# Knee finding based Optimized K-Means for Web Search

S. Poomagal[1], T. Hamsapriya[2]
[1]Department of Computer and Information Sciences
[2]Department of Information Technology
PSG College of Technology, Coimbatore
Tamilnadu, India
poomagal_sam@yahoo.co.in

***ABSTRACT:*** *With the vast amount of information available online, searching results for a given query requires the user to go through many titles and snippets. This searching time can be reduced by clustering search results into clusters so that the user can select the relevant cluster at a glance by looking at the cluster labels. For web page clustering, terms (features) can be extracted from different parts of the web page. Giansalvatore, Salvatore and Alessandro [1] have extracted terms from entire web page for clustering. Number of terms returned in this case is more and it produces lengthy vectors. To reduce the size of the vector, Stanis law Osinski et al.,[2] and Ahmed Sameh and Amar Kadray [3] have considered terms from the snippets.*

*In this work, we extracted terms from Uniform Resource Locator* (*URL*)*, Title tag and Meta tag and we used optimized K-means algorithm for clustering. Optimization of K-means algorithm is done by selecting the number of clusters using knee finding algorithm instead of selecting it randomly. We compared our method with existing methods in terms of Intra-cluster distance and Inter-cluster distance.*

**Keywords:** Stemming, Stop words, Snippets, Clustering, Centroid, Queries, Intra-cluster distance, Inter-cluster distance

**Received:** 29 December 2010, Revised 5 February 2011, Accepted 11 February 2011



## 1. Introduction

The growth of World Wide Web (WWW) is voluminous in the past decades. Huge volume of information is becoming available electronically over the Web. This tremendous volume poses challenges to the performance and scalability of web search engines. In addition to these technical issues, it also creates a problem for the user to browse.

When the user gives a query, search engine returns millions of documents and sorts it using the rank of the documents. Many algorithms [4-9] were introduced in the literature to calculate the rank based on the popularity of a document.

Even after ranking the documents using algorithms, searching is very tedious; since the number of search results displayed on each result page of a search engine is only 10 and the users have the habit of seeing only the first 10 or 20 results. Sometimes it is not possible to display the relevant documents within 20 results.

These problems can be overcome by clustering the search results and assigning labels to each cluster so that the user can select the relevant cluster by looking at the cluster labels.

### 1.1 Overview of KOPKWS

In our method, we have extracted terms from URL, Title tag and Meta tag. URL is selected since it provides identification for a web page and the terms in it are informative. Title tag is selected since it includes the title of a page and always the title of a page has useful terms in it. Meta tag has greater significance in clustering the web pages since the major use of Meta tag on any web page is to include the keywords which are used in that page.

Once the terms are extracted from the web pages, removal of stop words is done and stemming is performed. TFIDF (Term Frequency and Inverse-Document Frequency) values of the stemmed terms are calculated and a vector space model is constructed. Optimized K-means algorithm is applied on it.

The problem with K-means algorithm is that the random selection of number of clusters (K). Since the number of clusters selection affects the overall performance of the algorithm, it must be selected using a defined method. We optimized K-means algorithm by selecting the number of clusters using knee finding algorithm.

Knee finding algorithm finds the number of clusters by executing the K-means algorithm repeatedly with varying values of K. At the convergence time of K-means algorithm with different K values, Error value ($E_K$) is calculated and it is plotted on a graph against the possible number of clusters. Once the execution of K-means algorithm with different number of clusters is over, the Knee point of the $E_K$ curve is identified and the number of clusters at that point is taken as the best number of clusters.

## 2. Related Work

Clustering has been considered in many different ways as alternatives to the ranked list presentation [10]. Some of them apply traditional clustering algorithms which first cluster documents into topically-coherent groups according to content similarity and then generate descriptive labels for clusters [11]. However, these labels are often unreadable, which makes it difficult for the users to identify the clusters. In our work, labels are produced by considering the terms from the title and Meta tag so that the labels are more informative and readable.

For search results clustering, terms can be extracted from different parts of the web page. Giansalvatore, Salvatore and Alessandro [1] have extracted terms from entire web page for clustering. The clusters produced by this method are of good quality but the number of terms extracted is more and it produces lengthy vectors. Another difficulty with entire documents clustering is that the contents of some of the web pages could not be extracted.

Stanis law Osinski et al., [2] have considered terms from the snippets. The quality of clusters produced by this method is not acceptable. Since snippets are the sentences available in a web page which contain the query keywords in it. It is possible that the terms present inside the snippets may not be informative enough for clustering. HuskySearch [12] retrieves results from several popular Web search engines and extracts terms from those results.

### 2.1 Existing Techniques for Web Documents Clustering

Grouper [13] extracts terms from the snippets and clusters the results as they arrive using the Suffix Tree Clustering (STC) algorithm. Suffix Tree Clustering (STC) algorithm identifies the common phrases in the collection of documents. Steps in the algorithm are

i. Base Cluster identification

ii. Combining base clusters into final clusters

Initially, the meaningful terms are extracted by preprocessing the documents. Identification of base clusters is done by constructing the suffix tree. A suffix tree of a string *S* is a compact trie which contain all the suffixes of *S*. This algorithm considers documents as set of words, not as sequence of characters. A suffix tree is a rooted, directed tree in which each internal node has at least 2 children and each edge is labeled with a non-empty sub-string of *S*. The label of a node in defined to be the concatenation of the edge labels on the path from the root to that node. Each node in a suffix tree contains the document numbers in which the phrase presents. Using this, common phrases are identified. These common phrases are used to form clusters of similar documents.

Carrot2 applies Lingo clustering algorithm [2] and K-means clustering algorithm on the snippets to cluster the documents. In Lingo clustering algorithm, initially cluster labels are found and then the documents are assigned to the respective clusters. The steps in the algorithm are,

i.Frequent phrase extraction

ii.Cluster label induction and

iii.Content assignment

This algorithm uses terms from snippets. After extracting the meaningful terms from the snippets by performing snippet preprocessing, term-document matrix is formed and the TFIDF values are calculated and filled in the matrix. Next step of the algorithm is to find a group of cluster label candidates (Phrases or terms). It extracts frequent phrases using another algorithm namely shoc algorithm. A word-based suffix array is constructed and extended with an auxiliary data structure called Longest Common Prefix.

During the cluster label induction phase, this algorithm identifies the abstract concepts that best describe the input collection of snippets. The steps involved in this are abstract concept discovery, phrase matching and label pruning. Once the cluster labels are produced, cluster content assignment is done using the traditional vector space model. But instead of comparing the snippets with a single query, the input snippets are matched with set of queries which are the cluster labels. Similarity among the cluster labels and the document snippet is calculated and the document is assigned to most similar cluster labels.

K-means clustering is the simplest and most commonly used clustering algorithm. Initially the algorithm takes the collection of documents, number of clusters (K) and centroids of each cluster as input. Algorithm finds the distance of documents from the initial centroids and documents are assigned to nearer clusters. This process continues until some stopping criterian is met. Selection of initial centroids and the number of clusters (K) is done randomly. Once the clusters are formed, the cluster labels are generated by finding the terms with high frequency inside the documents.

Problem with the existing clustering algorithms is that the repetition of documents in different clusters. K-means algorithm avoids this problem by assigning each document to a single cluster. Among the existing clustering algorithms, K-means clustering algorithm is the most widely used algorithm [14]. It partitions the documents into clusters by minimizing the sum of the squared distances between the documents and the centroid of the clusters.

**2.2 Existing Techniques for Optimizing K-means**

Major drawback of K-means algorithm is that it selects the initial parameters randomly [15][16]. Since the performance of a clustering algorithm may be affected by the chosen value of the number of clusters (K), major problem with K-means clustering is the determination of the correct number of groups in a data set. Several approaches to this problem have been suggested in the literature.

Two older approaches are a Gaussian model-based technique using approximate Bayes factors [17] and the gap statistic which compares the change in within cluster dispersion with that expected under an appropriate null distribution [18]. There were several recent papers devoted to the selection of number of clusters (K) in the engineering literature where it is known as the cluster validation problem. Reported studies [19-23] on K-means clustering and its applications usually do not contain any explanation or justification about selecting particular values for K. First, a number of researchers [24, 25] used only one or two values for K. Second, several other researchers [26-28] utilized relatively large K values compared with the number of objects. Jian Wan et al.,[29] have improved the efficiency and effectiveness of K-Means clustering algorithm by parallelizing the tasks done using map and reduce functions .

**3. KOPKWS – Knee finding based Optimized K-means for Web Search**

The proposed method comprises the following phases:

1. Term Extraction (URL, Title tag and Meta tag)
2. Preprocessing (Stop word removal and Stemming)
3. TFIDF calculation
4. Applying Optimized K-means algorithm
5. Cluster label generation

Initially, the search results returned by a web search engine are retrieved. These web pages are analyzed by a parser and the

contents are extracted from the URL and the specified tags. These contents are informative enough because most search engines are well designed to facilitate users' relevance judgement only by the tag contents. Stop words are removed from the collection of words by checking it against the database of stop words and stemming is then performed to convert the words into their root form. In this paper, Porter's algorithm is used for stemming.

TFIDF matrix is formed by measuring the Term Frequency and Inverse Document Frequency values. Clustering algorithm is applied on the matrix formed.

In traditional K-means clustering algorithm, initial parameters are selected randomly. Distance matrix is formed using the distance function. Every document is assigned to the cluster for which the document is having the minimum distance. After forming clusters, new centroids are calculated for the clusters with more than one document. New centroids are calculated by taking the average of the term frequencies of the documents. This process is continued until the results of the two iterations become the same.

In the proposed optimized K-means algorithm, initial cluster centroids are selected randomly and K-means algorithm is executed as mentioned in the traditional K-means clustering with varying values of K. At the convergence time of K-means, the error value is calculated and plotted on the graph. Once the execution of K-means algorithm with different number of clusters is over, the Knee point of the $E_K$ curve is identified and the number of clusters at that point is taken as the best number of clusters.

**3.1 Term Extraction**

Text document clustering is different from web document clustering. Web documents are unstructured. In addition to text contents it contains tag information. This tag information and the punctuations should be removed from the document to extract meaningful terms. Tokenization is used to perform this operation.

Given a document, tokenization is the task of breaking into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

Input: Friends, Romans, Country men, lend me your ears;

Output: Friends Romans Country men lend me your ears

Contents from the URL and the specified tags are extracted by removing the tags and special symbols with the use of tokenization. Since the number of terms extracted is less, there is no need to perform feature selection. Instead, all the extracted terms can be considered as features for clustering the web documents.

**3.2 Preprocessing (Stop word removal and Stemming)**

Stop words does not provide any useful information to perform clustering. So they can be removed to avoid confusions. Some common stop words are is, was, are, were, what etc., For stop word removal, first the database of stop words are created and the terms extracted from the web pages are compared with the stop words list. Stop words found in the extracted terms are removed.

The goal of stemming is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

am, are, is $\Rightarrow$ be

car, cars, car's, cars' $\Rightarrow$ car

the boy's cars are different colors $\Rightarrow$ the boy car be differ color

Stemming usually refers to a crude heuristic process that chops off the ends of words and often includes the removal of derivational affixes.

Usually the stemmed words are not meaningful. For example, the stemmed word of "computation" is "comput". While stemming the words, two things have to be considered.

i. Different words with the same base meaning are converted to the same form

and

ii. Words with distinct meanings are kept separate.

For stemming, Porter's algorithm is used in this paper. It is a simple utility that reduces English words to their word stems - without the "ing", "ings";"s" etc.,

| Word | Base Word | Word | Base Word |
|---|---|---|---|
| Consigned | Consign | Consolation | Consol |
| Consigning | Consign | Consolations | Consol |
| Consignment | Consign | Console | Consol |
| Consisted | Consist | Consoled | Consol |
| Consistency | Consist | Consoles | Consol |
| Consistently | Consist | Consonant | Conson |
| Consisting | Consist | Consorted | Consort |
| Consists | Consist | Conspirator | Conspir |

Table 1. Words and their equivalent base

**3.3 TFIDF Calculation**

TFIDF is frequently used to construct a term vector space model. It evaluates the importance of a word in a document. The importance score increases proportionally with the number of times a word appears in the document but is offset by the frequency of a word in the entire collection of documents. Suppose there are set of documents, each with collection of terms. A simple way to group those documents using terms is to count the number of times a term occurs in a document. Calculated count is called as a term frequency. However, some terms are more common such as "contain" and these terms get more weight when term frequency is used. Also the terms like "contain" are not good keywords to identify the similarity among the documents. On the other side, the keywords that occur rarely are good to find the relevancy among the documents. Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the collection and increases the weight of terms that occur rarely.

This assigns to term i a weight in document j given by

$$TFIDF_{i,j} = TF_{i,j} \text{ X } IDF_i \quad (1)$$

$TF_{i,j}$ is calculated as:

$$TF_{i,j} = \frac{N_{i,j}}{NT_j} \quad (2)$$

$N_{i,j}$ is the number of times the term i appears in the document j and $NT_j$ is the total number of terms in the document j.

The inverse document frequency ($IDF_i$) is calculated as:

$$IDFi = \log \left( \frac{|D|}{|d: t_i \in d|} \right) \quad (3)$$

Where |D| is the total number of documents and $|d : t_i \in d|$ is the number of documents in which the term $t_i$ appears. These TFIDF values and the list of documents are then formed as a vector space.

**3.4 Optimized K-means Clustering Algorithm**

K-means clustering is the simplest and most commonly used clustering algorithm. Initially the algorithm takes collection of documents, number of clusters (K) and centroids of each cluster as input. Algorithm finds the distance of documents from the initial centroids and documents are assigned to nearer clusters. This process continues until some stopping criterian is met.

Selection of number of clusters decides the purity of the produced clusters. For the same document collection, different number of clusters will produce different kinds of clusters. This can be avoided by selecting the number of clusters using the following strategy.

**Steps in the Optimized K-means Algorithm**

1. Assign the initial value of K (Number of clusters to be created) as $\sqrt[2]{n/2}$.
2. Randomly select K initial centroids.
3. Assign each document to the nearest cluster by finding the distance between the document and the cluster centroid using Euclidean distance as,

$$Dist_{i,j} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad (4)$$

4. Recompute the cluster centroids using the current cluster memberships.
5. If there is a reassignment of documents to the new cluster, go to step 3.
6. Calculate the Error (Ek) based on the distance values at the convergence time of K-means.
7. Plot the error value on the graph, increment the value of K and continue from step 2.
8. Find the knee point of the curve and consider the number of clusters at this point as the best number of clusters.

### 3.4.1 Selection of Number of Clusters

**Steps in the selection of Number of Clusters**

1. Execute K-means algorithm with starting number of clusters as .
2. Calculate Error value (Ek) based on the distance values at the convergence time of K-means.
3. Plot this error value on a graph against the possible number of clusters.
4. Finally the number of clusters in the data set is chosen by examining and selecting the knee of the Ek curve.

The number of clusters to be found, along with the initial centroid values are specified as input parameters to the clustering algorithm. Given the initial centroid values, the distance from each document to each initial centroid value is found using Euclidean distance. Each document is then placed in the cluster associated with the nearest centroid. New cluster centroids are calculated after all documents have been assigned to a cluster. Suppose that Cim represents the centroid of the mth term of the ith cluster. Then,

$$C_{im} = \frac{\sum_{j=1}^{n_i} F^*_{i,jm}}{n_i} \qquad (5)$$

where $F^*_{i,j\,m}$ is the mth TFIDF value of the jth document assigned to the ith cluster and where ni is the number of documents in cluster i.

An error function can be defined that describes how much error is there between all of the documents in the collection and their respective cluster centroids. Assuming that there are k clusters, the error function (Ek) is defined as:

$$E_k = \sum_{i=1}^{k}\sum_{j=1}^{n_i}\sum_{m=1}^{M}(F^*_{i,jm} - C_{im})^2 \qquad (6)$$

where $(F^*_{i,jm}-C_{im})2$ is the distance measure between a document and the cluster centroid to which it is assigned.

Typically, to determine the most descriptive number of clusters, the K-means clustering algorithm is run on the data set for a range of possible clusters, say, from k =2 to N (N is assumed as Total number of documents / 3). At the end of each run, the error value Ek is calculated using equation (2). These error values are then plotted on a graph against the possible number of clusters. As the number of clusters K increases, it is reasonable to expect that the value of Ek decreases, since with more possible cluster centroids to choose from, it is more likely that any given document will be assigned to a potentially closest cluster. The final number of clusters in the document set is chosen by examining and selecting the knee of the Ek curve.

### 3.5 Cluster Label Generation

Cluster labels are generated to provide identification for the clusters. Generation of labels is done by extracting the terms from the documents in each cluster. Since the extraction of terms from the entire document may produce more number of terms, only

Title tag and Meta tag contents are extracted. For all the extracted terms, term frequency is calculated and the term which is having highest term frequency is taken as the label for the cluster.

### 4. Experimental Results

For our experiment, we considered 200 queries. For all the 200 queries, we collected first 200 results from yahoo, google and bing. PHP is used to write the code and the collected web documents are stored using SQL Server. The measures such as intra-cluster distance and inter-cluster distance are used to compare the proposed method with existing methods.

#### 4.1 Stop Word Removal

Initially, the words are extracted from URL and Tag contents and the stop words are removed. Table 2 shows the list of stop words considered in this paper.

| | |
|---|---|
| a | a, able, about, above, according, accordingly, across, actually, after, afterwards, again, against, ain't, all, allow, allows, almost, alone, along, already, also, although, always, am, among, amongst, an, and, another, any, anybody, anyhow, anyone, anything, anyway, anyways, anywhere, apart, appear, appreciate, appropriate, are, aren't, around, as, aside, ask, asking, associated, at, available, away, awfully |
| b | be, became, because, become, becomes, becoming, been, before, beforehand, behind, being, believe, below, beside, besides, best, better, between, beyond, both, brief, but, by |
| c | Common, came, can, can't, cannot, cant, cause, causes, certain, certainly, changes, clearly, co, com, come, comes, concerning, consequently, consider, considering, contain, containing, contains, corresponding, could, couldn't, course, currently |
| d | definitely, described, despite, did, didn't, different, do, does, doesn't, doing, don't, done, down, downwards, during, |
| e | each, edu, eg, eight, either, else, elsewhere, enough, entirely, especially, et, etc, even, ever, every, everybody, everyone, everything, everywhere, ex, exactly, example, except |
| f | far, few, fifth, first, five, followed, following, follows, for, former, formerly, forth, four, from, further, furthermore |
| g | Get, gets, getting, given, gives, go, goes, going, gone, got, gotten, greetings |
| h | had, hadn't, happens, hardly, has, hasn't, have, haven't, having, he, he's, hello, help, hence, her, here, here's, hereafter, hereby, herein, hereupon, hers, herself, hi, him, himself, his, hither, hopefully, how, howbeit, however |
| i | i'd, i'll, i'm, i've, ie, if, ignored, immediate, in, inasmuch, inc, indeed, indicate, indicated, indicates, inner, insofar, instead, into, inward, is, isn't, it, it'd, it'll, it's, its, itself |
| j | Just |
| k | keep, keeps, kept, know, knows, known |
| l | last, lately, later, latter, latterly, least, less, lest, let, let's, like, liked, likely, little, look, looking, looks, ltd |
| m | mainly, many, may, maybe, me, mean, meanwhile, merely, might, more, moreover, most, mostly, much, must, my, myself |
| n | name, namely, nd, near, nearly, necessary, need, needs, neither, never, nevertheless, new, next, nine, no, nobody, none, nor, normally, not, nothing, novel, now, nowhere |
| o | obviously, of, off, often, oh, ok, okay, old, on, once, one, ones, only, onto, or, other, others, otherwise, ought, our, ours, ourselves, out, outside, over, overall, own |
| p | particular, particularly, per, perhaps, placed, please, plus, possible, presumably, probably, provides |
| q | Que, quite, qv |
| r | rather, rd, re, really, reasonably, regarding, regardless, regards, relatively, respectively, right |
| s | said, same, saw, say, saying, says, second, secondly, see, seeing, seem, seemed, seeming, seems, seen, self, selves, sensible, sent, serious, seriously, seven, several, shall, she, should, shouldn't, since, six, so, some, somebody, somehow, someone, something, sometime, sometimes, somewhat, somewhere, soon, sorry, specified, specify, specifying, still, sub, such, sup, sure |

| | |
|---|---|
| t | t's, take, taken, tell, tends, th, than, thank, thanks, thanx, that, that's, thats, the, their, theirs, them, themselves, then, thence, there, there's, thereafter, thereby, therefore, therein, theres, thereupon, these, they, they'd, they'll, they're, they've, think, third, this, thorough, thoroughly, those, though, three, through, throughout, thru, thus, to, together, too, took, toward, towards, tried, tries, truly, try, trying, twice, two |
| u | un, under, unfortunately, unless, unlikely, until, unto, up, upon, us, use, used, useful, uses, using, usually |
| v | value, various, very, via, viz, vs |
| w | want, wants, was, wasn't, way, we, we'd, we'll, we're, we've, welcome, well, went, were, weren't, what, what's, whatever, when, whence, whenever, where, where's, whereafter, whereas, whereby, wherein, whereupon, wherever, whether, which, while, whither, who, who's, whoever, whole, whom, whose, why, will, willing, wish, with, within, without, won't, wonder, would, would, wouldn't |
| y | Yes, yet, you, you'd, you'll, you're, you've, your, yours, yourself, yourselves |
| z | Zero |

Table 2. List of Stopwords

Since only the contents of URL and specific tags are selected, number of terms after stop word removal is lesser than the number of terms produced by extracting terms from the entire document. Table 3 shows the number of terms before stop word removal and after stop word removal.

| **KEYWORDS / NUMBER OF DOCUMENTS** | **NUMBER OF TERMS EXTRACTED** | | | **NUMBER OF TERMS AFTER STOPWORD REMOVAL** | | |
|---|---|---|---|---|---|---|
| | 50 | 100 | 200 | 50 | 100 | 200 |
| APPLE | 905 | 1787 | 2626 | 756 | 1338 | 1997 |
| MYSQL | 452 | 854 | 1565 | 377 | 688 | 1233 |
| FREE ANTIVIRUS | 492 | 763 | 1291 | 408 | 613 | 1024 |
| BRIDGE | 470 | 921 | 1951 | 397 | 790 | 1670 |
| CLUSTER | 803 | 1233 | 2272 | 679 | 1017 | 1780 |
| DICTIONARY | 1134 | 1469 | 2229 | 982 | 1232 | 1833 |
| PROCESSOR | 483 | 979 | 2257 | 407 | 809 | 1760 |
| NIMCET | 532 | 819 | 1451 | 423 | 642 | 1142 |
| MOUSE | 921 | 1515 | 2445 | 766 | 1237 | 1973 |
| TAMIL MP3 | 948 | 1407 | 2137 | 843 | 1234 | 1847 |
| MOBILE | 630 | 875 | 1392 | 461 | 646 | 1088 |
| CAMERA | 457 | 847 | 1828 | 358 | 673 | 1448 |
| HARDWARE | 397 | 684 | 1303 | 346 | 578 | 1701 |
| GRID COMPUTING | 433 | 653 | 1394 | 344 | 513 | 1106 |
| NETWORKING | 581 | 929 | 1780 | 487 | 767 | 1404 |
| ORANGE | 653 | 1490 | 3019 | 566 | 1266 | 2434 |
| DATASTRUCTURES | 363 | 763 | 1349 | 288 | 592 | 1028 |
| DOCOMO | 553 | 1089 | 1940 | 460 | 879 | 1525 |
| CLOUD COMPUTING | 381 | 613 | 1145 | 312 | 516 | 927 |
| CAT | 645 | 1298 | 2274 | 533 | 1063 | 1865 |

Table 3. Number of terms after Stop word removal

**4.2 Stemming**

Once the stopwords are removed from the collection of extracted terms, stemming is performed to convert the words into their base forms. Table 4 shows the stemmed words produced.

| |
|---|
| redplaid, cloud, computing, oracl, weblogic, comput, easy, google, chelsio, communications, host, free, trial, wikipedia, alpha, wmf, definit, whati, word, http, sdefinit, sid, gci, html, noodp, concept, articl, describ, impact, variety, compani, countri, industri, refer, amazon, amzn, blackbaud, blkb, blackboard, bbbb, concur, technolog, cnqr, geographi, page, googl, goog, hp, intel, intc, intern, busi, machin, ibm, new, result, salesforc, forc, platform, leader, servic, instant, time, valu, complet, independ, noydir, really, mean, infoworldth, big, trend, sound, nebul, fuzzy, view, proposit, perspect, profession, index, follow, web, app, engin, applic, architectur, infrastructur, soa, orient, softwar, sun, microsystem, open, sourc, discov, brought, world, popular, starlight, howstuffwork, quot, work, smart, essenti, futur, let, file, internet, learn, benefit, drawback, computingbarry, warwick, raleigh, ibmu, public, v, templat, gener, corpor, copyright, expo, sy, media, journalsy, computingcloud, n, june, stamford, ct, youtub, explain, computingexplain, video, produc, present, christoph, barnatt, author, explainingcomput, associ, professor, compu, utility, saa, haa, explainedcloud, confus, term, want, talk, technology, buzzword, boil, section, ec, gogrid, netflix, gmail, server, dedic, vmware, virtualizationturn, datacent, secur, privat, virtual, add, capacity, demand, deploy, extern, built, vm, ware, gklqyx, kxtfwei, optim, data, center, lab, consolid, lifecycl, provid, continuity, pc, solut, dell, increas, capabl, flexibility, choic, eiwatch, content, enterpris, aspx, ng, publish, elast, ecamazon, aw, deliv, set, form, reliabl, scalabl, inexpens, pay, includ, simpledb, sq, fp, simpl, storag, grid, mechan, turk, queue, flexibl, payment, devpay, alexa, develop, forum, user, group, tool, api, unveil, centr, india, nist, gov, security, divis, csrc, websit, involv, differ, project, webpag, base, visit, resourc, clearinghous, event, driver, mobileag, pki, encrypt, ae, mode, ipv, alert, virus, organ, link, mission, staff, itl, divisiion, network, policy, manag, oper, solutions, ubuntu, plain, english, common, craft, product, computerworld, knowledg, voic, inform, c, right, reserv, action, command, viewpag, amp, pagepath, polici, eng, document, faq, demystifi, worldnetwork, bring, great, promis, industry, key, question, answer, level, agreement, supplement, jon, brodkin, schneier, onlin, rackspac, mosso, power, site, support, live, chat, today |

Table 4. Stemmed words

**4.3 TFIDF Matrix Formation**

Once the useful terms are extracted reduced, TFIDF matrix is formed. Table 5 shows the sample TFIDF matrix for 10 documents

| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |
|---|---|---|---|---|---|---|---|---|---|---|
| D1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| D2 | 0.2 | 0 | 0 | 0 | 0 | 0 | 0 | 0.1 | 0.1 | 0.1 |
| D3 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D4 | 0.2 | 0 | 0 | 0 | 0 | 0 | 0 | 0.1 | 0.1 | 0.1 |
| D5 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0.1 | 0.1 | 0.1 |
| D6 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D7 | 0.4 | 0.1 | 0 | 0.1 | 0.3 | 0.2 | 0.4 | 0 | 0 | 0 |
| D8 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D9 | 0.9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D10 | 0.3 | 0 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 5. TFIDF Matrix

**4.4 Results Comparison**

Quality of the produced clusters is identified using intra-cluster distance and inter-cluster distance.

Intra-cluster distance is calculated by finding square of the summation of the absolute differences between each point in a

cluster and adding a constant with it and finally dividing the result by 2.

$$L1 = \frac{1}{2}\left(\forall \sum_{k_{i,j} \in C_k} |X_i - X_j|^2 + Constant\right) \quad (7)$$

Where k is the number of clusters
i and j are the documents in kth cluster
Xi is a point of document i
Xj is a point of document j
Ck represents the kth cluster
0.5 is taken as the constant value

Inter-cluster distance is calculated by finding the square of distance between every point with all the other points in other clusters and adding a constant with it and finally dividing it by 2.

$$L2 = \frac{1}{2}\left(\forall \sum_{\substack{k_{i \in} C_k \& \\ j \notin C_k}} |X_i - X_j|^2 + Constant\right) \quad (8)$$

Where k is the number of clusters
i is the document belonging to the kth cluster
j is the document not in kth cluster
Xi is a point of document i
Xj is a point of document j
Ck represents the kth cluster
0.5 is taken as the constant value

Figure 1 show the intra-cluster distances produced by Snippet based STC algorithm, Snippet based Lingo algorithm, Snippet based traditional K-means algorithm and the proposed method. From the results it is proved that the proposed method produces clusters with less intra-cluster distance so that the documents in each cluster are strongly related.

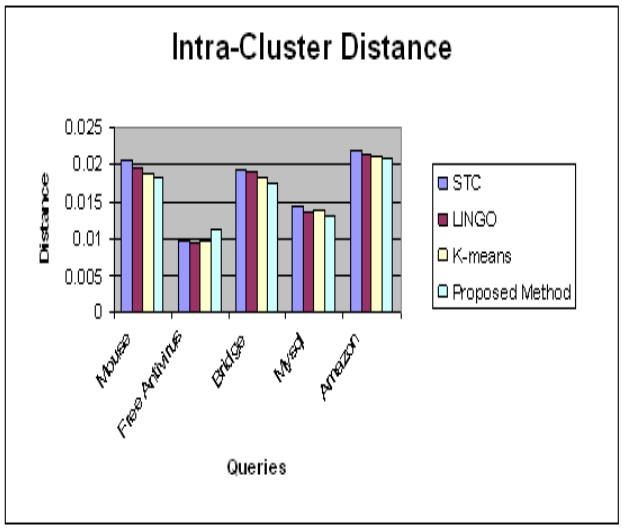


Figure 1. Intra-cluster Distance

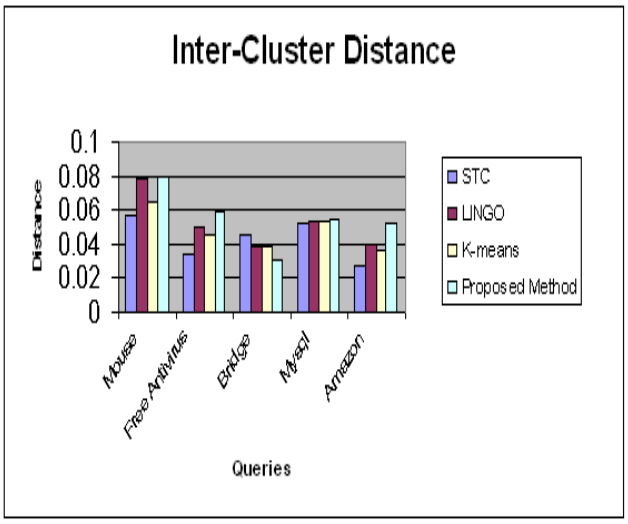


Figure 2. Inter-cluster Distance

Figure 2 show the inter-cluster distances produced by Snippet based STC algorithm, Snippet based Lingo algorithm, Snippet based traditional K-means algorithm and the proposed method. It proves that the proposed method have produced clusters such that the documents in one cluster are highly independent of the documents in other clusters.

## 5. Conclusion

In this paper, we extracted terms from URL, Title tag and Meta tag so that the vectors produced by this method are not too long when compared to entire document contents. Also this method produced high quality clusters than snippet based clustering since the terms in URL and specified tags are having greater significance in providing identification for the web page. Among the existing clustering algorithms, K-means algorithm is the best algorithm for clustering larger data sets. The major drawback of this algorithm is the random selection of initial parameters such as centroid selection and the number of clusters selection. In this work, we used Knee finding algorithm to select the number of clusters in order to optimize K-means clustering algorithm. The results show that KOPKWS has produced high quality clusters where the documents inside each cluster are tightly coupled. In future, this work can be extended by introducing a new method for the selection of the initial centroids or number of clusters and also the terms in other parts of the web pages can also be considered for clustering.

## References


[1] Mecca, Giansalvatore., Raunich, Salvatore., Pappalardo, Alessandro (2007). A new algorithm for clustering search results, *ACM Transactions on Data & Knowledge Engineering*, 62 (3).

[2] Osinski, Stanis law., Stefanowski, Jerzy., Weiss, Dawid (2004). Lingo: Search results clustering algorithm based on Singular Value Decomposition, *In:* Proceedings of Intelligent Information Systems Conference, Zakopane, Poland.

[3]Sameh,Ahmed., Kadray, Amar (2010). Semantic Web Search Results Clustering Using Lingo and WordNet, I*nternational Journal of Research and Reviews in Computer Science* (IJRRCS), 1 (2) 71-76.

[4] Brin,S.., Page, L (1998). The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems, 30 (1-7) 107-117.

[5] Kleinberg, J. (1998). Authoritative sources in a hyperlinked environment, *In*: Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms, p. 668-677.

[6] Lempel, R., Moran, S (2001). SALSA: The Stochastic Approach for Link-Structure analysis, *ACM Transactions on Information Systems*, 19 (2) 131–160.

[7] Haveliwala, Taher H. (2002). Topic Sensitive Page Rank, World Wide Web Conference, May 7–11.

[8] Xing, Wenpu., Ghorbani, Ali (2004). Weighted PageRank Algorithm, *In*: CNSR, Second Annual Conference on Communication Networks and Services Research (CNSR'04), 305-314.

[9] Yen, Chia-Chen., Jih-Shih, Hsu, (2010). Pagerank Algorithm Improvement by Page Relevance Measurement, *Journal of Convergence Information Technology*, 5 (8).

[10] Leouski , A., Allan, J. (2000). Improving Interactive Retrieval by Combining Ranked List and Clustering, *In*: Proceedings of RIAO, 665-681.

[11] Kalashnikov, Dmitri, V., Zhaoqi (Stella) Chen, Mehrotra, Sharad and Nuray-Turan, Rabia (2008). Web People Search via Connection Analysis, *IEEE Transactions on Knowledge and Data Engineering*, 20 (11) 1441-1457.

[12] Selberg, E., Etzioni, O. "Multi-service search and comparison using the MetaCrawler", Proceedings of the 4th World Wide Web Conference (WWW4), pp. 195-208, 1995.

[13] Zamir, O., Etzioni, O. (1998). Grouper - A Dynamic Clustering Interface to Web Search Results, *Computer Networks*, 31 (11-16) 1361-1374 (1995).

[14] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations, *In*: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probabilities, 281-297.

[15] Kanungo, T. , . Mount, D. M. Netanyahu, N Piatko, C.. Silverman , Rand . Wu, A. Y (2002). An efficient K-means clustering algorithm: Analysis and implementation", IEEE Trans. Pattern Analysis and Machine Intelligence, 24 (7) 881-892, 2002.

[16] Pelleg, D., Moore, A (1999). Accelerating exact K-means algorithms with geometric reasoning, *In:* Proceedings of the Conference on Knowledge Discovery in Databases (KDD 99), San Diego, California, p. 277–281.

[17] Kass, Robert E., Raftery, Adrian E. (2002). Bayes factors, *Journal of the American Statistical Association*, 90 (430) 773–795.

[18] Tibshirani, Robert., Walther,Guenther., Hastie,Trevor (2001). Estimating the number of clusters in a data set via the gap statistic, *Journal of the Royal Statistical Society*, Series B, 63, p. 411–423.

[19] Al-Daoud, M. B. , Venkateswarlu ,N. B., Roberts, S. A. (1995). Fast K-means clustering algorithms, Report 95.18, School of Computer Studies, University of Leeds.

[20] Alsabti, K., Ranka, S., Singh, V (1998). An efficient K-means clustering algorithm, *In:* Proceedings of the First Workshop on High-Performance Data Mining, Orlando, Florida.

[21] Bottou, L., Bengio, Y (1995). Convergence properties of the K-means algorithm, *Advanced Neural Information Processing Systems*, 7, 585–592.
[22] Bradley, S., Fayyad, U.M (1998). Refining initial points for K-means clustering, *In:* Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98) (Ed. J. Shavlik), Madison, Wisconsin, p. 91–99.
[23] Du, Q., Wong, T.W (2002). Numerical studies of MacQueen's K-means algorithm for computing the centroidal Voronoi tessellations, *International Journal of Computers and Mathematics with Applications*, 44 ( 3-4) 511–523.
[24] Bilmes, J., Vahdat, A.H., Im, E.J (1997). Empirical observations of probabilistic heuristics for the clustering problem, Technical Report TR-97-018, International Computer Science Institute, Berkeley, California.
[25] Castro,V. E., Yang, J (2000). A fast and robust general purpose clustering algorithm, *In:* Proceedings of the Fourth European Workshop on Principles of Knowledge Discovery in Databases and Data Mining (PKDD 00), Lyon, France, p. 208–218.
[26] Al-Daoud, M. B., Venkateswarlu, N. B., Roberts, S. A. (1996). New methods for the initialisation of clusters, *Pattern Recognition Letters*, 17 (5) 451–455.
[27] Fritzke, B. (1997). The LBG-U method for vector quantization – an improvement over LBG inspired from neural networks, *Neural Processing Letters*, 5 (1) 35–45.
[28] Hansen, L. K., Larsen, J. (1996). Unsupervised learning and generalization, *In:* Proceedings of the IEEE International Conference on Neural Networks, Washington, DC, p. 25–30.
[29] Wan, Jian., Yui, Wenming., Xu, Xianghua (2009). Design and Implement of Distributed Document Clustering based on MapReduce, *In*: Proceedings of the Second Symposium International Computer Science and Computational Technology (ISCSCT '09), p. 278-280.

**Author Biography**

Ms. S. Poomagal, obtained her B.Sc (Applied Science – Computer Technology) from Bharathiar University, Coimbatore, India in the year 2001. She completed her M.Sc (Applied Science – Computer Technology) from Bharathiar University, Coimbatore, India in the year 2003. She completed her M.Phil (Computer Science) from Bharathiar University in the year 2004. She is pursuing her Ph.D (Information Technology) in Anna University, Coimbatore, India. She worked as a lecturer in Dr. SNS Rajalakshmi College of Arts & Science, Coimbatore from Jan 2005 to April 2007. Currently she is working as an assistant professor in the Department of Computer and Information Sciences, PSG College of Technology, Coimbatore since June 2007. She has published 11 papers in various conferences and journals. Her research interests include Data mining, Compiler design and Programming Languages.

Dr. T. Hamsapriya, obtained her BE (Electronics and Communication Engineering) from Bharathiar University, Coimbatore, India in the year 1988. She completed her ME(Communication systems) from Bharathiar University, Coimbatore, India in the year 1992. She completed her Ph.D (Parallel Computing) from Anna University, Chennai in the year 2008. She worked in Kumaraguru College of Technology, Coimbatore, India as an associate lecturer from 1993-94. She worked in BPL Electronics as a consultant engineer from 1994-96. She was acting as a managing partner in Computer Software College fro 1996-99. She is working as a professor in PSG College of Technology, Coimbatore, India since 1999. She is the head of the Department of Information Technology, PSG College of Technology, Coimbatore. She is also a research cordinator in PSG College of Technology. She has published more than 25 papers in various international conferences and journals.