Clustering Algorithms For Intelligent Web

Kanna Al Falahi

Faculty of Information Technology,
UAE University,
Al Ain, UAE
E-mail: harous@uaeu.ac.ae

Saad Harous

Faculty of Information Technology,
UAE University,
Al Ain, UAE
E-mail: harous@uaeu.ac.ae

Yacine Atif*

Faculty of Information Technology,
UAE University,
Al Ain, UAE
E-mail: Yacine.Atif@uaeu.ac.ae
*Corresponding author

Abstract Detecting communities of users and data in the Web are important issues as the social Web evolves and new information is generated every day. In this paper, we discuss six different clustering algorithms that are related to the intelligent Web. These algorithms help us to identify communities of interest in the Web, which is very necessary to perform certain actions on specific group such as targeted advertisement. We consider the following algorithms: Single-Link algorithm, Average-Link algorithm, Minimum-Spanning-Tree Single-Link algorithm, K-means algorithm, ROCK algorithm and DBSCAN algorithm. These algorithms are categorized into three groups: Hierarchical, Partitional and Density-based algorithms. We show how each algorithm works and discuss potential advantages and shortcomings. We then compare these algorithms against each other and discuss their ability to accurately identify communities of interest based on social Web data which are large datasets with high dimensionality. Finally, we illustrate and discuss our findings through a case study, which involves clustering in online social networks.

Keywords: Algorithms, clustering, intelligent web, social networks.

1 Introduction

The Web is a huge repository of information of all kinds and types. Through the years, the Web has evolved dramatically. From the static Web 1.0, where webmasters create and upload web-pages with limited interaction possibilities, to the more dynamic Web 2.0, where contents are collaboratively generated and communicated across blogs, feeds and social networks. The advent of Web 3.0 brought more intelligence to Web contents through the evolution of the Semantic Web and more automation of services over the Web to support machine-to-machine interactions (SemanticWeb, 2010).

The Semantic Web provides novel models for retrieving and analyzing Web information. Using this platform, intelligent Web applications are able to analyze users' inputs and behaviors to respond accordingly to different contextual considerations. These applications analyze interactions and profile users based on past history or pre-established records. Another possibility follows a case-based reasoning approach to match users with similar assets and aspirations to common Web experiences. The opportunity to analyze similarities within social contexts empowers Web experiences through identifying the commons to recommend preferential Web contents and services (Adomavicius & Tuzhilin, 2005).

Connectivity is a core feature of intelligent web applications, where users share files, publish articles, comment on others' blogs or forums, view users' profiles and add new members to their connections. These are typical operations within today's social networks such as Facebook[1], MySpace[2] and Twitter[3]. To make useful inferences over social connections, intelligent Web applications need the following three typical modules in classical knowledge-based systems (Marmanis & Babenko, 2009):

1. **Content**: represented by the hypermedia data of the considered domain and composed of inter-linked resources.
2. **Reference**: or the knowledge base that tags and annotates domain content through rules which categorize contents into meaningful folksonomies.
3. **Algorithms**: which form the inference engine modules which on the domain content.

People feed the Web with information everyday. This continuous flow of information may result in some inconsistencies as users will have a myriad of choices that need to be organized in an efficient manner. Data classification and clustering facilitate the process of analyzing and building meaningful inferences, for example grouping similar Web-pages could help finding serious problems such as mirrored Web-pages or detecting copyright violation (Haveliwala et al., 2000). In the intelligent Web, there are two algorithmic approaches to categorize data: Clustering and Classification (Marmanis & Babenko, 2009). These approaches are useful in performing targeted advertisements or enhancing user Web experience by allowing users to view posts that are of interest to them (Adomavicius & Tuzhilin, 2005) and increase individual personalized Web experiences by performing special recommendations for specific users or providing page categorizations for individual users such as Google News[4].

The objective of this paper is to provide users with means to identify other users and data groups in the Web. We focus on clustering algorithms in the social Web context based on the following approaches categorization: Hierarchical, Partitional and Density-based algorithms. Each of these approaches features intrinsic techniques such as threshold or centroid techniques. We also discuss and compare Six important algorithms used for Clustering purpose namely: Link-based (Single-Link, Average-Link and MST Single-Link), K-means, ROCK and DBSCAN algorithms.

The rest of the paper is organized as follows: Section 2 defines clustering. Section 3 discusses different types of clustering techniques. Section 4 introduces some important terms and concepts related to clustering. Section 5 reveals different clustering algorithms. Section six presents a comparison among candidate clustering algorithms. Section 7 describes a case study related to the use of clustering algorithms in social networks where we evaluate some of the candidate algorithms' clustering performance and finally Section 8 concludes the paper with a summary of works and some suggested future works.

2 Clustering in Social Web

Dividing people into different groups is one of human natures. Previously, people used clustering in order to study phenomena and compare them with other phenomena based on a certain set of rules. Clustering refers to grouping similar things together. It is a division of data into groups of similar objects where each group is called a cluster. It consists of objects that embody some similarities and are dissimilar to objects of other groups (Berkhin, 2002). We can find many definitions for clustering in the literatures (Jain et al., 1999; Xu & Wunsch, 2005; Gower, 1971; Jain & Dubes, 1988; Mocian, 2009; Tan et al., 2005) but the most common definition consists in partitioning data into groups (called clusters), based on some criteria so that the data grouped in one cluster should share common similarities calculated using some distance measurements. We can define clustering in the context of social network by cliques of individuals with high friendship relations internally and scattered friendship externally (Mishra et al., 2007). With clustering, we can find groups of interest that contain useful properties that are helpful to study these groups and understand their behaviors. Amazon[5] for example provide users with recommendations based on their shopping experience. Twitter also started lately to recommend new friends (people to follow) to their users based on several factors, including the people these users follow and the people they follow (Follow, 2012).

Clustering can be used for summarizing large inputs. So instead of applying algorithms on an entire dataset, we can reduce the dataset based on specific clustering criteria (Marmanis & Babenko, 2009). Clustering analysis has been used in many research fields such as image analysis, data mining, pattern recognition, information retrieval and machine learning (Tan et al., 2005). In the Web, identifying groups of data or users would facilitate the availability and accessibility of data. Using clusters in the Web is a must because of the tremendous potential which results from identifying groups of users out of a huge number of users on the internet, especially in social networks. Similarly, filtering out relevant information out of the tremendous amount of information that are linked together can lead to personalized exposure to useful information. However in both cases, the huge size makes it hard to analyze this information.

3 Clustering Types

There are many kinds of clustering algorithms available in the literatures (Jain et al., 1999; Xu & Wunsch, 2005; Mocian, 2009; Berkhin, 2002). They can be categorized based on the cluster structure (Hierarchical, Partitional), data types and structure (numerical, categorical) or data size (large datasets) (Marmanis & Babenko, 2009). In general, clustering approaches can be divided into four main types: hierarchical, partitional, density-based and meta-search controlled (Stein & Busch, 2005). In our paper, we discuss hierarchical, partitional and density-based clustering.

The Hierarchical and Partitional algorithms partition the data into different non-overlapping subsets. A partition of a dataset $X = \{x_1, x_2, \ldots, x_N\}$, where $x_j = (x_{j1}, x_{j2}, ..., x_{jd}) \in \Re^d$ with each measure x_{ji} called a feature (attribute, dimension or variable) and d is the input space dimensionality(Xu & Wunsch, 2005), is a collection $C = \{ C_1, C_2, \ldots, C_k\}$ of k non-overlapping data subsets. $Ci \neq \oslash$ (non-null clusters) such that $C_1 \cup C_2 \cup \ldots \cup C_k = X$, where X is the super cluster and $Ci \cap Cj = \oslash$ for $i \neq j$ (Hruschka et al., 2009). The data partition is overlapping if the condition ($Ci \cap Cj = \oslash$ for $i \neq j$) is ignored and in that case the cluster will have sub clusters of different levels inside it (Hruschka et al., 2009).

3.1 Hierarchical Clustering

In hierarchical clustering, the clusters are represented as a tree called *dendrogram* (Xu & Wunsch, 2008). They can be either top-down (divisive) or bottom-up (agglomerative). Most of these algorithms need a threshold parameter that tells the algorithm when to stop looking for subgroups. Figure 1 shows a graphical representation of divisive and agglomerative algorithms.

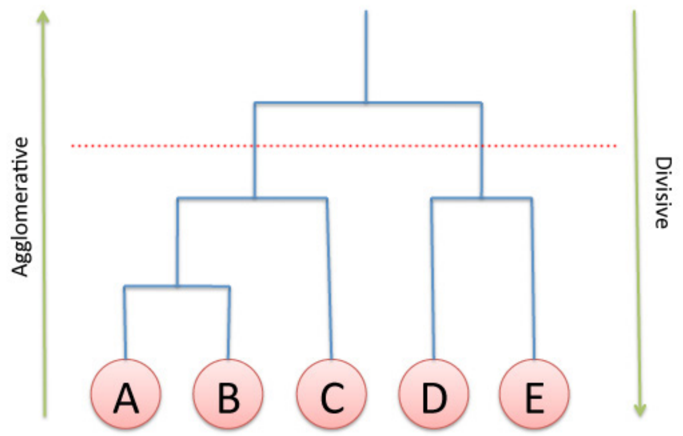

Figure 1 A dendrogram that represents Divisive vs Agglomerative clustering. Two clusters are generated when cutting the dendrogram at a specific level

In divisive hierarchical clustering, the algorithm starts from the global cluster that contains all the elements and then the data is divided into sub clusters. We need to find out which clusters to split and how to perform the splitting (Hammouda, 2002). While in agglomerative hierarchical clustering, the algorithm starts from a single cluster and then each two clusters are merged together until the global cluster is reached. DHSCAN is a hierarchical clustering algorithm represented in (Naughton et al., 2006) and used to group articls that refer to the same event and have similar sentences together.

The basic idea behind clustering is to find a distance/similarity measure between any two points such as Euclidean distance, cosine distance etc. In particular, this would be the shortest path in linkage algorithms that are based on linkage metric. To calculate the distance between two points, those algorithms include single-link, average-link and MST link techniques.

Hierarchical algorithms are represented using proximity matrix (distance matrix) assuming it is symmetric which means that it require the storage of $\frac{1}{2}$ n^2 proximities, where n is the number of elements (Tan et al., 2005). The total space complexity is $O(n^2)$ and the time required for computing the proximity matrix is $O(n^2)$ (Xu & Wunsch, 2008). In general, agglomerative hierarchical clustering do not have difficulties in selecting initial points as the algorithm will starts from single clusters. But they are expensive algorithms in terms of time and space which limit their usage with large scale datasets(Xu & Wunsch, 2005). We will focus on agglomerative hierarchical algorithms in this paper such as Single-Link, Average-Link and MST Single-Link algorithms.

3.2 Partitional Clustering

The Partitional algorithms have fixed number of clusters where the data is divided into a number of subsets (Mocian, 2009). The most common example is the K-means algorithm that starts by selecting random means for K clusters and assign each element to its nearest mean. K-means algorithms are O(tkn), where t is the number of iterations(Xu & Wunsch, 2008), k denotes the number of clusters and n the size of the data being clustered. These algorithms use a number of relocation schemes that provide optimization to the clusters, which means the clusters can be refined at each revisiting step and thus giving an advantage over hierarchical clustering (Mocian, 2009).

3.3 Density-Based Clustering

In density based algorithms, the cluster is a dense region of data objects. The points density is higher inside the cluster than outside the cluster. It is used the most when the shapes of the clusters are irregular and contain noise and outliers(Ester et al., 1996). DBSCAN is an example of density-based algorithms. In the worst case, the time complexity for this algorithm is $O(n^2)$, but in low dimensional spaces the time would be reduced to O(n logn) (Tan et al., 2005).

3.4 Meta-Search Controlled Clustering

The meta-search controlled clustering approach treats clustering as an optimization problem where a global goal criterion is to be minimized or maximized (Ester et al., 1996). Even though these algorithms provide flexibility, their runtime is unacceptably high. Cluster detection can be performed using genetic algorithms or two-phase greedy strategy (Ester et al., 1996).

In our paper, we will focus on Hierarchical, Partitional and Density-based algorithms. Next, we discuss these algorithms in details.

4 Concepts and terms

4.1 Distance and similarity measures

Any clustering algorithm has a similarity factor (proximity matrix) in order to organize similar objects together. It is important to understand the measures of similarity. What makes two clusters join? what makes two points similar? and how to calculate the distance (dissimilarity)?

Rui Xu and Donald Wunsch defined the function of distance or dissimilarity on a dataset X in their survey paper of clustering algorithms [11] by representing an $n * n$ symmetric proximity matrix for a dataset of n elements where the (i, j)th element represents the similarity or dissimilarity measure for the ith and the jth pattern (Xu & Wunsch, 2005).

The family of Minkowski distances is a very common class of distance functions (Hammouda, 2002) and can be represented as follows:

$$D(p_i, p_j) = \sqrt[w]{\sum (p_i - p_j)} \tag{1}$$

Where w is a parameter with a value greater than or equal to 1. Based on the value of w different distance functions can be represented such as Hamming distance (w=1), Euclidean distance (w=2) and Tschebyshev distance (w=∞). Other similarity measures are *cosine correlation* measure and *Jaccard* measure (Hammouda, 2002) further discussion can be found in (Xu & Wunsch, 2005).

4.2 Dendrogram data structure

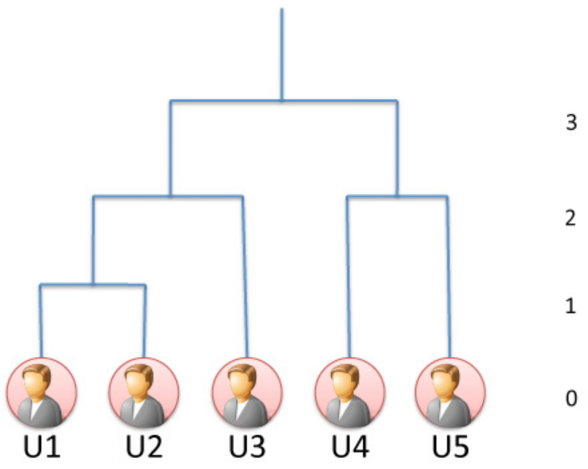

Figure 2 Dendrogram Structure

One of the basic structures in the clustering environment is the dendrogram, which is a tree data structure that is used to form the hierarchical cluster. Figure 2 shows a sample dendrogram with four levels. The dendrogram can be represented as a set of triples $S = \{[d, k, \{\dots\}]\}$ where d represents the threshold, k is the number of clusters and $\{\dots\}$ is the set of clusters. Figure 2 shows a dendrogram of detecting a cluster in a group of five users based on their distance similarities. The dendrogram could be represented by the following set $S = \{ [0, 5, \{\{U1\}, \{U2\}, \{U3\}, \{U4\}, \{U5\}\}], [1, 4, \{\{U1, U2\}, \{U3\}, \{U4\}, \{U5\}\}], [2, 2, \{\{U1, U2, U3\}, \{U4, U5\}\}], [3, 1, \{U1, U2, U3, U4, U5\}] \}$ (Marmanis & Babenko, 2009). The dendrogram represents a set of clusters. Most of the algorithms considered in this paper are hierarchical algorithms.

4.3 Proximity between clusters

Proximity calculation is the most important step in identifying clusters. It is used to measure how close are the data object to each other or how far they are, and differs according to the algorithm in use. For example, agglomerative hierarchical clustering techniques such as single-link, complete link and group average have different ways to determine the proximity threshold. The single-link defines the proximity as the closest distance between two elements in two different clusters or simply the shortest path between the two nodes in different clusters. Complete link calculates the largest distance between two points in two different clusters or the largest edge between the two nodes in different clusters. In the group average the proximity is defined to be the average length distance of all elements from the two different clusters (Tan et al., 2005). Figure 3 illustrates the three approaches.

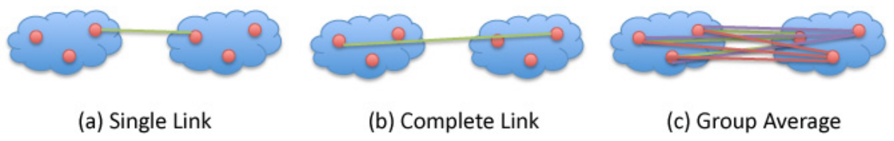

Figure 3 Cluster Proximity

5 An overview of clustering algorithms

In this section, we discuss and compare the following six clustering algorithms:

1. Link-based Algorithms
   (a) The Single-Link Algorithm
   (b) The Average-Link Algorithm
   (c) The Minimum-Spanning-Tree Single-Link Algorithm
2. The K-means Algorithm
3. The Robust Clustering Using Links algorithm (ROCK)
4. The Density-Based Spatial Clustering of Applications with Noise algorithm (DBSCAN)

5.1 Link-based algorithms

The link-based algorithms are agglomerative hierarchical algorithms where the dendrogram starts with individual objects and the proximity threshold is set to zero. Then the value of threshold is increased and based on that value the algorithm checks if two elements should be merged in one cluster or be kept disjoint. After a number of iterations all the elements will belong to a single super cluster.

The general algorithm for the hierarchical agglomerative algorithms can be described as shown in Algorithm 1.

Algorithm 1 General Hierarchical Agglomerative Algorithm

```
1. Set the proximity threshold and calculate the proximity matrix
2. Start with individual clusters
3. Based on the threshold merge the closest clusters
4. Update the threshold according to the new clusters
5. Repeat steps 3 and 4 until all the elements are in one super cluster
```

The Single-link, Average-link and Minimum-Spanning-Tree algorithms, which are link-based algorithms of type agglomerative hierarchical, will be discussed next.

5.1.1 The Single-Link

Single –Link Approach

The single link algorithm is based on the distance between clusters that are connected by at least one edge. First, It calculates the distance between the elements of the clusters. Then the proximity threshold is compared to the minimum distance to determine whether to merge the two clusters or not. The single–link distance between two clusters C$_i$ and C$_j$ could be represented by the following formula Hammouda (2002) :

$$D(C_i, C_j) = min_{x \in C_j, y \in C_i}(x - y) \tag{2}$$

The Single-Link Algorithm

The Single-link follows the general approach of the Linked-based algorithms described in Algorithm 1. The time and space complexity of the Single-Link algorithm is O(n^2) (Xu & Wunsch, 2005). This complexity will be a problem when working with very large data which is the case when clustering large real Web datasets such as social networks.

The single link is sensitive to noise and outliers actually suffers from the chain effect(Everitt et al., 2009). This effect occurs when the single link algorithm merges two clusters based on two points in these two clusters that are close to each other, regardless of the other points of the clusters that are far away. Single-link does not provide a solution for this problem(Marmanis & Babenko, 2009), but algorithms such as ROCK could provide a solution.

5.1.2 The Average-Link

Average –Link Approach

The average link algorithm is similar to the single-link algorithm but it uses different techniques to merge two clusters. It uses the average distance between any two points in the two different clusters and checks if it is less than the proximity threshold in order to merge the clusters.

The Average-Link Algorithm

As with single-link algorithm, we start with individual clusters and merge them until one cluster is formed, but unlike the single link the distance of all pairs of points between the two different clusters need to be calculated.

The average distance between two clusters C$_i$ and C$_j$ could be represented by the following formula:

$$D(C_i, C_j) = \frac{\sum_{x \in C_j, y \in C_i}(x - y)}{C_i.C_j} \tag{3}$$

The time and space complexity of the Average-Link algorithm is O(n^2) (Xu & Wunsch, 2005). Which is similar to the single-link algorithms so it has the same problem.

5.1.3 The Minimum-Spanning-Tree Single-Link

Minimum-Spanning-Tree Single-Link Approach

In this approach, a minimum spanning tree connects all the elements of a given set in a way that minimizes the sum of the adjacency values for the connected elements(Wu & Chao, 2004). The MST single link algorithm is a combination between the single link algorithms and the minimum spanning tree.

The Minimum-Spanning-Tree Single-Link Algorithm

The Prim-Jarnik algorithm (Wu & Chao, 2004) is used in this approach for the minimum spanning tree with single technique. This algorithm builds the minimum spanning tree starting from a single cluster (root) as expressed in Algorithm 2.

Algorithm 2 MST Single-Link Algorithm

```
1. Mark all elements of the graph as not visited
2. Choose any element you like as the root and mark it visited (cluster C created)
3. The smallest-weight edge e= (v, u) that connects one vertex v inside the
   clustering C is chosen and added to the spanning tree T.
4. Repeat until all vertices are visited and the minimum spanning tree is formed
```

The time and space complexity of the MST Single-Link algorithm is O(n^2)(Marmanis & Babenko, 2009). This is similar to the single-link and average-link algorithms. The MST single link algorithms results in fewer clusters than the single link algorithm because the proximity circle do not expand as much as it did in single link.

5.2 The K-means

5.2.1 K-means Approach

K-means is a Partitional algorithm. It uses the idea of centroid, which is the mean of a group of points. It has high performance characteristics and it is one of the oldest and most used clustering algorithms. Figure 4 illustrates the idea of centroid.

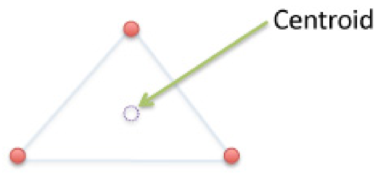

Figure 4 The Centroid Approach

5.2.2 The K-means Algorithm

The K-means algorithm starts by choosing the K initial centroids. The most simple approach is to choose random centroids. Then the points are assigned to their closest centroid to form K clusters (Kanungo et al., 2002). Depending on the points assigned to the cluster the centroid position is updated. We repeat the update until no more points to add or the centroids remain unchanged. The K-means algorithm can be represented as shown in Algorithm 3.

Algorithm 3 K-means Algorithm

```
1. Select K points as initial centroids
2. Form K clusters by assigning each point to its closest centroid
3. Re-compute the centroid of each cluster
4. Repeat steps 2 and 3 until centroids are not changed
```

The K-means is fast compared to other clustering algorithms. Its computational time is O(tkn) where t is the number of iterations, k represents the number of clusters and n is the number of data points we want to cluster. The space complexity for K-means is O(n) which is much better than link based algorithms.

Different runs of the K-means will produce different results since we randomly initialize the centroids actually this will produce poor clustering results. So choosing the right initial centroids is very important in order to create a good quality clusters (Kotsiantis & Pintelas, 2004). It is better to choose centroids in regions with high concentration of data points as proposed by David Arthur and Sergei Vassilvitskii in their K-mean++ article (Arthur & Vassilvitskii, 2007).

The K-means is efficient for large datasets (Kotsiantis & Pintelas, 2004) and works well with numerical data. But the challenge occurs when it is used with categorical data such as strings since we need to find a good way to represent the nonnumeric values in a numerical way.

5.3 The Robust Clustering Using Links (ROCK)

5.3.1 ROCK Approach

ROCK is an agglomerative hierarchical algorithm. It uses links as similarity measure rather than measures based on distance. It clusterS the points that have many common links. As an agglomerative hierarchical algorithm it starts from single clusters and merges these clusters until a super single cluster is formed. For the ROCK algorithm we need to define a minimum number of clusters that we want to form in order to stop the algorithm before all the elements are grouped in one single cluster.

5.3.2 Goodness measure

In the process of merging clusters in the ROCK algorithm, we need to determine the best pair of clusters to merge together. Thus the goodness measure is used. Actually for ROCK algorithm the best clusters are those that maximize the goodness measure. The goodness measure for two clusters C_i and Cj is represented as follows (Rajeev et al., 1999):

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}} \tag{4}$$

where *link[C_i, C_j]* represents the number of links between clusters C_i and C_j that is

$$\sum link(p_a, p_r) \tag{5}$$

Any pairs of clusters that will maximize the goodness measure will be the best pairs to merge. With algorithms that are based on similarity distance only, it will be difficult to determine if two clusters are separate because this kind of measurement may merge two clusters if there are two points close together even though these points do not have large number of common neighbors (Xu & Wunsch, 2008). Thus the ROCK algorithm uses links as its name implies. There is a link between two data points if a common neighbor exists between them. For the ROCK algorithm to merge two clusters the focus will be on the number of links n_i, n_j between all pairs points of the two clusters C_i, C_j. The large number of links should indicate a higher probability that the two points belong to the same cluster and should give the best cluster.

5.3.3 The ROCK Algorithm

The ROCK algorithm needs the following arguments:

1. *The set of points that we want to cluster*
2. *The minimum number of clusters to have to stop the ROCK algorithm before all points are merged in one cluster*
3. *Proximity that is required between two points in order to form a link between them*

The ROCK algorithm could be expressed as shown in Algorithm 4.

Algorithm 4 ROCK Algorithm

```
1. Create a cluster for each point
2. Use goodness measure to evaluate if two clusters should be merged or not
   (the best are the one those maximize the value of goodness measure)
3. Repeat step 2 until the number of clusters formed is equal to the minimum
   number to stop or the number of cluster doesn't change between iterations
```

The space complexity of the ROCK algorithm is O(n^2)(Marmanis & Babenko, 2009). While the time complexity is O(n^2 log n)(Rajeev et al., 1999).

ROCK algorithm is best used with categorical data such as keywords, Boolean attributes it uses the Jaccard coefficient to measure the similarity (Kotsiantis & Pintelas, 2004) and it works well on large data set. One of the advantages of using the ROCK algorithm is its ability to handle outliers effectively. Outliers are points that lies in a far distance from the other points. Which means these points can be easily discard, as they will never participate in the clustering process (Rajeev et al., 1999).

5.4 The Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

5.4.1 DBSCAN Approach

The DBSCAN algorithm is a density-based algorithm that uses density as a measurement other than links or distance between points.

Density-based algorithms are based on using density to identify the boundaries of objects. So clusters are identified based on points density within a specific region. Figure 5 explains this concept where we can identify three clusters in the figure. The points that don't belong to the clusters are identified as noise and DBSCAN is used to discover clusters and noise in a dataset.

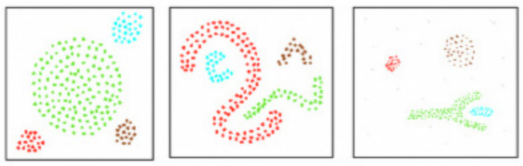

Figure 5 Density-based clustering

The DBSCAN can be described as follows (Figure 6): any two core points should be put in the same cluster if they are close to each other within a distance *Eps* (Xu & Wunsch, 2008). Where *Eps*, stands for epsilon, is a value that helps to define an epsilon neighborhood for any given data point *p* (Ester et al., 1996).

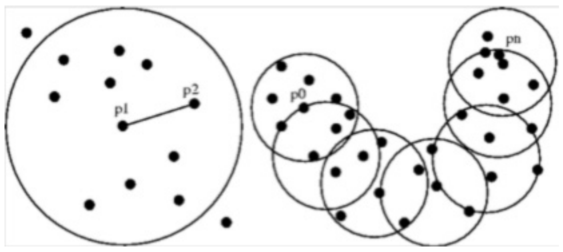

Figure 6 DBSCAN core points, border points and noise

To understand the concept of center points let us check Figure 6 that is similar to the figure in (Ester et al., 1996). The large circles are the epsilon neighborhood for points p and q each of them is a center of one of the circles. The circle radius is *Eps* and *minPoints* represents the minimum number of points that must be inside the circle for a data point to be considered a core point. The points that are on the border of the cluster are called border points. A point p_1 is directly density-reachable from a data point p_2 with respect to *Eps* and *minPoints* if there is a list of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i (Ester et al., 1996). The following two conditions should be met:

1. *p_1 inside the epsilon neighbor of p_2*
2. *There are more than minPoints data points inside the epsilon neighborhood of p_2*

5.4.2 The DBSCAN Algorithm

The DBSCAN algorithm is expressed in Algorithm 5.

Algorithm 5 DBSCAN Algorithm

1. Define the points as core, border, or noise points.
2. Eliminate noise points.
3. Put an edge between all core points that are within Eps of each other.
4. Make each group of connected core points into a separate cluster.
5. Assign each border point to one of the clusters of its associated core points.

The time complexity for the DBSCAN algorithm is $O(n^2)$(DBSCAN, 2012), where n is the number of points. DBSCAN can handle noise and different shape clusters because it is based on density. It can discover many clusters that are not found by the K-means algorithm. But this algorithm will have problems with clusters of very different densities as the algorithm requires that the object neighbors have enough high density(Xu & Wunsch, 2005) and with high-dimensional data (Tan et al., 2005). The DBSCAN uses R*-tree in order to improve the queries of determining the points within the Eps distance(Kotsiantis & Pintelas, 2004). R*-tree will reduce the time complexity of the DBSCAN to O(n logn)(Ester et al., 1996).

6 Analysis and Discussion

In this section, we discuss the complexity of clustering algorithms and other related issues. There are many criteria that decide the use of one algorithm over the others. The two main criteria are time complexity of these algorithms and do they handle data with high dimensionality.

6.1 Large datasets

To deal with a large number of elements we need to evaluate the computational complexity of the algorithm under consideration, in other words how long does this algorithm take to construct the cluster. There is a big difference between clustering groups of people on Facebook with millions of registered users and clustering a local newsgroup of some hundred users. To understand how crucial is the data size we need to understand how each algorithm deals with the memory size (space complexity) and the number of operations performed to cluster a set of data (time complexity). Table 1 shows both of these metrics for the algorithms discussed. Here k denotes the number of clusters, t the number of iterations, and n the size of the data being clustered. It is obvious that the problem is with the O(n^2) algorithms specially when n is large. In (Xu & Wunsch, 2005) Rui Xu and Li Wunsch compared the time and space complexities of these algorithms and provided an additional algorithms that can handle very large data sets such as CLARA, CLARANS and BIRCH.

Algorithm name	Space complexity	Time complexity
Single-Link	$O(n^2)$	$O(k\,n^2)$
Average-Link	$O(n^2)$	$O(k\,n^2)$
MST Single-Link	$O(n^2)$	$O(n^2)$
K-means	$O(n+k)$	$O(t\,k\,n)$
ROCK	$O(n^2)$	$O(n^2\,log(n))$
DBSCAN	$O(n^2)$	$O(n^2)$ or $O(n\,log(n))$ with R*-tree

Table 1 Space and Time complexities for clustering algorithms

It is obvious that hierarchical clustering algorithms are not suitable for large datasets because of their complexities. The K-means is the most efficient algorithm among them as the complexity is almost linear (Xu & Wunsch, 2005), but it cannot handle categorical data which is very important when clustering the web. DBSCAN can be improved by using spatial indices on the data points such as R*-tree that will reduce the time complexity for it from O(n^2) to O(n log(n)) and generates more efficient queries (Xu & Wunsch, 2005). It is important to mention that indexing spatial data faces difficulties in high dimensions and this subject is an active area of research(Marmanis & Babenko, 2009).

6.2 High dimensionality

The world that we deal with is of three-dimensionality and if we want to cluster worlds of higher dimensionalities we need to know that these worlds are governed by different rules and different proximities (Marmanis & Babenko, 2009). Actually higher dimensionality means larger computation which will slow the algorithm down.

High dimensionality produces a problem in data separation as the distance between the point and its nearest neighbor has no difference than the distance from that point to other points when the dimensionality is high enough(Xu & Wunsch, 2008). The "curse of dimensionality" is a problem that is related to high dimensionality. The term was introduced by Bellman to indicate the exponential growth of complexity in a high dimensionality situation (Xu & Wunsch, 2005), which indicates that the distance between any set of points in high dimensions are the same. In such situation there will be no effect for clustering algorithms that are based on distance measurements. Aggarwal provided a solution to this problem in (Aggarwal, 2002).

7 Related Works

Many algorithms for community detection have been proposed in the past. In (Newman & Girvan, 2004), Newman and Girvan proposed a method for discovering communities based on hierarchical divisive algorithms where an edge is removed iteratively from the network to split it into communities. Divisive algorithms were rarely used for community detection at that time as most of the studies were based on agglomerative algorithms. The main idea of their algorithm is to remove the edge with the highest betweenness. One fast method to measure edge betweenness is the shortest-path betweenness by measuring all the shortest paths passing through a giving link. Once an edge removed a recalculation of the edge betweenness is needed for all edges which leads to high computational costs. The process continues until the final communities consist of single nodes.

Fortunato et al (Fortunato et al., 2004) implemented a hierarchical clustering algorithm based on the work of Newman and Girvan (Newman & Girvan, 2004). Their work was based on using centrality measure to iteratively find and remove edges in the network. Their work shows that even the algorithm runs in $O(n^4)$, it is powerful when dealing with mixed and hard detectable communities.

Newman (Newman, 2004) also proposed another qualitative method for identifying communities called modularity. The *modularity* method uses a quantity function Q to measure if a specific division is meaningful when identify communities. The algorithm used was simple with reasonable running time than the other proposed algorithms at that time with worst time complexity of $O(n^2)$ on sparse network of nodes n. This method has become very popular and widely used. However, it has limits as claimed by Fortunato and Barthélemy (Fortunato & Barthelemy, 2007) who showed in their paper that modularity has scale that depends on the size of the network and any module that is smaller than the scale might not be determined. Zhenping Li et al Li et al. (2008) proposed a quantitative modularity density measure to get over the limits of the regular modularity method proposed by Newman and Girvan. They used both the nodes and the edges in their method and showed that the optimization of the quantitative modularity density measure will not affect the network division process which provides better detection for communities. But the method is still NP-hard.

Clauset, Newman and Moore (Clauset et al., 2004) proposed another algorithm called the CNM algorithm. The CNM algorithm is a bottom-up agglomerative clustering method that uses greedy techniques to combine and merge clusters in a network. The algorithm works similarly as in (Newman, 2004) and gives similar results for the communities found. But It is more efficient in identifying communities since its time performance in worst case drops to $O(m * d * logn)$.

8 Case Study and Evaluation

In this section, we use a case study to further explain clustering algorithms. We will explain the scripting language we have used and how we setup the environment to run the algorithms and obtain the results. Also we will discuss the results obtained using each algorithm.

The issue of identifying articles of similar topics is of great potential in the intelligent web environment, so in our case study we will use clustering algorithms to help in grouping similar articls. We collected the data from Delicious.com, which is a social bookmarking service that allows users to share, store and discover web bookmarks (Delicious.com, 2012). Since we are dealing with categorical data and keywords, represented by the articles titles, we will use ROCK and DBSCAN algorithms to define the different clusters and group the similar titles together.

8.1 Data Collection

For our two experiments, we have collected a list of 48 titles for different articles from Delicious.com and saved them in CSV file. For each title we assigned a unique ID and a username of the person who bookmarked that title. Two or more users can bookmark the same title. A sample of the dataset (11 out of 48 titles) is illustrated in Table 2.

ID	Name	Title
776	user01	Google sites to add social networking in "layers"
774	user01	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
740	user01	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
770	user01	Twitter is NOT a Social Network, Says Twitter Exec
722	user01	An open source collaborative network
744	user02	Google sites to add social networking in "layers"
710	user02	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
730	user03	Google sites to add social networking in "layers"
777	user03	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
756	user03	An open source collaborative network
733	user03	How To Discover Your Money Making Niche

Table 2 Sample dataset collected from Delicious.com

8.2 Setting Up the Environment

The two algorithms are implemented in Java language and to execute and debug them we used BeanShell, which is a free Java interpreter. The latest Java JDK and Apache Ant should be installed in order for the BeanShell interpreter to work correctly. All of the code commands were executed through the command line in Windows OS environment.

8.3 ROCK Algorithm

First we used the ROCK Algorithm to cluster the dataset. The algorithm uses the Jaccard coefficient to measure the similarities between different titles. It compares the common terms or keywords in the titles and based on that similarity titles are grouped together.

To start the experiment we have loaded Delicious titles using the 15 most common terms only and stored them in an array. The ROCK algorithm invoked to cluster the dataset with a minimum number of clusters equal to 5. This parameter will allow the ROCK algorithm to stop before grouping all the data in one cluster. The threshold of 0.2 is used to represent the needed proximity between two points to be linked. Algorithm 6 represents the code used to execute the algorithm and print the result.

Algorithm 6 ROCK Algorithm Execution Code

```
1. DeliciousDataset ds = DeliciousData.createDataset(15);
2. DataPoint[] dps = ds.getData();
3. ROCKAlgorithm rock = new ROCKAlgorithm(dps, 5, 0.2);
4. Dendrogram dnd = rock.cluster();
5. dnd.print(16);
```

The results of our experiment for the ROCK algorithms at level 16 shows 8 clusters, (Table 3). We noticed that the algorithm clustered similar titles such as title ID 799 and title ID 688 together. On the other hand, there are articles with similar titles but the algorithm did not merge them in one cluster such as title ID 520 that is in cluster 4 and title ID 681 that is in cluster 7. The algorithm also defined the non-obvious clusters such as the titles in cluster 4; which contains different titles that are grouped together because they contain similar terms related to "social network" topic. The ROCK algorithm will compare titles based on important keywords in these titles.

8.4 DBSCAN Algorithm

We applied the DBSCAN algorithm to the same dataset. The algorithm used also the Jaccard coefficient to measure the similarities between different titles. To start the experiment we have loaded Delicious titles using the most 15 common terms only and stored them in an array. A Cosine distance is used as a distance metric. The DBSCAN algorithm invoked to cluster the dataset with a *distance metric*, *Eps* (*neighbor threshold*), *minPoints* and *the term frequency.* Algorithm 7 represents the code used to execute the algorithm and print the result.

Clusters for: level-16, Goodness = 1.973889261532508		
Cluster No.	ID	Title
1	799	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
1	688	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
2	708	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	774	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	710	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
3	722	An open source collaborative network
3	715	An open source collaborative network
4	520	Twitter is NOT a Social Network, Says Twitter Exec
4	566	Twitter is NOT a Social Network, Says Twitter Exec
4	744	Google sites to add social networking in "layers"
4	730	Google sites to add social networking in "layers"
4	776	Google sites to add social networking in "layers"
4	770	Twitter is NOT a Social Network, Says Twitter Exec
5	740	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
5	720	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
6	777	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
6	795	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
7	681	Twitter is NOT a Social Network, Says Twitter Exec
7	500	Twitter is NOT a Social Network, Says Twitter Exec
7	790	Twitter is NOT a Social Network, Says Twitter Exec
7	780	Google sites to add social networking in "layers"
8	735	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
8	726	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials

Table 3 ROCK Algorithm Results

Algorithm 7 DBSCAN Algorithm Execution Code

```
1. DeliciousDataset ds = DeliciousData.createDataset(15);
2. DataPoint[] dps = ds.getData();
3. CosineDistance cosD = new CosineDistance();
4. DBSCANAlgorithm dbscan = new DBSCANAlgorithm(dps,cosD,0.7,2,true);
5. dbscan.cluster();
```

The results of our experiment for the DBSCAN algorithms are shown in Table 4. The results were more accurate than the ROCK algorithm results. All similar titles are clustered together such as clusters 2 and 3 as you can notice the titles are exactly similar to each other. Non-similar titles in cluster 1 (title ID 776, title ID 566) and cluster 4 (title ID 722, title ID 711) also defined by the algorithm, where in cluster 1 it grouped the titles based on the keyword "social networks" and in cluster 4 it grouped all the titles related to "open source" topic. The algorithm also was able to recognize the noise elements where these points do not belong to any cluster.

Table 4: DBSCAN Algorithm results

DBSCAN Clustering with NeighborThreshold = 0.7 minPoints = 2		
Cluster No.	ID	Title
1	776	Google sites to add social networking in "layers"
1	566	Twitter is NOT a Social Network, Says Twitter Exec
1	744	Google sites to add social networking in "layers"
1	780	Google sites to add social networking in "layers"
1	790	Twitter is NOT a Social Network, Says Twitter Exec
1	500	Twitter is NOT a Social Network, Says Twitter Exec
1	730	Google sites to add social networking in "layers"
1	770	Twitter is NOT a Social Network, Says Twitter Exec
1	681	Twitter is NOT a Social Network, Says Twitter Exec
1	520	Twitter is NOT a Social Network, Says Twitter Exec
2	774	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	708	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	777	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	726	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials

2	795	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	735	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
2	710	40 Best And Highly Useful Websites For Adobe Photoshop Tutorials
3	740	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	530	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	688	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	685	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	799	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
3	720	Nikon D7000: Camera Road Test With Chase Jarvis | Chase Jarvis Blog
4	722	An open source collaborative network
4	590	An open source collaborative network
4	711	XWiki - Open Source Wiki and Content-Oriented Application Platform
4	600	An open source collaborative network
4	715	An open source collaborative network
4	756	An open source collaborative network
5	690	Apple: Sorry, Steve Jobs Isn't a Ninja
5	736	Apple: Sorry, Steve Jobs Isn't a Ninja
6	499	How To Discover Your Money Making Niche
6	733	How To Discover Your Money Making Niche
7	743	How To Create WordPress Themes From Scratch Part 1
7	533	How To Create WordPress Themes From Scratch Part 3b
7	694	How To Create WordPress Themes From Scratch Part 2
7	510	How To Create WordPress Themes From Scratch Part 3a
Noise	540	iPhone SDK 3.0 Playing with Map Kit - ObjectGraph Blog
Noise	742	This Is The Second Time A Google Engineer Has Been Fired For Accessing User Data
Noise	577	Enhance your web forms with new HTML5 features
Noise	745	Resize or Move a Complete Flash Animation in One Go
Noise	746	10 Things You Didn't Know About the New #Twitter /via @gigaom #news #sm

Noise	732	How To Handle Customers During Virtual Assistant Problems
Noise	705	article on social media ad campaigns
Noise	587	The Business Plan
Noise	791	CSS Color Names
Noise	601	Typography : Web Style Guide 3
Noise	753	in 4 U.S. Adults Now Use Mobile Apps [STATS]

8.5 Results

From both experiments, the DBSCAN and the ROCK algorithms produce good clustering results for the dataset. We noticed that the DBSCAN advanced the ROCK algorithm as it able to find the correct clusters and outliers as shown in Table 4. The ROCK is based on measuring similarity between two clusters as it finds the common neighbors for the two clusters. But relaying on similarity might make the algorithm merge two clusters due to closeness even if they contain outliers or noise.

9 Conclusion

Clustering algorithms are an important approach to divide and analyze data. There are many different types of clustering each with its own technique. We can apply them on many phenomena in this world actually any dataset that consists of elements are qualified for applying clusters. We have discussed six clustering algorithms in this paper: single link, average link, MST single link, k-means, ROCK, and DBSCAN. The discussion was based on how accurate is to apply them on the social web.

The single-link, average-link, and MST single-link algorithms are agglomerative hierarchical algorithms. They do not perform efficiently (both with respect to time and space) on large data sets even though they are easily implemented algorithms.

The k-means algorithm is a partitional algorithm, which is more efficient than link-based algorithms. But it does not work with categorical data since it relies on the idea of centroid. Moreover it cannot handle outliers (the points that are far away from the main clusters) (Xu & Wunsch, 2005).

The ROCK algorithm is a hierarchical agglomerative algorithm that can handle categorical data since it relies on the links as measures more than the distance. But it has high time and space complexities.

The DBSCAN algorithm is a density-based algorithm that uses point density to identify clusters in a space. It can handle outliers even though its time and space complexity are high.

The algorithms discussed in this paper are used for the identification of groups of users and data on a website. We can combine algorithms such as the K-means with other algorithms. K-means is the preferred to use since it is simple and fast and can run on parallel computational platforms. Combining different algorithms together will maximize the benefits of these algorithms and guarantees the quality of resulted clusters (Kotsiantis & Pintelas, 2004). One example would be combining the efficient K-means algorithm with the powerful ROCK algorithm (if the data is Boolean or categorical) or DBSCAN algorithm (if the data is spatial). One scenario would be using K-means on the high-level clusters then process them with the ROCK or the DBSCAN algorithm.

References

Adomavicius, G. & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Data and Knowledge Engineering*, 17(6), 734–749.

Aggarwal, C. C. (2002). Towards meaningful high-dimensional nearest neighbor search by human-computer interaction. *Proceedings of the18th International Conference on Data Engineering.*

Arthur, D. & Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (pp. 1027–1035). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Berkhin, P. (2002). *Survey Of Clustering Data Mining Techniques.* Technical report, Accrue Software, San Jose, CA.

Clauset, A., Newman, M. E. J., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(5).

DBSCAN (2012). Wikipedia, retrieved on April 2, 2012.

Delicious.com (2012). Wikipedia, retrieved on April 2, 2012.

Ester, M., Kriegel, H.-p., Jörg, S., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).*

Everitt, B. S., Landau, S., & Leese, M. (2009). *Cluster Analysis.* Wiley, 4th edition.

Follow, D. W. T. (2012). Twitter, retrieved on april 2, 2012.

Fortunato, S. & Barthelemy, M. (2007). Resolution limit in community detection. *Proc. Natl. Acad. Sci. USA*, 104(1), 36–41.

Fortunato, S., Latora, V., & Marchiori, M. (2004). Method to find community structures based on information centrality. *Physical Review E*, 70(5).

Gower, J. C. (1971). A General Coefficient of Similarity and Some of Its Properties. *Biometrics*, 27(4), 857–871.

Hammouda, K. M. (2002). Web Mining: Identifying Document Structure for Web Document Clustering. *Master's Thesis, University of Waterloo, Waterloo, Ontario, Canada.*

Haveliwala, T. H., Gionis, A., & Indyk, P. (2000). Scalable Techniques for Clustering the Web. *ACM Computing Surveys*, 31((3)), 264–323.

Hruschka, E. R., Campello, R. J. G. B., Freitas, A. A., & Leon, P. (2009). A Survey of Evolutionary Algorithms for Clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 39(2), 133–155.

Jain, A. K. & Dubes, R. C. (1988). *Algorithms for clustering data.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data Clustering: A Review. *ACM Computing Surveys*, 31(3).

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C., Silverman, R., & Wu, A. Y. (2002). An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 881–892.

Kotsiantis, S. B. & Pintelas, P. E. (2004). Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science and Applications*, 1, 73–81.

Li, Z., Zhang, S., Wang, R.-S., Zhang, X.-S., & Chen, L. (2008). Quantitative function for community detection. *Phys Rev E*, 77(3).

Marmanis, H. & Babenko, D. (2009). Algorithms of the Intelligent Web. *Manning Publications Co.*

Mishra, N., Schreiber, R., Stanton, I., & Tarjan, R. (2007). Clustering Social Networks. In A. Bonato & F. R. K. Chung (Eds.), *Algorithms and Models for the Web-Graph*, volume 4863 of *Lecture Notes in Computer Science* chapter 5, (pp. 56–67). Berlin, Heidelberg: Springer Berlin Heidelberg.

Mocian, H. (2009). Survey of distributed clustering techniques. *MSc Internal Research Project, Computer Science, Imperial. College London.*

Naughton, M., Kushmerick, N., & Carthy, J. (2006). Clustering sentences for discovering events in news articles. *Advances in Information Retrieval*, 3936/2006, 535–538.

Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. *Phys Rev E*, 69(6).

Newman, M. E. J. & Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys Rev E*, 69(2).

Rajeev, S. G., Rastogi, R., & Shim, K. (1999). ROCK: A Robust Clustering Algorithm for Categorical Attributes. *15th International Conference on Data Engineering.*

SemanticWeb (2010). Wikipedia, retrieved on April 2, 2012.

Stein, B. & Busch, M. (2005). Density-based cluster algorithms in low dimensional and high-dimensional application. *Second International Workshop on Text-Based Information Retrieval.*

Tan, P.-N., Steinbach, M., & Kumar, V. (2005). Introduction to Data Mining. *Addison Wesley.*

Wu, B. Y. & Chao, K.-M. (2004). *Spanning Trees and Optimization Problems.* Chapman and Hall/CRC Press.

Xu, R. & Wunsch, D. (2008). Clustering. *Wiley-(IEEE Press Series on Computational Intelligence).*

Xu, R. & Wunsch, I. (2005). Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, 16(3), 645–678.