K-Linkage: A New Agglomerative Approach for Hierarchical Clustering

Pelin YILDIRIM[1], Derya BIRANT[2]
[1]*The Graduate School of Natural and Applied Sciences, Dokuz Eylul University, 35390, Turkey*
[2]*Department of Computer Engineering, Dokuz Eylul University, 35390, Turkey*
pelin@cs.deu.edu.tr

***Abstract*—In agglomerative hierarchical clustering, the traditional approaches of computing cluster distances are single, complete, average and centroid linkages. However, single-link and complete-link approaches cannot always reflect the true underlying relationship between clusters, because they only consider just a single pair between two clusters. This situation may promote the formation of spurious clusters. To overcome the problem, this paper proposes a novel approach, named *k*-Linkage, which calculates the distance by considering *k* observations from two clusters separately. This article also introduces two novel concepts: *k*-min linkage (the average of *k* closest pairs) and *k*-max linkage (the average of *k* farthest pairs). In the experimental studies, the improved hierarchical clustering algorithm based on *k*-Linkage was executed on five well-known benchmark datasets with varying *k* values to demonstrate its efficiency. The results show that the proposed *k*-Linkage method can often produce clusters with better accuracy, compared to the single, complete, average and centroid linkages.**

***Index Terms*—clustering, data mining, data processing, knowledge discovery, unsupervised learning.**

I. INTRODUCTION

Clustering, which is one of the data mining techniques, combines a set of objects into clusters based on a certain similarity measure. Clustering algorithms can be basically grouped under three categories: partitioning, hierarchical and density-based methods. *Partitioning clustering* is an iterative method which divides a dataset into disjoint clusters. *Hierarchical clustering* is characterized by the development of a hierarchy by either repeatedly merging small clusters into a larger one (agglomerative strategy) or splitting a larger cluster into smaller ones (divisive strategy). *Density-based clustering* is to discover clusters of arbitrary shape based on the density of the region surrounding the data point. This paper focuses on hierarchical clustering problems.

Hierarchical clustering has been commonly used in many applications by applying either divisive or agglomerative method. *Divisive hierarchical clustering* is a top down approach which starts with a single cluster and splits the cluster into two dissimilar clusters recursively until specified condition is satisfied. *Agglomerative hierarchical clustering* is a bottom up approach and starts with clusters containing single observations and continuously merges them based on a similarity strategy until all clusters are merged into one big cluster, or a stopping criteria is met. The traditional strategies of computing cluster distances are *single*, *complete*, *average*, and *centroid linkages*. However, these strategies can remain incapable of merging correct clusters, because small perturbations in the data can lead to large changes in hierarchical clustering assignments. There is no guarantee that single linkage or complete linkage will individually give the optimal clusters, because they consider only a single distance between two clusters. The calculation of distances between clusters based on a single pair may not always reflect the true underlying relationship between clusters and so it returns clusters that are only locally optimal. The main aim of this paper is to overcome this drawback by proposing a new approach.

This article proposes a novel linkage method for hierarchical clustering, named *k*-Linkage. The proposed *k*-Linkage method evaluates the distance between two clusters by calculating the average distance between *k* pairs of observations, one in each cluster. This paper also introduces two novel concepts: *k*-min linkage and *k*-max linkage. While *k*-min linkage considers *k* minimum (closest) pairs from points in the first cluster to points in the second cluster, *k*-max linkage takes into account *k* maximum (farthest) pairs of observations.

In the experimental studies, the proposed *k*-Linkage method was tested on five well-known benchmark datasets. The results show that the proposed approach can often produce more accurate clustering results, when compared with the traditional linkage methods in terms of accuracy rate. In addition, to determine the optimal number of pairs, we ran the algorithm several times using different *k* values, varying from 3 to 9 in increments of 2, and we selected the optimal one with the highest accuracy rate.

The remainder of the article is structured as follows: Section 2 summarizes the related literature and previous works on the subject. Section 3 gives background information on hierarchical clustering and explains the traditional linkage methods. This section also defines a novel method, named *k*-Linkage, and two novel concepts (*k*-min linkage and *k*-max linkage). Section 4 explains the improved version of agglomerative hierarchical clustering algorithm based on *k*-Linkage scheme. In Section 5, the experimental study is presented and the obtained experimental results are discussed. Finally, Section 6 gives some concluding remarks and future directions.

II. RELATED WORK

In data mining, hierarchical clustering is one of the most widely used cluster analysis method that groups a set of objects according to their similarities by building a cluster tree or dendrogram. Hierarchical clustering has been applied

Digital Object Identifier 10.4316/AECE.2017.04010

on a very broad range of fields, including textile [1-2], bioinformatics [3-4], production [5], text mining [6], data summarization, network security [7], pattern recognition [8] and health [9]. In this paper, our method is proposed for general purpose and so it can be applied in many fields.

Hierarchical clustering provides many motivational advantages for the application of clustering process on raw data efficiently. The first advantage is that hierarchical clustering is capable of identifying nested clusters, so it can show "cluster within clusters" and it allows the user to explore deeper insights from the data. An additional practical advantage in hierarchical clustering is the possibility of representing clusters via a two-dimensional diagram known as dendrogram. Dendrogram can also be very useful in understanding dataset and help identify outliers. It allows the user to traverse clusters in depth-first search or breadth-first search order. Another significant advantage of hierarchical clustering is that it does not require us to prespecify the number of clusters as the input. The number of clusters can be determined by observing levels in dendrograms and cutting at any level.

In the literature, the most commonly implemented linkage types for agglomerative hierarchical clustering are single linkage [10-11], complete linkage [12-13], average linkage [14] and centroid linkage [1]. While some studies [15] use and compare several linkage types and try to determine the best one, some studies [16] combine more than one method to produce better clustering results. For example, Lu and Liang [15] use both single linkage and the average linkage methods to reveal associations among features in the heart disease and mushroom data. Another study [16] combines single, complete and average linkage methods for the construction of phylogenetic tree.

Some studies have been proposed and applied different linkage criterion such as Ward's linkage [17], minimax linkage [18] and Genie linkage [19]. *Ward's linkage* uses an analysis of variance approach to evaluate the distances between clusters. Hirano et al. [17] applied Ward's method on the practical diagnosis dataset containing 140 instances. The obtained results presented in their paper shows that Ward's linkage produces the best clusters among the others (single and complete-linkage). *Minimax linkage* attempts to find clusters that have small radius. Bien and Tibshirani [18] presented and implemented minimax linkage method on the dataset containing 400 gray-scale, 64×64pixel images of 40 distinct people faces. Gagolewski et al. [19] also proposed a new hierarchical clustering linkage criterion, called *Genie*, in which the algorithm links two clusters according to the Gini index of the clusters. In contrast to existing works, our study proposes a different approach to calculate the distances between clusters.

The hierarchical clustering studies in the literature generally use Euclidean metric [20-22] to find distances between sets of observations. However, some studies preferred different metric measurements such as Baire distance [23], cosine similarity [24], and Manhattan distance [25-26]. Hirano et al. [17] used both Mahalanobis and Hamming distance measures for numerical and nominal attributes of the dataset respectively. Similarly, Bouguettaya et al. [27] developed an improved agglomerative hierarchical clustering algorithm by utilizing Euclidean and Canberra distance measures. Dynamic Time Warping (DTW) and Derivative Dynamic Time Warping (DDTW) are another common distance measures for clustering time series data. Luczak [28] implemented the combination of DTW and DDTW methods and performed on 84 datasets from a very broad range of fields, including medicine, finance, multimedia and engineering.

To reduce time complexity of clustering process and to get more accurate clustering results, modified hierarchical clustering algorithms [29] were realized such as potential-based hierarchical agglomerative (PHA) [30], fastcluster [31], and clustering using binary splitting (CLUBS) [32]. Davidson and Ravi [33] presented the use of instance and cluster level constraints in hierarchical clustering algorithms to improve performance.

There are several studies [34-35] that use ensemble clustering by combining partitions produced by different clustering algorithms into a single partition to increase clustering success. Zheng et al. [35] developed a framework for hierarchical ensemble clustering which compounds both partitional clustering and hierarchical clustering results and proposed a novel method for fitting ultra-metric distance from the aggregated distance matrices.

Differently from existing studies, our work focuses on the application of agglomerative hierarchical clustering algorithm using a novel linkage method, called *k*-Linkage. It is the first study that *k*-min and *k*-max linkage concepts are implemented to be able to produce more accurate clustering results than the traditional linkage methods.

III. AGGLOMERATIVE HIERARCHICAL CLUSTERING

Hierarchical clustering is one of the major cluster analysis techniques that construct hierarchical structure of clusters through a two-dimensional diagram known as dendrogram. The main steps in the agglomerative hierarchical clustering (AHC) are presented in Figure 1. Each observation in the dataset is assigned to one distinct cluster, then distances between each pair of the objects of the clusters are calculated and the closest pair of clusters according to the linkage criteria is merged into one cluster continuously.

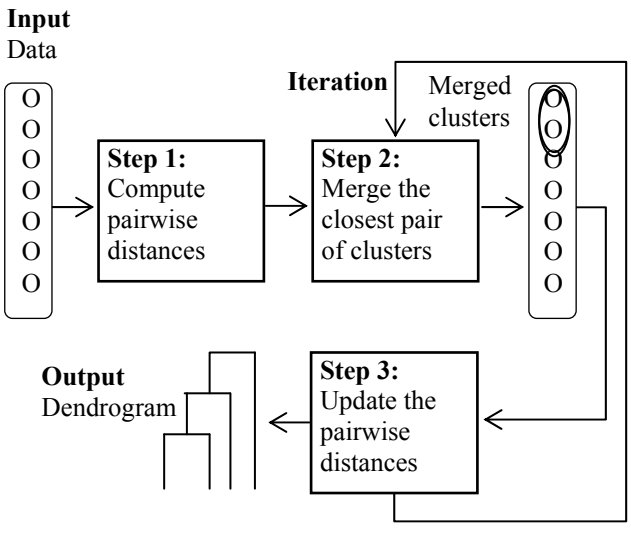


Figure 1. The step-by-step process of AHC algorithm

A. Classical Linkage Methods

While a hierarchical clustering algorithm is being computed on a given transactional dataset $T = \{t_1, t_2, ..., t_p\}$, there are p clusters such that $C = \{C_1, C_2, ..., C_p\}$, where $\cup_{i=1}^{p} C_i = T$ and $C_i \neq \emptyset$. A linkage method begins with p clusters and then the most similar clusters C_u and C_v are found and merged into one cluster. At the j-th step of the procedure, $j = 0, 1, ..., p-1$, the clustering procedure decides which of two clusters $C_u^{(j-1)}$ and $C_v^{(j-1)}$ are to be merged so that we get $C_w^{(j)} = C_u^{(j-1)} \cup C_v^{(j-1)}$, where $C_u^{(j-1)} \cap C_v^{(j-1)} = \emptyset$ for $u \neq v$.

Let $\{x_1, x_2, ..., x_m\}$ be a set of m observations from cluster C_u and $\{y_1, y_2, ..., y_n\}$ be a set of n observations from cluster C_v. The distance between clusters C_u and C_v is denoted by $d_{Cu,Cv}$ and it is formulated by $D(x,y)$ which is the distance between every possible observation x from C_u and observation y from C_v. To calculate $D(x,y)$, the Euclidian distance is usually used for numerical attributes, while Jaccard distance can be preferred for categorical variables.

Hierarchical clustering controls linkage strategies for iterative optimization, each of which repeatedly merges the most similar clusters. As an optimization problem, the main objective of our study is to minimize the differences within each cluster and maximize the differences between the clusters. It is possible to solve an optimization problem by using different techniques such as ant colony [36], fuzzy models [37], particle swarm optimization [38, 39], and simulated annealing [38, 39]. The agglomerative hierarchical clustering is an example of greedy algorithms in that it does the locally optimal thing at each step, but this doesn't guarantee producing a globally optimal solution. The optimization problem in this study can be defined by objective functions, the variables and the constraints as follows.

Objective Functions: minimize d_{c_u,c_v}

Variables: d_{c_u,c_v} is the distance between clusters C_u and C_v

$u,v = 1...p$ for p clusters

$C_u = \{x_1, x_2, ..., x_m\}$ for m items

$C_v = \{y_1, y_2, ..., y_n\}$ for n items

$D(x,y)$ is the distance between items $x \epsilon C_u$ and $y \epsilon C_v$

Constraints: $C_i \neq \emptyset$, $C_u \cap C_v = \emptyset$ for $u \neq v$

$D(x, y) \geq 0$, $D(x, x) = 0$ and $D(x, y) = D(y, x)$

The objective function of the optimization problem varies according to linkage method. There are mainly four linkage methods to evaluate the distances between clusters: single, complete, average and centroid. At each stage of the clustering process, two clusters that have the smallest linkage distance according to the selected linkage method are merged.

Single Linkage: (Figure 2a) Single linkage, also called nearest-neighbor technique, selects the distance between closest observations in clusters as shown in Equation (1).

Objective function for single linkage:

$$d_{C_u,C_v} = \arg\min_{(u,v)} \left(\min_{x \in C_u, y \in C_v} D(x,y) \right) \quad (1)$$

Complete Linkage: (Figure 2b) Complete linkage, also called furthest-neighbor technique, selects distance between farthest observations in clusters as shown in Equation (2).

Objective function for complete linkage:

$$d_{C_u,C_v} = \arg\min_{(u,v)} \left(\max_{x \in C_u, y \in C_v} D(x,y) \right) \quad (2)$$

The main objective of this method is to minimize the maximum inter-cluster distance, as an optimization problem.

Average Linkage: (Figure 2c) Average linkage calculates distances between all pairs of observations in clusters and averages all of these distances as shown in Equation (3).

Objective function for average linkage:

$$d_{C_u,C_v} = \arg\min_{(u,v)} \left(\frac{1}{|C_u|} \frac{1}{|C_v|} \sum_{x \in C_u} \sum_{y \in C_v} D(x,y) \right) \quad (3)$$

where $|C_u|$ and $|C_v|$ are the number of objects in the clusters C_u and C_v respectively.

Centroid Linkage: (Figure 2d) Centroid linkage method finds the distance between two mean vectors of the clusters. As an optimization problem, the goal of centroid linkage method is to minimize the objective function given in Equation (4).

Objective function for centroid linkage:

$$\begin{aligned} d_{C_u,C_v} &= \arg min\left(D(\bar{x}, \bar{y}) \right) \\ &= \arg\min_{(u,v)} \left(D\left(\left(\frac{1}{|C_u|} \sum_{x \in C_u} x \right), \left(\frac{1}{|C_v|} \sum_{y \in C_v} y \right) \right) \right) \end{aligned} \quad (4)$$

where $\bar{x}$ and $\bar{y}$ are the centroids (mean) of the clusters Cu and Cv respectively.

Theoretical properties of a distance measure D(x,y) between two objects x and y are as follows:

- $D(x, y) \geq 0$. The distance between two objects must be strictly greater than 0.
- $D(x, x) = 0$. The distance between an object and itself must be 0.
- $D(x, y) = D(y, x)$. The distance between object x and y must be the same as the distance between y and x.

Given a dataset that consists of five instances, assume that there are two clusters $C_u = \{x_1, x_2\}$ and $C_v = \{y_1, y_2, y_3\}$. The calculation of distances between clusters in terms of four linkage methods is as follows:

- ***Single Linkage***

$D(x, y) = \min\{D(x_1, y_1), D(x_1, y_2), D(x_1, y_3), D(x_2, y_1), D(x_2, y_2), D(x_2, y_3)\}$

- ***Complete Linkage***

$D(x, y) = \max\{D(x_1, y_1), D(x_1, y_2), D(x_1, y_3), D(x_2, y_1), D(x_2, y_2), D(x_2, y_3)\}$

- ***Average Linkage***

$$D(x,y)=\frac{D(x_1,y_1)+D(x_1,y_2)+D(x_1,y_3)+D(x_2,y_1)+D(x_2,y_2)+D(x_2,y_3)}{6}$$

- ***Centroid Linkage***

$$D(x,y)=D\left(\left(\frac{x_1+x_2}{2}\right),\left(\frac{y_1+y_2+y_3}{3}\right)\right)$$

To unify all of these methods, the Lance - Williams procedure provides a generalization in which all methods are special cases, as given in Equation (5).

Objective Function:

Minimizing

$$d_{C(AB)}=\alpha_A d_{CA}+\alpha_B d_{CB}+\beta d_{AB}+\gamma\,|\,d_{AC}-d_{BC}\,| \quad (5)$$

Variables:

α_A, α_B, β, γ are parameters

A, B, C are clusters.

d_{ij} is the distance of cluster (or object) pairs.

$d_{C(AB)}$ is the distance between cluster C and the new cluster AB.

n_i refers to the number of items in cluster i, $i \in \{A, B, C\}$.

Constraints:

$\alpha_A + \alpha_B + \beta = 1$

$\alpha_A = \alpha_B$

$\beta < 1$

Single Linkage:

α_A=1/2, α_B=1/2, β=0, γ=*-1/2*

Complete Linkage:

α_A=1/2, α_B=1/2, β=0, γ=*1/2*

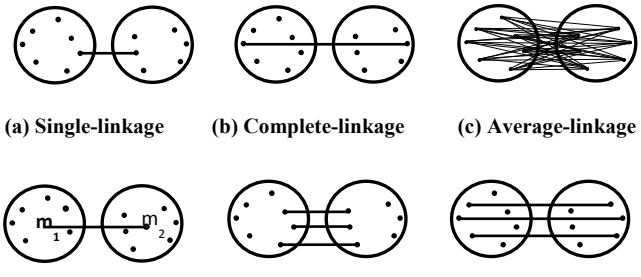
Average Linkage:

$\alpha_A=n_A/(n_A+n_B)$, $\alpha_B=n_B/(n_A+n_B)$, β=*0*, γ=*0*

Centroid Linkage:

$\alpha_A=n_A/(n_A+n_B)$, $\alpha_B=n_B/(n_A+n_B)$, $\beta=-n_An_B/(n_A+n_B)^2$, γ=*0*

In order to compensate the drawbacks of the current linkage schemes (given in the III.B. section), we propose a new linkage criterion: *k*-Linkage. Instead of considering only one pair like single and complete linkage methods, the *k*-min linkage (Figure 2e) and *k*-max linkage (Figure 2f) methods take into account more than one pairs, i.e. *k* closest or *k* furthest pairs respectively.



(a) Single-linkage **(b) Complete-linkage** **(c) Average-linkage**

(d) Centroid-linkage **(e) *k*-min linkage (k=3)** **(f) *k*-max linkage (k=3)**

Figure 2. Classical and proposed linkage methods

B. Drawbacks of Classical Linkage Methods

The classical linkage methods have the following drawbacks [19, 36]:

- The *single linkage* method suffers from a chaining effect and produces long chains and it has a tendency to produce clusters that are straggly or elongated. Figure 3a demonstrates chaining problem in single-link clustering. The single-link method only compares *d1* and *d2* distances, and the distance between left-right clusters (*d1*) is smaller than the distance between up-down clusters (*d2*). Since the merge criterion considers one pair and, a chain of points can be extended for long distances without regard to the overall shape of the emerging cluster [40]. In addition, the single linkage method tends to construct clusters of unbalanced sizes and produces highly skewed dendrograms. Furthermore, it has limitations on the detection of clusters that are not well separated. On the other hand, it might be useful to detect outliers in the dataset, because outliers often appear as clusters with only one member.

- The *complete-linkage* method in general tends to produce tightly bound clusters. Clusters tend to be compact and roughly equal in diameter. In addition, complete linkage method is sensitive to outliers which are points that do not fit well into the global structure of the cluster. Figure 3b demonstrates outlier problem in complete-link clustering. The outlier at the left edge splits the optimal cluster because the smallest furthest distance is *d2* among alternative distances. It tends to break large clusters, often resulting in a single large cluster and a number of singletons or ones with a very low cardinality.

- The *average linkage* method is somewhere between single linkage and complete linkage. However, it takes long time to calculate the distances between all pairs and average all of these distances. The time needed to apply a hierarchical clustering algorithm is most often dominated by the number of computations of a pairwise distance measure. Time constraint is an important issue for large datasets.

- In the *centroid linkage* method, the centre will move as clusters are merged. As a result, the distance between merged clusters may actually decrease between steps, making the analysis of results problematic. In other words, clustering with centroid linkage is not monotonic and can contain an inversion, which means that similarity can increase during clustering, instead of monotonically decreasing from iteration to iteration. In the case of an inversion in a dendrogram, a horizontal merge line shows up lower than the previous merge line. Increasing similarity in clustering steps contradicts the fundamental assumption that small clusters are more coherent than large clusters [40]. Therefore, the algorithm causes problems that some instances may need to be switched from their original clusters.

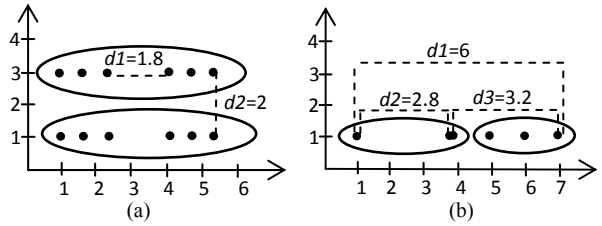


Figure 3. (a) Chaining problem in single-link clustering (b) Outlier problem in complete-link clustering

Solution quality in hierarchical clustering may vary depending on how clusters are fused. There is no guarantee that single or complete linkage will collectively or individually give the optimal clusters. They sometimes do not reflect the true underlying data structure. Because of the greedy nature of the single and complete linkages in hierarchical clustering, the algorithm returns clusters that are only locally optimal. The presence of local minima leads to incorrect clustering results. To overcome the limitations of the current linkage methods, this paper proposes a new linkage method, named *k*-Linkage.

C. K-Linkage Method

K-Linkage is a novel linkage method which aims to find similarity of clusters by considering *k* observations from a cluster C_u and *k* observations from another cluster C_v. In the subject of *k*-Linkage method, this study proposes two novel concepts, named *k-min linkage* and *k-max linkage*, to evaluate distances between clusters.

1) K-min Linkage Method

The *k*-min linkage method calculates the sum of distances between *k* closest observations in clusters and finds the average of them as a similarity measure. Definition 1 defines *k*-min linkage concept for the first time.

Definition 1. Let $\{x_1, x_2, ..., x_m\}$ be a set of *m* observations from cluster C_u and $\{y_1, y_2, ..., y_n\}$ be a set of *n* observations from cluster C_v. The distance between clusters C_u and C_v is denoted by $d_{Cu,Cv}$ and it is formulated by taking the average of *k* closest observation pairs *(x,y),* where $x \in C_u$ and $y \in C_v$, as shown in Equation (6).

Objective function for k-min linkage:

$$d_{C_u,C_v} = \frac{1}{|k|}\sum_{i=1}^{k} \underset{(i)}{\arg\min}_{(u,v)} \left(\min_{x \in C_u, y \in C_v} D(x,y) \right) \quad (6)$$

On the basis of *k*-min linkage method, two clusters which have the most *k* similar members on average are merged at each stage of the process. For the case where more than one pairs of observations have the same similarity, some of them can easily be selected, because overall average distance doesn't change in the case of ties.

2) K-max Linkage Method

The *k*-max linkage method calculates the sum of distances between *k* farthest observations in clusters and finds the average of them as a similarity measure. Definition 2 defines *k*-max linkage concept for the first time.

Definition 2. Let $\{x_1, x_2, ..., x_m\}$ be a set of *m* observations from cluster C_u and $\{y_1, y_2, ..., y_n\}$ be a set of *n* observations from cluster C_v. The distance between clusters C_u and C_v is denoted by $d_{Cu,Cv}$ and it is formulated by taking the average of *k* farthest observation pairs *(x,y),* where $x \in C_u$ and $y \in C_v$, as shown in Equation (7).

Objective function for k-max linkage:

$$d_{C_u,C_v} = \frac{1}{|k|}\sum_{i=1}^{k} \underset{(i)}{\arg\min}_{(u,v)} \left(\max_{x \in C_u, y \in C_v} D(x,y) \right) \quad (7)$$

On the basis of *k*-max linkage method, two clusters which have the most *k* dissimilar members on average are merged at each stage of the process.

The agglomerative hierarchical clustering algorithm has been improved in this research. The steps of the application of the proposed methods are given below.

Step 1. Assign each object in the dataset to a separate cluster so that for *n* objects we have *n* clusters each containing just one object.

Step 2. Calculate the distances between the clusters.

Step 3. Find the closest pair of clusters based on a similarity strategy and merge them into a single cluster.

- ***K*-min Linkage:** Select the average distance of *k*-closest objects between clusters.
- ***K*-max Linkage:** Select the average distance of *k*-farthest objects between clusters.

Step 4. Compute the distances between new cluster and each of other clusters.

Step 5. Repeat steps 3 and 4 until all objects are clustered into a single cluster.

D. Advantages of K-Linkage Method over Traditional Hierarchical Methods

Proposed k-Linkage method has several advantages over traditional linkage methods. First, considering *k* observations instead of single observation prevents the greedy nature of the single and complete linkages. It also achieves greater speed-up than average and centroid linkage, so as to reduce the number of computations of a pairwise distance measure, because a spatial index can be used for quick neighborhood lookup.

Another advantage of the proposed *k*-Linkage method is that it can be used to detect clusters with arbitrary shapes, because it prevents both chaining and rounding effects by considering several pairs. While the single linkage method produces elongated clusters and the complete linkage method tends to construct spherical clusters, *k*-Linkage method is more robust to local optimal decisions.

The main advantage behind the *k*-Linkage clustering lies in the fact that its solution is similar to the well-known technique K-nearest neighbor (KNN). How the KNN algorithm takes into account several objects when classifying a new instance, similarly, *k*-Linkage method also considers several pairs to make sure about the relationship among clusters.

E. Advantages of K-Linkage Method over Non-Hierarchical Methods

There are many advantages of *k*-Linkage method in comparison with non-hierarchical clustering algorithms such as k-Means, DBSCAN. First, *k*-Linkage method presents hierarchical structure of clusters, so it gives more information about the clusters than the unstructured set of clusters returned by non-hierarchical clustering. Thus, the output of the k-linkage method is easy to interpret and very

useful in understanding the dataset.

Another significant advantage of *k*-Linkage method is that it is deterministic (non-random) which means that it does not include any random parameter or random initialization technique. Thus, it produces the same results when run several times on the same data. However, some non-hierarchical clustering algorithms (i.e. k-Means) depend on random initialization so that clustering results may vary across runs.

Another advantage of our *k*-Linkage method is that it is appropriate for clustering high-dimensional data. Besides these advantages, *K*-linkage method also supports different forms of similarity and distance, thus it can be used with many attribute types. It does not even require a distance, any measure can be used, including similarity functions, such as Euclidean distance for numerical data, Jaccard distance for categorical data, Levenshtein distance for strings, and Gower distance for time series and mixed type data, even semantic similarity measures. However, some non-hierarchical clustering algorithms are limited to Euclidean distance.

F. An Example for K-Linkage Method

In this section, the application of the proposed linkage metrics (*k*-min and *k*-max linkage) in agglomerative hierarchical clustering is illustrated by two datasets. Table I shows sample datasets that contain *X* and *Y* coordinates of the instances and consist of 18 records that are uniquely identified by an ID. Only *Y* value of the element "*N*" is different between two datasets. In this study, Euclidean distance was used as distance measurement between instances.

TABLE I. SAMPLE DATASETS

Dataset 1		
ID	**X**	**Y**
A	2	7
B	2	6
C	3	6
D	3	8
E	4	6
F	5	8
G	7	9
H	12	16
I	13	16
J	14	16
K	14	17
L	15	16
M	16	16
N	15	**18**
O	12	9
P	13	9
Q	15	9
R	16	8

Dataset 2		
ID	**X**	**Y**
A	2	7
B	2	6
C	3	6
D	3	8
E	4	6
F	5	8
G	7	9
H	12	16
I	13	16
J	14	16
K	14	17
L	15	16
M	16	16
N	15	**24**
O	12	9
P	13	9
Q	15	9
R	16	8

Table II shows the clustering steps of AHC algorithm with single and *k*-min linkage methods on dataset 1, while Table III shows the steps for complete and *k*-max linkage methods on dataset 2, where the user defined *k* parameter was selected as 3. In the first step, each observation in the dataset is assumed as one distinct cluster. After that, the most similar pair of clusters according to the selected linkage criteria are merged into one cluster in each step. For example; the closest clusters in the first step are cluster {A} and cluster {B}, so clusters {A} and {B} are merged as {A, B} in step 2. This process is continued until all clusters are merged into one cluster. The first sixteen steps, each producing a new cluster by merging two existing clusters, are identical. At the step 17 in Table II, single-link clustering joins clusters {A,B,C,E,D,F,G} and {O,P,Q,R} because on the maximum similarity definition of cluster similarity, those two clusters are closest. On the other hand, *k*-min linkage method joins clusters {H,I,J,K,L,M,N} and {O,P,Q,R} because those are the closest clusters according to top three similar pairs among them. Similarly, complete and *k*-max linkage methods make a difference at the step 17.

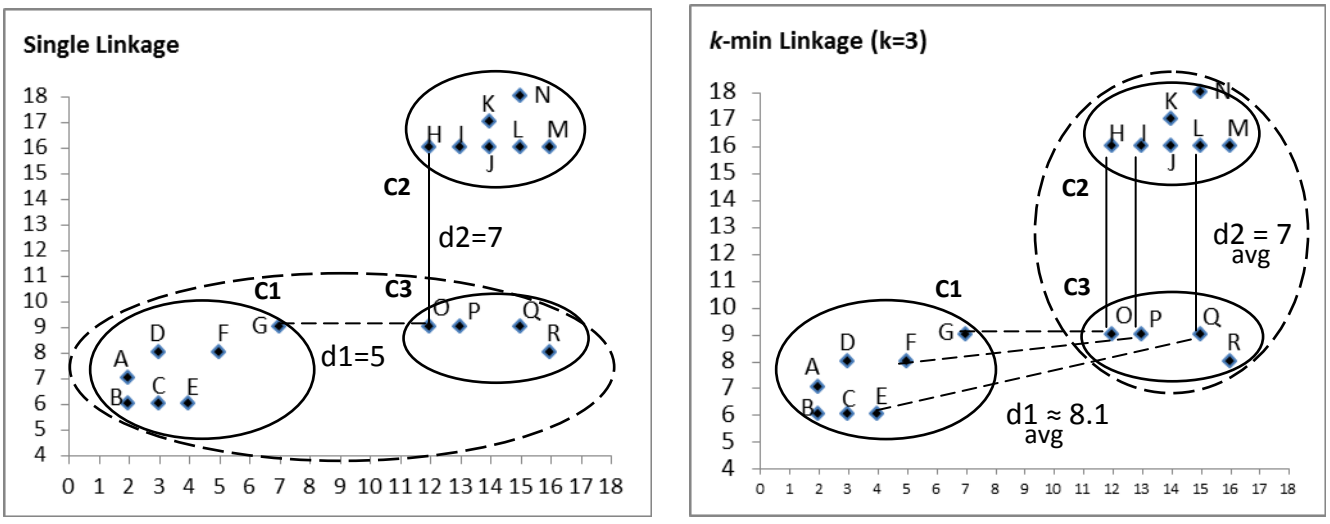
TABLE II. CLUSTERING STEPS BASED ON SINGLE AND K-MIN LINKAGE METHODS FOR DATASET 1

Step	Clusters using Single Linkage	Clusters using *k*-min Linkage (k=3)
1	{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
2	**{A,B}**,{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	**{A,B}**,{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
3	**{A,B,C}**,{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	**{A,B,C}**,{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
4	**{A,B,C,E}**,{D},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	**{A,B,C,E},**{D},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
5	{A,B,C,E},{D},{F},{G},**{H,I}**,{J},{K},{L},{M},{N},{O},{P},{Q},{R}	{A,B,C,E},{D},{F},{G},**{H,I}**,{J},{K},{L},{M},{N},{O},{P},{Q},{R}
6	{A,B,C,E},{D},{F},{G},**{H,I,J}**,{K},{L},{M},{N},{O},{P},{Q},{R}	{A,B,C,E},{D},{F},{G},**{H,I,J}**,{K},{L},{M},{N},{O},{P},{Q},{R}
7	{A,B,C,E},{D},{F},{G},**{H,I,J,K}**,{L},{M},{N},{O},{P},{Q},{R}	{A,B,C,E},{D},{F},{G},**{H,I,J,K}**,{L},{M},{N},{O},{P},{Q},{R}
8	{A,B,C,E},{D},{F},{G},**{H,I,J,K,L}**,{M},{N},{O},{P},{Q},{R}	{A,B,C,E},{D},{F},{G},**{H,I,J,K,L}**,{M},{N},{O},{P},{Q},{R}
9	{A,B,C,E},{D},{F},{G},**{H,I,J,K,L,M}**,{N},{O},{P},{Q},{R}	{A,B,C,E},{D},{F},{G},**{H,I,J,K,L,M}**,{N},{O},{P},{Q},{R}
10	{A,B,C,E},{D},{F},{G},{H,I,J,K,L,M},{N},**{O,P}**,{Q},{R}	{A,B,C,E},{D},{F},{G},{H,I,J,K,L,M},{N},**{O,P}**,{Q},{R}
11	**{A,B,C,E,D}**,{F},{G},{H,I,J,K,L,M},{N},{O,P},{Q},{R}	**{A,B,C,E,D}**,{F},{G},{H,I,J,K,L,M},{N},{O,P},{Q},{R}
12	{A,B,C,E,D},{F},{G},**{H,I,J,K,L,M,N}**,{O,P},{Q},{R}	{A,B,C,E,D},{F},{G},**{H,I,J,K,L,M,N}**,{O,P},{Q},{R}
13	{A,B,C,E,D},{F},{G},{H,I,J,K,L,M,N},{O,P},**{Q,R}**	{A,B,C,E,D},{F},{G},{H,I,J,K,L,M,N},{O,P},**{Q,R}**
14	**{A,B,C,E,D,F}**,{G},{H,I,J,K,L,M,N},{O,P},{Q,R}	**{A,B,C,E,D,F}**,{G},{H,I,J,K,L,M,N},{O,P},{Q,R}
15	{A,B,C,E,D,F},{G},{H,I,J,K,L,M,N},**{O,P,Q,R}**	{A,B,C,E,D,F},{G},{H,I,J,K,L,M,N},**{O,P,Q,R}**
16	**{A,B,C,E,D,F,G}**,{H,I,J,K,L,M,N},{O,P,Q,R}	**{A,B,C,E,D,F,G}**,{H,I,J,K,L,M,N},{O,P,Q,R}
17	**{A,B,C,E,D,F,G,O,P,Q,R}**,{H,I,J,K,L,M,N}	{A,B,C,E,D,F,G},**{H,I,J,K,L,M,N,O,P,Q,R}**
18	{A,B,C,E,D,F,G,O,P,Q,R,H,I,J,K,L,M,N}	{A,B,C,E,D,F,G,O,P,Q,R,H,I,J,K,L,M,N}

TABLE III. CLUSTERING STEPS BASED ON COMPLETE AND K-MAX LINKAGE METHODS FOR DATASET 2

Step	Clusters using Complete Linkage	Clusters using *k*-max Linkage (k=3)
1	{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
2	**{A,B}**,{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	**{A,B}**,{C},{D},{E},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
3	{A,B},**{C,E}**,{D},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}	{A,B},**{C,E}**,{D},{F},{G},{H},{I},{J},{K},{L},{M},{N},{O},{P},{Q},{R}
4	{A,B},{C,E},{D},{F},{G},**{H,I}**,{J},{K},{L},{M},{N},{O},{P},{Q},{R}	{A,B},{C,E},{D},{F},{G},**{H,I}**,{J},{K},{L},{M},{N},{O},{P},{Q},{R}
5	{A,B},{C,E},{D},{F},{G},{H,I},**{J,K}**,{L},{M},{N},{O},{P},{Q},{R}	{A,B},{C,E},{D},{F},{G},{H,I},**{J,K}**,{L},{M},{N},{O},{P},{Q},{R}
6	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},**{L,M}**,{N},{O},{P},{Q},{R}	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},**{L,M}**,{N},{O},{P},{Q},{R}
7	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},{L,M},{N},**{O,P}**,{Q},{R}	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},{L,M},{N},**{O,P}**,{Q},{R}
8	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},{L,M},{N},{O,P},**{Q,R}**	{A,B},{C,E},{D},{F},{G},{H,I},{J,K},{L,M},{N},{O,P},**{Q,R}**
9	{A,B},{C,E},**{D,F}**,{G},{H,I},{J,K},{L,M},{N},{O,P},{Q,R}	{A,B},{C,E},**{D,F}**,{G},{H,I},{J,K},{L,M},{N},{O,P},{Q},{R}
10	**{A,B,C,E}**,{D,F},{G},{H,I},{J,K},{L,M},{N},{O,P},{Q,R}	**{A,B,C,E}**,{D,F},{G},{H,I},{J,K},{L,M},{N},{O,P},{Q},{R}
11	{A,B,C,E},{D,F},{G},**{H,I,J,K}**,{L,M},{N},{O,P},{Q,R}	{A,B,C,E},{D,F},{G},**{H,I,J,K}**,{L,M},{N},{O,P},{Q},{R}
12	**{A,B,C,E,D,F}**,{G},{H,I,J,K},{L,M},{N},{O,P},{Q,R}	**{A,B,C,E,D,F}**,{G},{H,I,J,K},{L,M},{N},{O,P},{Q,R}
13	{A,B,C,E,D,F},{G},**{H,I,J,K,L,M}**,{N},{O,P},{Q,R}	{A,B,C,E,D,F},{G},**{H,I,J,K,L,M}**,{N},{O,P},{Q,R}
14	{A,B,C,E,D,F},{G},{H,I,J,K,L,M},{N},**{O,P,Q,R}**	{A,B,C,E,D,F},{G},{H,I,J,K,L,M},{N},**{O,P,Q,R}**
15	**{A,B,C,E,D,F,G}**,{H,I,J,K,L,M},{N},{O,P,Q,R}	**{A,B,C,E,D,F,G}**,{H,I,J,K,L,M},{N},{O,P,Q,R}
16	{A,B,C,E,D,F,G},**{H,I,J,K,L,M,N}**,{O,P,Q,R}	{A,B,C,E,D,F,G},**{H,I,J,K,L,M,N}**,{O,P,Q,R}
17	**{A,B,C,E,D,F,G,O,P,Q,R}**,{H,I,J,K,L,M,N}	{A,B,C,E,D,F,G},**{H,I,J,K,L,M,N,O,P,Q,R}**
18	{A,B,C,E,D,F,G,O,P,Q,R,H,I,J,K,L,M,N}	{A,B,C,E,D,F,G,O,P,Q,R,H,I,J,K,L,M,N}

Figure 4 visualizes the clusters at the step 16 in Table II. The figure shows that the application of AHC algorithm based on different linkage methods (single-link and *k*-min-link) can produce different clustering results at the next step. In the single-link clustering process (Figure 4a), the distances between "G-O" denoted by *d1* and "O-H" denoted by *d2* are compared, and then clusters *C1* and *C3* are merged, because *d1* is smaller than *d2*. However, this causes the chaining problem as discussed in section 3.1. On the other hand, *k*-min linkage method merges clusters *C2* and *C3* according to the average of top three closest pairs: "O-H", "I-P", and "L-Q". Thus, *k*-min linkage method avoids the construction of long chains and the production of clusters that are elongated as shown in Figure 4b. In addition, *k*-min linkage method reduces the sum of squared errors (SSE) from 3.32 to 2.85 in this example. So, the SSE results show that it is possible to get more optimal clustering results by using *k*-min linkage based agglomerative clustering algorithm.

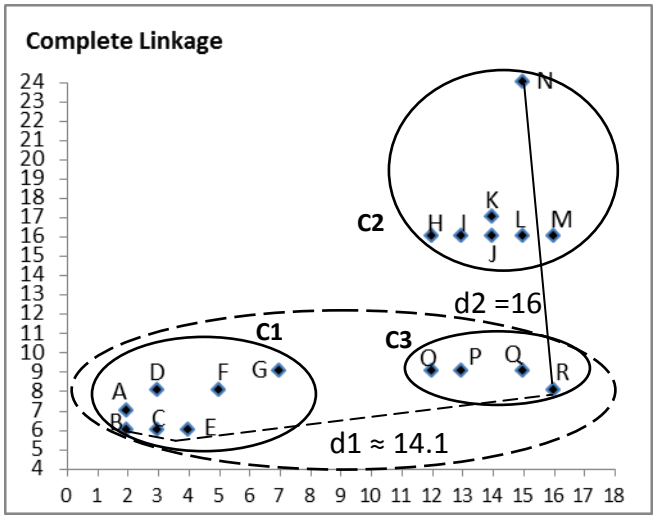


(a) Single Linkage merges clusters C1 and C3

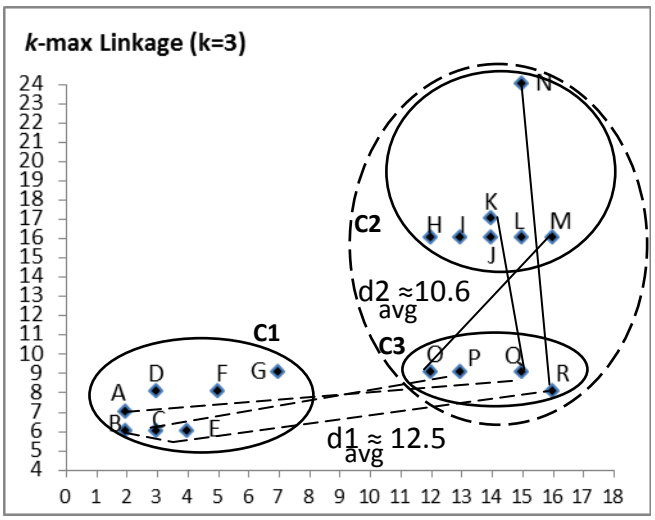
(b) *K*-min Linkage merges clusters C2 and C3

Figure 4. Merging clusters with *k*-min linkage and single linkage methods

Figure 5 is an example of a complete linkage clustering of the set of points given in Table I and the *k*-max linkage clustering of the same set. It visualizes the clusters at the step 16 in Table III. In complete-link clustering, the distances between "B-R" denoted by *d1* and "N-R" denoted by *d2* are compared, and then clusters *C1* and *C3* are merged, because *d1* is smaller than *d2*. On the other hand, *k*-max linkage method merges clusters *C2* and *C3* according to the average of top three farthest pairs: "N-R", "O-M", and "K-Q". The sum of squared errors (SSE) of the clusters that are constructed by the complete and k-max linkages are 3.32 and 2.24 respectively. This means that complete-link clustering didn't find the most optimal cluster structure in this example, because it pays too much attention to outliers as explained in section 3.1. It can be affected by points at a great distance in a cluster where two merge candidates meet. However, *k*-max linkage method avoids the greedy nature of the complete-link by considering several pairs.



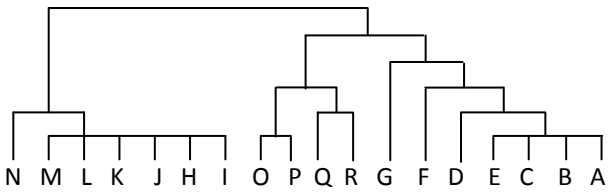
(a) Complete Linkage merges clusters C1 and C3



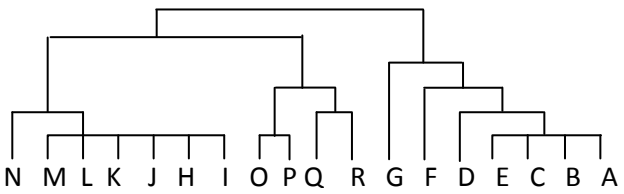
(b) *K*-max Linkage merges clusters C2 and C3

Figure 5. Merging clusters with *k*-max linkage and complete linkage method

The clustering steps of four different methods (single, complete, *k*-min and *k*-max linkage) for the datasets given in Table I are summarized via dendrograms in Figure 6 and Figure 7. The first clusters are the same for single linkage and *k*-min linkage methods. However, the dendrogram differs in the last steps. Complete linkage and *k*-max linkage lead to the similar dendrogram pattern, but differs towards the end. The hierarchy needs to be cut at some point. A number of criteria can be used to determine the cutting point: (i) cut the dendrogram at a prespecified level of similarity, (ii) cut the hierarchy where the gap between two successive similarities is largest, (iii) cut at the point that a target number of clusters is reached [40].

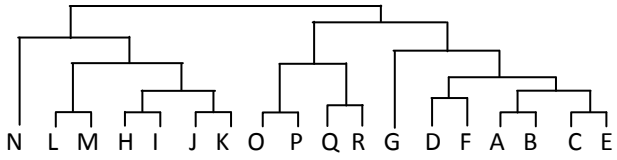


(a) Single linkage method

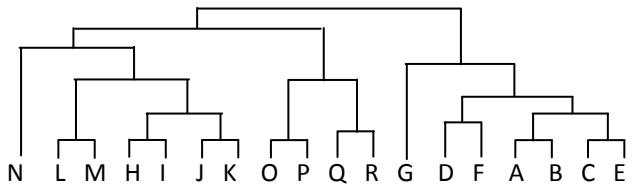


(b) *k*-min linkage method (*k*=3)

Figure 6. Dendrograms of single-link and *k*-min link clustering



(a) Complete linkage method



(b) *k*-max linkage method (*k*=3)

Figure 7. Dendrograms of complete-link and *k*-max link clustering

IV. The Algorithm of *K*-Linkage Method

This study presents an improved version of agglomerative hierarchical clustering algorithm that has the ability to cluster objects in a dataset using *k*-Linkage similarity metric. The pseudocode of the proposed *k*-Linkage method is presented in Figure 8. This algorithm only finds the distance between two clusters *ClusterA* and *ClusterB*, so it should be called from a native AHC algorithm that executes the steps of merging the currently most similar clusters. In the pseudocode, a "*Link*" structure is constituted to define a pair of two objects; one belonging to the first cluster and the other belonging to the second cluster. The algorithm stores a list of pairs by calculating the distances between objects in the pairs using *findDistance()* function. If the linkage type is *k*-min, the list is sorted by ascending order considering *distance* values. On the contrary, in the case of *k*-max, the list is sorted by descending order. After that, the obtained top-*k* distances in the list according to user defined *k* value are averaged. The pairs including selected objects are removed from the list to consider distinct objects in the next step. The output value (*kLinkageDistance*) represents the distance between two clusters based on *k*-Linkage method.

```
struct Link
begin
   int startPoint
   int endPoint
  double distance
end

Algorithm K-Linkage
  Inputs: ClusterA: the first cluster,
          ClusterB: the second cluster,
          k: the number of object pairs,
          linkageType: the type of linkage (k_min or k_max)
  Output: kLinkageDistance
begin
   List list = new List()
   k = min(ClusterA.numberofObjects(), ClusterB.numberofObjects(), k)
   for i = 0 to ClusterA.numberofObjects() - 1
      for j = 0 to ClusterB.numberofObjects() - 1
         list.Add(new Link(i, j, findDistance(ClusterA[i], ClusterB[j])
      end for
   end for
   if linkageType == k_min
      list.OrderBy(d => d.distance)
   else
      list.OrderByDescending(d => d.distance)
   end if
   double totalDistance = 0
   for i = 0 to k-1
      totalDistance += list[0].distance
      int tempStart = list[0].startPoint
      int tempEnd = list[0].endPoint
      list.RemoveAll(s => s.startPoint == tempStart  ||  e => e.endPoint == tempEnd)
   end for
   double kLinkageDistance = totalDistance / k
   return kLinkageDistance
end
```

Figure 8. The pseudocode of the proposed *k*-Linkage method

AHC is one of the most commonly used hierarchical clustering algorithms but it needs a significant amount of time to cluster considerably large datasets. The complexity of the naive AHC algorithm is $O(n^3)$, because it exhaustively requires to scan the $n \times n$ matrix to find the most similar clusters in each of n-1 iterations, where n is the number of instances [40]. To handle this problem and to reduce time complexity to $O(n^2)$, several improved algorithms are proposed, such as SLINK and CLINK for single-linkage and complete-linkage criterions respectively. Another study [41] uses kd-tree (k-dimensional tree) with locally-ordered and heap-based versions in which empirical performance is better than $O(n^2)$ and closer to linear scaling with input size. The time complexity of the proposed *k*-Linkage method is also $O(n^2)$ with a proper data structure and index-assisted searching mechanism, where n is the number of instances in the dataset.

V. Experimental Study

In this study, the proposed linkage types were compared with traditional linkage types such as single, complete, centroid and average linkages. We have expanded an application that can be accessed from GitHub repository: https://github.com/gyaikhom/agglomerative-hierarchical-clustering. The application was implemented for agglomerative hierarchical clustering in C programming language. Our expanded application reads data from a file and includes six different methods to cluster data: single, complete, average, centroid, *k*-min linkage and *k*-max linkage. The application was executed on five different benchmark datasets with varying *k* numbers to determine the optimal solution. In order to evaluate the cluster results and to compare our method with the existing methods, accuracy rate is calculated by comparing output cluster labels with previously known class labels.

A. Dataset Description

In the experimental study, five different datasets which are well-known and broadly used in data mining were selected to demonstrate the capabilities ok *k*-min and *k*-max linkage methods. The datasets, named Iris, Wine, Haberman, Diabetes and Banknote were obtained from UCI Machine Learning Repository that can be accessed from the web site https://archive.ics.uci.edu/ml/datasets.html.

B. Comparison of K-Linkage Method with Traditional Methods

In this experimental study, *k*-min and *k*-max linkage methods have been used for the first time to improve clustering validation and quality. To measure cluster validity, cluster labels that match externally previously known class labels are evaluated and regarded as accuracy rate of clustering result. In simple terms, *accuracy* is the ratio of the number of correctly clustered data points to the total number of data points. Accuracy is calculated using the formula, accuracy = (TP + TN)/ (TP + FN + TN + FP), where TP, FP, TN, and FN denote the number of true positives, false positives, true negatives, and false negatives, respectively.

In order to evaluate the proposed *k*-linkage scheme, we tested it in various datasets. Figure 9 shows the comparative

results of the *k*-min linkage method with single linkage method in terms of accuracy rate. After trying different alternatives, the best value for the *k* parameter was determined as 5. The obtained results show that the proposed *k*-min linkage method is generally more successful than the single linkage method in terms of accuracy rate. Even though single-link clustering may seem preferable at first, it is optimal with respect to the wrong criterion in many clustering applications. Single-link clustering reduces the assessment of cluster quality to a single similarity between a pair of observations. Since the merge criterion is strictly local, it cannot recognize the overall distribution of the clusters. On the other hand, *k*-min linkage method can reflect the true underlying relationship between clusters by considering several pairs and so it can find the better merge candidates.

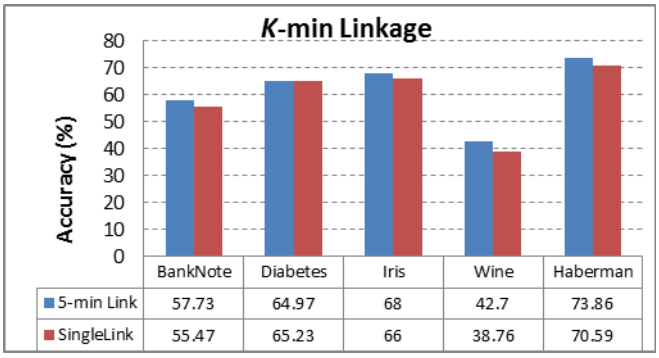


Figure 9. Comparison of *k*-min linkage with single linkage

Figure 10 shows a comparison between complete and *k*-max linkage methods, where *k* is equal to 5. The results show that the proposed *k*-max linkage method has a potential to outperform the complete-link approach. A measurement based on only one pair cannot fully reflect the distribution of instances in a cluster. It is therefore not surprising that complete-link algorithm can produce undesirable clusters. Considering *k* pairs in each step of clustering, instead of only one pair, can improve cluster validation. Taking into account our algorithm's accuracy performance, the proposed method may be recommended for practical use.

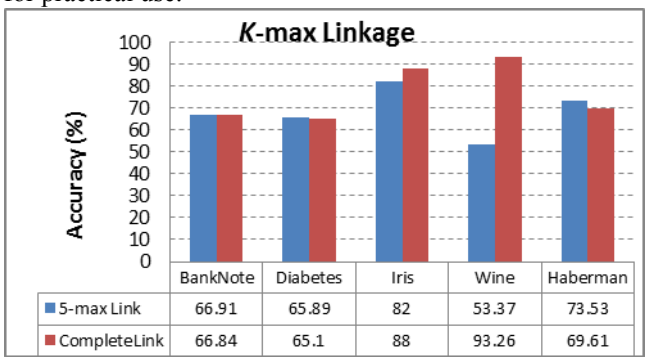


Figure 10. Comparison of *k*-max linkage with complete linkage

Table IV gives comparison results of *k*-linkage metric with traditional similarity metrics (single-link, complete-link, average-link, and centroid linkage) on the datasets in terms of accuracy rate. According to the results, its performance is comparable with other linkage methods. The proposed *k*-linkage method outperforms the current linkage methods in three of the five datasets in terms of the clustering quality. Centroid clustering is not optimal for any dataset because inversions can occur. Rather than average-link, *k*-linkage method can be used, because its similarity measure is conceptually simpler than the average of all pairwise similarities. On the other hand, a spatial indexing mechanism can be used for *k*-linkage methods to determine top-*k* closest or farthest pairs faster.

TABLE IV. COMPARISON OF K-LINKAGE METHODS WITH CLASSICAL METHODS

	Accuracy Rate (%)					
Dataset	***k*-min Link (k=5)**	***k*-max Link (k=5)**	**Single Link**	**Complete Link**	**Average Link**	**Centroid**
BankNote	57.73	**66.91**	55.47	66.84	64.5	63.78
Diabetes	64.97	**65.89**	65.23	65.1	65.23	65.23
Iris	68	82	66	88	**88.67**	66
Wine	42.7	53.37	38.76	**93.26**	38.76	38.76
Haberman	**73.86**	73.53	70.59	69.61	69.61	71.24

C. The Effect of Parameter on K-Linkage Method

K-linkage method requires a user defined parameter *k* which is the number of pairs of instances between clusters. To achieve optimal *k* value, several experiments can be performed as trial-and-error approach and the value which gives the highest accuracy rate can be selected as *k*. The graphs in Figure 11 and Figure 12 show the accuracy rate changes for *k*-min and *k*-max linkage, where *k* is ranging from 3 to 9 in increments of 2. It is possible to see from the results that when the value of the *k* parameter increases, the accuracy rate remains the same or becomes a little bit higher. However, the rate of increase differs from dataset to dataset.

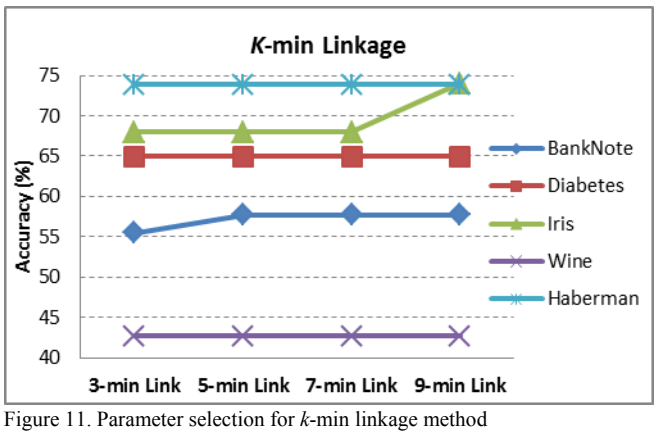


Figure 11. Parameter selection for *k*-min linkage method

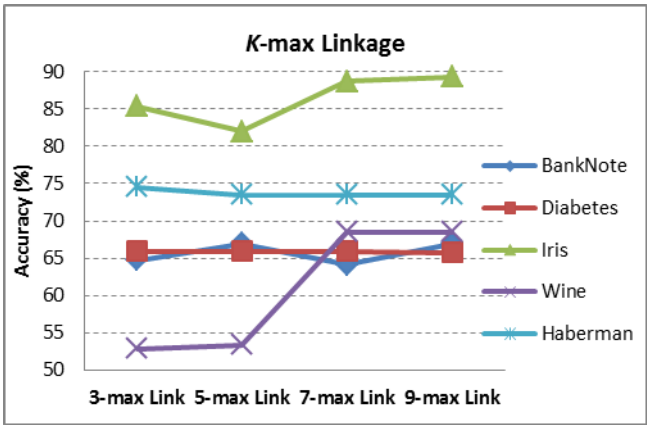


Figure 12. Parameter selection for *k*-max linkage method

VI. Conclusion and Future Work

Agglomerative hierarchical clustering groups objects using a bottom-up approach that starts with clusters that contain only one item and merges two most similar clusters continuously based on a similarity metric. In some cases, traditional metrics can remain incapable of merging optimal clusters. Because of this reason, our study proposes a novel similarity metric, named *k*-linkage that evaluates similarity between pair of clusters considering *k* observations from them. This article also proposes two novel concepts: *k*-min linkage and *k*-max linkage that find *k* closest and *k* farthest pair of observations to specify similarity between clusters.

In the experimental studies, the proposed linkage method was executed on five different benchmark datasets with varying *k* number and compared with traditional linkage methods. The results show that the proposed approach can generally produce more accurate clustering results than classical similarity metrics. According to the results, we recommend *k*-linkage method for hierarchical clustering because it is the method that usually produces the clusters with the higher accuracy and lower SSEs. It does not suffer from chaining, from sensitivity to outliers and from inversions.

In the future, the following studies can be carried out:

- In this study, Euclidean distance was used as a measure to find the similarity between clusters, since datasets consist of numerical values. However, our proposed approach can also be applied on a different dataset, which includes categorical values, by using a different metric such as Jaccard distance.
- In the experimental studies, the optimal value of *k* was determined by trial-and-error method and the value which gives the highest accuracy rate was selected as *k*. Instead of this method, a new algorithm can be developed to find an optimal *k* value.
- It is also possible to use a spatial index such as KD-tree or R-tree for quick neighborhood lookup. In addition, different data structures such as heap can be used in the solution.
- The algorithm of *k*-Linkage can be easily parallelizable and thus may be run on multiple threads to speed up its execution further on. It is possible to perform the parallel computations of different cluster pairs in order to obtain a desired clustering.

To apply the proposed approach on the huge amount of data, a cloud-based framework can be developed to process data in an efficient and low-cost way.

References

[1] H. Yoon, S. Park, "Determining the structural parameters that affect overall properties of warp knitted fabrics using cluster analysis," Textile Research Journal, vol. 72, no. 11, pp. 1013-1022, 2002. doi: 10.1177/004051750207201114

[2] P. Prada, A. Curran, K. Furton, "Characteristic human scent compounds trapped on natural and synthetic fabrics as analyzed by SPME-GC/MS," Journal of Forensic Science & Criminology, vol. 1, no. 1, pp. 1-10, 2014. doi: 10.15744/2348-9804.1.s101

[3] Y. Loewenstein, E. Portugaly, M. Fromer, M. Linial, "Efficient algorithms for accurate hierarchical clustering of huge datasets: tackling the entire protein space," Bioinformatics, vol. 24, no. 13, pp. i41-i49, 2008. doi: 10.1093/bioinformatics/btn174

[4] D. Wei, Q. Jiang, Y. Wei, S. Wang, "A novel hierarchical clustering algorithm for gene sequences," BMC Bioinformatics, vol. 13, no. 174, pp. 1-15, 2012.doi: 10.1186/1471-2105-13-174

[5] Y. Bang, C. Lee, "Fuzzy time series prediction using hierarchical clustering algorithms," Expert Systems with Applications, vol. 38, no. 4, pp. 4312-4325, 2011. doi: 10.1016/j.eswa.2010.09.100

[6] H. Gao, J. Jiang, L. She, Y. Fu, "A new agglomerative hierarchical clustering algorithm implementation based on the Map Reduce framework," International Journal of Digital Content Technology and its Applications, vol. 4, no. 3, pp. 95-100, 2010. doi: 10.4156/jdcta.vol4.issue3.9

[7] S. Horng, M. Su, Y. Chen, T. Kao, R. Chen, J. Lai, C. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," Expert Systems with Applications, vol. 38, no. 1, pp. 306-313, 2011. doi: 10.1016/j.eswa.2010.06.066

[8] J. Almeida, L. Barbosa, A. Pais, S. Formosinho, "Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering," Chemometrics and Intelligent Laboratory Systems, vol. 87, no. 2, pp. 208-217, 2007. doi: 10.1016/j.chemolab.2007.01.005

[9] S. Deininger, M. Ebert, A. Fütterer, M. Gerhard, C. Röcken, "MALDI imaging combined with hierarchical clustering as a new tool for the interpretation of complex human cancers," Journal of Proteome Research, vol. 7, no. 12, pp. 5230-5236, 2008. doi: 10.1021/pr8005777

[10] A. Shalom, M. Dash, "Efficient partitioning based hierarchical agglomerative clustering using graphics accelerators with Cuda," International Journal of Artificial Intelligence & Applications, vol. 4, no. 2, pp. 13-33, 2013. doi: 10.5121/ijaia.2013.4202

[11] H. A. Dalbouh, N. M. Norwawi, "Bidirectional agglomerative hierarchical clustering using AVL tree algorithm," International Journal of Computer Science Issues, vol. 8, no. 5, pp. 95-102, 2011.

[12] E. Althaus, A. Hildebrandt, A. K. Hildebrandt, "A Greedy algorithm for hierarchical complete linkage clustering," in International Conference on Algorithms for Computational Biology, Tarragona, 2014, pp. 25-34. doi: 10.1007/978-3-319-07953-0_2

[13] A. Mamun, R. Aseltine, S. Rajasekaran, "Efficient record linkage algorithms using complete linkage clustering," PLOS ONE, vol. 11, no. 4, pp. 1-21, 2016. doi: 10.1371/journal.pone.0154446

[14] O. Yim, K. Ramdeen, "Hierarchical Cluster Analysis: Comparison of three linkage measures and application to psychological data," The Quantitative Methods for Psychology, vol. 11, no. 1, pp. 8-21, 2015. doi: 10.20982/tqmp.11.1.p008

[15] Y. Li, L. R. Liang, " Hierarchical clustering of features on categorical data of biomedical applications," in Proceedings of the ISCA 21st International Conference on Computer Applications in Industry and Engineering, Hawaii, 2008.

[16] E. Nasıbov, C. Kandemır-Cavas, "OWA-based linkage method in hierarchical clustering: Application on phylogenetic trees," Expert Systems with Applications, vol. 38, no. 10, pp. 12684-12690, 2011. doi: 10.1016/j.eswa.2011.04.055

[17] S. Hirano, X. G. Sun, S. Tsumoto, "Comparison of clustering methods for clinical databases," Information Sciences, vol. 159, no. 3-4, pp. 155-165, 2004. doi: 10.1016/j.ins.2003.03.011

[18] J. Bien, R. Tibshirani, "Hierarchical clustering with prototypes via minimax linkage," Journal of the American Statistical Association, vol. 106, no. 495, pp. 1075-1084, 2011. doi: 10.1198/jasa.2011.tm10183

[19] M. Gagolewski, M. Bartoszuk, A. Cena, "Genie: A new, fast, and outlier-resistant hierarchical clustering algorithm," Information Sciences, vol. 363, pp. 8-23, 2016. doi: 10.1016/j.ins.2016.05.003

[20] S. Dasgupta, P. Long, "Performance guarantees for hierarchical clustering," Journal of Computer and System Sciences, vol. 70, no. 4, pp. 555-569, 2005. doi: 10.1016/j.jcss.2004.10.006

[21] J. Wu, H. Xiong, J. Chen, "Towards understanding hierarchical clustering: A data distribution perspective," Neurocomputing, vol. 72, no. 10-12, pp. 2319-2330, 2009. doi: 10.1016/j.neucom.2008.12.011

[22] A. Mirzaei, M. Rahmati, "A novel hierarchical-clustering-combination scheme based on fuzzy-similarity relations," IEEE Transactions on Fuzzy Systems, vol. 18, no. 1, pp. 27-39, 2010. doi: 10.1109/tfuzz.2009.2034531

[23] P. Contreras, F. Murtagh, "Fast, linear time hierarchical clustering using the Baire metric," Journal of Classification, vol. 29, no. 2, pp. 118-143, 2012. doi: 10.1007/s00357-012-9106-3

[24] A. Barirani, B. Agard, C. Beaudry, "Competence maps using agglomerative hierarchical clustering," Journal of Intelligent Manufacturing, vol. 24, no. 2, pp. 373-384, 2011. doi: 10.1007/s10845-011-0600-y

[25] H. Clifford, F. Wessely, S. Pendurthi, R. Emes, "Comparison of clustering methods for investigation of genome-wide methylation array data," Frontiers in Genetics, vol. 2, no. 88, pp. 1-11, 2011. doi: 10.3389/fgene.2011.00088

[26] Y. M. Yacob, H. A. M. Sakim, N. A. M. Isa, "Decision tree-based feature ranking using Manhattan hierarchical cluster criterion," International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering, vol. 6, no. 2, pp. 765-771, 2012.

[27] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, A. Song, "Efficient agglomerative hierarchical clustering," Expert Systems with Applications, vol. 42, no. 5, pp. 2785-2797, 2015. doi: 10.1016/j.eswa.2014.09.054

[28] M. Łuczak, "Hierarchical clustering of time series data with parametric derivative dynamic time warping," Expert Systems with Applications, vol. 62, pp. 116-130, 2016. doi: 10.1016/j.eswa.2016.06.012

[29] D. Eppstein, "Fast hierarchical clustering and other applications of dynamic closest pairs," Journal of Experimental Algorithmics, vol. 5, p. 1-10, 2000. doi: 10.1145/351827.351829

[30] Y. Lu, Y. Wan, "PHA: A fast potential-based hierarchical agglomerative clustering method," Pattern Recognition, vol. 46, no. 5, pp. 1227-1239, 2013. doi: 10.1016/j.patcog.2012.11.017

[31] D. Müllner, "fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python," Journal of Statistical Software, vol. 53, no. 9, 2013. doi: 10.18637/jss.v053.i09

[32] E. Masciari, G. M. Mazzeo, C. Zaniolo, "A new, fast and accurate algorithm for hierarchical clustering on Euclidean distances," in Pacific-Asia Conference on Knowledge Discovery and Data Mining, Gold Coast, 2013.doi: 10.1007/978-3-642-37456-2_10

[33] I. Davidson and S. S. Ravi, "Towards efficient and improved hierarchical clustering with instance and cluster level constraints", Technical Report, Department of Computer Science, University at Albany, 2005.

[34] S. Bobdiya, K. Patidar, "An efficient ensemble based hierarchical clustering algorithm," International Journal of Emerging Technology and Advanced Engineering, vol. 4, no. 7, pp. 661-666, 2014.

[35] L. Zheng, T. Li, C. Ding, "A framework for hierarchical ensemble clustering," Acm Transactions on Knowledge Discovery from Data, vol. 9, no. 2, 2014. doi:10.1145/2611380

[36] Z. Chen, S. Zhou, J. Luo, "A robust ant colony optimization for continuous functions," Expert Systems with Applications, vol. 81, pp. 309-320, 2017. doi: 10.1016/j.eswa.2017.03.036

[37] J. Vaščák, "Adaptation of fuzzy cognitive maps by migration algorithms," Kybernetes, vol. 41, no. 3, pp. 429-443, 2012. doi: 10.1108/03684921211229505

[38] R. Precup, M. Sabau, E. M. Petriu, "Nature-inspired optimal tuning of input membership functions of Takagi-Sugeno-Kang fuzzy models for anti-lock braking systems," Applied Soft Computing, vol. 27, pp. 575-589, 2015. doi: 10.1016/j.asoc.2014.07.004

[39] S. Vrkalovic, T. Teban, I. Borlea, "Stable Takagi-Sugeno fuzzy control designed by optimization," International Journal of Artificial Intelligence, vol. 15, no. 2, pp. 17-29, 2017.

[40] C. D. Manning, P. Raghavan, H. Schütze, "Hierarchical clustering", An Introduction to Information Retrieval, pp. 377-402, Cambridge University Press, 2012.

[41] B. Walter, K. Bala, M. Kulkarni, K. Pingali, "Fast agglomerative clustering for rendering," in The IEEE Symposium on Interactive Ray Tracing, Los Angeles, 2008.