Web-Page Clustering Using Cemetery Organization Behavior of Ants

Prepared By

Samir Kariya

11MCEC29

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2013

Web-Page Clustering using Cemetery Organization Behavior of Ants

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

Prepared By

Samir Kariya

(11MCEC29)

Guided By

Prof. Priyank Thakkar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2013

Certificate

This is to certify that the Major Project Report entitled ""**Web-Page Clustering Using Cemetery Organization Behavior of Ants"** submitted by **Kariya Samir R. (Roll No: 11MCEC29)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part-I, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Priyank Thakkar
Guide & Assistant Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Prof. Vijay Ukani
Associate Professor
Coordinator M.Tech - CSE
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. Sanjay Garg
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr K Kotecha
Director,
Institute of Technology,
Nirma University, Ahmedabad

Undertaking for Originality of the Work

I, **Kariya Samir R.**, Roll. No. **11MCEC29**, give undertaking that the Major Project entitled "**Web-Page Clustering Using Cemetery Organization Behavior of Ants**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering** of Nirma University, Ahmedabad, is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

_____

Signature of Student

Date:

Place:

Endorsed by

Prof. Priyank Thakkar

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof. Priyank B. Thakkar**, Professor, Computer Science Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout this work. The appreciation and continual support he has imparted has been a great motivation to me in reaching a higher goal. His guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

My deepest thank you is extended to **Prof. Vijay Ukani**, PG CSE - Coordinator, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for an exceptional support and continual encouragement throughout the Major Project.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr K Kotecha**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, and Ahmedabad for their special attention and suggestions towards the project work.

The blessings of God and family members make the way for completion of Project. I am very much grateful to them.

- Samir Kariya

11MCEC29

Abstract

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. Due to the increasing amount of data available online, the World Wide Web has become one of the most valuable resources for information retrievals and knowledge discoveries. Web mining technologies are the right solutions for knowledge discovery on the Web.

In this dissertation, we focus on web page clustering based on their content. A web page clustering system can be useful in web search for grouping search results into closely related sets of documents. It can improve similarity search by focusing on sets of relevant documents. At the same time, web page clustering is much more difficult than pure-text clustering due to a large variety of noisy information embedded in web pages.

In this dissertation, web page clustering problem is addressed by the technique inspired by cemetery organization behavior of ants. Impact of dimensionality reduction and feature selection is also studied.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 General

It is well known that the world wide web may be considered as a huge and global information center. A web site usually contains great amounts of information distributed through hundreds of pages. Without proper guidance, a visitor often wanders aimlessly without visiting important pages, loses interest and leaves the site sooner than expected. This consideration is at the basis of the great interest about web information mining both in the academic and the industrial world.

The term Web mining was coined by Etzioni (1996) to denote the use of data mining techniques to automatically discover Web documents and services, extract information from Web resources, and uncover general patterns on the Web. Over the years, Web mining research has been extended to cover the use of data mining and similar techniques to discover resources, patterns, and knowledge from the Web and Web related data (such as Web usage data or Web server logs). Defined as, "the discovery and analysis of useful information from the World Wide Web", web mining go through large volumes of web page data quickly and attempt to transform that data in a way that will enhance the discovery of new knowledge. Mining uses a combination of many disciplines: from statistics and pattern recognition to artificial intelligence and machine learning.

Though web mining has been successful in many of its applications, it is still a relatively new field. New techniques and innovations are found frequently, and yet there is

much potential for growth.

1.2 Motivation

Data analysis underlies many computing applications, either in a design phase or as part of their on-line operations. Data analysis procedures can be dichotomized as either exploratory or confirmatory, based on the availability of appropriate models for the data source, but a key element in both types of procedures (whether for hypothesis formation or decision-making) is the grouping, or classification of measurements based on either (i) goodness-of-fit to a postulated model, or (ii) natural groupings (clustering) revealed through analysis. Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity. Intuitively, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster.

Clustering is useful in several exploratory pattern-analysis, grouping, decision-making, and machine-learning situations, including data mining, document retrieval, image segmentation, and pattern classification. However, in many such problems, there is little prior information (e.g., statistical models) available about the data, and the decision-maker must make as few assumptions about the data as possible. It is under these restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data points to make an assessment (perhaps preliminary) of their structure.

1.3 Scope of the Work

To achieve the objective of developing a cemetery organization behavior of Ant for effective clustering, the work seeks to develop an optimal number of clusters using Lumer-Faieta algorithm. The parameter for optimization is the accuracy and additionally, the number of clusters so as to address the important issue of ease in comprehension. The approach considers designing a Lumer-Faieta algorithm and add some parameters in it. Thus the scope is limited to proposing a new clustering algorithm that is a Modification

of L-F algorithm of cemetery organization behavior of Ant.

1.4 Fundamental of Web Mining

Web Mining is the extraction of interesting and potentially useful patterns and implicit information from artifacts or activity related to the WorldWide Web. There are roughly three knowledge discovery domains that pertain to web mining: Web Content Mining, Web Structure Mining, and Web Usage Mining. Web content mining is the process of extracting knowledge from the content of documents or their descriptions. Web document text mining, resource discovery based on concepts indexing or agentbased technology may also fall in this category. Web structure mining is the process of inferring knowledge from the WorldWide Web organization and links between references and referents in the Web. Finally, web usage mining, also known as Web Log Mining, is the process of extracting interesting patterns in web access logs.

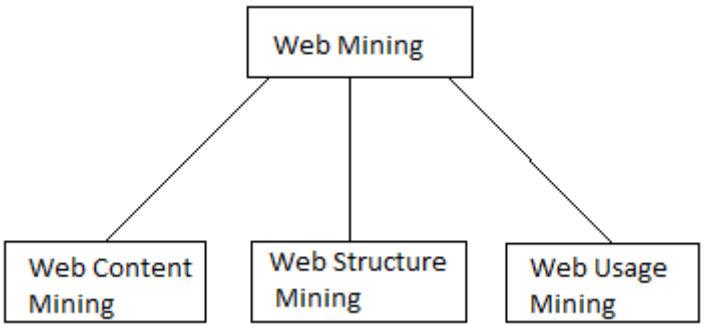

Figure 1.1: Structure of Web mining

1.4.1 Web Content Mining

Web content mining is an automatic process that goes beyond keyword extraction. Since the content of a text document presents no machinereadable semantic, some approaches have suggested to restructure the document content in a representation that could be exploited by machines. The usual approach to exploit known structure in documents is to use wrappers to map documents to some data model. Techniques using lexicons for content interpretation are yet to come. There are two groups of web content mining strategies: Those that directly mine the content of documents and those that improve on the content search of other tools like search engines.

1.4.2 Web Structure Mining

WorldWide Web can reveal more information than just the information contained in documents. For example, links pointing to a document indicate the popularity of the document, while links coming out of a document indicate the richness or perhaps the variety of topics covered in the document. This can be compared to bibliographical citations. When a paper is cited often, it ought to be important. The PageRank and CLEVER methods take advantage of this information conveyed by the links to find pertinent web pages. By means of counters, higher levels cumulate the number of artifacts subsumed by the concepts they hold. Counters of hyperlinks, in and out documents, retrace the structure of the web artifacts summarized.

1.4.3 Web Usage Mining

Web servers record and accumulate data about user interactions whenever requests for resources are received. Analyzing the web access logs of different web sites can help understand the user behaviour and the web structure, thereby improving the design of this colossal collection of resources. There are two main tendencies in Web Usage Mining driven by the applications of the discoveries: General Access Pattern Tracking and Customized Usage Tracking. The general access pattern tracking analyzes the web logs to understand access patterns and trends. These analyses can shed light on better structure and grouping of resource providers. Many web analysis tools exist but they are limited and usually unsatisfactory.

1.5 Organization of Thesis

Following this introduction, chapter 2 presents an overview of the traditional clustering techniques, followed by introduction of clustering and its application and requirements.

Chapter 3 presents a literature survey of previous work to date in the domains of clustering of Web-pages and clustering using Ant behavior that is relevant to the work presented in the remainder of the thesis.

Chapter 4 proposes an Lumer-Faieta algorithm based on Cemetery Organization of Behavior of Ant accompanied with the detailed explanation of the algorithm.

Chapter 5 describes the methodology used in implementing each step of the algorithm whereas chapter 6 shows the experimental results and proves how the proposed algorithm outperforms Lumer-Faieta on datasets.

The thesis ends with discussing the conclusions derived from the work and explores some future enhancements that can be made to the algorithm.

Chapter 2

Web-Page Clustering

unsupervised learning refers to the problem of trying to find hidden structure in unlabeled data. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. Unsupervised learning is closely related to the problem of density estimation in statistics. However unsupervised learning also encompasses many other techniques that seek to summarize and explain key features of the data. Many methods employed in unsupervised learning are based on data mining methods used to preprocess data. Clustering is often called an unsupervised learning task as no class values denoting an a priori grouping of the data instances are given, which is the case in supervised learning.

Web page clustering is one of the major and most important preprocessing steps in web mining analysis. In this context (Web Usage/Context Mining) items to be studied are web pages. Web page clustering puts together web pages in groups, based on similarity or other relationship measures.

2.1 Clustering

Clustering [11, 13] is a data mining technique used to place data elements into related group without advance knowledge of the group definitions.

Clustering is a process which partitions a given data set into homogeneous groups

based on given features such that similar objects are kept in a group whereas dissimilar objects are in different groups. It is the most important unsupervised learning problem. It deals with finding structure in a collection of unlabeled data.

Clustering is the process of assembling the data into classes or clusters so that objects within a cluster have high similarity in relationship to another, but are very dissimilar to objects in other clusters. Data clustering is under vigorous development and is applied to many application areas including business, biology, medicine, chemistry, etc. Owing to the huge amounts of data collected in databases, cluster analysis has recently become a highly active topic in data mining research. For cluster analysis to work efficiently and effectively, as many literatures have presented, there are the following typical *requirements* of clustering in data mining: [14]

1. **Scalability:** Many clustering algorithms work well on small data sets containing fewer than several hundred data objects; however, a large database may contain millions of objects. Clustering on a sample of a given large data set may lead to biased results. Highly scalable clustering algorithms are needed.

2. **Ability to deal with different types of attributes:** Many algorithms are designed to cluster interval-based (numerical) data. However, applications may require clustering other types of data, such as binary, categorical (nominal), and ordinal data, or mixtures of these data types.

3. **Discovery of clusters with arbitrary shape:** Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density. However, a cluster could be of any shape. It is important to develop algorithms that can detect clusters of arbitrary shape.

4. **Minimal requirements for domain knowledge to determine input parameters:** Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters). The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects. This not only burdens users, but it also makes the quality of clustering difficult to control.

5. **Ability to deal with noisy data:** Most real-world databases contain outliers or missing, unknown, or erroneous data. Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

6. **Insensitivity to the order of input records:** Some clustering algorithms cannot incorporate newly inserted data (i.e., database updates) into existing clustering structures and, instead, must determine a new clustering from scratch. Some clustering algorithms are sensitive to the order of input data. That is, given a set of data objects, such an algorithm may return dramatically different clusterings depending on the order of presentation of the input objects. It is important to develop incremental clustering algorithms and algorithms that are insensitive to the order of input.

7. **High dimensionality:** A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions. Human eyes are good at judging the quality of clustering for up to three dimensions. Finding clusters of data objects in high dimensional space is challenging, especially considering that such data can be sparse and highly skewed.

The research is focused on finding user behavior by using efficient and effective cluster analysis.

Problems encountered with clustering algorithm:

- Dealing with a large number of dimensions and large number of objects can be prohibitive due to time complexity
- The effectiveness of an algorithm depends on the definition of similarity (distance)
- The outcomes of an algorithm can be interpreted in different ways

2.1.1 K-means Clustering

k-means[11] is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a

given data set through a certain number of clusters (assume k clusters) fixed apriori.

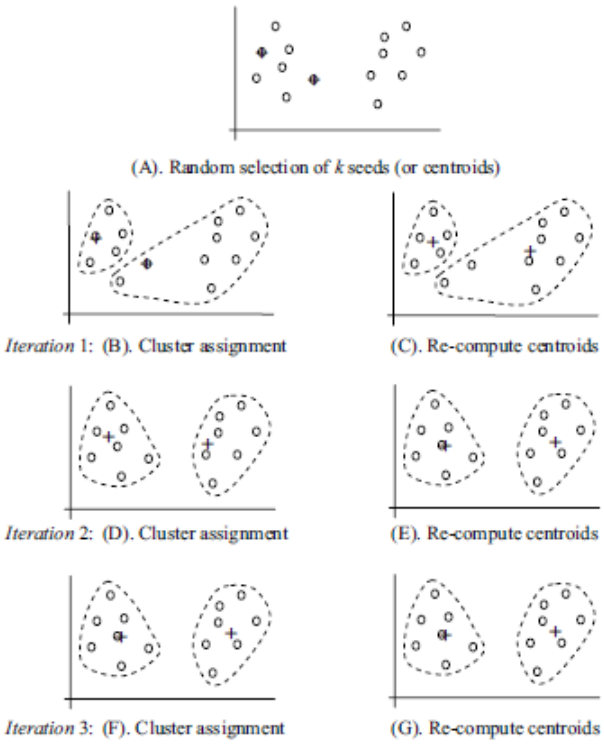

Figure 2.1: K-means Clustering

Algorithm: The k-means algorithm for partitioning, where each clusters center is represented by the mean value of the objects in the cluster.

Input: k is the number of clusters and D is a data set containing n objects.

Output: A set of k clusters.

Method:

1. arbitrarily choose k objects from D as the initial cluster centers;
2. repeat
3. (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
4. update the cluster means, i.e., calculate the mean value of the objects for each cluster;
5. until no change;

Advantages

- Fast, robust and easier to understand
- Relatively efficient: O(tknd), where n is # objects, k is # clusters, d is # dimension of each object, and t is # iterations. Normally, k, t, d $\ll$ n.

Disadvantages

- The learning algorithm requires apriori specification of the number of cluster centers
- The use of Exclusive Assignment - If there are two highly overlapping data then k-means will not be able to resolve that there are two clusters
- Randomly choosing of the cluster center cannot lead us to the fruitful result
- Unable to handle noisy data and outliers

2.1.2 Application of Clustering

- Market Research
  Marketing: Finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
- Pattern Recognition
  WWW: Clustering weblog data to discover groups of similar access patterns
- Image Processing: In image segmentation coding techniques, image is segmented to different regions separated with contours, and coded with different coding techniques. Region growing, c-means, and split and merge methods are used generally for image segmentation.

2.2 Web Page Clustering

A clustering [9] system can be useful in web search for grouping search results into closely related sets of documents. Clustering can improve similarity search by focusing on sets of relevant documents. Clustering is also a valuable technique for analyzing the Web.

Matching the content-based clustering and the hyperlink structure can reveal patterns, duplications, and other interesting structures on the Web.

2.2.1 Representation of web page:

Clustering can be applied to any set of objects as long as a suitable representation of these objects exists. The most common representation, which also works for other machine learning and data mining methods (such as classification), is the attribute-value (or feature-value) representation. In this representation a number of attributes (features) are identified for the entire population, and each object is represented by a set of attribute-value pairs. Alternatively, if the order of the features is fixed, a vector of values (data points) can be used instead. The document vector space model is exactly the same type of representation, where the features are terms.

Vector Space Model

It defines documents [9] as vectors (or points) in a multidimensional Euclidean space where the axes (dimensions) are represented by terms. Depending on the type of vector components (coordinates), there are three basic versions of this representation: Boolean, term frequency (TF), and term frequency-inverse document frequency (TFIDF).

Assume that there are n documents $d_1, d_2, ..., d_n$ and m terms $t_1, t_2, ..., t_m$. Let us denote as n_{ij} the number of times that term t_i occurs in document d_j. In a Boolean representation, document d_j is represented as an m-component vector $\overrightarrow{d_j} = (d_j^1 d_j^2 ... d_j^m)$, where

$$d_j^i = \begin{cases} 0 & if \;\; n_{ij} = 0 \\ 1 & if \;\; n_{ij} > 0 \end{cases} \tag{2.1}$$

As we mentioned earlier, the Boolean representation is simple, easy to compute, and works well for document classification and clustering. However, it is not suitable for keyword search because it does not allow document ranking. Therefore, we focus here on the TFIDF representation.

In the term frequency (TF) approach, the coordinates of the document vector $\overrightarrow{d_j}$ are

represented as a function of the term counts, usually normalized with the document length. For each term t_i and each document d_j , the TF (t_i ,d_j) measure is computed. This can be done in different ways; for example:

- Using the sum of term counts over all terms (the total number of terms in the document):

$$TF(t_i, dj) = \begin{cases} 0 & if\ n_{ij} = 0 \\ \frac{n_{ij}}{\sum_{k=1}^{m} n_{kj}} & if\ n_{ij} > 0 \end{cases} \tag{2.2}$$

- Using the maximum of the term count over all terms in the document:

$$TF(t_i, dj) = \begin{cases} 0 & if\ n_{ij} = 0 \\ \frac{n_{ij}}{max_k n_{kj}} & if\ n_{ij} > 0 \end{cases} \tag{2.3}$$

- Using a log scale to condition the term count:

$$TF(t_i, dj) = \begin{cases} 0 & if\ n_{ij} = 0 \\ 1 + log(1 + log(n_{ij})) & if\ n_{ij} > 0 \end{cases} \tag{2.4}$$

In the Boolean and TF representations, each coordinate of a document vector is computed locally, taking into account only the particular term and document. This means that all axes are considered to be equally important. However, terms that occur frequently in documents may not be related to the content of the document. This is in turn increases the size of the resulting set and makes document ranking difficult if this term is used in the query. The same effect is caused by stopwords such as a, an, the, on, in, and at and is one reason to eliminate them from the corpus.

The basic idea of the inverse document frequency (IDF) approach is to scale down the coordinates for some axes, corresponding to terms that occur in many documents.For each term ti the IDF measure is computed as a proportion of documents where t_i occurs with respect to the total number of documents in the collection. Let D = $\bigcup_1^n d_j$ be the document collection and D_{ti} the set of documents where term t_i occurs. That is, D_{ti} =

$\{ d_j \mid n_{ij} > 0\}$. As with TF, there are a variety of ways to compute IDF; some take a simple fraction $|D|/|D_{ti}|$, others use a log function such as

$$IDF(t_i) = log\frac{1+|D|}{|D_{ti}|} \tag{2.5}$$

In the TFIDF representation each coordinate of the document vector is computed as a product of its TF and IDF components:

$$d_j^i = TF(t_i, d_j)IDF(t_i) \tag{2.6}$$

The types of parameters determine the type of document representation:

- The simplest way to use a term as a feature in a document representation is to check whether or not the term occurs in the document. Thus, the term is considered as a Boolean attribute, so the representation is called **Boolean**.
- The value of a term as a feature in a document representation may be the number of occurrences of the term (**term frequency**) in the document or in the entire corpus. Document representation that includes the term frequencies but not the term positions is called a bag-of-words representation because formally it is a multiset or bag (a type of set in which each item may occur numerous times).
- Term positions may be included along with the frequency. This is a complete representation that preserves most of the information and may be used to generate the original document from its representation.

Relevance Ranking

- The Boolean keyword search is simple and efficient, but it returns a set (unordered collection) of documents (cannot uniquely identify the resulting documents).
- The solution is to rank documents in the response set by relevance to the query and present to the user an ordered list with the top-ranking documents first.
- The Boolean termdocument matrix cannot, however, provide ordering within the documents matching the set of keywords.

- Therefore, additional information about terms is needed, such as counts, positions, and other context information. One straightforward approach is TF-IDF.

2.2.2 Document Representation:

To facilitate the process of matching keywords and documents, some preprocessing steps are taken first:

- Documents are tokenized; that is, all punctuation marks are removed and the character strings without spaces are considered as tokens (words, also called terms).
- All characters in the documents and in the query are converted to upper or lower case.
- Words are reduced to their canonical form (stem, base, or root). For example, variant forms such as is and are are replaced with be, various endings are removed, or the words are transformed into their root form, such as programs and programming into program. This process, called stemming, uses morphological information to allow matching different variants of words.
- Articles, prepositions, and other common words that appear frequently in text documents but do not bring any meaning or help distinguish documents are called stopwords. Examples are a, an, the, on, in, and at. These words are usually removed.

Chapter 3

Literature Survey

Kai-Cheng Hu et. al. [1], have described a novel pheromone update strategy for improving the clustering results of ant colony optimization (ACO).The proposed algorithm is motivated by the observation that most of the ACOs only keep track of the promising foraging information, which has the potential to lead to better solutions than all the other search directions in the pheromone table. This eventually makes the search converge to particular search directions in later iterations because the pheromone values on good routing paths will be reinforced. As such, the breadth of search (diversity) will be reduced, thus limiting the clustering results of ACO.

The proposed algorithm (ACODPT - Ant Colony Optimization Dual Pheromone Tables) adds a second pheromone table to ACO for recording the unpromising foraging information that is worse than all the other search directions and using a novel construction method to explore the new search directions. In other words, by leveraging the strengths of diversification and intensification, the proposed algorithm can find better solutions than traditional ACO.

A query directed web page clustering (QDC) [12], which uses the user's query as part of a reliable measure of cluster quality. The new algorithm has five key innovations: a new query directed cluster quality guide that uses the relationship between clusters and the query, an improved cluster merging method that generates semantically coherent clusters by using cluster description similarity in additional to cluster overlap, a new cluster splitting method that fixes the cluster chaining or cluster drifting problem, an improved heuristic for cluster selection that uses the query directed cluster quality guide, and a

new method of improving clusters by ranking the pages by relevance to the cluster.

Initially there is a singleton cluster for each base cluster. QDC merges clusters using single-link clustering over a relatedness graph. Single-link clustering merges together all clusters that are part of the same connected component on the graph. The relatedness graph has the clusters as vertices and has an edge between any two clusters that are sufficiently similar.

QDC defines two clusters to be sufficiently similar only if both the cluster contents and cluster descriptions are sufficiently similar. Requiring the cluster descriptions to match in addition to the contents dramatically reduces the merging of semantically unrelated clusters and increases cluster quality. Additionally, the cluster contents similarity threshold can be significantly reduced, which allows more semantically related clusters to merge.

Each cluster now contains at least all the base clusters that relate to one idea; this is assured as single-link clustering merges all related clusters. But single-link clustering, even with our improved similarity function, can produce clusters containing multiple ideas and irrelevant base clusters due to cluster chaining (drifting). Such clusters need to be split.

QDC uses a hierarchical agglomerative clustering algorithm to identify the sub-cluster structure within each cluster. The algorithm uses a distance measure to build a dendrogram for each cluster starting from the base clusters in the cluster. Each cluster is split by cutting its dendrogram at an appropriate point - when the distance between the closest pair of sub-clusters falls below a threshold. This threshold means that any groups of base clusters that are not tightly interconnected with each other will be split. Using a higher threshold will lower the split point and increase the splitting frequency.

QDC uses a distance measure with three components: the number of paths between the two sub-clusters on the relatedness graph of length one (onelinks), or of length two (twolinks), and the average distance from base clusters in one sub-cluster to base clusters in the other sub-cluster.

QDC identifies better clusters using a query directed cluster quality guide that considers the relationship between a cluster's descriptive terms and the query terms. Secondly, it increases the merging of semantically related clusters and decreases the merging of semantically unrelated clusters by comparing the descriptions of clusters in addition to comparing the overlap of page contents between clusters. Thirdly, it fixed the cluster chaining (drifting) problem using a new cluster splitting method. Fourthly, it chooses better clusters to show the user by improving the ESTC cluster selection heuristic to consider the number of clusters to select and cluster quality. Finally, it improves the clusters by ranking the pages according to cluster relevance.

Wen Xiong et. al. [3], presents a novel hybrid clustering approach, which uses adaptive ant colony optimization (ACO) to optimize the partition of data set, and utilizes enhanced particle swarm optimization (PSO) to refine the result of the adaptive ACO. The paper says that this algorithm works well for small data set and for large data set it may consumes much time to obtain optimum.

Kate A. Smith et. al. [15], Evaluate the feasibility of using a self-organizing map (SOM) to mine web log data and provide a visual tool to assist user navigation. The resulting map not only provides a meaningful navigation tool (for web users) that is easily incorporated with web browsers, but also serves as a visual analysis tool for webmasters to better understand the characteristics and navigation behaviors of web users visiting their pages.

There are several advantages to using the SOM to cluster documents, rather than people, due to the objectivity of the process. In addition, the process is automatic (hence the name "self-organizing"). It can thus be done on a large scale and therefore saves labour costs. It also facilitates search by concept instead of search by keyword.

The organization of the web documents is based solely on the user's navigation behavior. It has been demonstrated that the resulting map of this system is very meaningful and can be easily incorporated with a web browser to assist user navigation.

B. Praveen et. al. [7], compares various clustering algorithms such as K-means, Fuzzy c-means, Subtractive Clustering and K-modes used for grouping of web user sessions. The clusters formed as a result of applying these algorithms are aggregated to form web user profiles. The recommendation engine uses these profiles, to generate pages for recommendation. The recommendation effectiveness is evaluated using standard measures such as coverage, precision and F1 measure. Subtractive Clustering fared well on msnbc data set and Fuzzy c-means on msweb data. K-modes performed well for recommendation.

In [16] document clustering, each document corresponds to an item and each possible feature corresponds to a transaction. A frequent item set found using the association rule discovery algorithm corresponds to a set of documents that have a sufficiently large number of features in common. These frequent item sets are mapped into hyperedges in a hypergraph. A hypergraph H = (V,E) consists of a set of vertices V and a set of hyperedges E. A hypergraph is an extension of graph in the sense that each hyperedge can connect more than two vertices. In this model, the set of vertices V corresponds to the documents and each hyperedge e E corresponds to a set of related documents found. The weight of hyperedge is calculated as the average confidence of all the association rules involving the related documents of the hyperedge. E.g: If $\{d_1, d_2, d_3\}$ is a frequent item set, then the hypergraph contains a hyperedge that connects d_1, d_2 and d_3 . The confidence of an association rule involving documents like $\{d_1, d_2\} \Rightarrow \{d_3\}$ is the conditional probability that a feature occurs in document d_3 whenever it occurs in d_1 and d_2. Given all the possible association rules as $\{d_1\} \overset{0.8}{\Rightarrow} \{d_2, d_3\}$, $\{d_1, d_2\} \overset{0.4}{\Rightarrow} \{d_3\}$, $\{d_1, d_3\} \overset{0.6}{\Rightarrow} \{d_2\}$, $\{d_2\} \overset{0.4}{\Rightarrow} \{d_1, d_3\}$, $\{d_2, d_3\} \overset{0.8}{\Rightarrow} \{d_1\}$ and $\{d_3\} \overset{0.6}{\Rightarrow} \{d_1, d_2\}$, the weight of the hyperedge is 0.6.

Next a hypergraph partitioning algorithm is used to partition the hypergraph such that the weight of hyperedges that are cut by the partitioning is minimized. Note that by minimizing the hyperedge-cut we essentially minimize the relations that are violated by partitioning the documents into different clusterds. Similarly this method can be applied to a word clustering. In this setting each word corresponds to an item and each document corresponds to a transaction.

Ms. Vinita Shrivastava et. al. [19], Propose a new technique to enhance the learning capabilities and reduce the computation intensity of a competitive learning multi-layered neural network using the K-means clustering algorithm. The proposed model use multi-layered network architecture with a back propagation learning mechanism to discover and analyze useful knowledge from the available Web log data.

In [8] clustering algorithms based on Harmony Search (HS) optimization method that deals with web document clustering. By modeling clustering as an optimization problem, first, we propose a pure HS based clustering algorithm that finds near global optimal clusters within a reasonable time. Then we hybridize K-means and harmony clustering to achieve better clustering. Experimental results on five different data sets reveal that the proposed algorithms can find better clusters when compared to similar methods and the quality of clusters is comparable.

The globally optimal partition of a given set of web pages is considered and novel algorithms, named HSCLUST and HK CLUST, by modeling clustering problem as an optimization of an objective function, are proposed. HSCLUST is appropriate for finding near global regions within a reasonable time, but not as good as K-means at fine-tuning within those regions. So, second improvement was hybridization of K-means and HSCLUST algorithm sequentially. The hybrid algorithm combines the power of the HSCLUST with the speed of a K-means and the global searching stage and local refine stage are accomplished by these two modules, respectively.

Karunesh Gupta et. al. [6], A Hybrid FCM with GA is to be a powerful extension of the famous Fuzzy C-Means algorithm. Essentially the Hybrid FCM with GA algorithm is a learning algorithm which can mine databases to build an optimized and efficient clustering results. Hybrid FCM with GA improved the global search capability of GA and also the local search capability of FCM, and hence can better solve the clustering problem.

Chapter 4

The Proposed Algorithm

Swarm Intelligence[4] is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. Examples, bird flocking, fish schooling, etc...

SI is concerned with the study of systems comprising many individuals that:
◇ are relatively simple (with respect to task they are asked to solve)
◇ coordinate via distributed control
◇ sense their environment locally
◇ can communicate directly only with their neighbors
◇ can communicate indirectly via stigmergy
◇ have stochastic behavior
◇ exploit positive and negative feedback mechanisms

4.1 Preprocessing Steps

Input a Data file in ARFF format

- Convert the web documents into text documents. For example, use the "Save As . . . " option of the Internet Explorer with "Save as type: Text File (*.txt)."
- Save all documents in a single file (action: concatenate, on pages: text). Convert it to text format [as done in part (b)] and examine its content.

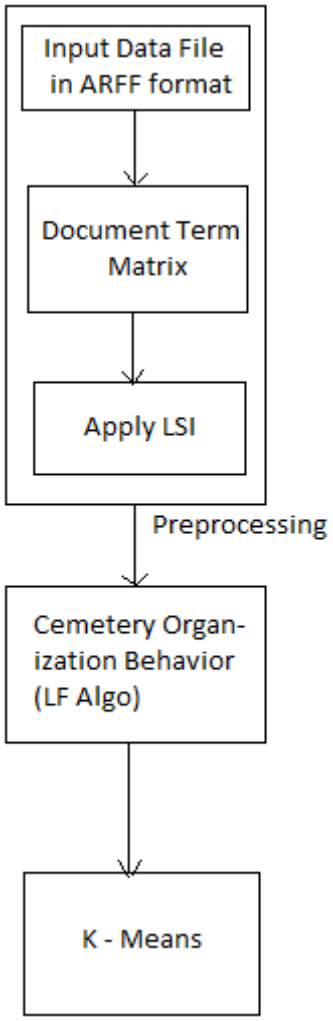

Figure 4.1: Proposed Algorithm

- Use the concatenation of the web documents and create a text file where each document is represented on a separate line in plain text format.

- Enclose the document content in quotation marks (") and add the document name at the beginning of each line and a file header at the beginning of the file.
  Anthropology, " Anthropology consists of four ...
  ...

- This representation uses two attributes: document-name and documentcontent, both of type string.

Document-Term Matrix

- Load the file in the Weka system using the "Open file" button in "Preprocess" mode.

- After successful loading the system shows some statistics about the number of attributes, their type, and the number of instances (rows in the data section or documents).

- Choose the StringToNominal filter and apply it to the first attribute, document name. Then choose the StringToWordVector filter and apply it with "outputWordCounts= true" (you may also change the setting of "onlyAlphabeticTokens" and "useStoplist" to see how the results change).

- Now you have a document-term matrix loaded in Weka. Use the "Edit" option to see it in a tabular format, where you can also change its content or copy it to other applications (e.g., MS Excel). Once created in Weka the table can be stored in an ARFF file through the "Save" option.

Latent semantic indexing (LSI) Latent semantic indexing (LSI) is an indexing and retrieval method that uses a mathematical technique called singular value decomposition (SVD) to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text. LSI is based on the principle that words that are used in the same contexts tend to have similar meanings. A key feature of LSI is its ability to extract the conceptual content of a body of text by establishing associations

between those terms that occur in similar contexts.

LSI is also an application of correspondence analysis, a multivariate statistical technique, to a contingency table built from word counts in documents.

Called Latent Semantic Indexing because of its ability to correlate semantically related terms that are latent in a collection of text, it was first applied to text at Bell Laboratories in the late 1980s. The method, also called latent semantic analysis (LSA), uncovers the underlying latent semantic structure in the usage of words in a body of text and how it can be used to extract the meaning of the text in response to user queries, commonly referred to as concept searches. Queries, or concept searches, against a set of documents that have undergone LSI will return results that are conceptually similar in meaning to the search criteria even if the results don't share a specific word or words with the search criteria.

Benefits of LSI

LSI overcomes two of the most problematic constraints of Boolean keyword queries: multiple words that have similar meanings (synonymy) and words that have more than one meaning (polysemy). Synonymy is often the cause of mismatches in the vocabulary used by the authors of documents and the users of information retrieval systems. As a result, Boolean or keyword queries often return irrelevant results and miss information that is relevant.

LSI is also used to perform automated document categorization. In fact, several experiments have demonstrated that there are a number of correlations between the way LSI and humans process and categorize text. Document categorization is the assignment of documents to one or more predefined categories based on their similarity to the conceptual content of the categories. Dynamic clustering based on the conceptual content of documents can also be accomplished using LSI.

Because it uses a strictly mathematical approach, LSI is inherently independent of language. This enables LSI to elicit the semantic content of information written in any

language without requiring the use of auxiliary structures, such as dictionaries and thesauri. LSI can also perform cross-linguistic concept searching and example-based categorization. For example, queries can be made in one language, such as English, and conceptually similar results will be returned even if they are composed of an entirely different language or of multiple languages.

Text does not need to be in sentence form for LSI to be effective. It can work with lists, free-form notes, email, Web-based content, etc. As long as a collection of text contains multiple terms, LSI can be used to identify patterns in the relationships between the important terms and concepts contained in the text.

4.2 Ant Based Clustering Based on Cemetery Organization

Deneubourg et al. [18] have proposed the general idea is that isolated items should be picked up and dropped at some other location where more items of that type are present. Let us assume that there is only one type of item in the environment. The probability p_p for a randomly moving, unladen agent (representing an ant in the model) to pick up an item is given by:

$$p_p = (\frac{k_1}{k_1 + f})^2 \tag{4.1}$$

where f is the perceived fraction of items in the neighborhood of the agent, and k_1 is a threshold constant. When f $\ll k_1$, p_p is close to 1, that is, the probability of picking up an item is high when there are not many items in the neighborhood. p_p is close to 0 when f $\gg k_1$, that is, items are unlikely to be removed from dense clusters. The probability p_d for a randomly moving loaded agent to deposit an item is given by:

$$p_d = (\frac{f}{k_2 + f})^2 \tag{4.2}$$

where k_2 is another threshold constant: for $f \ll k_2, p_d$ is close to 0, whereas for $f \gg k_2, p_d$ is close to 1. As expected, the depositing behavior obeys roughly opposite rules.

Exploratory Data Analysis

The Algorithm introduced by Lumer and Faieta [15] consists of projecting the space of attributes onto some lower dimensional space, typically of dimension z = 2, so as to make clusters appear with the following property: intracluster distances should be small with respect to inter-cluster distances, that is, attribute distances between objects that belong to different clusters.

The LF algorithm works as follows. Let us assume that z = 2. Instead of embedding the set of objects into $\Re^2$ or a subspace of $\Re^2$, they approximate this embedding by considering a grid, that is, a subspace of Z^2, which can also be considered a discretization of a real space. Agents that are moving in this discrete space can directly perceive a surrounding region of area s^2 (a square $Neigh_{(s\times s)}$ of $s \times s$ sites surrounding site r). Direct perception allows a more efficient evaluation of the state of the neighborhood than the memory-based procedure used in the BM: while the BM was aimed to a robotic implementation, the LF algorithm is to be implemented in a computer, with significantly fewer material constraints. Let $d(o_i, o_j)$ be the distance between two objects o_i and o_j in the space of attributes. Let us also assume that an agent is located at site r at time t, and finds an object o_i at that site. The "local density" $f(o_j)$ with respect to object o_i at site r is given by

$$f(o_i) = \begin{cases} \frac{1}{s^2} \sum_{o_j \epsilon Neigh_{s\times s}(r)} \left[1 - \frac{d(o_i,o_j)}{\alpha}\right] & if \;\; f > 0, \\ 0 & otherwise \end{cases} \tag{4.3}$$

$f(o_i)$ is a measure of the average similarity of object O_i with the other objects o_j present in the neighborhood of o_i.$f(o_i)$ replaces the fraction f of similar objects in the basic model (BM). α is a factor that defines the scale for dissimilarity: it is important for it deter-

mines when two items should or should not be located next to each other. For example, if is too large, there is not enough discrimination between different items, leading to the formation of clusters composed of items which should not belong to the same cluster. If, on the other hand, α is too small, distances between items in attribute space are amplified to the point where items which are relatively close in attribute space cannot be clustered together because discrimination is too high.

Let us briefly examine the behavior of f by looking at the extreme cases. If, for example, all s^2 sites around r are occupied by objects that are similar to o_i ($\forall o_j \epsilon Neigh_{(s\times s)}(r)$, $d(o_i, o_j) = 0$), then $f(o_i) = 1$, and the object should be picked up with low probability. If all s^2 sites around r are occupied by objects that are maximally dissimilar to o_i ($\forall o_j \epsilon Neigh_{(s\times s)}(r)$, $d(o_i, o_j) = d_{max}$), then $f(o_i)$ is small, and the object should be picked up with high probability. Finally, if all sites around r are empty, then, obviously, $f(o_i) = 0$, and the object should be picked up with a high probability. An easy generalization shows that o_i should be picked up with high (respectively, low) probability when f is close to 0 respectively, close to 1). Lumer and Faieta define picking up and dropping probabilities as follows:

$$p_p(o_i) = (\frac{k_1}{k_1 + f(o_i)})^2 \tag{4.4}$$

$$p_d(o_i) = \begin{cases} 2f(o_i), & if \ \ f(o_i) < k_2 \\ 1, & if \ \ f(o_i) \geq k_2 s \end{cases} \tag{4.5}$$

Where k_1 and k_2 are two constant that play a role similar to k_1 and k_2 in Basic Model.

Algorithm High-level description of the Lumer Faieta algorithm

/* **Initialization** */

For every item o_i **do**

place o_i randomly on grid

End For

For all agents **do**

place agent at randomly selected site

End For

/* **Main loop** */

For t = 1 to t_{max} **do**

For all agents **do**

If ((agent unladen) and (site occupied by item o_i)) **then**

Compute $f(o_i)$ and $p_p(o_i)$

Draw random real number R between 0 and 1

If (R $\leq p_p(o_i)$) **then**

Pick up item o_i

End If

Else If ((agent carrying item o_i) and (site empty)) **then**

Compute $f(o_i)$ and $p_d(o_i)$

Draw random real number R between 0 and 1

If (R $\leq p_d(o_i)$) **then**

Drop item

End If

End If

Move to randomly selected neighboring site not occupied by other agent

End For

End For

Print location of items

To illustrate the functioning of their algorithm, Lumer and Faieta use a simple example in which the attribute space is $\Re^2$, and the values of the two attributes for each object correspond to its coordinates (x,y) in $\Re^2$. Four clusters of 200 points are generated in attribute space, with x and y distributed according to normal (or gaussian) distributions $N(\mu, \sigma)$ of average μ, and variance σ^2:

1. x $\propto$ N(0.2,0.1), y $\propto$ N(0.2,0.1),
2. x $\propto$ N(0.8,0.1), y $\propto$ N(0.2,0.1),
3. x $\propto$ N(0.8,0.1), y $\propto$ N(0.8,0.1),
4. x $\propto$ N(0.2,0.1), y $\propto$ N(0.8,0.1),

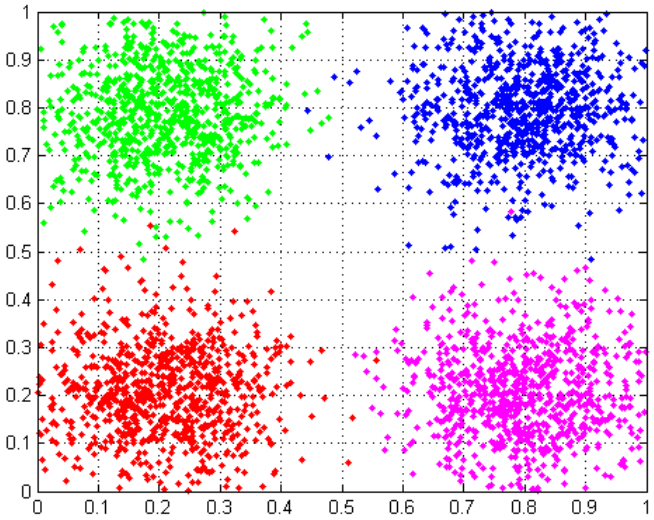

Figure 4.2: Distribution of points in "attribute space": 4 clusters of 200 points each are generated in attribute space, with x and y distributed according to normal (or gaussian) distributions $N(\mu, \sigma)$.

for clusters 1, 2, 3, and 4, respectively. The data points were then assigned to random locations on a 100 $\times$ 100 grid, and the clustering algorithm was run with 10 agents. Figures show the system at t = 0, t = 500,000, and t = 1,000,000. At each time step, all agents have made a random move and possibly performed an action. Grid coordinates have been scaled to [0,1] $\times$ [0,1]. Items that belong to different clusters are represented by different symbols: o, +, *, and x. Objects that are clustered together belong to the same initial distribution, and objects that do not belong to the same initial distribution are

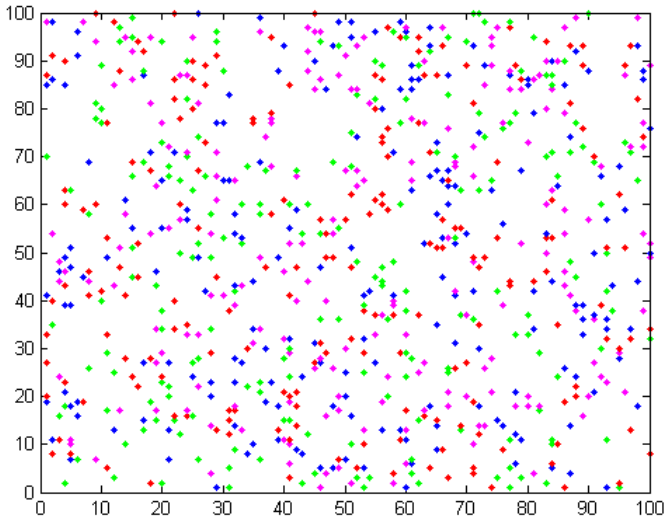

Figure 4.3: Initial spatial distribution of 800 items on a 100 × 100 grid. Grid coordinates have been scaled to [0,1] × [0,1]. Items that belong to different clusters are represented by different symbols: $\circ, +, *$, and $\times$. Distribution of items at t = 500,000. $k_1 = 0.1$, $k_2 = 0.15$, $\alpha = 0.5$, $s^2 = 9$.

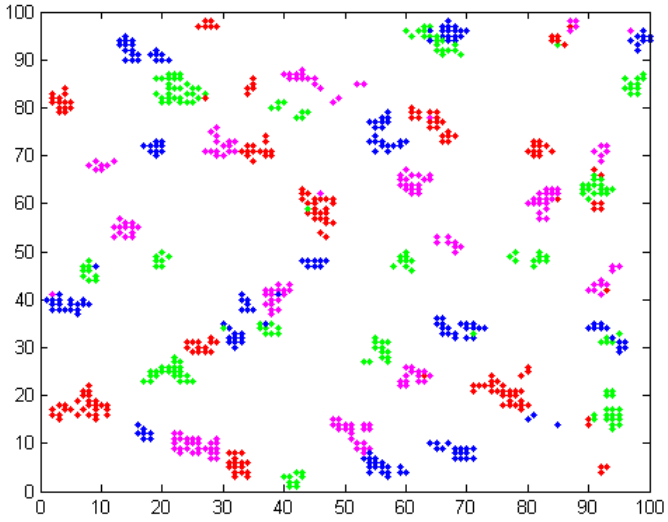

Figure 4.4: Items that belong to different clusters are represented by different symbols: $\circ, +, *$, and $\times$. Distribution of items at t = 500,000. $k_1 = 0.1$, $k_2 = 0.15$, $\alpha = 0.5$, $s^2 = 9$.

found in different clusters. *But there are generally more clusters in the projected system than in the initial distribution.*

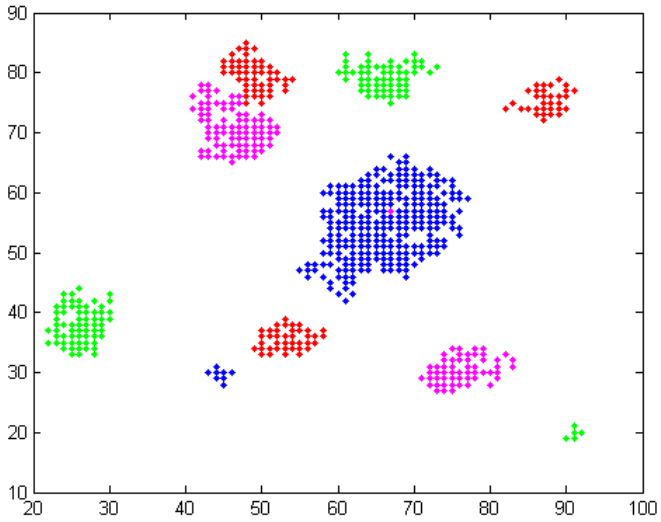

Figure 4.5: Items that belong to different clusters are represented by different symbols: $\circ, +, *$, and $\times$. Distribution of items at t = 1,000,000. $k_1 = 0.1$, $k_2 = 0.15$, $\alpha = 0.5$, $s^2 = 9$.

Modifications to Lumer - Faieta Algorithm

The algorithm developed by Lumer and Faieta [15] showed success in a number of applications. The algorithm does, however, have the tendency to create more clusters than necessary. A number of modifications have been made to address this problem, and to speed up the search:

1. **Agents with different moving speeds.** Let v be the speed of an agent (v is the number of grid units walked per time unit by an agent along a given grid axis). v is distributed uniformly in [l,v_{max}]. The simulations use $v_{max} = 6$. v also influences, through the function f, the tendency of an agent to either pick up or drop an object:

$$f(o_i) = \begin{cases} \frac{1}{s^2} \sum_{o_j \epsilon Neigh_{s \times s}(r)} \left[1 - \frac{d(o_i,o_j)}{\alpha(1+\frac{v-1}{v_{max}})} \right] & if \;\; f > 0, \\ 0 & otherwise \end{cases} \tag{4.6}$$

   Therefore, fast moving ants are not as selective as slow ants in their estimation of the average similarity of an object to its neighbors. The diversity of agents allows

the formation of clusters over various scales simultaneously: fast agents form coarse clusters on large scales, that is, drop items approximately in the right coarse-grained region, while slow agents take over at smaller scales by placing objects with more accuracy. Since v serves as a sort of temperature, the presence of agents with different values of v corresponds to a system operating at different temperatures at the same time.

2. **A short-term memory.** Agents can remember the last m items they have dropped along with their locations. Each time an item is picked up, the agent compares the properties of the item with those of the m memorized items and goes toward the location of the most similar instead of moving randomly. This behavior leads to a reduction in the number of equivalent clusters, since similar items have a low probability of initiating independent clusters.

3. **Behavioral switches.** The system exhibits some kind of self-annealing since items are less and less likely to be manipulated as clusters of similar objects form. Lumer and Faieta have added the possibility for agents to start destroying clusters if they haven't performed an action for a given number of time steps. This procedure allows a "heating up" of the system to escape local nonoptimal configurations.

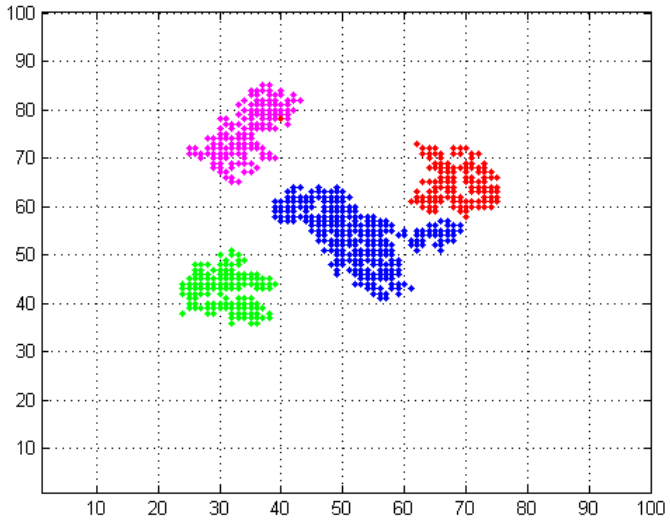

Figure 4.6: Spatial distribution of 800 items on a 100 x 100 grid at t = 1,000,000. Items that belong to different clusters are represented by different symbols: o, +, *, and x. k_1 = 0.1, k_2 = 0.15, α = 0.5, s^2 = 9, m = 8, V_{max} = 6.

Chapter 5

Simulation Results

This chapter covers the details of the tool used for implementation of the project and other relevant details. The chapter also presents the snapshots of the running algorithm and explains the significance of some steps on the final output.

5.1 Tool Used

The proposed algorithm Lumer-Faieta have been implemented in MATLAB 7.8.0 (R2009a). Data preprocessing performed in Weka 3.6.6. All the experiments are performed on Windows Platform.

The reason behind choosing MATLAB for Implementation of the proposed algorithm is as follow:

- Interactive interface
- Debugging facilities
- High quality graphics and visualization facility
- MATLAB's add on feature in the form of toolboxes: making it possible to extend the existing capabilities of the language with ease
- Can manipulate large amount of data: which is the basic requirement in Data Mining

- Researchers in [7] suggest that a way to speed up the algorithm is to make a faster implementation using C/C++/MATLAB rather than JAVA: Hence the well known tool WEKA is not used.
- MATLAB has an innumerable in-built functions ready in it which makes the implementation work very easy.

Weka

Weka (Waikato Environment for Knowledge Analysis) is a comprehensive toolbench for machine learning and data mining. It contains implementations of algorithms for classification, clustering, and association rule mining, along with graphical user interfaces and visualization utilities for data exploration and algorithm evaluation.

5.2 Performance Measure

The following 4 terms are used for performance measure quantity.

- True Positive (TP): the number of webpage correctly classified to that cluster
- False Positive (FP): the number of webpage incorrectly classified to that cluster
- True Negative (TN): the number of webpage correctly rejected from that cluster
- False Negative (FN): the number of webpage incorrectly rejected from that cluster

The easiest way to see the structure of the error is to include the number of documents falling in each of the categories above in a matrix called a confusion matrix (also, contingency table) as follows:

	Predicted (Clusters)	
Actual(Classes)	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Table 5.1: Confusion Matrix

In these terms the overall classes to clusters error and accuracy are the following:

$$error = \frac{FP + FN}{TP + FP + TN + FN} \tag{5.1}$$

$$accurcy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.2}$$

$$precision = \frac{TP}{TP + FP} \tag{5.3}$$

$$recall = \frac{TP}{TP + FN} \tag{5.4}$$

A confusion matrix can be built with more than two classes and clusters by using more rows (for the classes) and more columns (for the clusters). Thus, we can define a generalized confusion matrix for m classes and k clusters as shown in Table 5.2. The number n_{ij} in each cell indicates the number of documents from cluster j that belong to class i. Now we can define recall and precision with respect to class i and cluster j as follows:

	Clusters				
Classes	1	...	j	...	k
1	n_{11}	...	n_{1j}	...	n_{1k}
.					
.					
.					
i	n_{i1}	...	n_{ij}	...	n_{ik}
.					
.					
.					
m	n_{m1}	...	n_{mj}	...	n_{mk}

Table 5.2: Confusion Matrix for m classes and k clusters

$$P(i,j) = \frac{n_{ij}}{\sum_{i=1}^{m} n_{ij}} \tag{5.5}$$

$$R(i,j) = \frac{n_{ij}}{\sum_{j=1}^{k} n_{ij}} \tag{5.6}$$

$$F(i,j) = \frac{2P(i,j)R(i,j)}{P(i,j) + R(i,j)} \tag{5.7}$$

To get rid of the indices we take the maximum of F(i,j) over all clusters and then sum across classes. As classes generally include different numbers of documents, we weight

their contribution to the sum with the proportion of documents in each. Thus, we obtain the F-measure for the entire clustering.

$$F = \sum_{i=1}^{m} \frac{n_i}{n} \quad j = \overset{\max}{1, ..., k} \quad F(i, j) \tag{5.8}$$

where $n_i = \sum_{j=1}^{k} n_{ij}$ (the number of documents belonging to class i, or row i total) and n = $\sum_{i=1}^{m} \sum_{j=1}^{k} n_{ij}$ (the total number of documents in the sample).

The F-measure provides a more precise account for the error than does the overall accuracy. The explanation is that the F-measure is actually the harmonic mean of precision and recall. The harmonic mean is used for averaging rates (ratios of two quantities specified in different units, such as distance/time and price/count). The precision and recall can be seen as rates, although not of the same kind, but it seems that the harmonic mean works well for averaging them.

5.3 Data Sets

To verify the effectiveness of the algorithm proposed in this paper, we conducted experiments on different datasets from various real domains. The aim behind using different datasets for experimentation is to prove the consistency of the proposed algorithm in different domains. The details about these datasets are listed in Table 5.3. The number of instances in these datasets range from hundreds to more than a million, verifying the performance of proposed algorithm on datasets with different sizes.

	Documents	Attributes	Clusters
Bank Search	600	22513	2
All Text	662	21231	4
All Text Combine	364	25927	8

Table 5.3: Composition of Data Sets

In preprocessing, make term document matrix using StringToNominal filter and StringToWordVector filter. In StringToWordVector filter, use TF-IDF document representa-

tion, set OutputWordCount True, set LovinsStemmer as Stemmer, set useStoplist True.

5.4 Experimental Results

In this phase, Some initial runs were performed and The results showing F-measure for different Data Sets for two distance measures euclidean and cosine is shown in Fig 5.4.

Fig 5.1 shows result of Cemetery Organization algorithm for Bank search data with Cosine distance measure.

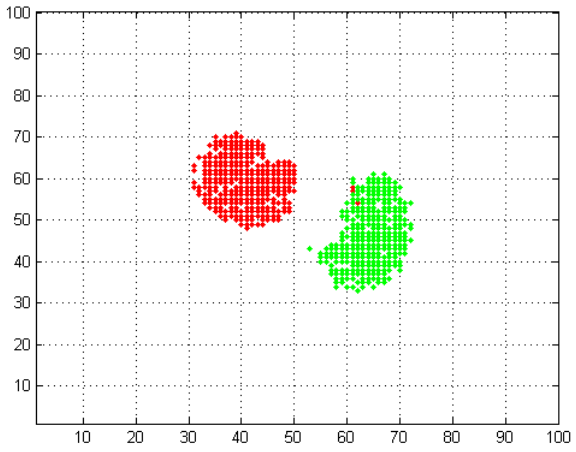

Figure 5.1: Bank Data Sets

Fig 5.2 shows result of Cemetery Organization algorithm for All Text data with Euclidean distance measure.

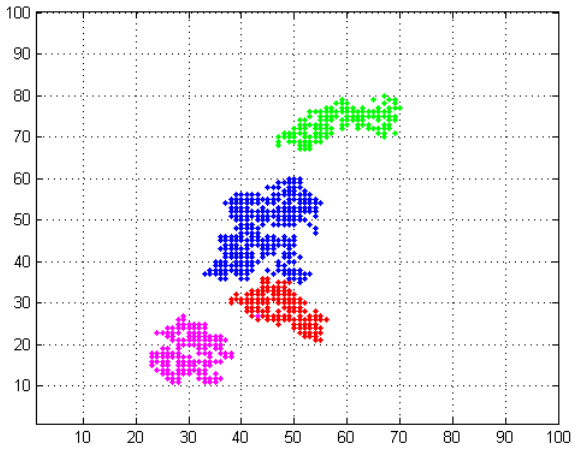

Figure 5.2: All Text Data Sets

Fig 5.3 shows result of Cemetery Organization algorithm for All Text Combine data

with Euclidean distance measure.

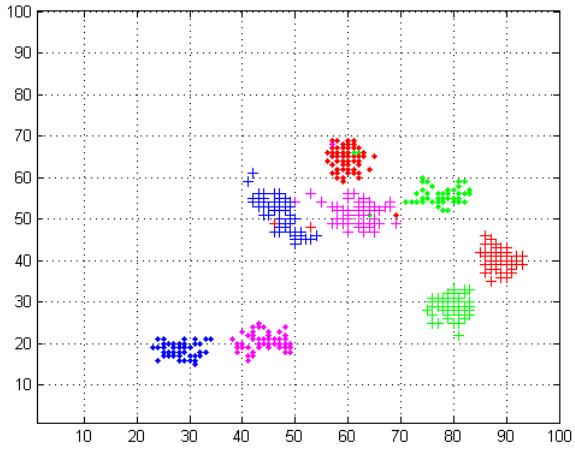

Figure 5.3: All Text Combine Data Sets

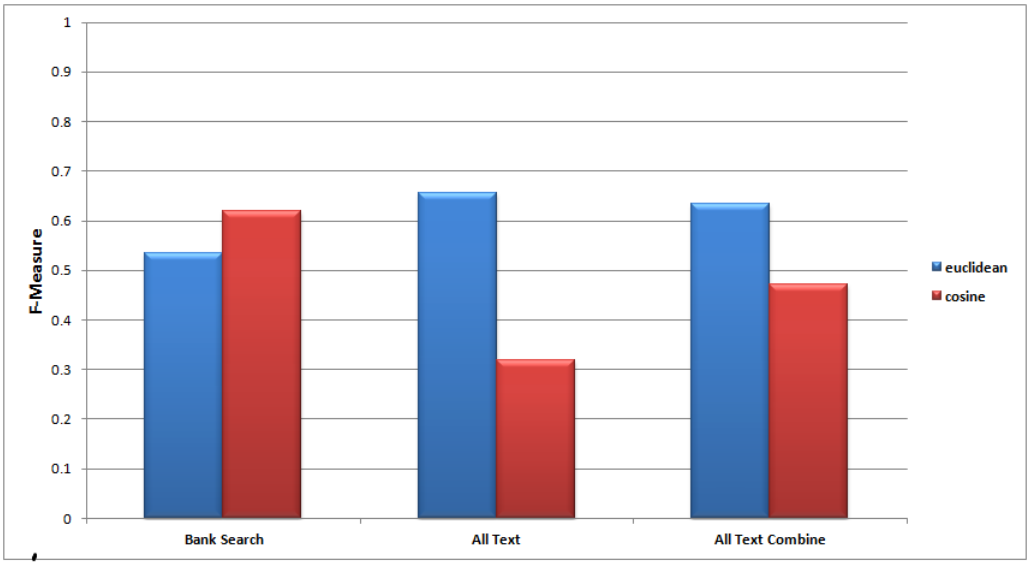

Figure 5.4: F-measure for All Attributes in Data Sets

For each run, the accuracy keeps on varying even with same options. To reduce the stochastic noise in accuracy measurements we need to perform multiple runs. The tables show the readings for averaging over 10 runs.

Table 5.4 shows result in terms of F-measure for Bank search data. The tables consist of various distance measures (i.e. Euclidian and cosine) for different feature selection methods (LSI and Information gain).

From above table5.4 we conclude the Best result in cosine distance, No. of attributes

No of Attributes	LSI		Info. Gain	
	Euclidean	Cosine	Euclidean	Cosine
300	0.63663	0.83401	0.64824	0.76717
400	0.65656	0.87471	0.67828	0.83476
500	0.66340	**0.88590**	0.69985	**0.83780**
600	0.66865	0.87996	0.69924	0.83667
700	0.63137	0.85229	0.65427	0.81357

Table 5.4: F-measures for Bank Search Data

500 in LSI feature selection for Bank Search Data is 0.88590 and in Info. Gain No. of attributes 500 in cosine is 0.83780.

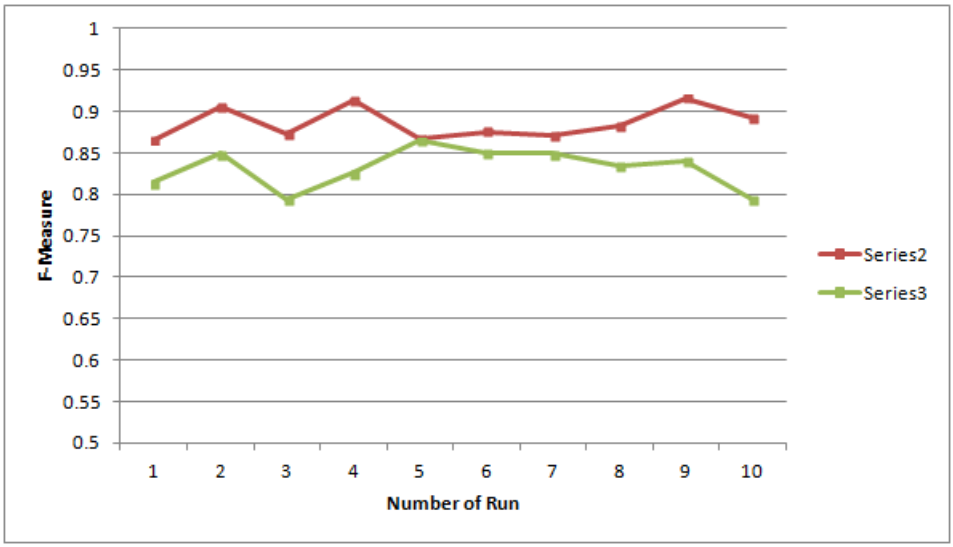

Figure 5.5: Best F-measure for Bank Search Data across 10 runs for as LSI=500 cosine and as Info Gain=500 cosine

Table 5.5 shows result in terms of F-measure for All Text data. The tables consist of various distance measures (i.e. Euclidian and cosine) for different feature selection methods (LSI and Information gain).

No of Attributes	LSI		Info. Gain	
	Euclidean	Cosine	Euclidean	Cosine
300	0.88822	0.29102	0.85845	0.36142
400	0.90492	0.40115	**0.85685**	0.46612
500	**0.91554**	0.50836	0.85531	0.53608
600	0.90076	0.51133	0.85601	0.53980
700	0.89035	0.47876	0.84965	0.51722

Table 5.5: F-measures for All Text Data

From above table5.5 we conclude the Best result in cosine distance, No. of attributes 500 in LSI feature selection for All Text Data is 0.91554 and in Info. Gain No. of

attributes 400 in cosine is 0.85685.

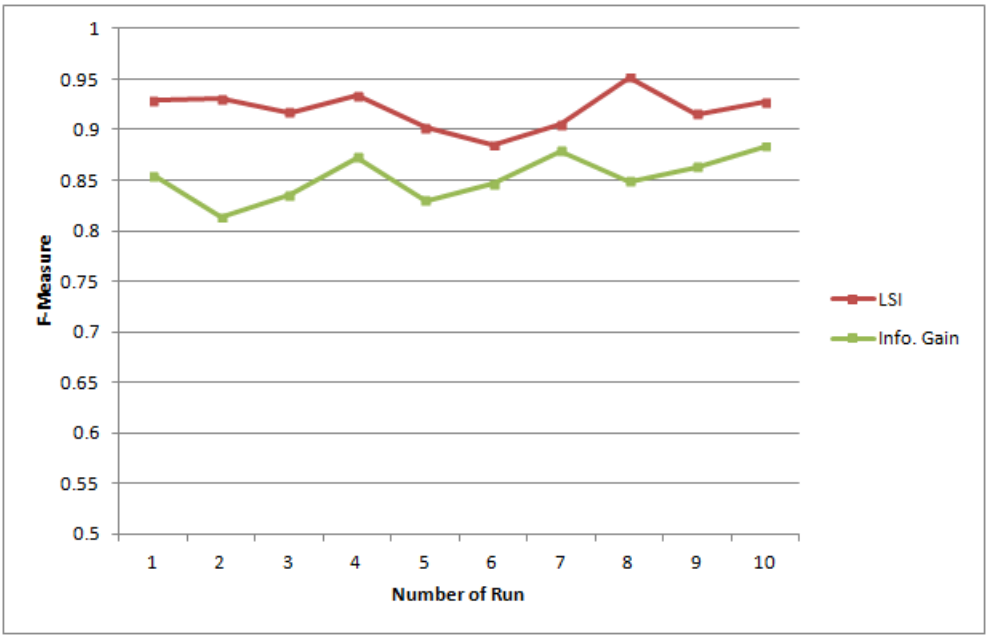

Figure 5.6: Best F-measure for All Text Data across 10 runs for as LSI=500 cosine and as Info Gain=400 cosine

Table 5.6 shows result in terms of F-measure for All Text Combine data. The tables consist of various distance measures (i.e. Euclidian and cosine) for different feature selection methods (LSI and Information gain).

No of Attributes	**LSI**		**Info. Gain**	
	Euclidean	**Cosine**	**Euclidean**	**Cosine**
300	0.87588	0.65267	0.80001	0.66492
400	0.89277	0.68594	**0.79825**	0.68923
500	**0.89635**	0.69105	0.79716	0.69443
600	0.89394	0.69657	0.79950	0.70126
700	0.88012	0.67590	0.77356	0.67837

Table 5.6: F-measures for Bank Search Data

From above table5.6 we conclude the Best result in cosine distance, No. of attributes 500 in LSI feature selection for All Text Combine Data is 0.89635 and in Info. Gain No. of attributes 400 in cosine is 0.79825.

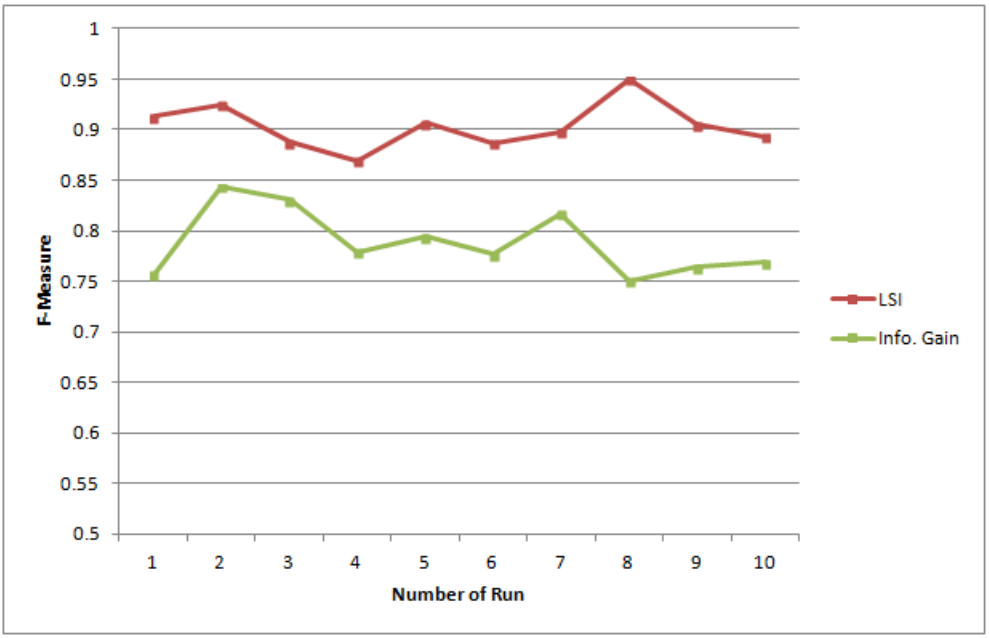

Figure 5.7: Best F-measure for All Text Data across 10 runs for as LSI=500 cosine and as Info Gain=400 cosine

Chapter 6

Conclusion and Future Work

In this thesis, algorithm inspired by cemetery organization behavior of ant is used to cluster web pages. Web pages are preprocessed and represented in low dimensional space using Latent Semantic Indexing. Algorithm implemented by us first clusters the web pages and maps them on to two dimensional grid space. We have decided optimal number of clusters by analyzing this grid space. Optimal number of clusters and web pages represented on this two dimensional grid space are then used in k-means algorithm to achieve final clustering of web pages. Implementation results are promising and shows the effectiveness of the proposed framework.

Developed Algorithm will be tested on some other benchmark dataset. Particle Swarm Optmization will also be used to cluster the web pages.

Bibliography

[1] K.-C. H. Chun-Wei Tsai and M.-C. Chiang, "Ant colony optimization with dual pheromone tables for clustering," IEEE International Conference on Fuzzy Systems, June 2011.

[2] H. K. and A. K., "Clustering Algorithm Employ in Web Usage Mining: An Overview," Bharati Vidyapeeths Institute of Computer Applications and Management, New Delhi, March 2011.

[3] W. Xiong and C. Wang, "A novel hybrid clustering based on adaptive aco and pso," IEEE, 2011.

[4] M. V. S. G. Mr. Pankaj K. Bharne and M. S. K. Yewale, "Data clustering algorithms based on swarm intelligence," IEEE, 2011.

[5] O. M. Jafar and R. Sivakumar, "Ant-based clustering algorithms: A brief survey," International Journal of Computer Theory and Engineering, Octomber 2010.

[6] K. Gupta and M. Shrivastava, "Web usage mining clustering using hybrid fcm with ga," International Journal of Advanced Computer Research, June 2010.

[7] V. B. Praveen, "Influence of various clustering algorithms on web personalization," Proceeding of the International Workshop on Machine Intelligence Research, 2009.

[8] Rana Forsati, Mehrdad Mahdavi, Mohammadreza Kangavari and Banafsheh Safarkhani, "WEB PAGE CLUSTERING USING HARMONY SEARCH OPTIMIZATION", Department of Computer Engineering, Tehran Azad University, Tehran, Iran, IEEE, 2008.

[9] Z. Markov and D. T. Larose, DATA MINING THE WEB Uncovering Patterns in Web Content, Structure, and Usage. John Wiley & Sons, 2007.

[10] B. Liu, Web DataMining: Exploring Hyperlinks, Content, and Usage Data, Springer, Department of Computer Science, University of Illinois at Chicago, 2007.

[11] J. Han and M. Kamber, Data Mining: Concepts and Techniques, Elsevier Inc. 2006.

[12] Daniel Crabtree, Peter Andreae and Xiaoying Gao, "Query Directed Web Page Clustering", Victoria University of Wellington New Zealand, IEEE, 2006.

[13] R. Xu and D. W. II, "Survey of Clustering Algorithms," IEEE Trans. Neural Networks, May 2005.

[14] L. Wanner, "Introduction to Clustering Techniques," July 2004

[15] Kate A. Smith and Alan Ng, "Web page clustering using a self-organizing map of user navigation patterns", Monash University, P.O. Box 63B, Victoria 3800, Australia, Elsevier Science, 2003.

[16] jerome Moore and Eui-Hong, "Web Page Categorizing and feature selection using Association Rule and Principal Component Clustering", University of Minnesota, IEEE, 2000.

[17] A.K. Jain, M.N.Murty, and P.J.Flynn, "Data clustering: A review," ACM Computing Surveys, September 1999.

[18] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm Intelligence : From Natural to Artificial Systems, Sante Fe Institute Studies in the Sciences of Complexity, Oxford University, 1999.

[19] M. V. Shrivastava and M. N. Gupta, "Performance improvement of web usage mining by using learning based k-mean clustering," International Journal of Computer Science and its Applications.