. An approach for web search results clustering and labeling is based on suffix tree data structure. The approach is an incremental and linear time algorithm, with significantly lower memory requirements. This approach also labels every final cluster a common phrase, thus it is suitable for quickly browsing by users.

STC finds clusters based on the common phrases shared by documents. A suffix tree is a very efficient way to identify common phrases in documents, but suffix trees suffer from large memory requirements and poor locality characteristics. Our work presented in this paper is mainly focused on improving the effectiveness of document clustering algorithms. Efficiency optimization of the algorithm has been a target of our current work, both the time efficiency and the space efficiency. Moreover, Suffix arrays have been applied to clustering, are similar to suffix trees and perform the same function, but have significantly lower memory requirements. In the future work, mathematically well founded orthogonal clustering method can be introduced that improves the STC method. Clustering performance can also be improved further by using other types of information such as hyperlink information.

VI. REFERENCES

[1] W. B. Croft, "Organising and Searching Large Files of Documents". PhD thesis, University of Cambridge, October (1978).

[2] C.J.van Rijsbergen, "Information Retrieval", Butterworths, London, 2nd ed.(1979).

[3] W. B. Croft and R. H. Thompson, (I3R: A new approach to the design of document retrieval systems", Journal of the American Society for Information Science, pp.389-404, (1987).

[4] P. Willett, Recent, "Trends in Hierarchical Document Clustering", A Critical Review, Information Processing & Management Vol. 24, No. 5, (1988).

[5] Anil K. Jain, Richard C. Dubes, Algorithms for clustering, Prentice Hall (1988).

[6] R.B.Allen, P.Obry and M.Littman, "An interface for navigating clustered document set returned by queries", In Proceddings of ACM Conference on Organisational Computing Systems, pp.166-171, (1993).

[7] D. R. Cutting, D. R. Karger, and J. O. Pedersen, "Constant interaction-time Scatter/Gather browsing of very large document collections", In Proceedings of ACM SIGIR, pp.126-134, (1993).

[8] M. A. Hearst and J. O. Pedersen, "Reexamining the cluster hypothesis: Scatter/gather on retrieval results". In

Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 76-84, (1996).

[9] O. Zamir and O. Etzioni, "Grouper: A dynamic clustering interface to web search results". In Proceedings of the 8th International World Wide Web Conference, Toronto, Canada, May (1999).

[10] O. Zamir and O. Etzioni, "Web Document Clustering: A Feasibility Demonstration", In Proceedings of the 19th International ACM SIGIR Conference on Research and Development of Information Retrieval (SIGIR'98),(1998).

[11] C.W. Wen, et al., "A distributed hierarchical clustering system for web mining, International Conference on Web-Age Information Management (WAIM2001)", pp. 103–113, (2001).

[12] Y.Wang, M. Kitsuregawa, "Use link-based clustering to improve web search results, International Conference on Web Information Systems Engineering (WISE)", pp.119–128, (2001).

[13] Meyer zu Eissen and Stein, "Analysis of Clustering Algorithms for Web-based Search", Paderborn University, pp.168-178, (2002).

[14] K. Malde, E. Coward, and I. Jonassen, Fast sequence clustering using a suffix array algorithm, In *BIOINFORMATICS*, volume 19 (10), pp1221–1226, 2003.

[15] H. Zeng, Q. He, Z. Chen, and W. Ma, "Learning to cluster web search results", Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (2004).

[16] S. Osinski, J. Stefanowski and D. Weiss, Lingo: "Search results clustering algorithm based on singular value decomposition", In Proceedings of the International Conference on Intelligent Information Systems (IIPWM'04), Zakopane, Poland, (2004).

[17] Daniel Crabtree, Xiaoying Gao, Peter Andreae, Improving Web Clustering by Cluster Selection, *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI'05),* 2005

[18] Sven Meyer zu, Eissen Benno Stein, and Martin Potthast, The Suffix Tree Document Model Revisited, *Proceedings of the I-KNOW '05, 5th International Conference on Knowledge Management Journal of Universal Computer Science*, pp. 596-603, 2005.

[19] Yue Xu, Hybrid, "Clustering with Application to Web Mining", (2005).

[20] S. Sambasivam and N.Theodosopoulus, "Advanced Data Clustering Methods of Mining Web Documents", Information Science and Information Technology, Vol 3, pp.563-578, (2006).

[21] J.W.R. Li, "A New Clustering Merging Algorithm of Suffix Tree Clustering", In IFIP International Federation for Information Processing, Intelligent Information Processing III, eds. Z.Shi, Shimohara K, Feng D, Boston:Springer, vol.228, pp.197-203, (2006).

[22] Hung Chim and Xiaotie Deng, A New Suffix Tree Similarity Measure for Document Clustering, the International World Wide Web Conference Committee (IW3C2), WWW 2007 Banff, Alberta, Canada, May 8.12, 2007.

[23] Y. Zhang and B. Feng, "A Co-occurrence based Hierarchical Method for Clustering Web Search Results", Information Technology Journal, vol.7, (2008).

[24] M. H. Chehreghani, H. Abolhassani and M.H. Chehreghani, "Density link-based methods for clustering web page", Decision Support Systems (2009) Article in Press.

[25] Ok-Ran Jeong, Sang-Won Lee, "An Effeicient Clustering Framework for Relevant Web Information", (2009).

[26] Steve Branson and Ari Greenberg, Clustering Web Search Results Using Suffix Tree Methods, Stanford University, Final Project Report or cite to

http://www.stanford.edu/class/archive/cs/cs276a/cs276a.1032/projects/reports/arigreen-branson.pdf