# Topical Clustering of Search Results using Suffix Tree Clustering

Sunil D. Jejurkar
Government
Engineering College,
Aurangabad, India

Vivek P. Kshirsagar
Government
Engineering College,
Aurangabad, INDIA

## ABSTRACT
In Today's world, with the increased use of internet the large volume of data is stored on World Wide Web. To use this large data the different search engines are provided. But the accuracy of the data is again based on the appropriate search query submitted by the user to search engine. Depending on the search query the search engine retrieves the massive amount of relevant data by using different algorithms such as page rank algorithm or relevancy algorithm. Further, the returned results decide the performance as well as the efficiency of the search engine. Search result clustering problem means clustering the search results returned by the search engine.

In this paper a comparative analysis of Suffix Tree Clustering algorithms is done to decide the how accurately it clusters the search results i.e. an empirical analysis which is done by using standard datasets.

## General Terms
Algorithms, Experimentation.

## Keywords
Suffix Tree Clustering, Search Results.

## 1. INTRODUCTION
The existing search engines are generally used to access vast amount of information which is stored on the World Wide Web. If anyone want to search for something then an search engine is used to get required results from the internet. The user has to specify the proper search query to any common search engine and depending on the search query a flat list of relevant search results is computed using page rank or relevance algorithm and same displayed to user. Then user gets required information. Any common search engine is focuses on the availability of different search results. The another problem with search result is it shows trending results at top level, because of it non-trending but important information is get lower ranking in the search results.

Information retrieval and ranking functions are very important to the search engines. The organization and presentation of the results is also very important and could considerably have an effect on the utility of the search engine.

The general concept of search engine is to focus on words rather than its meaning. When user search for query a flat list of relevant documents is retrieved, user has to scan this list from top to bottom to get the required results. But in clustering the search results are grouped under some cluster label. User has to scan all cluster labels and then select the relevant cluster and scan that cluster results only. The overall time required to scan whole list is gets reduced [1][14].

The different search engines available are facing the problem of lexical ambiguity. It can be defined as the consequences of the low number of query results entered on average by web user.

This paper is organized in Six sections, Section 1 deals with Introduction Section 2 gives The related Study, Section 3 briefs the Implementation of existing Algorithms and Section 4 shows System Analysis and graphs and Section 5 presents the conclusion.

## 2. RELATED WORK
The suffix tree is used to perform different operations on string. The suffix tree clustering gives the linear time complexity of *O(n),* because of linear complexity the response time of suffix tree clustering is very low and it is on the top of all other clustering algorithms.

## 2.1 SUFFIX TREE CLUSTERING ALGORITHM
STC algorithms consist of following logical steps:

a. *Document "Cleaning":* Retrieving the document snippets from Google and parsing and stemming the results.

b. *Generalized Suffix Tree Construction:* The Suffix Tree construction is done in this phase.

c. *Identifying Base Clusters:* In the Suffix Tree each node represents a base cluster.

d. *Combining Base Clusters into Clusters:* The nearly identical base clusters are combined into one cluster.

### *2.1.1 Document "Cleaning"*
The document cleaning is the basic step of any document processing algorithm. The document cleaning mainly consist of following phases

i. HTML tag cleaning: The input to system is HTML web pages. So, in this step all HTML tags from documents are removed.

ii. Then the sentence boundries are identified using punctuation and HTML tags.

iii. All non-word tokens are removed from documents such as numbers, punctuation marks, etc.

iv. Steaming is also important step of document processing. Steaming algorithm is a process of linguistic normalization in which the variant forms of the word are reduced to a common form. The main advantage of stemming is to improve retrieval effectiveness and to reduce the size of indexing. As shown in figure 1, the

various forms of "accept" words are converted to its original plural to singular form.

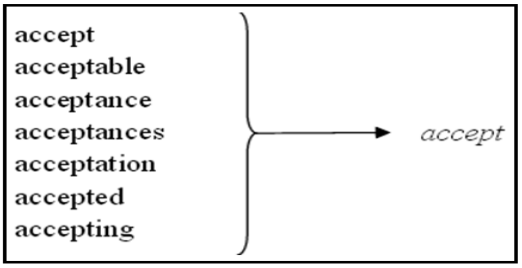


**Fig 1: Process of stemming**

### 2.1.2 Generalized Suffix Tree Construction

In the process of constructing generalized suffix tree the first step is construction of suffix tree. The suffix tree is a data structure which contains all the suffixes of a given string, so as to run many important string operations more efficiently. The string may be a string of characters or string of words. The suffix tree for the string S is defined as a tree such that: the paths from the root to the leaves have a one-to-one relationship with the suffixes of S, all edges are labelled with nonempty strings, all internal nodes (except perhaps the root) have at least two children. In this paper documents treats String as collection of words, not characters.

- A suffix tree is a rooted, directed tree in which every internal node has at least two children nodes.
- Every edge in the tree is labelled with a non-empty sub-string of S (Original string).
- The label of a node is defined as the path from root node to the node. The node name is the common phrase shared by all lower level documents which originated from same node.
- The different edges originating from same node never has similar edge-labels, this feature makes the tree mere generalized as well as compact.

The suffix tree in this paper is constructed using the Ukkonen's algorithm. It is best algorithm for compact suffix tree construction. Figure 2, shows the example of generalizes suffix tree for strings "cat ate cheese", "mouse ate cheese too" and "cat ate mouse too".

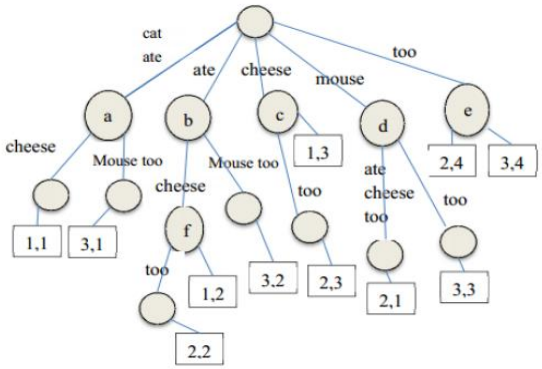


**Fig 2: Example of generalized suffix tree for the givenstrings.**

In this figure, nodes are represented by circles; the document list is enclosed in the square bracket. The every node of the suffix tree represents common phrase shared by a group of documents.

### 2.1.3 Identifying Base Clusters

The suffix tree is constructed for all the documents present in document collection. The every node of suffix tree represents a common phrase shared by a group of documents. The node label is nothing but common phrase. These nodes are called as base clusters.

Each base cluster assigned a score *s(B)* that is a function of the number of documents it contains and the words that make up its phrase. The function is given in (1):

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

Here |B| indicates the number of document in a base cluster *B* and |P| is the number of word in a phrase *P*. The base clusters identified in the Figure (Fig 2) are shown in table (Table 1)

**Table 1: Representation of nodes and corresponding clusters**

| Node | Phrase | Documents |
|---|---|---|
| a | cat ate | 1,3 |
| b | ate | 1,2,3 |
| c | cheese | 1,2 |
| d | mouse | 2,3 |
| e | too | 2,3 |
| f | ate cheese | 1,2 |

### 2.1.4 Combining Base Clusters into Clusters

A binary similarity measure is defined between base clusters to decide whether to allow merging of it or not. The binary similarity will be 1 if the conditions in formulas (2) and (3) are fulfilled.

$$|B_m \frown B_n| / |B_m| > 0.5 \qquad (2)$$
$$|B_m \frown B_n| / |B_n| > 0.5 \qquad (3)$$

Otherwise their similarity will be defined as 0. This step is presented on the figure (Fig 3). Each cluster consists of the union of the document of all its base clusters. This figure explains the base cluster graph of the six base clusters.

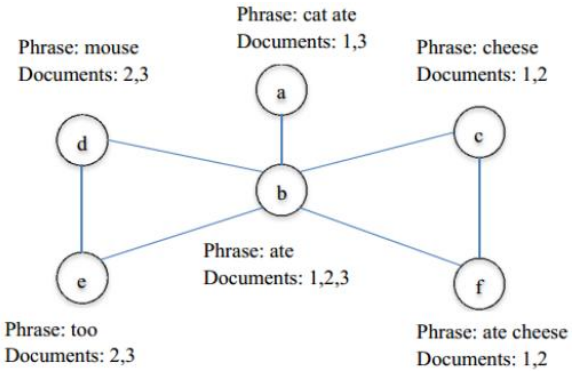


**Fig 3: Result of combining based clusters**

## 3. SYSTEM IMPLEMENTATION

In this paper, three popular algorithm of search result clustering are implemented. The different steps of the execution of the procedure are as follows:

1. Retrieve search results from any common search engine providing a search query.
2. Then select the algorithm using which the search result clustering to be done.
   As shown in Figure 1, depending on the search query and maximum number of search results expected, the selected

search engine returns the search results. Then an proper algorithm is selected for clustering the search results.

3. Then run that algorithm gives the clustered search results. The cluster name is nothing but the generalized topic name of the group of search results
   As shown in Figure 2, the "Clustered Search Results" tab gives the all information about the number of cluster s formed with their names as well as contents of each cluster with search result title and link belongs to each cluster. (1. The total number of clusters formed with their names and attributes, 2. The detailed view of the cluster formed with search title and search link.)

Then depending on the different computed clusters the result and analysis is performed.

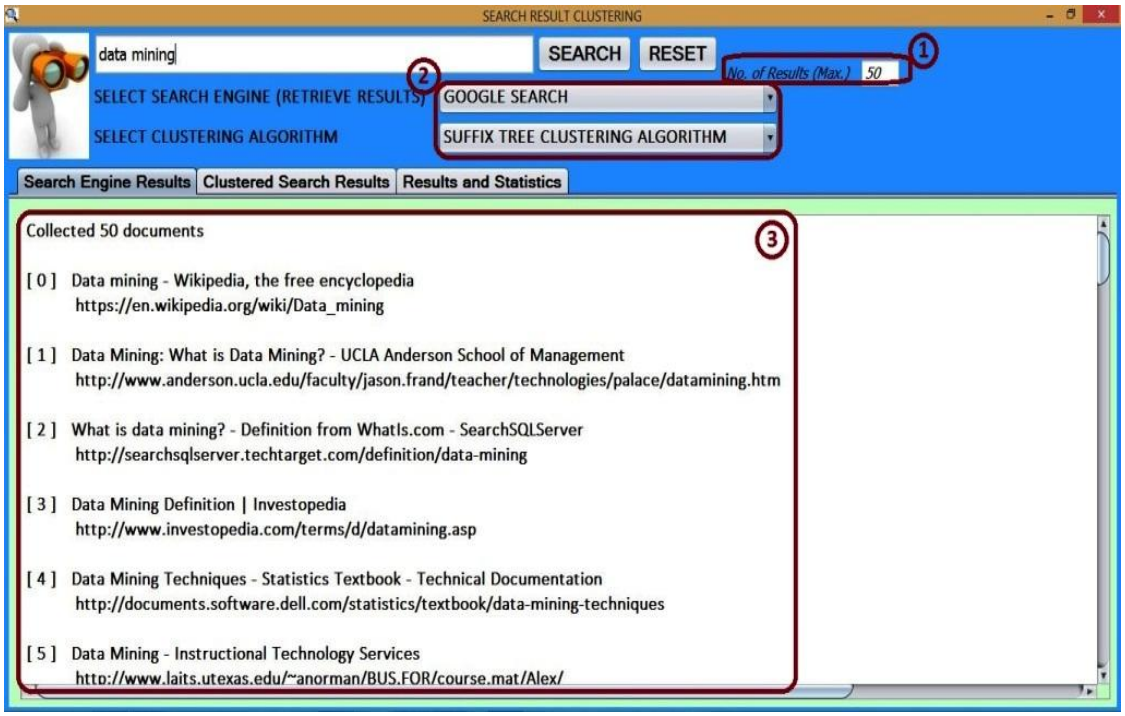


**Fig 4: Retrieve the search results from any common search engine and pass it to algorithm for search result clustering.**

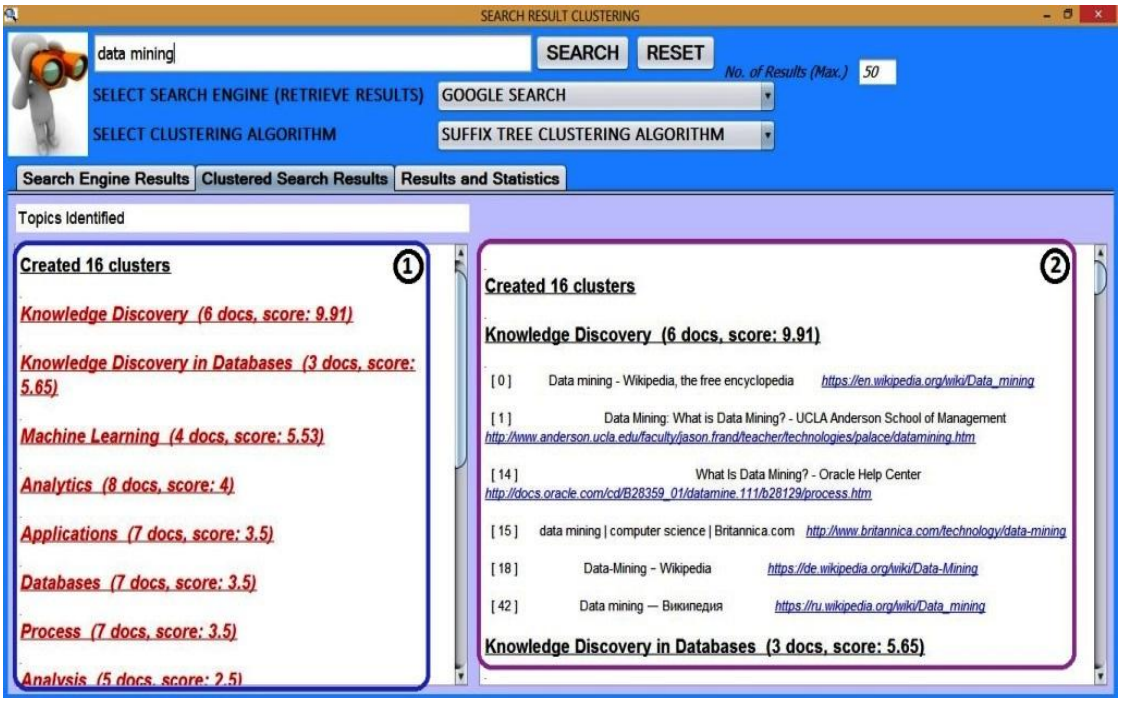


**Fig 5: Clusters formed for query "data mining" using the returned search results.**

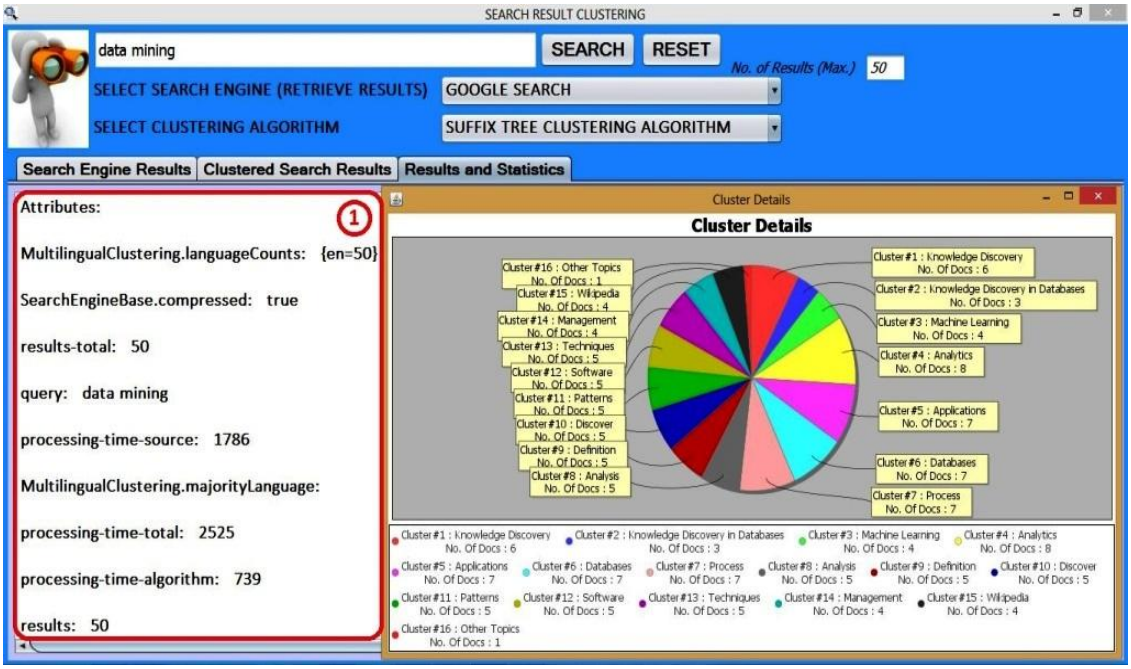


**Fig 6: Results and statistics of the clustering algorithm w.r.to the time and names as well as snippets covered.**

## 4. ANALYSIS AND GRAPHS

The different types of results analysis are performed such as number of cluster formed with unique name, the time required to retrieve the data from source, the time required to execute the algorithm, the overlapping clusters, the coverage of small outlier, format of cluster label and so on.

### 4.1 Number of Clusters Formed

As shown in the Fig 7, the number of cluster created in Suffix Tree Clustering algorithm is less. The clusters formed in STC algorithm are overlapping in nature. The clusters label created in STC are small and accurate while in K-Means it creates descriptive as well as lengthy cluster labels.

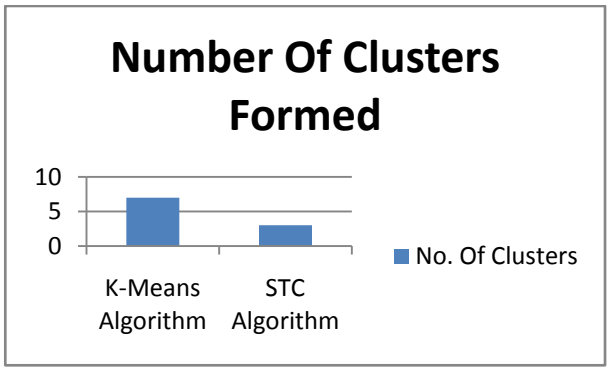


**Fig 7: Number of clusters formed with respect to clustering algorithm.**

### 4.2 The Processing Time Required

As shown in Fig 8, the algorithm processing time graph is shown (i.e. the time required for clustering the search results returned from search engine). The time required to retrieve the results from any common search engine is not considered. The processing time required for the execution of K-Means clustering algorithm is more than Suffix Tree Clustering algorithm. So, here Suffix Tree Clustering algorithm is working efficiently in minimum time frame.

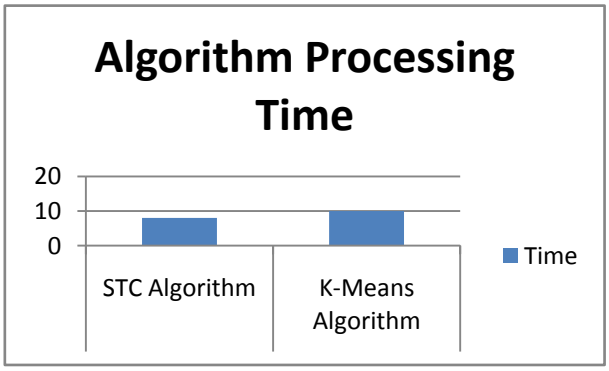


**Fig 8: Processing Time Required for Algorithm Execution.**

## 5. CONCLUSION

The numbers of algorithms are available for clustering the documents having their own merits and demerits. The Suffix Tree Clustering algorithm produces meaningful clusters with respect to the search query. The Suffix Tree Clustering algorithms returns less clusters compared to others but, the formed clusters are small as well as very appropriate in nature. The cluster labels produced in other algorithms such as the Lingo algorithm, K-Means algorithm are descriptive but lengthy. The Scalability is high in Suffix Tree Clustering compared to K-Means algorithm. Also the clusters formed in the Suffix Tree Clustering are overlapping in nature.

Future work may include the increase in cluster label quality. The cluster labels are directly decided using common phrases of suffix tree nodes. By considering the stop-words list the cluster labels can be made more descriptive. The another advancement is possible i.e. applying the algorithm to original web documents rather than the returned snippets to get more relevant results in one cluster.

## 6. REFERENCES

[1] Oren Zamir and Oren Etzioni. Document Clustering: A Feasibility Demonstration. Proceedings of the 19th International ACM SIGIR Conference on Research and Development of Information Retrieval, 1998, pp 46-54.

[2] Oren Zamir and Oren Etzioni. Grouper: A Dynamic Clustering Interface to Web Search Results. WWW8/Computer Networks, Amsterdam, Netherlands, 1999.

[3] Oren E. Zamir. Clustering Web Documents: A Phrase-Based Method for Grouping Search Engine Results. Doctoral Dissertation, University of Washington, 1999.

[4] Scatter/gather a cluster based approach to browsing large document collections. Douglassr cutting,David R.Karger ,Jan O Pederson,15 annual International SIGIR 92,ACM 0-89791-542-0912/0006/0318.

[5] Antonio Di Marco and Roberto Navigli, Clustering Web Search Results with Maximum Spanning Trees other publication details.

[6] Ke,W., Sugimoto, C.R., Mostafa, J.: Dynamicity vs. effectiveness: studying online clustering for scatter/gather. In: Proc. of SIGIR 2009, MA, USA, 2009, pp. 19–26.

[7] Carpineto, C., Osinski, S., Romano, G.,Weiss, D.: A survey of web clustering engines. ACM Computing Surveys 41(3), 2009, pp. 1–38.

[8] Kamvar, M., Baluja, S.: A large scale study of wireless search behavior: Google mobile search. In: Proc. of CHI 2006, New York, NY, USA, 2006, pp. 701–709.

[9] Osinski, S., Weiss, D.: A concept-driven algorithm for clustering search results. IEEE Intelligent Systems 20(3), 2005, 48–54.

[10] Sanderson, M.: Ambiguous queries: test collections need more sense. In: Proc. of SIGIR 2008, Singapore, 2008, pp. 499–506.

[11] Chen, J., Za¨ıane, O.R., Goebel, R.: An unsupervised approach to cluster web search results based on word sense communities. In: Proc. Of WI-IAT 2008, Sydney, Australia, (2008),. pp. 725–729.

[12] Zhang, X., Hu, X., Zhou, X.: A comparative evaluation of different link types on enhancing document clustering. In: Proc. of SIGIR 2008, Singapore, 2008,. pp. 555–562.

[13] iBoogie – meta search engine with automatic document clustering. http://www.iboogie.tv/.14. Inducing word senses to improve web search result clustering.

[14] Robert Navigli and Giuseppe Crisafulli department of Informatics,Rome,Proceedings osf the 2012 Conference on EMpherical Methods in Natural Language Processing,Pg 116-126 MIT,USA OCT9-11 2010 @)ACL.