A MIXED-INTEGER PROGRAMMING APPROACH TO THE CLUSTERING PROBLEM WITH AN APPLICATION IN CUSTOMER SEGMENTATION

by

Burcu Sağlam

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

Koç University
July, 2005

Koç University
Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Burcu Sağlam

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Assist. Prof. Metin Türkay

Assist. Prof. F. Sibel Salman

Assoc. Prof. Serpil Sayın

Assist. Prof. Deniz Aksen

Assist. Prof. Atilla Gürsoy

Assoc. Prof. Fikri Karaesmen

Assist. Prof. E. Lerzan Örmeci

Date:

ABSTRACT

Clustering is an important data mining problem that identifies groups of entities that are similar to each other with respect to a certain number of attributes. In this thesis, two mathematical programming based approaches to clustering are presented. The proposed mathematical programming based approaches are applied to a digital platform company's customer segmentation problem involving transactional attributes related to the customers. In the first model, the clustering problem is formulated as a mixed-integer linear programming problem with the objective of minimizing the maximum cluster diameter among all clusters. In order to overcome difficulties related to computational complexity of this model, a heuristic clustering approach is developed that improves computational times dramatically without compromising from optimality in most of the cases that were tested. In the second model, the clustering problem is modeled as a mixed-integer nonlinear programming problem with the objective of minimization of sum of within-group distances. We show that every extreme point solution of the nonlinear relaxation of the model is integer due to the unimodularity property of the constraint set of the model. We solve the continuous relaxation of the model, obtain integer solutions and observe that solution time of the model is very short. Although the model is solved to local optimality, the interpretations derived from the solution of the model are promising. The performance of the approaches is tested both on an illustrative example and on a real problem. The analysis of the results indicates that the approaches are computationally efficient and create meaningful segmentation of data.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisors Assist. Prof. Metin Türkay, Assist. Prof. F. Sibel Salman and Assoc. Prof. Serpil Sayın for their supervision and encouragement in all steps of the development of this thesis.

I am grateful to Assist. Prof. Deniz Aksen, Assist. Prof. Atilla Gürsoy, Assoc. Prof. Fikri Karaesmen and Assist. Prof. E. Lerzan Örmeci for taking part in my thesis committee, for critical reading of this thesis and for their valuable suggestions and comments.

I wish to thank Salih Eren, Devrim Melek Tunç and Kemal Özden at the Information Technologies Department of Digiturk for providing the data for the experiments and their support in the development of this thesis.

And I would like to express my appreciation to my friends Semra Ağralı, Aysun Aker, Alper Altan, Ali Selim Aytuna, Ferit Ozan Akgül, Ayşegül Azder, Kıvılcım Büyükhatipoğlu, Ertan Çetingül, Ferhan Ercengiz, Umut Küçükkabak, Zeynep Orman, Müge Pirtini, Fadime Üney Yüksektepe and Işıl Yıldırım for their unforgettable friendships during my master study at Koç University.

Finally, I would like to thank Uğur Kalaba for his support and encouragement in all steps of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

MWGD	maximum within group distance
SWGD	sum of within-group distances
TWGD	total within-group distances
WGSD	within group sum of distances
WGSS	within group sum of squares
WGAD	sum of average within group distances
MIP	mixed integer programming
NLP	nonlinear programming
MINLP	mixed integer nonlinear programming
i	the instance i
k	the cluster k
n	the number of instances
K	the number of clusters
m	the number of dimensions
R^m	m - dimensional space
d_{ij}	the distance between the instance i and j in any norm on R^m
D_k	the diameter of the cluster k
$D_{\max}$	the maximum diameter among the generated clusters
x_{ik}	the binary decision variable that represents the assignment of an instance to a particular cluster
v	the vertex v
V	the set of vertices
E	the set of edges

$G = (V, E)$	the incomplete graph in which V denotes the set of vertices and E denotes the set of edges
R	the distance parameter used in the construction of parametric $G_R = (V, E_R)$ graph
$G_R = (V, E_R)$	the parametric graph constructed where an edge between two vertices exist if the weight of the edge is less than the parameter R
$u(t)$	upper bound in iteration t of the seed finding algorithm which is used in calculation of the R distance value
$l(t)$	lower bound in iteration t of the seed finding algorithm which is used in calculation of the R distance value
S	the maximal independent set
$\lvert S \rvert$	the cardinality of the maximal independent set
C_k	the set of instances assigned to cluster k
$\lvert C_k \rvert$	the cardinality of the cluster k
$C(i)$	the cluster label of the instance i
AP_i^k	the average distance between the instance i and instances assigned to cluster k
AT	the sum of within-cluster average distances
β	multiplier which controls the size of the diameter in the reassignment algorithm
$neighbor(i)$	the number of neighbor nodes the instance i has in the parametric graph $G_{R_1} = (V, E_{R_1})$ in the determination of the $SF2$ algorithm
dis_{ik}	the distance between instance i and seed k
rr_{ikm}	the ratio of the distance between instance i and seed k and the distance between instance i and seed m
c	the cutoff parameter value for setting inclusion and exclusion rules

Chapter 1

INTRODUCTION

Customer Relationship Management (CRM) is one of the most important topics in marketing. Today, companies are looking for answers to questions such as who are the most preferred customers, who are loyal to the company, who have a tendency to churn, which product may attract more customers, which products should be placed together on the shelves of a retail outlet to increase the sales, etc. CRM is a marketing concept which aims to create new ways to answer these kinds of questions that the companies are seeking to respond. It can be defined as improved understanding of customer needs and desires, and better targeting of marketing efforts to satisfy the expectations of ongoing customers, to attack the potential ones and to increase the market share of a company. When we purchase something in the store, we leave more than our money behind us. Our transaction contains unknown facts about us and since the early eighties companies have the opportunity to store these transactions that may be instrumental in understanding the behavior of their customers. This fact led to an increased dimensionality of the stored data in data warehouses of the companies. As a result, the abundance of large data collections and the need to extract hidden knowledge within them have triggered the development of another new research area, Data Mining (DM). DM is the development of new methods to analyze large databases in order to find out hidden information. Researchers from different branches of science, such as machine learning, statistics, databases, visualization and graphics, optimization, computational mathematics and theory of algorithms are working in this interdisciplinary field.

Although DM encompasses statistical techniques which necessitate statisticians and experts to explore and to analyze the data, the state of the art indicates that there is more to DM than statistical analysis. Marketing people are new owners of the job and they are asking for more actionable results. More importantly, new owners of the data are asking new questions that cannot be answered by existing tools. Furthermore, traditional techniques are based on the fact that one may first develop a hypothesis and then validate it on the data, but new era is asking for unknowns which should be driven from data.

DM has several application areas and therefore several techniques suitable for application are being explored by researchers. Segmentation is the general name of a descriptive DM model class with the purpose of improving the efficiency of marketing strategies. In marketing literature it is defined as the grouping of customers into previously unknown categories. Clustering analysis which originates from mathematics is a DM technique developed for the purpose of identifying groups of entities that are similar to each other with respect to certain similarity measures which are utilized in segmentation applications. It is useful in several exploratory pattern analysis, grouping, decision-making, and machine-learning situations, including DM, document retrieval, image segmentation and pattern recognition.

Clustering is particularly applied when there is a need to partition the instances into natural groups, but predicting the class of objects is almost impossible. There are a large number of approaches to the clustering problem, including optimization based models that employ mathematical programming for developing efficient and meaningful clustering schemes. It has been widely emphasized that clustering and optimization may help each other in several aspects, leading to better methods and algorithms with increased accuracy and efficiency. Exact and heuristic mathematical programming based clustering algorithms have been proposed in recent years. However, most of these algorithms suffer from scalability as the size and the dimension of the data set increases. In most cases, mathematical programming based clustering models belong to the group of NP-complete problems that makes them have exponential solution times. Another important discussion in clustering is the definition of best partitioning of a data set, which is difficult to predict since it is a relative and subjective topic. Different models may result in different solutions subject to the selected clustering criteria and the developed clustering model. Therefore, the development of efficient and robust clustering algorithms is a need in both DM and CRM areas.

This thesis studies a mathematical programming based approach to the clustering problem with a real application. For this purpose, two-mixed integer nonlinear mathematical programming models are proposed with the criteria of minimization of the maximum cluster diameter and minimization of sum of within group pair-wise distances respectively. We refer to these models as *MIP-Diameter* and *MINLP-SWGD* in the rest of the thesis. The first model is linearized and reduced to a mixed integer linear mathematical programming model. The second nonlinear model is solved as a continuous relaxation and reduced to a nonlinear programming model. The solution of the relaxed *NLP-SWGD* model is shown to be integer due to the total unimodularity property. The second model results in a local optimal solution in a very reasonable time. The accuracy of the proposed models is analyzed on an illustrative data set, and the computational performance of the models is analyzed on a real data set. Furthermore, in order to overcome the computational difficulties of the *MIP-Diameter* model, the graph theoretic maximal independent set approach is utilized to provide initial assignments of data points to clusters. Two seed finding algorithms, *SF*1 and *SF*2, are developed with the purpose of reducing the solution time of the model while preserving the near-optimality of the final solution. As shown in our experimental results, the solution time of the model is reduced drastically by applying the proposed seed finding algorithms to the model while the solution quality worsens slightly. In addition to seed finding algorithms, a preprocessing methodology is proposed to further improve the solution time of the model by reducing the size of the branch and bound tree. This preprocessing methodology is an extension of the seed finding algorithms. It is a heuristic approach to determine the inclusion of some instances to clusters initially and also exclusion of some instances from clusters. The experiments on illustrative and real data sets are also reported for this methodology to analyze the effect of these rules on the solution quality of the proposed model and algorithm. Previous studies conducted upon minimization of maximum diameter criterion deal only with the value of the objective function and the solution time of the model. In this study, we have shown that the solution of the *MIP-Diameter* model is still open to improvements even in the case of the model being solved to optimality. There exist alternative optimal solutions due to the structure of the model since it considers only the assignment of instances which determines the value of the objective function of the model. In order to improve the solution quality of the *MIP-Diameter* model, a reassignment algorithm is proposed which relies on improving two criteria, the average of distances between each data point and other data points assigned to the same cluster and the sum of average within-cluster distances. The proposed seed finding algorithm, the *MIP-Diameter* model and the reassignment algorithm constitute the proposed clustering algorithm. The experimental results of the algorithm are reported to analyze its efficiency and performance. The results of the proposed algorithm are compared with the results of the well-known *K-Means* algorithm, and interpretations derived from the solutions of both algorithms are given.

This thesis is organized as follows: The literature survey on customer relationship management, data mining, the clustering problem and optimization based clustering models is given in Chapter 2. In Chapter 3, the proposed *MIP-Diameter* model, the proposed seed finding algorithms *SF*1 and *SF*2, the reassignment algorithm and the preprocessing rules are given and experiments conducted on a small illustrative data set are reported to analyze the accuracy and performance of the models visually. Experimental results and findings on the Digiturk data set are reported in Chapter 4. In Chapter 5, the proposed *MINLP-SWGD* model's formulation is given and experiments with the proposed model applied to the Digiturk problem to analyze its performance and efficiency are reported. In Chapter 6, the two proposed clustering models are compared in terms of studied clustering performance measures and benchmarked with the solution of the *K-Means* algorithm. Also, interpretations derived from the solution of the algorithm, the nonlinear model and the *K-Means* algorithm are given. Finally, the thesis is concluded by summarizing the results and discussing ideas for future work.

Chapter 2

LITERATURE SURVEY

Customer relationship management (CRM) is described by several professionals as efforts done by companies with the purpose of understanding their customers successfully, satisfying their needs and anticipating their desires without compromising from increasing their profitability and enlarging their market shares in today's competitive markets. Today, companies are in search of transactions that possess key information about their customers which may lead them to success and revenue. As transactional and categorical data accumulate and are stored in data warehouses, extracting knowledge out of huge data sets becomes a crucial problem. Data mining (DM) has been an integral part of CRM studies, with the premise that companies can achieve successful customer relations if they understand their customers' characteristics and desires in an improved fashion with respect to the traditional CRM implementations as also pointed out by Nemati and Barko [1].

DM is a recently developing research area which incorporates the development of new methods, algorithms and technology to explore patterns and relationships to gain insight and to extract hidden knowledge in data warehouses. Briefly, Grossman et al. [2] defined DM as the process of finding hidden patterns, associations, rules and statistically significant structures in large databases. Owners of data are asking new questions that cannot be answered by existing data analysis models. DM sometimes seems to be an automatic information extraction system but in fact this approach is contradictory to the current state of the art in DM where human intervention is inevitable. The importance of interaction between human and data mining is also pointed out in Grossman et al. [2] and Fayyad et al. [3].

Rygielski et al. [4] gave an overview of data mining; its applications in industry and the techniques used under this topic, and explained the relation and interaction between DM and CRM applications from various aspects. Interested reader in applications of DM may refer to Hwang et al. [5].

The role of optimization in DM has been elaborated by Padmanabhan and Tuzhilin [6] who discussed that DM and optimization can help each other in the development of new CRM applications, such as maximizing customer lifetime value, customer analysis and customer interactions. In the same study, it is stated that optimization and DM may interact in one of the following two ways, either optimization may state a step of DM process, or optimization may be the main frame of the development of DM techniques.

Developing a DM model based on optimization is typically difficult and furthermore it is often intractable to obtain an exact solution of the model. In most DM problems, there will be no unique solution and problem sizes give rise to computational problems where the number of variables and constraints increase drastically. In the literature, there exist studies which incorporate mathematical programming approaches into DM but their main limitation always remains as problem size and solution times.

Clustering is a descriptive DM technique with applications in data exploration, segmentation, targeted marketing, and cross-selling [3], [5]. Originally, it is a mathematics problem which has been studied for years. The definition of the clustering problem is to group similar instances into a number of categories to characterize and to classify the data. These categories, in other words, clusters can be formed in various forms with various objectives. In clustering, there exist two objectives which should be considered simultaneously: the similarity of instances within a cluster should be maximized while the similarity of instances between clusters should be minimized. Roiger and Geatz [7] pointed out benefits of clustering such as observing meaningful and hidden relations that may exist in the data set, analyzing the set of input attributes in order to find out the best set of attributes to build a promising business model, and even in some cases determining outliers.

In this thesis we refer to the term instance as a point in a single or a multi-dimensional space. An instance consists of attributes and each attribute constitutes a dimension in space.

There exist different clustering approaches in the literature such as distance-based, model-based and partition-based clustering approaches. Since the proposed models in this thesis are distance-based special attention is given to distance-based clustering approaches. In the distance-based approach, similarity and dissimilarity between instances is captured by a distance function. Attributes of an instance are quantified and the similarity of instances is formulated using a metric. These metric measures are then aggregated using averages and correlations. On categorical data it is harder to propose metric measures since it is possible to deteriorate data while transforming it from categorical scale to numerical scale. Assume that we have an attribute for the brand of our customers' cars. The values of this attribute are categorical and we want to use a distance-based clustering scheme in our application. And further assume that this attribute has three different values such as BMW, Volkswagen and Mercedes-Benz. In the transformation of this attribute into numerical scale assume that we denote BMW with 1, Volkswagen with 2 and Mercedes-Benz with 3. If we apply such a transformation, we accept that Volkswagen is more similar to BMW and Mercedes-Benz than BMW and Mercedes-Benz are similar to each other. But actually, the brands BMW and Mercedes-Benz belong to the group of luxury automobiles and Volkswagen belongs to a lower automobile class. Thus, due to this type of transformation difficulties it is hard to apply metric measures on data sets including categorical attributes.

In DM, the dimensionality of data is one of the main concerns since it directly influences the efficiency, performance and accuracy of the proposed solution methodology. At this point, definitions of data, model and solution method terms are given to make the things clearer. Data refers to a dataset which consists of instances. The model defines according to which criterion and constraints the partitioning of the data set will be obtained. Solution method of a model is the way the model is solved. Assume that we have an optimization based model. We can solve it to optimality if the computational complexity of the model allows by solving it with the appropriate solver or if it does not allow we may apply a heuristic approach to solve the problem.

Reduction of dimensionality constitutes another research field in DM. It aims to reduce data size in a model and achievement of stability and robustness in the behavior of the model solution [3]. The reduction should be unique for the combination of data, model and results. Depending on the data or the model, there are several ways to diminish the dimensionality of data. Derivation of new more manageable attributes that compromise information stored in attributes which are used in the derivation of these new attributes is one of techniques used under this topic. New attribute derivation is highly put into practice in case of having a lot of time-series and sequential records. Removal of dependent or similar attributes is a must since the existence of highly dependent attributes in the data set may lead to skewed results and this process is also classified under data cleaning step of the overall DM process.

In some cases, dimensionality reduction may result in an easier fitting of the model since there can be less number of attributes and/ or instances in a reduced data set. However this may affect the accuracy of the model. In time-series and sequential data analysis, one may apply different techniques in reducing the dimensionality of data such as Kalman filtering, hidden markov models and database methods like detection of episodes and frequent sequences [3].

Another important point in clustering is how to initiate the partitioning of the instances. According to this criterion, most commonly, analytical clustering techniques are analyzed in two types: hierarchical and nonhierarchical [8], [9], [10].

The hierarchical clustering (HC) methods proceed as a series of partitioning operations starting with a single cluster containing all instances and ending when a predefined terminating criterion is achieved. These techniques are classified as agglomerative and divisive methods [8]. The HC methods do not require the number of clusters to be known at the beginning, which constitutes a robust advantage over nonhierarchical methods. On the other hand in these methods, once an instance is assigned to a cluster, the assignment is irrevocable. Therefore, Sharma [11] stated that the HC methods are applied to generate some interpretations over the data set and the solution of a HC may be used as an input for a nonhierarchical method in order to improve the resulting cluster solution.

Nonhierarchical clustering (NHC) methods, also called partitioning clustering [10], refers to the case where the number of partitions is known a priori. Usually, the data is divided into K clusters initially and the NHC algorithm iterates for all possible movements of data points between the formed clusters until a stopping criterion is met. According to Sharma [11], the nonhierarchical algorithms have two main differences; the selection of the initial cluster representatives, such as mean, centroid or median, and the way it assigns the instances to the clusters. In these methods, each cluster can be represented by the center of the cluster (*K-Means*) or by one instance located in the cluster center (K-Medoids). The NHC algorithms are sensitive to initial partitions and due to this fact, there exist too many local minima [11].

On the other hand, there are also NHC algorithms that build the clusters incrementally, such as the heuristic global *K-Means* method of Likas et al. [12]. They state that the optimal K cluster solution can be reached from the optimal solution of $(K-1)$ cluster problem by utilizing a local search over all data set by running the *K-Means* algorithm repetitively to come up with the best possible K^{th} cluster center. Experimentally, the algorithm is shown to be superior to the classical *K-Means* algorithm. Although the algorithm may lead to promising solutions, it does not guarantee to reach to the global optimal solution.

Perhaps the most commonly used clustering algorithm is the *K-Means* algorithm. In this thesis, our proposed method is benchmarked with the *K-Means* algorithm, and therefore a literature review on this approach will follow now. Fisher [13] modeled the objective function of minimization of the sum of within-cluster distances for single dimension case and proposed a least squares algorithm without a stopping criterion. In later years, *K-Means* algorithm is implemented by applying this criterion as an error function. *K-Means* is an iterative distance-based clustering algorithm [10], [14] which was first suggested by Forgy [15]. In *K-Means* algorithm, the similarity measure is the squared Euclidean distance. In the widely known *K-Means* approach, the iterative objective is to minimize the summation of 2-norm distances between each data point and the center of the cluster which it belongs to [16]. The *K-Means* algorithm has two main steps: the assignment of instances to the proper clusters and the updating of the cluster centers [9]. Each instance is assigned to a cluster that it is most similar subject to the criteria defined. Cluster centers are updated after the assignment of all instances is finished by computing the mean for each cluster. This may continue until no reassignment takes place or a predefined terminating criterion is achieved.

K-Means is reported to work well in practice, although its worst case time complexity is shown to be exponential [10]**.** The *K-Means* algorithm works well when the candidate clusters are approximately of equal size. Also, with *K-Means* it is impossible to interpret which attributes are significant and for this reason several irrelevant attributes may cause sub-optimal results. An important problem faced with *K-Means* algorithm is that the resulting solution is a local minimum which is sensitive to the initially selected cluster centers [14]. In their study, it is shown that the algorithm converges to a local minimum if differentiability conditions to a Kuhn-Tucker point hold. Thus, in order to find good minima, repeating the algorithm several times with different starting points is required, without guaranteeing the global optimality.

The determination of initial cluster centers constitutes another research area since it directly affects the accuracy and the performance of *K-Means*, a good initial solution may lead to a highly accurate and efficient clustering and a bad initial solution may lead to an inaccurate and late terminating clustering scheme. Sampling is pointed as an alternative to determine the set of initial cluster centers. There are various studies conducted on the topic of finding good seeds, where a seed is an instance of the data set which may possibly attract other data points around itself according to the defined clustering criterion, but none of them promises to find out the initial seeds that will achieve the global optimal solution [17].

Other interesting clustering algorithms are Birch [10], Cobweb [18] and Clarans [19]. Birch (Balanced Iterative Reducing and Clustering using Hierarchies) is suggested where available memory is limited which is first presented by Zhang et al. [10]. It is an incremental and hierarchical clustering algorithm applicable to large data sets. Characteristics of the Birch algorithm are that one iteration of the algorithm ends with a successful clustering, and additional steps improve the quality of the clustering. In Zhang et al.'s study, it is stated that the other clustering techniques ignore the fact that data points in a data set are not equally important and dense data points can be considered as a group. This clustering method is applicable to large datasets, and it achieves this by concentrating on dense portions of the dataset. The limitation of the technique is that, it is sensitive to parameter settings.

Cobweb, introduced by Fisher [18], is a well-known representative of conceptual clustering technique. It forms a concept hierarchy to capture knowledge [7]. Cobweb deals only with categorical (nominal) data and groups the instances in a hierarchical structure based on the cluster quality. The measure of quality of clusters is called category utility. Main limitations of the Cobweb are that it handles only nominal attributes, and computing the category utility measurement is an expensive task to accomplish [10], [20]. Also, Roiger and Geatz [7] emphasized the impact of instance ordering on the resulting clustering.

Clarans is a type of K-Medoids algorithm. K-Medoids algorithms are a variation of *K-Means* type algorithms in which the cluster information is captured by the medoid of the cluster instead of the mean of the cluster. Clarans differs from other K-Medoids algorithms by its randomized partial search strategy. In Clarans, the clustering problem is represented as a graph in which each node is a K-partition represented by K medoids, and two nodes are neighbors if only one medoid of them is different and the rest is same. The algorithm starts with a randomly selected node. Starting with this node, it checks the defined limited number of neighbor nodes and if a better neighbor is found, it switches to that neighbor and continues; if not it records the initial starting point as local minimum and selects a new random starting point. Limitations of the Clarans are that it may fail to converge to a definite local minimum due to random search and efficiency considerations [10].

In the literature, there are several studies that incorporate optimization in clustering. Optimization based clustering methods are developed to produce an optimal partition of instances into a specified number of groups by either minimizing or maximizing some numerical criterion. The criterion depends on the application and data being studied. It is typically NP-Hard to reach to global optimum in optimization based clustering problems [3], [21]. Examples for clustering criterions in optimization can be stated as follows: minimization of sum of within groups squared distances, minimization of sum of within group averages and minimization of maximum cluster diameter.

Researchers have been studying clustering problem with mathematical programming approaches since the early 70s. In an early study, Vinod [22] developed two mathematical programming formulations of the clustering problem. The first formulation originated from warehouse location problems. The idea of a group leader in which an instance of data set that identifies the group is applied and the objective of the proposed first integer programming model is to minimize the sum of within group distances between the leader and instances assigned to corresponding cluster. If we assume the group leader as the center of the cluster then the objective turns out to be the minimization of sum of distances between center of cluster and instances assigned to that cluster. Several definitions for the cost matrix are given. The second formulation is also an integer mathematical programming model with the objective of minimization of within-group sums of squares (*WGSS*). In order to make things clearer, in the model there is a set of constraints to compute cluster means and a set of constraints to count the number of instances assigned to each cluster. A nonlinear objective function for the model is constructed as the summation of distances between instances and their corresponding cluster means. A string property is defined and stated to be necessary to obtain the minimum for this model but in later years, Rao [23] showed that this statement is invalid. Both for the one and multi dimensional cases, by proper selection of cost function and validity of string property, it is shown that the nonlinear objective function can be reduced to a linear one by using 0-1 variables. Computational results are reported only for very small sized data sets and finally the study is concluded by stating that the integer programming formulations are flexible and efficient in grouping problems.

In the years following Vinod's study, Rao [23] formulated the clustering problem with different distance based mathematical programming models respectively with the objectives of minimization of within groups sums of distances (*WGS*D), minimization of the sum of average within group squared distances (*WGAD*), minimization of total within group distance (*TWGD*) and minimization of the maximum within-group distance (*MWGD*). Rao [23] proposed a number of linear and non-linear integer programming models for these clustering objectives. For the *WGSD* criterion, a dynamic programming algorithm was proposed for the 1-dimensional case. For the p-dimensional case, an integer linear programming model was proposed for this partitioning criterion. The formulation is stated to be valid under certain conditions. It is claimed that the model is reduced to a set partitioning problem with a predefined number of partitions. Validation condition of the model is the distance function should be Euclidean and/ or a defined string property in the study should be satisfied. The string property is not a necessary condition for optimality which stated as there exists a seed instance in each cluster and the distance between the leader of a cluster and any instance in other clusters rather than this cluster cannot be less than any distance between this leader and instances assigned to this cluster. A solution scheme is proposed for the reduced set partitioning problem and the solution time is stated to be reasonable. The second proposed model in the study is a nonlinear integer programming formulation of the criterion *WGAD*. In the special case where the number of entities in each cluster is specified, it is shown that the model is reduced to an integer linear programming model. But, for the reduced model it is stated that the number of constraints increases enormously with increased number of instances and clusters, and so that the model is computationally tractable only for small sized sets. An alternatively proposed solution scheme is partitioning instances into two most compact clusters and continuing this process until a predefined number of clusters is reached. As a third model, the *TWGD* is studied and a solution approach only for two cluster case is proposed. Finally, the fourth proposed model is an integer linear programming problem of the criterion *MWGD*. It is stated that the formulation is tractable only for data sets with small number of instances since the number of constraints of the model increase enormously. A heuristic solution approach for the 2-cluster problem is given for this model.

Later on, Brusco [24] studied the *TWGD* and *MWGD* criteria with a branch and bound methodology. The *TWGD* branch and bound algorithm is implemented by deriving better lower bounds with solving sub matrices sequentially and by deriving tighter upper bounds by utilizing an exchange algorithm. The improvement is achieved over the existing work of Klein and Aronson [25]. In their branch and bound algorithm, at any particular node of the branch and bound tree, instance set is divided into two parts, the first n_1 instances are assigned and the last n_2 instances are unassigned. The bound is derived from three components: *TGWD* of assigned instances, *TGWD* between assigned and unassigned instances and *TGWD* of unassigned instances. It is straightforward to calculate the first two components. Brusco proposed new branch and bound algorithm by altering the computing procedure of the third component by applying the exchange algorithm for partitioning developed by Banfield and Bassil [26]. The exchange algorithm is a local search procedure starting with an initial random partitioning of the data set using two types of reassignment operations. The algorithm is modified for *MWGD* criterion since *TGWD* model is criticized as resulting clusters tend to have approximately same number of instances. *MWGD* criterion is selected because to modify the algorithm according to the other criterions such as *WGAD* is stated to be difficult. The modified *MWGD* algorithm is implemented by deriving upper bounds with utilizing a randomized complete-link clustering algorithm. The computational results are reported for data sets consisting of up to 40 instances and the squared Euclidean distance is used. For *TGWD* criterion, the solution time reported for 40 instances and 6 clusters solution is 3152.12 seconds on a 666MHz Pentium III computer with 128MB of RAM running in a Windows 98 environment.The branch-and-bound methods reported in this study are not applicable on large data sets due to concerns on computational time. Moreover, the efficiency and performance of the algorithms are reported to be very sensitive to the number of clusters.

Bradley et al. [9] investigated integer programming formulations with the objective of minimizing the sum of distances between each data point and its corresponding cluster center, in other words implementation of *K-Means* algorithm within an optimization model. In their studies, they divided the clustering methods into three classes: metric-based methods, model-based methods and partition-based methods and focused on mathematical programming formulations of two popular nonhierarchical clustering techniques, *K-Means* and K-Median. Another objective function that has been of interest is the minimization of the sum of within cluster distances.

Koontz et al. [21] propose a branch and bound clustering algorithm that utilizes the *WGSS* criterion. The proposed algorithm is stated to determine sharper lower bounds than the basic branch and bound approach by proving the property, in a set of n instances divided into two sets with n_1 and n_2 instances in each, such that $n_1 + n_2 = n$, the sum of the minimal *WGSS* values for the two sets cannot be greater than the minimal value of *WGSS* for n instances. It is claimed that the efficiency of the proposed algorithm depends heavily on the selected clustering criterion's structure since the algorithm uses bounds based on the properties of the objective function.

Diehr [27] studies algorithm of Koontz et al. [21] with the objectives of minimization of the *WGSS* criterion and *WGAD* criterion for effectiveness and efficiency. Modifications of the previous algorithm are investigated and performance of modifications is studied as a function of number of instances, number clusters and separation degree of the generating distributions of the data set. It is stated that criterions applicable for the proposed algorithm should share the property that an added instance to a set of n instances should not decrease the value of the minimum objective function value. The distance function is defined as squared Euclidean distance. Diehr's algorithm uses the same bounds with Koontz algorithm but their difference is Diehr's algorithm changes the order of enumeration and applies heuristics in order to decrease the solution time. The study is concluded as the proposed algorithm is applicable for data sets up to 120 instances, the number of partitioning is less than 6 or less and the clusters are well separated.

Still, integer programming models for clustering problem, such as those developed by Vinod [22] and Rao [23] may be preferred as they add more flexibility to the model, and they consume less computing time than previous approaches such as dynamic programming.

Here we summarize our contributions with comparison to existing work that's closest to our work. In this thesis we study two mathematical programming models for clustering. Both of these models were proposed earlier by Rao [23]. The objective functions of the studied models are (1) minimization of the maximum cluster diameter (*MWGD*) and (2) minimization of sum of within-group distances (*TWGD*).

We formulate the first model as a mixed integer nonlinear programming (*MINLP*) model. We then study a linearization which corresponds to the model proposed by Rao [23]. For the *MWGD* model, Rao provided a heuristic approach only for the 2-cluster problem and no experimental findings are reported in the study in terms of solution quality and computational findings.

In later years, Brusco [24] studied the *MWGD* criteria with a similar *MINLP* formulation to the Rao's *TWGD* formulation. He proposed a branch and bound algorithm to solve the model. He applied the proposed algorithm to data sets consisting of up to 45 instances. In this study, we aimed at solving larger instances and considered three clustering criteria: (1) minimization of the *MWGD*, (2) minimization of the total within-group averages (*WGAD)* and (3) minimization of the *TWGD*. We first focused on solving the *MWGD* model. In order to improve the solution time of the model we propose two seed finding algorithms that utilize a maximal independent set in a parameterized graph. The solution time of the model is improved drastically by the proposed approach without compromising from optimality significantly. Next, we proposed and tested rules for reducing the size of the model again with minor deviations from optimal solution.

We improve the solution of the model in terms of the *WGAD* criterion by a reassignment algorithm. We also show that solving the $2,3,4,..,k-1,k,..$ clustering problems sequentially by providing the objective value of the previous solution as an upper bound to the next model improves the solution times significantly. The seed finding algorithm, the *MWGD* model and the reassignment algorithm constitute the proposed algorithm for clustering in this thesis based on the *MWGD* clustering performance criterion.

We next considered the *TWGD* criterion. We formulate the second model also as a *MINLP* model. Rao [23] proposed a solution methodology only for 2-cluser case and again no experimental results were reported in his study. Brusco proposed to solve the model by a branch and bound algorithm but the proposed algorithm is applied only on small data sets. We solve the continuous relaxation of the *SWGD MINLP* model after showing that the model satisfies the total unimodularity property. Thus, we find local optima in the order of seconds in our experiments with up to 500 instances partition up to 5 clusters.

Furthermore, we compare the solution of the proposed algorithm, the *TWGD* model and the *K-Means* algorithm and also derive some interpretations from these solutions. We observe that the interpretability of the solutions of the proposed algorithm and the nonlinear model *TWGD* is better than the interpretability of the solution of the *K-Means* algorithm. We apply hypothesis testing to analyze the solutions achieved by the proposed algorithm, the *TWGD* model and the *K-Mean*s algorithm statistically and show that the solution of the proposed algorithm and the *TWGD* model lead to significantly better solutions than the *K-Means* algorithm.

Chapter 3

THE *MIP-DIAMETER* MODEL AND THE PROPOSED HEURISTIC CLUSTERING ALGORITHMS

3.1 The Proposed *MIP-Diameter* Mathematical Programming Model

In this chapter, a mixed-integer programming model to partition the data set into exclusive clusters is studied. The objective function of the model is to *minimize the maximum diameter* of the generated clusters with the goal of obtaining evenly compact clusters. This criterion was previously studied by other researchers [23], [24]. Rao [23] proposed a very similar integer programming model with the difference that the maximum diameter variable is also set to be an integer. It is stated that the formulation is tractable only for small sized data sets since the number of constraints of the model increases exponentially. A heuristic solution approach for the 2-cluster case is given in the study. Brusco [24] studied the same criterion applied to the integer programming formulation of Klein and Aronson [25] and proposed a branch and bound algorithm. The algorithm utilizes a randomized complete-link clustering algorithm to obtain upper bounds and sub-matrices are solved sequentially to obtain lower bounds. In the study, the computational results seemed to be promising in terms of solution time and closeness to the optimality but no information is provided on the cluster quality of the solutions.

In the model in this study, it is assumed that the number of desired clusters K is known a priori since the determination of the number of clusters constitutes another subject of research in the clustering literature and it is out of the scope of this thesis. However, typically K will be small and the proposed method can be repeated for $k = 1,2,3,..,K$ for some k much smaller than number of instances, if necessary.

As previously stated by Rao [23], the mathematical programming formulation of the minimization of maximum diameter criterion is computationally demanding. Moreover, there exist alternative optimal solutions since the objective function of the model is insensitive to assignments except for the ones that occur in the "largest" cluster. Later, in order to improve the solution quality of the model, we propose a reassignment algorithm in Section 3.2.

The goal of the model is to find the optimal partitioning of the data set into K exclusive clusters given a data set of n data items in $m-$dimensions, i.e. a set of n points in R^m. The parameter d_{ij} denotes the distance between two data points i and j in R^m and can be calculated by any desired norm on R^m such as the Euclidean or the Tchebycheff.

The mathematical formulation of the model, *MIP-Diameter*, is given below.

MIP-Diameter:

Minimize $\quad Z = D_{\max} \qquad (3.1)$

subject to

$$D_k \geq d_{ij} x_{ik} x_{jk} \quad \forall i, j, k, i = 1,.., n, j = 1,.., n, k = 1,.., K \qquad (3.2)$$

$$\sum_{k=1}^{K} x_{ik} = 1 \quad \forall i, i = 1,.., n \qquad (3.3)$$

$$D_{\max} \geq D_k \quad \forall k, k = 1,.., K \qquad (3.4)$$

$$x_{ik} \in \{0,1\} \quad \forall i, k, i = 1,.., n, k = 1,.., K \qquad (3.5)$$

$$D_k \geq 0 \quad \forall k, k = 1,.., K \qquad (3.6)$$

In this model, the variable D_k denotes the diameter of the cluster k and the variable $D_{\max}$ denotes the maximum diameter among the generated clusters (MWCD), which forms the objective function of the model. The variable x_{ik} is a binary decision variable that represents the assignment of an instance to a particular cluster; it takes a value of 1 if the instance i is assigned to cluster k, and 0 otherwise. According to constraint (3.2), the diameter of cluster l is allowed to be at least the maximum distance between any two data points assigned to cluster k. Since the model aims to partition the data points into exclusive clusters, each instance should be assigned to only one cluster as given in equation (3.3). Constraint (3.4), in conjunction with the objective function, helps set variable $D_{\max}$ equal to the value of the maximum diameter. The model has kn binary variables, $k+1$ continuous variables and $O(kn^2)$ constraints.

The *MIP-Diameter* model is a nonconvex bilinear mixed-integer programming model and as such, it is almost impossible to solve this model to optimality in a reasonable amount of time even with small number of instances. Therefore, we linearize the *MIP-Diameter* constraint (3.2) that has a bilinear term without increasing the size of the formulation as follows:

$$D_k \geq d_{ij}(x_{ik} + x_{jk} - 1)$$

$$\forall i, j, k, i = 1,..,n, j = 1,..,n, k = 1,..,K \quad (3.7)$$

Here, if both of decision variables x_{il} and x_{jl} are equal to 1, then the constraint will be active, meaning that data points i and j are assigned to cluster k and the diameter of the cluster has to be at least as long as the distance between them. If one of the decision variables or both of them are equal to zero, the constraint will be redundant.

Although the linear *MIP-Diameter* model outperforms the nonlinear model in terms of CPU time and the number of iterations, the experimental findings given in Section 3.5 show that the computational performance of the model is not comparable with other non-optimization based suggested studies and algorithms in the literature such as the *K-Means* algorithm. Moreover, the experiments on synthetic data sets, where the global optimal solution is known a priori, show that the objective function of the model should be strengthened. Considering this, we develop a reassignment algorithm which we present in Section 3.2.2. The proposed algorithm is developed given in the following sections considering all these facts. In order to improve the solution time, a heuristic approach is developed that is based on solving the model with initial seeds, followed by a reassignment heuristic aimed at improving the cluster quality of the model by incorporating sum of within cluster distance averages as a secondary measure.

3.2 The Proposed Heuristic Clustering Algorithm

The proposed clustering algorithm is based on two approaches in addition to the *MIP-Diameter* model. The proposed seed finding algorithm is developed to achieve a reduction in the solution time of the model without compromising from the near-optimality of the solution and is stated in detail in Section 3.2.1. The reassignment algorithm is developed to improve the solution quality of the *MIP-Diameter* model due to the reason we stated in previous chapter. First the seed finding algorithm is applied on the data set and then the *MIP-Diameter* model initialized by the solution of the seed finding algorithm is solved. Finally the reassignment algorithm is applied to the solution of the model. The flowchart of the algorithm is given below.

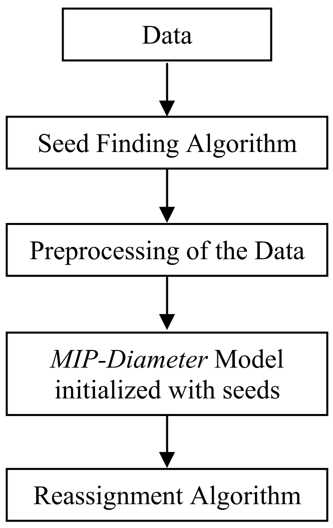
Data
Seed Finding Algorithm
Preprocessing of the Data
MIP-Diameter Model initialized with seeds
Reassignment Algorithm

Figure 3.1: The flowchart of the proposed clustering algorithm

3.2.1 Introducing Seeds

The idea of fixing the assignment of some instances to certain clusters has been used in clustering algorithms before with the goal of improving computational efficiency. These fixed assignments typically improve the computational performance of the algorithm; however, a new question of how to best determine the instances to be fixed, the seeds, is raised. In particular, selecting an initial seed for each cluster in such a way to ensure that the seeds are separated well from each other is desired. To achieve this, we use a graph-theoretic approach which utilizes the concept of a maximal independent set, which has been used in other clustering studies [28]. The proposed seed finding algorithm works as follows. A graph $G_R = (V, E_R)$ is constructed where the set of vertices V corresponds to instances. An edge between two vertices exists in E_R if the distance between the two corresponding instances is less than R, a distance parameter that is initialized as the average of minimum and maximum values of d_{ij} values. Given this graph G_R, an independent set S_R of G_R is a subset of V such that there is no edge in E_R among the members of S_R. A maximal independent set is constructed by starting initially with a randomly picked vertex v from the set V and eliminating the vertices adjacent to v. This step is repeated until no more vertices can be added to S_R. Then, the size of S_R is checked, and this procedure is repeated by adjusting the distance parameter R through a line search and by rebuilding the corresponding graph G_R until the maximal independent set S_R in the current graph consist of K points. Once a maximal independent set with size K is achieved, each vertex in the maximal independent set is assigned to a different cluster for initializing the *MIP-Diameter* model. Although the procedure is sensitive to the selection of the initial vertex and different maximal independent sets can be constructed with different initial vertices, the seed finding algorithm improve CPU times of the *MIP-Diameter* model significantly in our computational tests.

Once the K seeds to be assigned to the K clusters are determined as described above, the *MIP-Diameter* model can be solved with these instances fixed. Obviously, this constitutes a heuristic approach to solving the *MIP-Diameter* model. As it will be reported later, this heuristic approach is capable of obtaining near optimal solutions for the *MIP-Diameter* model with much less computational effort.

3.2.2 The Reassignment Algorithm

To address the problem of poor quality clusters due to the insensitivity of the *MIP-Diameter* model to the assignments that take place in the clusters that do not change the objective function, a reassignment procedure is proposed. The reassignment procedure relies on improving two new criteria, the average of distance between each data point and other data points assigned to the same cluster and the sum of averages of within-cluster distances.

Let C_k denote the set of all instances assigned to cluster k. Then for instance i that belongs to cluster k, the quantity

$$AP_i^k = \sum_{\substack{j \in C_k \\ j \neq i}} d_{ij} / (|C_k| - 1) \qquad (3.8)$$

is computed where $|C_k|$ denotes the number of elements in cluster k.

At each step of the reassignment algorithm, it is hypothesized that an assignment that yields a smaller AP_i^k is better than one with a larger AP_i^k since the instances within the clusters, on average, would be closer to each other. Once all the possible reassignments are checked for all data points, it is hypothesized that the whole reassignment procedure yields a better sum of within-cluster average distances calculated as AT where

$$AT = \sum_{k=1}^{K} \Big(\sum_{i \in C_k} \sum_{\substack{j \in C_k \\ j \neq i}} d_{ij} \Big) / (|C_k|(|C_k| - 1)) \qquad (3.9)$$

which is the sum of the average of within-cluster distances.

The reassignment algorithm works as follows: Given the solution of the *MIP-Diameter* model obtained with fixed seeds, each instance is considered for a possible reassignment different than the one generated by the model. An instance i is selected, and tentatively it is considered for reassignment to each cluster k. The improvement is computed, if any, that takes place in AP_i^k, the average of distances between point i and other points in cluster k, if such a reassignment were to be conducted. In addition, it is checked that such a reassignment would not worsen the objective function of the *MIP-Diameter* model,

$D_{\max}$, beyond an allowed range described by means of a multiplier β which has to be greater than or equal to 1. In other words, the reassignment procedure may allow for a deterioration of $D_{\max}$ to $\beta\, D_{\max}$ in order to achieve a better AT value eventually, if the parameter β is set equal to a value that is strictly greater than 1. The instance under study is reassigned to a cluster that yields the largest improvement in AP_i^k without worsening the maximum diameter beyond the allowed range. Once every instance is considered for reassignment, a second pass is conducted in the same way. The reassignment procedure stops when it is not necessary to reassign instances anymore. Finally, the new AT value is compared with the initial AT value that is obtained from the solution of the *MIP-Diameter* model with fixed seeds and if the new AT is less than the initial one, the reassignment of the data points is accepted. Otherwise, the solution of the *MIP-Diameter* model with the given seeds remains unchanged. This final control checks the AT value in order to eliminate undesired solutions which can be produced by the reassignment algorithm. A formal description of the algorithm is presented as follows.

Step 1: Seed Finding

```
Step 1.0    Set iteration number $t = 0$.
            Set lower and upper bounds to $l(t) = \min\{d_{ij}\}, u(t) = \max\{d_{ij}\}$.
Step 1.1    Set $R(t) = \frac{l(t) + u(t)}{2}$.
Step 1.2    Construct the graph $G_{R(t)}$ where instances correspond to vertices, and
            the edge $(i, j) \in E_{R(t)}$, if $d_{ij} \le R(t)$.
Step 1.3    Find a maximal independent set $S$ in $G_{R(t)}$.
            If $|S| = K$, then go to Step 2.
            If $|S| < K$, then set $l(t+1) = l(t)$, $u(t+1) = R(t)$, $t = t+1$ and
            Go to Step 1.1.
            If $|S| > K$, then set $l(t+1) = R(t)$, $u(t+1) = u(t)$, $t = t+1$.
```

Go to Step 1.1.

Step 2: Linear *MIP-Diameter* Model with Fixed Seeds

Let $S = \{v_1, v_2, \ldots, v_K\}$. Without loss of generality, set $x_{ii} = 1$ for each $v_i \in S$. Solve the *MIP-Diameter* model with these assignments fixed. For $i = 1 \ldots n$, define $C(i) = k$ for k such that $x_{ik}{}^0 = 1$ where $x_{ik}{}^0$ denotes the optimal value of the assignment variable x_{ik} according to the solution of the *MIP-Diameter* model.

Step 3: Reassignment

Step 3.0 Set $x^* = x^0$, where x^0 is the solution obtained in Step 2.

For $k = 1,.., K$, let $C_k(x^*)$ denote the instances assigned to cluster k in assignment matrix x^*. Let $AP(x^*) = \sum_{\substack{j \in C(x^*) \\ i \neq j}} d_{ij} / |C_k(x^*) - 1|$ denote the average distance between instance i and other data points in cluster k, $D_k(x^*) = \max_{i,j \in C_k(x^*)} \{d_{ij}\}$ denote the diameter of cluster k and

Initial_$AT(x^*) = \sum_{k=1}^{k} (\sum_{i \in C_k(x^*)} \sum_{\substack{j \in C_k(x^*) \\ j \neq i}} d_{ij}) / (|C_k(x^*)|(|C_k(x^*)| - 1))$ denote the sum of within cluster averages in solution x^*.

Set $R = \max_{l=1\ldots k} \{D_k(x^*)\}$ and set $\beta \geq 1$.

Step 3.1 Set $i = 1$. Set $update = 0$.

Step 3.2 Set $k = 1$.

Step 3.3 If $x_{ik}^* = 1$ and $k < K$, set $k = k + 1$ and go to Step 3.4.

If $x_{ik}^* = 1$ and $k = K$, set $i = i + 1$ and go to Step 3.5.

Step 3.4 Let x^C denote the solution in which instance i is moved from cluster $C(i)$ to cluster k.

If $AP(x^C) \leq AP(x^*)$ and $D_{\max}(x^C) \leq \beta R$, then

set $C(i) = k$ and $x^* = x^C$, $update = update + 1$.

If $k < K$, set $k = k + 1$ and go to Step 3.3. Otherwise go to Step 3.5.

Step 3.5 If $i < n$, set $i = i + 1$ and go to Step 3.2.

If $i = n$ and $update = 0$,

Set $Final\_AT(x^*) = \sum_{k=1}^{K} (\sum_{i \in C_k} \sum_{\substack{j \in C_k \\ j \neq i}} d_{ij}) / (|C_k(x^*)|(|C_k(x^*)| - 1))$,

Go to Step 3.6.

If $i = n$ and $update > 0$, go to Step 3.1.

Step 3.6 If $Final\_AT(x^*) < Initial\_AT(x^*)$, then STOP.

If $Initial\_AT(x^*) < Final\_AT(x^*)$, then set $x^* = x^0$, i.e. revert to the clustering solution at the end of Step 2.

3.3 Improved Seed Finding

In Section 3.2 of the thesis, the proposed clustering algorithm is stated and a formal description of the algorithm is given. In the next chapter of the thesis, several computational experiments are reported to indicate the efficiency and performance of the proposed algorithm, and the improvement achieved by applying seed finding and reassignment algorithms to the initialization and solution, respectively, of the *MIP-Diameter* model. An alternative seed finding algorithm is developed to improve the robustness of the seed finding step presented in the previous section. The seed finding algorithm proposed in the previous section is named as Seed Finding 1 (*SF*1) and the algorithm proposed in this section is named as Seed Finding 2 (*SF*2) in later parts of the study.

The *SF*1 algorithm has a random nature and it is known that the quality of initial seeds has a direct impact on the final solutions and reported solution times of the algorithm. In order to overcome this issue, alternative maximal independent set finding methods are studied to improve the performance of the proposed algorithm. Among these alternative seed finding algorithms, the most promising one is selected and, now, a detailed explanation of the alternative seed finding approach will be given.

The aim of the alternative seed finding algorithm is to select a set of vertices belonging to a maximal independent set in a way that the selected vertices have the largest number of neighbors around them. The expectation behind this algorithm is to select a set of vertices that will act as natural cluster centers. Recall that in the procedure of $SF1$, a parameter R is set and according to this parameter the graph G_R is constructed. Over this graph, a random selection procedure is applied to form a maximal independent set. The alternative proposed seed finding algorithm $SF2$ proceeds as follows. Consecutively two graphs $G_{R_1} = (V, E_{R_1})$ and $G_{R_2} = (V, E_{R_2})$ are constructed where the set of vertices V corresponds to instances and an edge between two vertices exists in E_{R_1} or E_{R_2} if the distance between the two corresponding instances is less than the corresponding parameter R_1 or R_2 respectively. R_1 and R_2 are distance parameters that are initialized as the average of minimum and maximum values of d_{ij} values existing at the beginning of the algorithm and the same line search procedure given in Section 3.2 applies to the determination of these parameters. Initially, the first graph $G_{R_1} = (V, E)$ is constructed and the number of neighbors of each vertex is counted. Next, the second graph $G_{R_2} = (V, E)$ is constructed according to the parameter R_2 and the vertex having the largest number of neighbors according to the $G_{R_1} = (V, E)$ graph is set as the initial element of the maximal independent set. A maximal independent set is constructed by starting initially with this the selected vertex v and eliminating the vertices adjacent to v, since an independent set S is a subset of V such that there is no edge among the members of S. If there will be a tie, the most populated vertex is selected arbitrarily. This step is repeated until no more vertices can be added to S. Then, the size of S is checked, and this procedure is repeated by adjusting the distance parameter R_2 through line search and by rebuilding the corresponding graph $G_{R_2} = (V, E_{R_2})$ until the maximal independent set S in the current graph consists of k points. If k points cannot be obtained due to the combination of these two parameters R_1 and R_2, the distance parameter R_1 is updated according to the number of seeds found. If the number of seeds obtained is less than the required number, then R_1

is decreased by recalculating its value in order to construct a denser graph. Similarly, if the number of seeds found is more than the required number, then R_1 is increased. The procedure continues until the desired number of seeds is reached.

Similarly, once a maximal independent set with size k is achieved, each vertex in the maximal independent set is assigned to a different cluster to initialize the *MIP-Diameter* model and the rest of the algorithm proceeds as stated in Section 3.2.

A formal description of the alternative seed finding algorithm is presented as follows.

Seed Finding 2 (*SF*2)

Step 1: Construction of the graph $G_{R_1} = (V, E_{R_1})$

```
Step 1.0    Set iteration number to $t_1 = 0$.
            Set lower and upper bounds to $l_1(t_1) = \min\{d_{ij}\}$, $u_1(t_1) = \max\{d_{ij}\}$.
Step 1.1    Set $R_1(t_1) = \frac{l_1(t_1) + u_1(t_1)}{2}$.
Step 1.2    Construct the graph $G_{R_1(t_1)}$ where instances correspond to vertices, and
            the edge $(i, j) \in \mathrm{E}_{R_1(t_1)}$, if $d_{ij} \le R_1(t_1)$.
Step 1.3    For $i = 1...n$, set $neighbor(i) = 0$.
Step 1.4    Set $i = 1$.
Step 1.5    Set $j = 1$.
Step 1.6    If edge $(i, j) \in \mathrm{E}_{R_1(t_1)}$, set $neighbor(i) = neighbor(i) + 1$.
            If $j < n$, then set $j = j + 1$ and repeat this step.
            If $j = n$ and $i < n$, then set $i = i + 1$ and go to Step 1.5.
            Otherwise go to Step 2.
```

Step 2: Construction of the graph $G_{R_2} = (V, E_{R_2})$

```
Step 2.0    Set iteration number to $t_2 = 0$.
            Set lower and upper bounds to $l_2(t_2) = \min\{d_{ij}\}$, $u_2(t_2) = \max\{d_{ij}\}$.
Step 2.1    Set $R_2(t_2) = \frac{l_2(t_2) + u_2(t_2)}{2}$.
```

Step 2.2 Construct the graph $G_{R_2(t_2)}$ where instances correspond to vertices, and the edge $(i,j) \in \mathrm{E}_{R_2(t_2)}$, if $d_{ij} \le R_2(t_2)$.

Step 2.3 Find a maximal independent set S in $G_{R_2(t_2)}$ such that

If $|S| = K$, then STOP.

If $|S| < K$, then set $l(t_2+1) = l(t_2)$, $u(t_2+1) = R(t_2)$, $t_2 = t_2 + 1$ and Go to Step 2.1.

If $|S| > K$, then set $l(t_2+1) = R(t_2)$, $u(t_2+1) = u(t_2)$, $t_2 = t_2 + 1$.

Go to Step 2.1.

If $|S|$ does not converge to k, then set $l(t_1+1) = l(t_1)$, $u(t_1+1) = R(t_1)$, $t_1 = t_1 + 1$ and Go to Step 1.1 or,

then set $l(t_1+1) = R(t_1)$, $u(t_1+1) = u(t_1)$, $t_1 = t_1 + 1$ and Go to Step 1.1.

In our computational experiments, the experiments conducted by applying the two seed finding algorithms *SF*1 and *SF*2 to the proposed clustering algorithm are reported. Both *SF*1 and *SF*2 algorithms produce promising results and show a positive improvement in the solution time of the *MIP-Diameter* model but although the improved solution time of the algorithm is reduced drastically in comparison to its solution without fixing the seeds initially, still they are open to improvements. In Section 3.4 we propose a preprocessing algorithm to reduce the solution time of the model by fixing some of the instances to clusters initially, while preserving the quality of the solution.

3.4 Preprocessing with Inclusion and Exclusion Rules

The computational results of the proposed algorithm in terms of accuracy and performance are promising compared to other mathematical programming clustering models and optimization based algorithms existing in the literature but in order to cope with other heuristic clustering algorithms, in this section of the study, an inclusion and exclusion rule extraction methodology is given.

As mentioned before, the aim of *SF*2 is finding seeds which are possibly close to being natural cluster centers. This idea can be extended to setting inclusion and exclusion rules without compromising from optimality significantly. Actually, this preprocessing methodology is an extension of the seed finding algorithm. The *SF*1 and *SF*2 algorithms produce a set of seeds which are assumed to be sufficiently away from each other and each one may lead to the formation of a cluster. At this point, we may assume these seeds as the gravitational center of the clusters and expect them to keep other points around them in the corresponding cluster. Some instances will be very close to the center and some will be away from it. The instance which is close to the seed of a cluster is expected to be far from the seed of the other clusters. By setting a cutoff value which represents being close to or away from the seeds, a set of instances are initially fixed to the clusters in addition to the seeds or excluded from some clusters which means the so-called instance will not be assigned to this cluster during the execution of the *MIP-Diameter* model.

First, the *SF*1 or *SF*2 algorithms are applied and a set of seeds are determined. For $i = 1,..,n$ and $k = 1,..,K$, let dis_{ik} denote the distance between the instance i and the seed k and for $m = 1,..,K$, and let rr_{ilm} denote the relative ratio

$$rr_{ikm} = \frac{dis_{ik}}{dis_{im}} \tag{3.10}$$

which is the ratio of the distance between instance i and seed k and the distance between instance i and seed m, where $k \neq m$. So, there exists $\binom{K}{2}$ number of relative distance ratios for each instance i.

Let c denote the cut off parameter value used in inclusion and exclusion rules. For instance i, for a fixed seed k and for $m = 1,..,K$, where $k \neq m$, if $rr_{ikm} \leq c$ then instance i is assigned to the cluster of seed k. Similarly, if $rr_{ikm} \geq c$, then instance i is not allowed be assigned to the cluster k by setting the required constraint in the *MIP-Diameter* model. The first rule applies for inclusion of instances to clusters and second rule applies for exclusion of instances from corresponding clusters.

A formal description of the inclusion / exclusion rules is presented as follows.

Inclusion / Exclusion Rules

Step 1: Calculation of the distances between instances and seeds

```
Step 1.0    Set $i = 1$.
Step 1.1    Set $k = 1$.
Step 1.2    Set $dis_{il}$ the distance between the instance $i$ and seed $k$,
            If $k < K$, then $k = k + 1$ and go to Step 1.2,
            If $k = K$ and $i < n$, then $i = i + 1$ and go to Step 1.1.
            Otherwise go to Step 1.3.
Step 1.3    If $i = n$, then go to Step 2.
```

Step 2: Calculation of the relative ratio rr values

```
Step 2.0    Set $i = 1$.
Step 2.1    Set $k = 1$.
Step 2.2    Set $m = 1$.
Step 2.3    Set $rr_{ikm} = \frac{dis_{ik}}{dis_{im}}$ for $m \neq k$.
            If $m < K$, set $m = m + 1$ and go to Step 2.3.
            If $m = K$ and $k < K$, set $k = k + 1$ and go to Step 2.2.
            If $m = K$, $k = K$ and $i < n$, set $i = i + 1$ and go to Step 2.1.
            If $m = K$, $k = K$ and $i = n$, then go to Step 3.
```

Step 3: Inclusion rules

```
Step 3.0    Set $c$ cutoff value.
Step 3.1    Set $i = 1$.
Step 3.2    Set $k = 1$.
Step 3.3    For $m = 1...k$ where $m \neq k$,
            If $rr_{ikm} \leq c$, then set $x_{ik} = 1$,
            If $k < K$, then set $k = k + 1$ and go to Step 3.3.
            If $k = K$ and $i < n$, then set $i = i + 1$ and go to Step 3.2.
            If $k = K$ and $i = n$, then go to Step 4.
Step 4: Exclusion rules
    Step 4.0    Set $c$ cutoff value.
    Step 4.1    Set $i = 1$.
    Step 4.2    Set $k = 1$.
    Step 4.3    For $m = 1,..,K$ where $m \neq k$,
                If $rr_{ikm} \geq c$, then set $x_{ik} = 0$,
                If $k < K$, then set $k = k + 1$ and go to Step 4.3.
                If $k = K$ and $i < n$, then set $i = i + 1$ and go to Step 4.2.
                If $k = K$ and $i = n$, then STOP.
```

In the next section, the proposed model and algorithms are applied to test their performance on a synthetic data set.

3.5 Experiments on an Illustrative Example

In this part of the thesis, the proposed mathematical programming *MIP-Diameter* model and the variations of the heuristic clustering algorithms are applied on a set of 81 data points shown in Figure 3.2. For the purpose of illustration, data points are represented in a 2-dimensional space and the distance between any two points is calculated by the Euclidean distance measure.

Experiments are performed on this synthetic data set, which consists of four distinct clusters to show how *MIP-Diameter* model may fail to reach an acceptable solution, and to evaluate the accuracy and the performance of the proposed clustering algorithms and as well as comparing the results of algorithms with the results of the well-known *K-Means* algorithm. Also, the proposed inclusion/ exclusion rules are applied on the data set to analyze their efficiency and their impact on the solution quality of the *MIP-Diameter* model and the proposed clustering algorithm in this thesis.

3.5.1 Experiments with the *MIP-Diameter* Model

For the data set given in Figure 3.2, the minimum value of the maximum diameter of the generated clusters is 8, since the horizontal and vertical distance between any two neighbor data point is equal to 1. The optimal solution to the linearized *MIP-Diameter* model is obtained using the CPLEX 8.1 solver in approximately three hours of CPU time on a PC with Pentium IV 3.06 GHz processor, 512 MB memory. There are 524,954 branch & bound nodes enumerated with respect to the 329 variables and 13,045 constraints for this illustrative example. The maximum diameter is found as 8. These computational findings indicate the difficulty of solving the proposed *MIP-Diameter* model exactly.

In Figure 3.3, the optimal solution of the model is illustrated on the data set and it is seen that there are 10 data points that could possibly be assigned better. These assignments are due to the fact that *MIP-Diameter* model considers only the assignments of instances which set the maximum diameter value and it is insensitive to other assignments being made. It is observed that similar instances are assigned to different clusters which may lead to a biased interpretation of the solution. These assignments do not deteriorate the minimum value of $D_{\max}$ but result in an increase in the value of AT value. Thus, the proposed reassignment algorithm is developed with the purpose of fixing these distorted assignments.

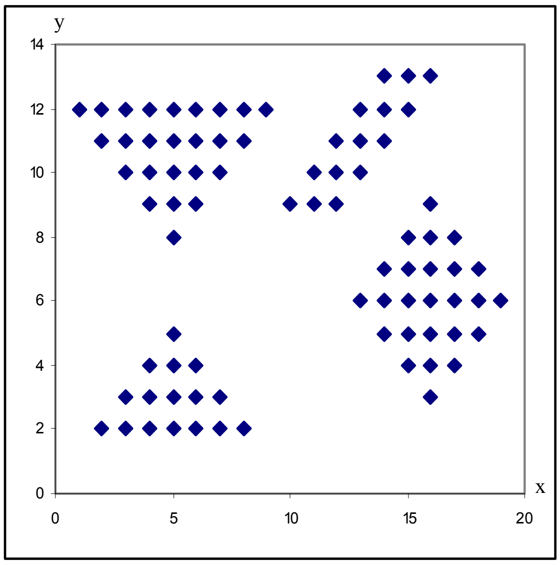

Figure 3.2: Data points and their graphical representation

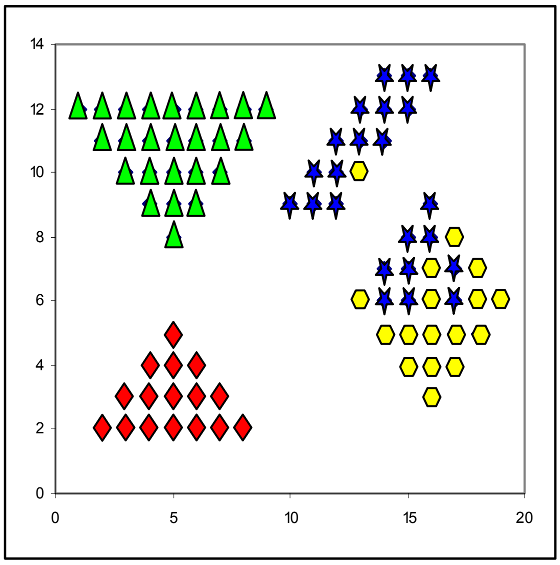

Figure 3.3: The solution of the *MIP-Diameter* model without seeds

3.5.2 Experiments with *SF*1

Next, the *MIP-Diameter* model initialized by the *SF*1 algorithm is applied on the same data set. A set of experiments are reported in order to observe accuracy, efficiency and performance of the *SF*1 and the effect of randomness of the algorithm on these parameters. In these experiments, initially, the algorithm starts by the seed finding algorithm *SF*1 to determine 4 seeds and then the *MIP-Diameter* model is solved with these seeds. The experiments are given in Table 3.1. It is seen that the *MIP-Diameter* model solved with seeds also reaches the optimal solution in all of the experiments as maximum diameter value is again found to be 8. However, by applying the seed finding algorithm the running time of the model is decreased from almost three hours to 4.08 to 13.41 seconds while preserving the optimality of the resulting solution. The achievement in the solution time of the model is promising but further investigation on the solution quality of the model should be done. In Figures 3.4 and 3.5, a good solution and a bad solution example of the model are illustrated to show the possible outcomes of the model initiated by the *SF*1.

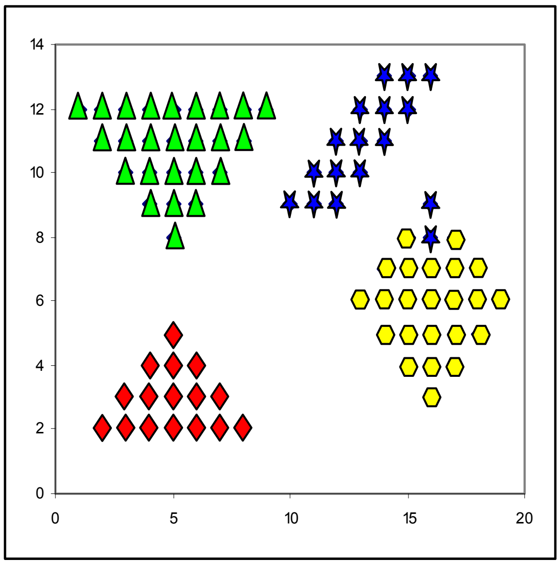

Figure 3.4: A solution of the *MIP-Diameter* model with seeds found by *SF*1

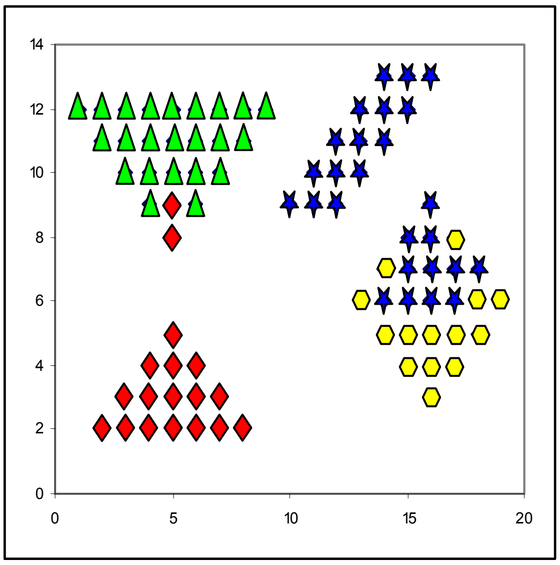

Figure 3.5: A possible alternative solution of the *MIP-Diameter* model with seeds found by *SF*1

As expected, when one examines Figure 3.4 and Figure 3.5, it is seen that there are still assignments of data points that are open to improvement. Figure 3.4 indicates that good initial seeds may lead to a successful assignment pattern. In this figure, there exist only 2 distorted assignments while preserving the optimality of the model and the reported solution time for this experiment is only 4.08 seconds. But, Figure 3.5 is shown to indicate the effect of randomness of the *SF*1 algorithm on the solution quality of the *MIP-Diameter* model. In this solution, there are 13 distorted assignments and the reported solution time is 11.98 seconds. The computational findings of these two solutions, in experiments 1 and 4 respectively, and alternative solutions of the *SF*1 algorithm applied to *MIP-Diameter* model are given in Table 3.1.

In below Table 3.1, the results of different solutions are given in order to observe the effect of randomness of the *SF*1 algorithm on the solution quality of the *MIP-Diameter* model. The number of variables is 329 and the number of constraints is 13,049 in all experiments reported in below table. Here, *Iter.* denotes the number of iterations the model executed and *Node* denotes the number of branch & bound nodes enumerated by the model. D_{max} is the optimal solution found and *Distorted* denotes the number of distorted assignments in the final solution of the model initialized by *SF*1 algorithm. In the last two columns, $t(\text{sec})$ denotes CPU time and *Solv (MB)* denotes the size of required solver memory in megabytes.

Exp. Number	Iter.	Node	D_{max}	Distorted	t(sec)	Solv. (MB)
1	859	0	8	2	4.08	6.12
2	855	11	8	9	7.17	6.05
3	1135	37	8	11	8.78	6.05
4	1267	56	8	13	11.98	6.05
5	1350	53	8	8	13.41	6.12

Table 3.1: Computational results for *SF*1 algorithm

The preservation of the optimal solution of the model is a good achievement of the *SF*1 algorithm but due to the random nature of the algorithm, several set of seeds can be determined and there is no guarantee on the selection of seeds which may end up with a meaningful solution. In later parts of the study, the findings from the real data set strengthen this hypothesis with respect to the *SF*1 algorithm.

3.5.3 Experiments with *SF2*

In this section of the study, the *MIP-Diameter* model initialized by the *SF*2 algorithm is applied on the illustrative data first to analyze the efficiency of the algorithm and then to compare the performance of the two seed finding algorithms and their impact on the *MIP-Diameter* model. Several solutions of *MIP-Diameter* model initialized by *SF*2 algorithm are reported in below Table 3.2. The solutions of these experiments are also the exact solution for the minimization of maximum diameter criterion. The running times of the experiments are from 9.45 seconds to 17.11 seconds. In Figure 3.6, an assignment pattern of the *MIP-Diameter* model initialized with seeds found by *SF*2 algorithm is shown.

Exp. Number	Iter.	Node	D_{max}	Distorted	t(sec)	Solv. (MB)
6	1331	3	8	7	9.45	6.05
7	1562	81	8	6	11.26	6.05
8	1649	80	8	8	12.14	6.05
9	2030	105	8	4	14.94	6.05
10	2269	117	8	4	17.11	6.05

Table 3.2: Computational results for SF2 algorithm

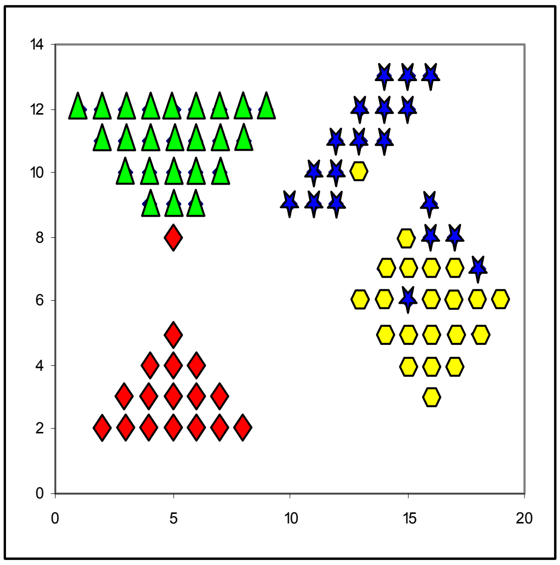

Figure 3.6: The solution of the *MIP-Diameter* model with seeds found by the *SF2* algorithm

Here we observe consistency and less variation among experiments. Note that *SF*2 is a deterministic algorithm which is a derandomized version of *SF1*. By derandomization, the run times increase slightly, the solution quality stays the same and most importantly robust solutions are obtained.

3.5.4 Experiments with Proposed Heuristic Clustering Algorithm and Its Comparison with *K-Means*

In this section of the illustrative example part of the study, the proposed seed finding algorithm is applied on the solution of the *MIP-Diameter* model initialized by two seed finding algorithms where the value of parameter β equals to 1. The 10 experiments reported in Tables 3.1 and 3.2 initialized by *SF*1 and *SF*2 algorithms converge to the same solution which is shown below in Figure 3.7 after the reassignment algorithm is applied to the solution of the *MIP-Diameter* model with seeds. In this solution, only one data point is assigned incorrectly. The CPU time of the reassignment algorithm on this data set is minimal, thus we ignore its computation time in this part of the study.

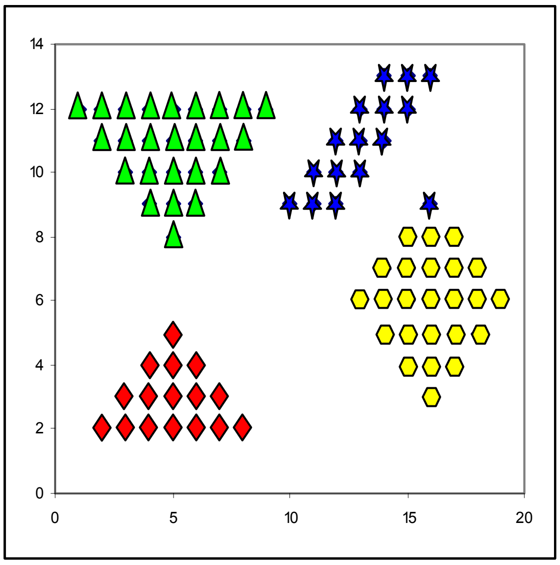

Figure 3.7: The solution of the proposed algorithm

Next, the solution of the proposed algorithm is compared with the solution of the well-known *K-Means* algorithm according to the measure AT, the sum of average distances of the generated clusters. In below Figure 3.8, the assignment pattern of the *K-Means* algorithm is illustrated and in Table 3.3 the numerical values for the exact solution, the *MIP-Diameter* model's solution, the *K-Means* algorithm's solution and the proposed algorithm's solution are given. Based on the values given in Table 3.3, the proposed algorithm solution is only 0.2% worse than the optimal solution of the *MIP-Diameter* model with respect to the sum of averages of within-cluster distances which aims to generate compact clusters. However, the defined clustering measures indicate that the solution of the *K-Means* algorithm is not close to the optimal partitioning of the data set.

In order to compare the two algorithms to the *K-Means* algorithm, we also discuss the quality of the solutions. The solution quality of the *K-Means* algorithm on this illustrative data set as seen from Figure 3.8 is not favorable. The clusters formed by *K-Means* algorithm are different from the original clusters existing in the data set. One of the resulting clusters generated by *K-Means* algorithm is the combination of two different clusters originally. The numerical observations constitute an important step in the clustering analysis. But another important expectation from clustering algorithms is the interpretability of the resulting solutions. In this example we show that *K-Means* algorithm may lead to a skewed interpretation of the resulting solution although the AT value of the solution of this algorithm seems comparable to those of the other algorithms.

Solution	AT	D_{max}
Optimal	10.8766	8
MIP-Diameter model with seeds	12.3127	8
K-Means algorithm	11.3838	10.05
Proposed algorithm	10.8988	8

Table 3.3: The comparison of results based on the sum of averages of within-cluster distances and $D_{\max}$

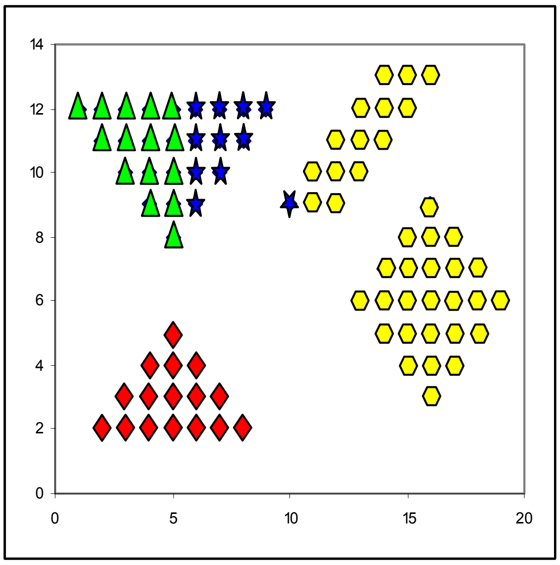

Figure 3.8: The solution of the *K-Means* algorithm

3.5.5 Effect of Inclusions and Exclusions on the Performance of the Proposed Clustering Algorithm

In this section, experimental results for the Inclusion/ Exclusion rules applied to the *MIP-Diameter* model is reported in order to analyze the efficiency of the proposed rules and their impact on the solution quality of the model. In Table 3.4, the experimental results are reported to observe the effect of the Inclusion/ Exclusion rules on the *MIP-Diameter* model solution. These experiments are conducted as follows. First, the *SF2* algorithm is applied on the data set and 4 seeds are selected, then these seeds are given as input to Inclusion/ Exclusion rule derivation algorithm and the value of the cutoff parameter c is set to a value depending on the data set and the type of the rule one prefers to apply on the model. The instances to be assigned to certain clusters are determined according to the value of c. Then, *MIP-Diameter* model is solved with these initial assignment and finally the reassignment algorithm is applied on the solution of the *MIP-Diameter* model in order to correct possible deterioration resulted by inclusion/ exclusion rules and to improve the solution quality of the *MIP-Diameter* model.

In experiments performed on these rules, we observe that the effect of including an instance has a bigger impact than the exclusion of an instance from a cluster. When the variable x_{ik} denoting the assignment of instance i to cluster k is fixed to 1, from constraint (3.3) the rest of x_{ik} for each k is fixed to 0. This eliminates K decision variables from the problem. When it is fixed to 0, only one of the decision variables is fixed and, $K-1$ decision variables for instance i are still free. For this reason, first the experiments conducted with inclusion rules are reported in Table 3.4 to analyze their impact on the solution of the model. In Table 3.4, the newly introduced expression $\%D_{\max}$ denotes the difference, in percentage, between the exact solution of the model and the optimal solution achieved by the model initialized by inclusions. Final $D_{\max}$ value denotes the value of the maximum diameter and also *Final AT* denotes the value of sum of averages within clusters occurring in the solution of the reassignment algorithm.

In Table 3.4, it is shown that when the value of c increases, there are more number of instances to be fixed to clusters initially. Until c takes the value of 0.4, the optimal solution of the model is equal to the exact solution of the model. If we increase the c value more than this level, $D_{\max}$ starts to deteriorate but we see that the reassignment algorithm fixes this deterioration and reduces the value of the AT value to the best possible value achievable by the proposed clustering algorithm which is previously reported as the solution of the clustering algorithm in Table 3.3. The inclusion/ exclusion rules are proposed in order to decrease the solution time of the *MIP-Diameter* model while preserving the accuracy of the solution to a possible extent. The solution time of the model is decreased to 0.25 CPU seconds while the solution of the model is still reported to be optimal according to the minimization of the maximum diameter criterion and only 0.2% worse than the optimal solution of the *MIP-Diameter* model with respect to the sum of averages of within-cluster distances.

Exp. Number	c	Number of Inclusions	Iter.	Node	$D_{\max}$	% $D_{\max}$	Final $D_{\max}$	Final AT	t (sec)
11	-	4	1331	3	8	-	8	10. 8988	9.45
12	0.2	16	1123	64	8	-	8	10. 8988	5.89
13	0.3	25	235	1	8	-	8	10. 8988	1.16
14	0.4	33	283	3	8	-	8	10. 8988	1.08
15	0.5	41	108	0	8.06	0.775	8	10. 8988	0.34
16	0.6	50	44	0	9.22	15.25	8	10. 8988	0.3
17	0.7	58	28	0	10.05	25.625	8	10. 8988	0.26
18	0.8	66	11	0	10.05	25.625	8	10. 8988	0.28
19	0.9	73	6	0	10.05	25.625	8	10. 8988	0.25

Table 3.4: The inclusion experiments for different values of c cutoff value

Finally, we study the impact of exclusions on the performance and solution quality of the *MIP-Diameter* model. To observe this, the c value for inclusion is set to be equal to 0.3, which preserves the exactness of the solution of the model on the illustrative data set. Experiments are performed on this setting with different values of c value for exclusion. Exclusion rules work as the opposite of the inclusion rules. A higher c value for exclusion means that a smaller number of instances will be excluded from clusters because as the c

value increases in exclusion, the further instances to seeds are excluded from corresponding clusters. In Table 3.5, the experimental results are given for this set of experiments.

From Table 3.5, it is seen that the exact solution for the illustrative data set is achieved for all c values in the exclusion rule. Also it is seen that the exclusions has a positive impact on the solution time of the model but when compared with inclusions, as stated previously their effect is less.

Exp. Number	c	Number of Exclusions	Iter.	Node	D_{max}	% D_{max}	t (sec)
11	1	81	201	2	8	-	1.03
12	1.1	70	207	2	8	-	0.76
13	1.2	65	217	2	8	-	0.81
14	1.3	50	255	2	8	-	0.92
15	1.4	40	257	1	8	-	0.94
15	1.5	29	232	1	8	-	1.05

Table 3.5: The exclusion experiments for different values of c cutoff value

Chapter 4

EVALUATION OF THE PROPOSED *MIP-DIAMETER* MODEL AND HEURISTIC CLUSTERING ALGORITHM ON A REAL DATA SET

In this part of the thesis, the performance and the accuracy of the *MIP-Diameter* model and the proposed heuristic clustering algorithm are examined on a real data set. In later parts of this section, results of various computational experiments carried on data sets drawn from the Digiturk database are reported.

In this thesis, real data from a satellite broadcasting company, Digiturk, is used in our computational experiments. The company, founded in 1999, is a private digital platform operating in Turkey. The firm has around 800,000 customers and provides five product packages, three pay-per-view services and also various channels, interactive channels and events to its customers. Digiturk is eager to find out the opportunities in customer relationship marketing, such as one–to–one marketing. The company would like to segment its customers based on the transactional factors, such as their package subscriptions, pay-per-view purchases, and interactive event interests. Also, in Chapter 6 of the study an interpretation of the customer segments obtained by the clustering algorithm applied to Digiturk data is given.

In this section of the study, the *MIP-Diameter* model is solved by using OPL Studio version 3.6.1 [29], CPLEX version 8.1 and seed finding, reassignment and *K-Means* algorithms are implemented using MATLAB 7.0 [30]. The experiments are performed on a notebook with 256 MB memory and 1.5 GHz speed Pentium (R) M processor running under Windows XP Professional.

4.1 Experiments with the *MIP-Diameter* Model

On the real data set consisting of 23 attributes and 2000 instances several sets of experiments are performed in order to analyze the performance and the accuracy of the proposed *MIP-Diameter* model. We selected 100, 200, 300, 400 and 500 instances out of 2000 instances to perform our experiments.

First, we performed a set of experiments on the linearized *MIP-Diameter* model without seeds to compare its results with the one initiated with seeds. One can observe the positive effect of the seed finding algorithm on the solution of the *MIP-Diameter* model, in terms of improvement in the number of iterations, the number of nodes and the CPU time required to partition the data set into 2 clusters by examining the results given in Table 4.1 and Figure 4.1. From Figure 4.1, one can see that the solution time of the *MIP-Diameter* model without seeds increases exponentially and if the model is solved with seeds, it is seen that the solution time reduces drastically.

In Table 4.1, k is the number of clusters, N is the number of instances, $Var.$ is the number of variables and $Cons.$ is the number of constraints in the model, $Iter.$ is the number of iterations the model performs and $Solv.(MB)$ is the required memory size, $t(\text{sec})$ denotes CPU time and $D_{\max}$ is the objective function value of the *MIP-Diameter* model.

					Without Seeds					With Seeds				
Exp. Number	k	N	Var.	Cons.	Dmax	Iter.	Node	t(sec)	Solv. (MB)	Dmax	Iter.	Node	t(sec)	Solv.(MB)
1	2	100	203	10002	10.8	3615	42	**8.92**	4.65	10.8	430	11	**1.84**	4.66
2	2	200	403	40002	10.8	14295	79	**149.03**	17.97	10.8	1257	31	**22.38**	17.97
3	2	300	603	90002	10.8	35943	251	**1895.45**	39.74	10.8	1761	39	**70.61**	39.74
4	2	400	803	160002	10.8	45988	122	**3787.94**	70.23	10.8	2346	55	**292.94**	70.23
5	2	500	1003	250002	10.8	50609	116	**8129.77**	100.95	10.8	2343	34	**893.34**	100.95

Table 4.1: A comparison of *MIP-Diameter* model with *MIP-Diameter* with seeds

It is seen in Table 4.1 that CPU times, number of nodes and iterations decrease significantly when seeds are used while the optimal solution of the model is preserved. We performed several experiments with varying size of data sets and number of clusters and from all those experiments the positive effect of the seed finding algorithm is approved. While partitioning the same data set with 100 instances into 4 clusters *without seeds* takes more than 2 weeks of time, by applying the seed finding algorithm we partition the same data set into 4 clusters only in 33.2 seconds.

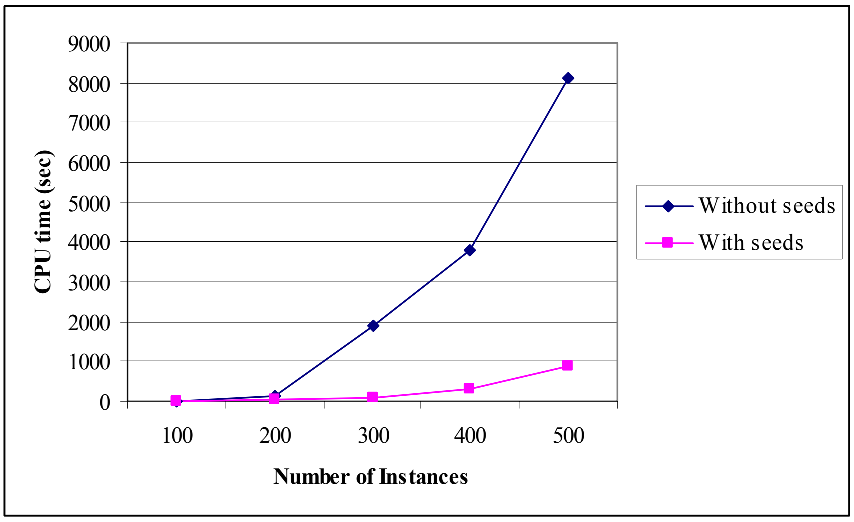

Figure 4.1: The effect of the seed finding algorithm on the solution time of the *MIP-Diameter* model

The positive effect of feeding seeds to the *MIP-Diameter* model is observed experimentally in above Table 4.1 and Figure 4.1. In the 2 cluster case, the optimality of the model is preserved while achieving an improvement in solution time of the model. In many clustering algorithms, the 2 cluster case is mentioned as a special case and in this proposed algorithm, selection of the 2 seeds for 2 clusters is done by determining the instances which have the longest distance between each other. Thus, the preservation of the optimality is an expected result for the 2 cluster case.

4.2 Comparison of the Performance of *SF*1 and *SF*2 Algorithms on the Real Data Set

In Chapter 3, we proposed two seed finding algorithms and later in the same chapter, we reported a set of experiments on an illustrative data set in order to show the performance of the proposed *SF*1 and *SF*2 algorithms. As discussed earlier, *SF*1 algorithm picks a randomly selected node to form a maximal independent set and each time the algorithm proceeds, it may produce a different seed set. In most cases tested, the performance and accuracy of the algorithm was seen to be promising and the results reported were optimal or very close to the optimal solution. But, giving initially seeds to algorithms is a widely studied area and it has been proven that the quality of the seeds directly affects the quality of the final solution and the performance of the solution algorithm. The *SF*2 algorithm is designed with the purpose of reducing the effect of the randomness that exists in the *SF*1 algorithm. The *SF*2 algorithm proceeds by selecting the first element of the candidate independent set according to the popularity of the nodes which is defined on a parametric graph. The rest of the algorithm proceeds the same way as *SF*1 does. Thus, the *SF*2 algorithm is expected to be more stable than the *SF*1 algorithm. In Tables 4.2, 4.3, 4.4 and 4.5 the experimental results of the *SF*1 and *SF*2 algorithms performed on the real data set of 100 instances partitioned into 3 and 4 clusters are reported respectively.

Exp. Number	k	N	Var.	Cons.	D_{max}	Iter.	Node	t(sec)	Solv. (MB)
6	3	100	304	14956	**9.8**	1103	110	9.8	6.98
7	3	100	304	14956	8.9	1866	169	14.09	6.97
8	3	100	304	14956	**9.4**	4666	344	17.5	6.97
9	3	100	304	14956	8.9	3251	304	17.8	6.97
10	3	100	304	14956	8.9	7471	749	25.61	6.97

Table 4.2: *SF*1 Algorithm applied to the *MIP-Diameter* model on the data set of 100 instances for 3 cluster solution

Exp. Number	k	N	Var.	Cons.	D_{max}	Iter.	Node	t(sec)	Solv. (MB)
11	3	100	304	14956	8.9	1899	247	14.14	6.97
12	3	100	304	14956	8.9	2892	285	18.17	6.90
13	3	100	304	14956	8.9	3467	377	18.55	6.97
14	3	100	304	14956	8.9	6659	433	23.19	6.98
15	3	100	304	14956	8.9	6053	395	24.27	6.90

Table 4.3: *SF*2 Algorithm applied to the *MIP-Diameter* model on the data set of 100 instances for 3 cluster solution

Exp. Number	k	N	Var.	Cons.	D_{max}	Iter.	Node	t(sec)	Solv. (MB)
16	4	100	405	19908	8.5	4535	686	33.2	9.27
17	4	100	405	19908	**8.8**	3417	647	33.47	9.17
18	4	100	405	19908	8.5	16195	2906	70.88	9.17
19	4	100	405	19908	**8.8**	14874	4163	95.34	9.17
20	4	100	405	19908	8.5	77334	20262	379.33	9.27

Table 4.4: *SF1* Algorithm applied to the *MIP-Diameter* model on the data set of 100 instances for 4 cluster solution

Exp. Number	k	N	Var.	Cons.	D_{max}	Iter.	Node	t(sec)	Solv. (MB)
21	4	100	405	19908	8.5	6661	678	35.09	9.16
22	4	100	405	19908	8.5	6035	1042	38.59	9.17
23	4	100	405	19908	8.5	11683	3249	76.33	9.17
24	4	100	405	19908	8.5	18940	7888	138.69	9.17
25	4	100	405	19908	8.5	33132	6799	142.34	9.17

Table 4.5: *SF*2 Algorithm applied to the *MIP-Diameter* model on the data set of 100 instances for 4 cluster solution

The objective function value of the *MIP-Diameter* model applied on the 100 instance data set without seeds for 3 clusters is found to be 8.9. As previously stated, the same data set could not be partitioned into 4 clusters in more than two weeks of time but based on our experiments with seeds we expect that the value of the optimal solution for the 4 cluster solution is 8.5. From above tables, it is seen that *SF*2 algorithm leads to better solutions than the $SF1$ algorithm. In Table 4.2, two of the experiments are reported to be suboptimal for the 3 cluster solution case and in Table 4.4 two of the experiments are reported to be suboptimal for the 4 cluster solution case. None of the experiments conducted with the $SF2$ algorithm is reported to be suboptimal which constitutes an advantage over the $SF1$ algorithm indicating its robustness. It would not be favorable to compare the solutions times of the models initialized by alternative seed finding algorithms since both algorithms involve a random part in their structure. However we can conclude that compared with the solution times more than two weeks of time, all of the reported solution times are very promising.

4.3 Experiments with the Proposed Heuristic Clustering Algorithm

In Tables 4.6, 4.7 and 4.8, the experimental findings with the proposed algorithm are given with the goal of illustrating the benefit of the reassignment procedure. For this part of the study, data sets with 100, 200 and 300 instances are partitioned into two to five clusters. In addition to the explanations in Section 4.1, the term AT in the below tables is the sum of averages of within-cluster distances.

In these experiments, we apply a property of the proposed *MIP-Diameter* model.

Property 4.1: The objective function value of the k cluster solution can be used as an upper bound in the $k+1$ solution.

Proof: The proof of this property follows easily from the following observation. In the *MIP-Diameter* model, the optimal solution of the k cluster model is feasible to the $k+1$ cluster model. Therefore the objective function value of the $k+1$ model is expected to be at least as good as the objective function value of the k cluster model which means that the objective function value of the k cluster model can be used as an upper bound for the $k+1$ cluster solution. □

Based on the above property, the experiments reported in Tables 4.6, 4.7 and 4.8 are conducted. CPU Times reported in the below tables are achieved by using the upper bound constraint. For this data set, the solution of the two cluster case has almost no impact on the solution time of the three cluster case since the gap between the two objective function values is large but it is seen that the improvement in solution times is promising for the increasing number of cluster as the gap between the values of the $D_{\max}$ values is small. The partitioning of the 300 instance data set into five clusters takes more than 1,000 second for several seed sets however by applying the stated property in above, the solution is achieved in only 10.72 seconds. By applying this property, we may partition the data set into 5 cluster recursively. In the below tables, the solution times are not reported as aggregate since we want to emphasize the positive effect of the Property 4.1 on the solution efficiency of the model. Due to this fact aggregate solution times also should be considered. The aggregate solution time of the 5-cluster solution on 300 instance data set is actually 1,004 seconds. When we compare this value with the above stated value, one can state there is not a significant achievement due to this property. But when we solve the 4-cluster solution, the benefits we achieve by applying this property are significant since the solution time of the proposed algorithm for 4-cluster case is more 1,000 seconds. When we apply this property, it reduces into only 53.72 seconds for 300 instance data set as seen from Experiment 36 in Table 4.7.

In all of these experiments, the coefficient β is taken as 1. We note that the sum of average within cluster distances almost always improves with reassignment and the improvement is more significant as the number of clusters increases. One can observe these findings from Figures 4.3, 4,4 and 4.5.

				MIP-Diameter						Proposed Algorithm						
Exp. Number	k	Var.	Cons.	Iter.	Node	Dmax	AT	t(sec)	Solv. (MB)	RIter.	Dmax	AT	%Dmax	%AT	t(sec)	Total t(sec)
26	2	203	10002	430	11	10.8	13.228	**1.84**	4.66	3	10.8	12.4425	**0.00**	**0.06**	**0.34**	**2.18**
27	3	304	14956	1899	247	8.9	17.791	**14.14**	6.97	4	8.9	17.6075	**0.00**	**0.01**	**0.61**	**14.75**
28	4	405	19909	31	0	8.5	23.273	**0.36**	9.17	4	8.5	22.3050	**0.00**	**0.04**	**0.81**	**1.17**
29	5	506	24861	180	0	8.4	29.566	**1.36**	11.48	10	8.4	27.3717	**0.00**	**0.08**	**1.92**	**3.28**

Table 4.6: The experimental results of *Proposed Algorithm* on the 100 instance data set

				MIP-Diameter						Proposed Algorithm						
Exp. Number	k	Var.	Cons.	Iter.	Node	Dmax	AT	t(sec)	Solv. (MB)	RIter.	Dmax	AT	%Dmax	%AT	t(sec)	Total t(sec)
30	2	403	40004	1257	31	10.8	13.199	**24.38**	17.97	4	10.8	12.387	**0.00**	**0.06**	**2.00**	**26.38**
31	3	604	59906	6131	505	8.9	18.111	**172.81**	26.72	3	8.9	17.4138	**0.00**	**0.04**	**2.90**	**175.71**
32	4	805	79809	181	0	8.5	22.704	**1.97**	35.44	3	8.5	21.974	**0.00**	**0.03**	**3.84**	**5.81**
33	5	1006	99711	-	-	8.4	27.073	**1.81**	44.18	3	8.4	26.724	**0.00**	**0.01**	**4.63**	**6.44**

Table 4.7: The experimental results of *Proposed Algorithm* on the 200 instance data set

				MIP-Diameter						Proposed Algorithm						
Exp. Number	k	Var.	Cons.	Iter.	Node	Dmax	AT	t(sec)	Solv. (MB)	RIter.	Dmax	AT	%Dmax	%AT	t(sec)	Total t(sec)
34	2	603	90004	1761	39	10.8	13.008	**75.75**	39.73	4	10.8	12.3663	**0.00**	**0.05**	**5.95**	**81.70**
35	3	904	134857	16607	954	8.9	17.841	**874.58**	59.35	3	8.9	17.3427	**0.00**	**0.03**	**8.77**	**883.35**
36	4	1205	179709	1602	19	8.5	23.571	**53.72**	78.95	5	8.5	21.8446	**0.00**	**0.07**	**17.34**	**71.06**
37	5	1506	224561	-	-	8.4	26.638	**10.72**	98.54	6	8.4	26.5557	**0.00**	**0.00**	**25.48**	**36.20**

Table 4.8: The experimental results of *Proposed Algorithm* on the 300 instance data set

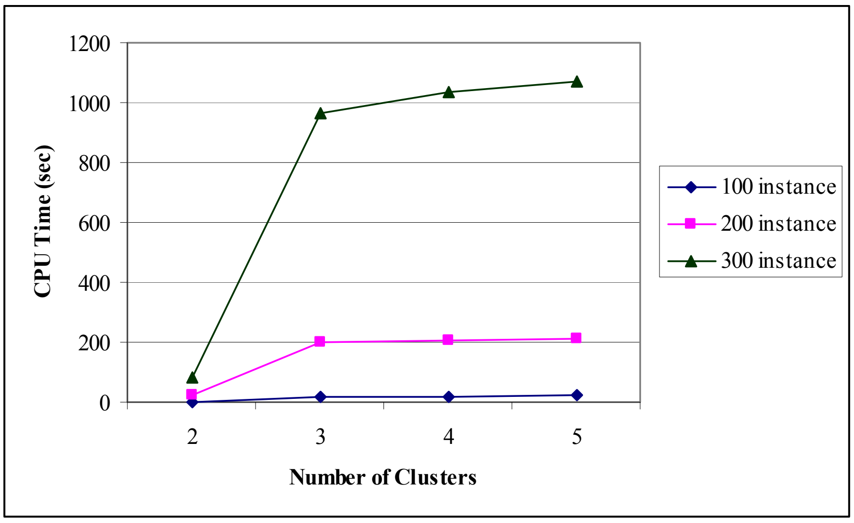

Figure 4.2: The performance of the algorithm in terms of CPU Time according to the number of instances and the number of clusters

In above Figure 4.2, one can observe the positive effect of the Property 4.1 on the performance of the proposed algorithm in addition to the seed finding algorithm. As a remark, the solution time of the 3-cluster case for all data sets is long compared to the 4 and more cluster cases because the gap between the values of the 2-cluster and 3-cluster solution is big. Therefore, the derived upper bound for 3-cluster case has no impact on the solution time of the algorithm. But as the number of clusters is increased, this gap gets smaller and the solution time of the algorithm improves dramatically. Moreover, the *MIP-Diameter* model loses its exponential solution time property.

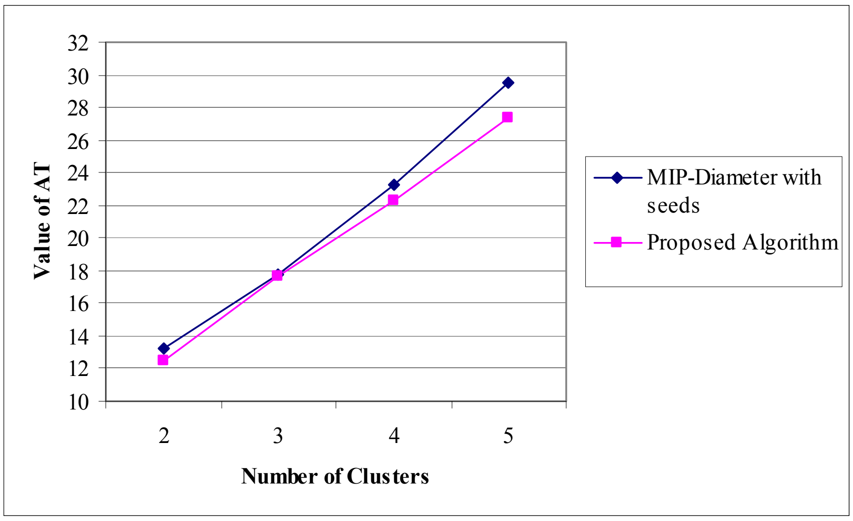

Figure 4.3: The effect of the proposed algorithm on the value of *AT* on the 100 instance data set

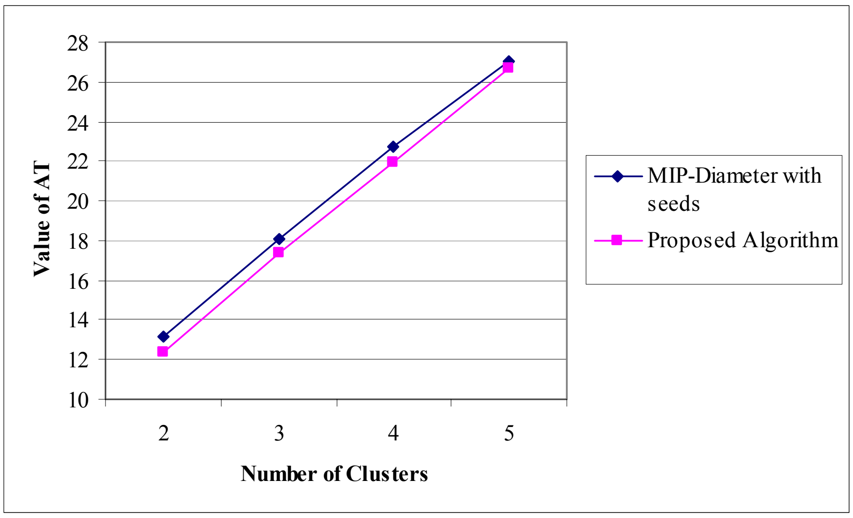

Figure 4.4: The effect of the proposed algorithm on the value of *AT* on the 200 instance data set

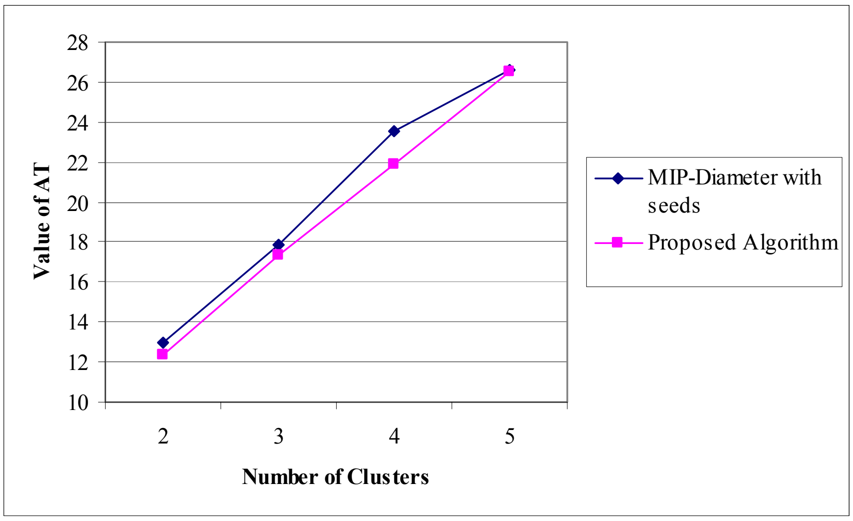

Figure 4.5: The effect of the proposed algorithm on the value of *AT* on the 300 instance data set

4.4 Experiments with Inclusion / Exclusion Preprocessing Rules

In Tables 4.9, 4.10, 4.11, 4.12, 4.13 and 4.14 the experimental results with the inclusion/ exclusion preprocessing rules applied to the proposed algorithm with the purpose of observing the benefits achieved are given. Data sets with 100, 200 and 300 instances are partitioned into two to five clusters for this part of the study. The tables are organized as follows: the first table for each data set shows the computational values in terms of number of iterations, number of nodes and solution time required and the value of $D_{\max}$ and AT criteria achieved by the *MIP-Diameter* model initialized solely with seeds and *MIP-Diameter* model initialized with included and excluded instances in order to observe the improvement achieved by the proposed inclusion/ exclusion rules. The second table reports the results of the reassignment algorithm applied to the solutions reported in first tables. Therefore, the proceeding two tables for each data set should be analyzed together to observe the improvements.

All experiments reported in this section are performed as follows: seeds are determined to partition the data set into two to five clusters by applying the *SF*2 algorithm. The *MIP-Diameter* model is solved as initialized by these seeds and the objective function value of each preceding solution is used as an upper bound in the solution of the consecutive experiment. The experiments with inclusions and exclusions use the same set of seeds to determine which instances will be assigned to certain clusters and which instances will be excluded from certain clusters. Next, the *MIP-Diameter* model is solved following the same procedure. We note that one should consider the reported solution times in these experiments as aggregate times as we stated in Section 4.3 of the thesis. Next, the reassignment algorithm is applied on both solutions for the same data set with equal number of clusters. The c value for inclusion is set to 0.7 and c value for exclusion is set to 1.5 in all experiments reported in this section in order to analyze possible outcomes of proposed methodology to improve the solution time of the *MIP-Diameter* model.

When we analyze the below tables, $D_{\max}$ deteriorates when the inclusion and exclusions are applied over a certain cutoff value for inclusions and below a certain cutoff value for exclusions. However, the AT values occurring in both solutions are reported to be very close to each other. The solution time of the model is decreased in all cases. In experiment 47, the value of $D_{\max}$ is increased 6% but from other side, the solution time is reduced drastically while at the same time an improvement in AT value is achieved. Next, from Tables 4.10, 4.12 and 4.14 it can be observed the final AT for two solutions are reported to be very close to each other. Also, one can observe these findings from Figures 4.6 and 4.7. Thus, we can conclude that the proposed inclusion/ exclusion rules have a positive impact on solution time of the proposed *MIP-Diameter* model while preserving the final quality of the solution, further to tell possibly in some cases resulting in an ignorable increase in final AT values.

			MIP-Diameter initialized with seeds only						*MIP-Diameter* initialized with Inclusions and Exclusions							
Exp. Number	k	N	Iter.	Node	$D_{\max}$	AT	t(sec)	Solv. (MB)	Inc.	Exc.	Iter.	Node	$D_{\max}$	AT	t(sec)	Solv. (MB)
38	2	100	430	11	10.81	13.23	**1.84**	4.66	28	23	94	9	10.8	13.05	**0.23**	4.71
39	3	100	1899	247	8.9	17.79	**14.14**	6.97	54	9	33	0	9.2	18.30	**0.31**	6.92
40	4	100	31	0	8.5	23.27	**0.36**	9.17	22	0	-	-	9.1	23.68	**0.31**	9.17
41	5	100	180	0	8.4	29.57	**1.36**	11.48	21	0	10	0	9	29.47	**0.47**	11.47

Table 4.9: The experimental results of *MIP-Diameter* model initialized by applying Inclusion and Exclusion rules on the 100 instance data set

			MIP-Diameter initialized with seeds only					*MIP-Diameter* initialized with Inclusions and Exclusion				
Exp. Number	k	N	RIter.	D_{max}	AT	$\%D_{max}$	%AT	RIter.	D_{max}	AT	$\%D_{max}$	%AT
42	2	100	3	10.8	12.44	**0.00**	**0.06**	2	10.8	12.44	**0.00**	**0.05**
43	3	100	4	8.9	17.61	**0.00**	**0.01**	5	8.9	17.61	**0.03**	**0.04**
44	4	100	4	8.5	22.31	**0.00**	**0.04**	8	8.9	22.50	**0.02**	**0.05**
45	5	100	10	8.4	27.37	**0.00**	**0.08**	8	9	27.20	**0.00**	**0.08**

Table 4.10: The experimental results of *Reassignment Algorithm* applied on the solutions of the *MIP-Diameter* model reported in Table 4.9

			MIP-Diameter initialized with seeds only						*MIP-Diameter* initialized with Inclusions and Exclusions							
Exp. Number	k	N	Iter.	Node	D_{max}	AT	t(sec)	Solv. (MB)	Inc.	Exc.	Iter.	Node	D_{max}	AT	t(sec)	Solv. (MB)
46	2	200	1257	31	10.8	13.19	**24.38**	17.97	57	47	218	1	10.8	12.97	**0.92**	17.99
47	3	200	6131	505	8.9	18.11	**172.81**	26.72	96	7	279	3	9.5	17.79	**1.95**	26.74
48	4	200	181	0	8.5	22.70	**1.97**	35.44	55	0	-	-	8.9	23.37	**1.84**	35.45
49	5	200	-	-	8.4	27.07	**1.81**	44.18	68	0	-	-	8.4	28.09	**1.92**	441.88

Table 4.11: The experimental results of *MIP-Diameter* model initialized by applying Inclusion and Exclusion rules on the 200 instance data set

			MIP-Diameter initialized with seeds only					*MIP-Diameter* initialized with Inclusions and Exclusions				
Exp. Number	k	N	RIter.	D_{max}	AT	$\%D_{max}$	%AT	RIter.	D_{max}	AT	$\%D_{max}$	%AT
50	2	200	4	10.8	12.39	**0.00**	**0.06**	4	10.8	12.39	**0.00**	**0.05**
51	3	200	3	8.9	17.42	**0.00**	**0.04**	2	9.5	17.40	**0.00**	**0.02**
52	4	200	3	8.5	21.98	**0.00**	**0.03**	4	8.9	22.33	**0.00**	**0.04**
53	5	200	3	8.4	26.73	**0.00**	**0.01**	4	8.4	26.78	**0.00**	**0.05**

Table 4.12: The experimental results of *Reassignment Algorithm* applied on the solutions of the *MIP-Diameter* model reported in Table 4.11

			MIP-Diameter initialized with seeds only						*MIP-Diameter* initialized with Inclusions and Exclusions							
Exp. Number	k	N	Iter.	Node	D_{max}	AT	t(sec)	Solv. (MB)	Inc.	Exc.	Iter.	Node	D_{max}	AT	t(sec)	Solv. (MB)
54	2	300	1761	39	10.8	13.01	**75.75**	39.73	86	71	363	3	10.8	12.94	**2.75**	39.78
55	3	300	16607	954	8.9	17.84	**874.58**	59.35	95	34	545	30	9.8	19.19	**11.11**	59.39
56	4	300	1602	19	8.5	23.57	**53.72**	78.95	118	6	201	0	8.5	22.81	**4.11**	78.99
57	5	300	-	-	8.4	26.64	**10.72**	98.54	101	0	-	-	8.4	27.86	**4.47**	98.56

Table 4.13: The experimental results of *MIP-Diameter* model initialized by applying Inclusion and Exclusion rules on the 300 instance data set

			MIP-Diameter initialized with seeds only					*MIP-Diameter* initialized with Inclusions and Exclusions				
Exp. Number	k	N	RIter.	D_{max}	AT	%D_{max}	%AT	RIter.	D_{max}	AT	%D_{max}	%AT
58	2	300	4	10.8	12.37	**0.00**	**0.05**	5	10.8	12.37	**0.00**	**0.04**
59	3	300	3	8.9	17.34	**0.00**	**0.03**	6	9.7	17.91	**0.01**	**0.07**
60	4	300	5	8.5	21.84	**0.00**	**0.07**	3	8.5	21.81	**0.00**	**0.04**
61	5	300	6	8.4	26.56	**0.00**	**0.00**	4	8.4	26.57	**0.00**	**0.05**

Table 4.14: The experimental results of *Reassignment Algorithm* applied on the solutions of the *MIP-Diameter* model reported in Table 4.13

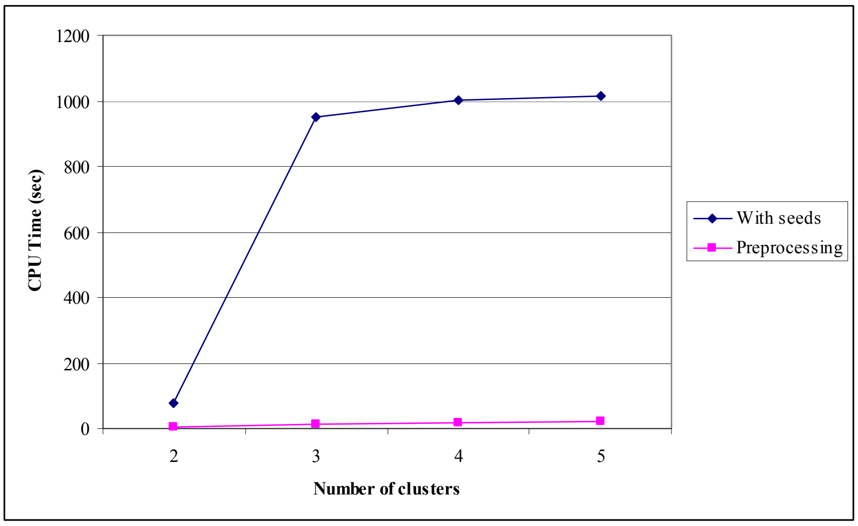

Figure 4.6: The effect of the preprocessing rules on the solution time of the proposed algorithm on 300 instance data set

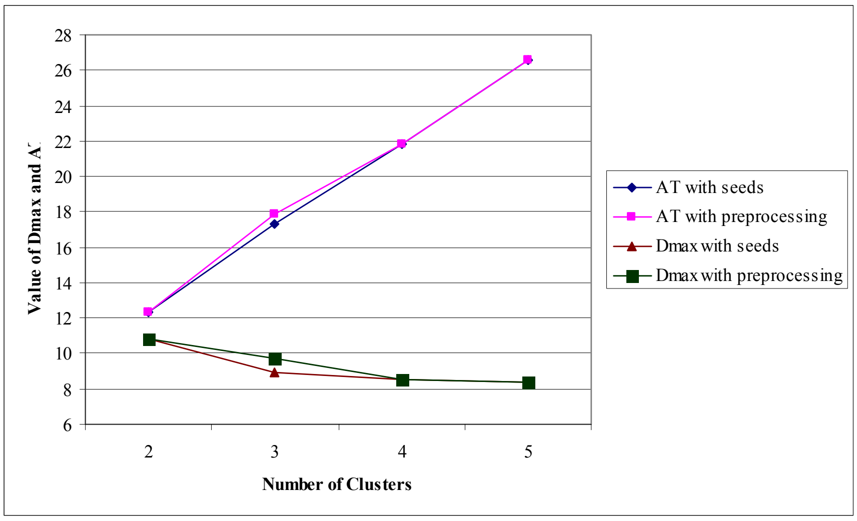

Figure 4.7: The effect of the preprocessing rules on the D_{max} and AT values of the proposed algorithm on 300 instance data set

Chapter 5

A MIXED-INTEGER NONLINEAR MATHEMATICAL PROGRAMMING MODEL TO MINIMIZE THE SUM OF WITHIN-GROUP DISTANCES (*MINLP-SWGD*)

5.1 Proposed Mathematical Programming Model

In this part of the thesis, a mixed-integer nonlinear programming model (*MINLP*) with the objective function of minimization of the sum of within-group distances (*SWGD*) is studied. Previously, clustering problem with this criterion is studied by Rao [23]. Rao [23] studied the same *MINLP* model, and named the partitioning criterion as total within-group distances (*TWGD*). He proposed the linearization of the model by adding a set of constraints. He proposed a solution methodology only for the two cluster case but reported no computational results. Brusco [24] studied the criterion of minimization of the sum of within-group distances in addition to the minimization of the maximum diameter criterion. In the study, he proposes a branch and bound algorithm in which tighter upper bounds are derived by utilizing an exchange algorithm. For the *TWGD* criterion, the solution time reported for 40 instances and 6 clusters is 3152.12 seconds. The branch-and-bound methods reported in his study are not applicable on large data sets due to limitations on computational time. Moreover, the efficiency and performance of the algorithms are reported to be very sensitive to the number of clusters.

The formulation of the problem with the objective defined above leads to a *mixed-integer nonlinear programming* problem.

Given a data set of n data items in m - dimensions, i.e. a set of n points is R^m, the objective of the proposed mathematical programming model is to find the optimal partitioning of the data set into K exclusive clusters, assuming that the number of desired clusters is known a priori.

The mathematical formulation of the model *MINLP-SWGD* is given below.

MINLP-SWGD:

$$\text{Minimize } Z = \sum_{k=1}^{K}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij}x_{ik}x_{jk} \tag{5.1}$$

subject to

$$\sum_{k=1}^{K} x_{ik} = 1 \qquad \forall i, i = 1,..,n \tag{5.2}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k, i = 1,..,n, k = 1,..,K \tag{5.3}$$

where the variable x_{ik} is assigned to the value 1 or 0 according to whether the i^{th} instance is assigned to cluster k or not. The objective function of the model is the summation of pairwise distances between the data points assigned to the same cluster (5.1). The model aims to partition the data points into exclusive clusters and therefore each instance should be assigned to only one cluster as given in equation (5.2). The model has kn binary variables, one continuous variable and $O(kn)$ constraints.

The *MINLP-SWGD* model is a nonconvex *MINLP* model which is difficult to solve exactly. With a bilinear objective function *MINLP* models inherit the combinatorial structure of *MIP* models and the difficulty of solving nonconvex *NLP* models. The subclasses of *MIP* and nonconvex models belong to the class of NP-Hard problems. The solution methods of *MINLP* models can be stated as Outer Approximation methods (OA) [31], Branch and Bound Algorithms [32], Extended Cutting Plane methods [33] and Generalized Bender's Decomposition [34].

We modeled the *MINLP-SWGD model* with GAMS IDE version 2.0.30.1 [35] and used the DICOPT solver to solve the problem. DICOPT, developed by Grossman and Kocis [36], is a general purpose *MINLP* solution algorithm which utilizes the OA method. The OA methods require the solution of a related *MIP* and a *NLP* problem successively. *MINLP* model is decomposed into a linear master *MIP* problem and a *NLP* subproblem.

Since our model is nonconvex, the DICOPT algorithm solves the problem to local optimality. BARON is a global optimization solver but it is applicable only on small-sized data sets, since convergence to the global optimal solution is a demanding process. In the next chapter of the thesis, the computational experiments performed on the real data set with the proposed model are given. The reported solution times for the model seem to be very promising.

In further investigation of the proposed model, we formed the continuous relaxation of the model as a nonlinear programming (*NLP*) model. The binary decision variables are set as positive continuous variables (5.3). Lower and upper bounds are given as 0 and 1 respectively, to limit the value of the assignment variables. In order to solve the continuous relaxation of the *MINLP-SWGD* model, we use the MINOS solver. It is an *NLP* solver which also does not guarantee global optimality. In the results of the experiments all the *NLP* model solutions are found to be integer. Further investigation on the proposed model leads us to the following property.

Property 5.1: Total Unimodularity Property [37]

Let A be an $m \times n$ integer matrix and it has a rank of m which means that it consists of m linearly independent rows. Then, we say that A is unimodular if the determinant of every basis matrix B of A has value $+1$ or -1. The proof of the unimodularity theorem is given by Ahuja et al. [37]. Thus, relying on this proof, we can state that if the integer valued matrix A is unimodular, then every basic feasible solution of the polyhedron defined by the constraints $Ax = b$ where $x \geq 0$, is integer for every integer valued right hand side vector b. The analyzed total unimodularity property applies to a special subclass of unimodular matrices. If every square submatrix of A matrix has a determinant of 0 or ± 1 then the matrix A is totally unimodular. Again, in the same reference it is stated that every totally unimodular matrix is unimodular since each basis matrix B of the matrix A has a determinant of ± 1 $\square$.

Proposition 5.1: The constraint set of the *MINLP-SWGD* model has the total unimodularity property.

Proof of the Proposition 5.1: For the constraint 5.3, n is the number of data points and k is the number of clusters, the corresponding A matrix of the *MINLP-SWGD* model can be stated algebraically as follows.

$$\begin{array}{cc} & \begin{array}{ccccccccccccccc} x_{11} & x_{12} & \cdots & x_{1k} & x_{21} & x_{22} & \cdots & x_{2k} & \cdots & \cdots & \cdots & & x_{n(k-1)} & x_{nk} \end{array} \\ A = \begin{array}{c} 1 \\ 2 \\ \vdots \\ \vdots \\ n \end{array} & \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 & \cdots & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 1 & 1 & \cdots & 1 & \cdots & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & \cdots & \cdots & \cdots & 1 & 1 \end{bmatrix} \end{array}$$

The above $n \times nk$ matrix A is $0-1$ matrix and its rank is equal to n since it consists of n linearly independent rows.

Claim 5.1: Every square submatrix of A has a determinant 0 or $+1$ and therefore it is a totally unimodular matrix.

Proof of the Claim 5.1: To prove the claim, we need to show every square submatrix M of A of size k has determinant of 0 or $+1$. We apply induction hypothesis on the size of the square submatrix to achieve this result. Since the matrix A is a $0-1$ matrix, the claim is true for $k=1$. Assume that the claim holds for some k and let M be any $(k+1)\times(k+1)$ submatrix of A. There exist two possibilities that the submatrix M satisfies. The first possibility is M contains a column with no nonzero element; the second possibility is some column M_l has exactly one nonzero element, let's say in the i^{th} row. In the first case, the determinant of M is 0 and the claim holds. In case two, let M' denote the submatrix of M obtained by deleting the i^{th} row and l^{th} column. This implies $\det(M) = +1\det(M')$. By the induction hypothesis we stated, $\det(M')$ is either 0

or $+1$, so $\det(M)$ is also 0 or $+1$. This establishes the claim. Thus, we can conclude that the matrix A is a unimodular matrix since every totally unimodular matrix is unimodular.

Corollary 5.1: There exists an extreme point solution of the NLP relaxation of the *MINLP-SWGD* model defined by Constraint 5.3 is integer.

Proof of the Corollary 5.1: Follows from Proposition 5.1 and Property 5.1, we conclude that an extreme point solution of the NLP relaxation of the *MINLP-SWGD* model defined by Constraint 5.3 is integer. Since our model is bilinear, we prefer to use the term extreme point in stead of using the term basic feasible solution since the stated term belongs to the linear programming.

Due to Corollary 5.1, we solve the continuous relaxation of the *MINLP-SWGD*, as stated earlier. Since the objective function of the *MINLP-SWGD* model is nonconvex, then the solutions cannot be guaranteed to be global optimal. In next chapter of the thesis, we report computational experiments conducted on real data set of the proposed model in order to analyze its efficiency and performance.

5.2 Experiments with *NLP-SWGD* on Illustrative Data Set

In this section of the thesis, the proposed *MINLP-SWGD* model is applied on the illustrative data set given in Section 3.5. We performed experiments on this data set with *MINLP-SWGD* model to evaluate the quality of the solution of the model. We also observe that the solution times of the model solved by MINOS are quite small for this synthetic data set.

In Figure 5.1, the solution of the model is shown. Visually, two data points seem to be assigned incorrectly but according to the objective function value of the model this assignment pattern results in the minimum value for the criterion of the model. The global optimal solution of the model is known a priori since the data set is synthetic. For this example, the solution of the *MINLP-SWGD* model is the global optimal solution but this finding does not imply that the global solution can be obtained in all experiments with different data sets due to nonconvexity of the objective function unless a global optimization solver is used. The value of the objective function is 2234, there are 325 decision variables and 324 constraints in the model. We modeled *MINLP-SWGD* problem with GAMS IDE version 2.0.30.1 [35] and performed our experiments on a workstation with 3.2 GHz Xeon processor and 2 GB memory. The reported solution time of the *MINLP-SWGD* model is 0.25 seconds.

Furthermore, this experiment clearly shows that different clustering algorithms may lead to different assignment patterns due to the different types of partitioning criteria applied in the models. Even in the case of exact partitioning of the models, as seen in Figure 5.1 the models may lead to different assignments of data points to clusters. We believe that, this observation is important to emphasize the subjectivity of clustering approaches. This constitutes one of the biggest difficulties in the clustering problem.

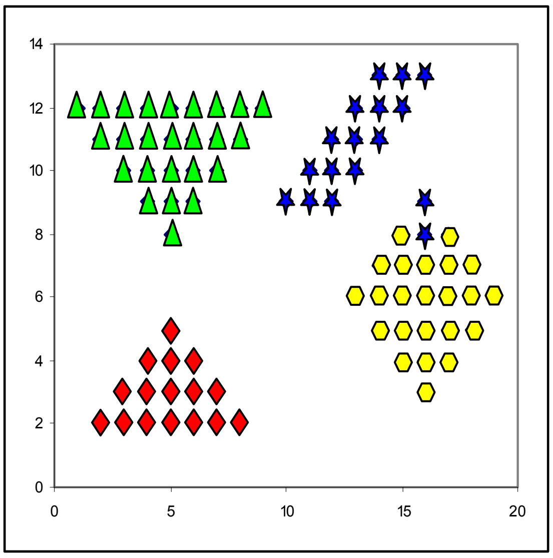

Figure 5.1: The solution of the *NLP-SWGD* model on the illustrative data set

5.3 Experiments with *MINLP-SWGD* on the Real Data Set

In this section of the thesis, computational experiments with the proposed *MINLP-SWGD* model are given in Tables 5.1, 5.2, 5.3, 5.4 and 5.5 on 100 to 500 instance data sets. Each data set is partitioned into two to five clusters. The CPU times of the models is very promising on all data sets and can be seen to be in the order of seconds.

Exp. Number	K	N	Var.	Cons.	SWGD	Iter.	t(sec)	Solv. (MB)
62	2	100	201	100	15152.25	66	**0.42**	3.9 Mb
63	3	100	301	100	9388.02	80	**0.56**	3.9 Mb
64	4	100	401	100	6649.29	88	**0.73**	3.9 Mb
65	5	100	501	100	5155.61	111	**1.02**	3.9 Mb

Table 5.1: Experiments with *MINLP-SWGD model* on 100 instance data set

Exp. Number	K	N	Var.	Cons.	SWGD	Iter.	t(sec)	Solv. (MB)
66	2	200	401	200	60916.59	130	**1.78**	5.0 Mb
67	3	200	601	200	37749.49	159	**3.33**	5.0 Mb
68	4	200	801	200	26725.59	173	**4.45**	5.0 Mb
69	5	200	1001	200	20855.44	253	**7.53**	5.0 Mb

Table 5.2: Experiments with *MINLP-SWGD model* on 200 instance data set

Exp. Number	K	N	Var.	Cons.	SWGD	Iter.	t(sec)	Solv. (MB)
70	2	300	601	300	137259.99	195	**5.72**	6.0 Mb
71	3	300	901	300	84986.94	260	**10.69**	6.0 Mb
72	4	300	1201	300	60198.26	254	**13.53**	6.0 Mb
73	5	300	1501	300	46913.18	320	**20.75**	6.0 Mb

Table 5.3: Experiments with *MINLP-SWGD* model on 300 instance data set

Exp. Number	K	N	Var.	Cons.	SWGD	Iter.	t(sec)	Solv. (MB)
74	2	400	801	400	244800.39	233	**10.83**	7.6 Mb
75	3	400	1201	400	152348.91	296	**20.72**	7.6 Mb
76	4	400	1601	400	108904.30	381	**33.86**	7.6 Mb
77	5	400	2001	400	84242.07	477	**52.31**	7.6 Mb

Table 5.4: Experiments with *MINLP-SWGD* model on 400 instance data set

Exp. Number	K	N	Var.	Cons.	SWGD	Iter.	t(sec)	Solv. (MB)
78	2	500	1001	500	383394.41	310	**21.94**	10.2 Mb
79	3	500	1501	500	238696.25	381	**39.74**	10.2 Mb
80	4	500	2001	500	171581.1	471	**64.53**	10.2 Mb
81	5	500	2501	500	132823.1	572	**97.70**	10.2 Mb

Table 5.5: Experiments with *MINLP-SWGD* model on 500 instance data set

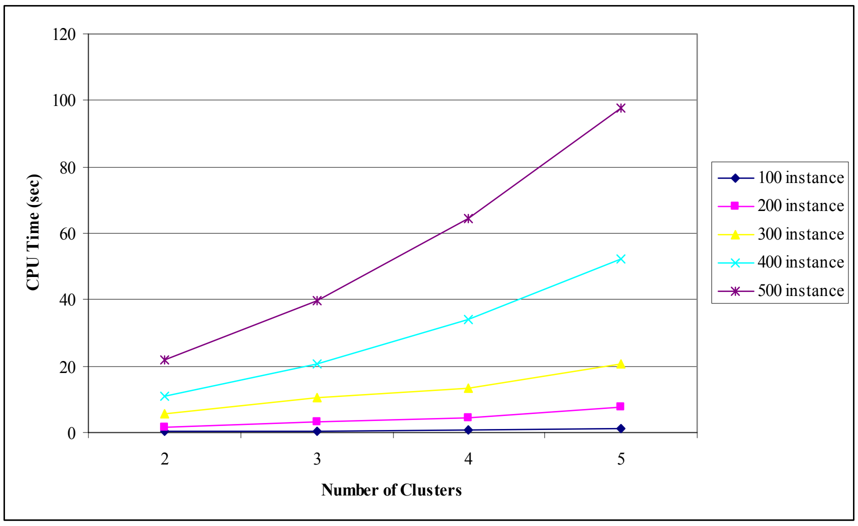

Figure 5.2: The solution time of the *SWGD* model versus the increased number of clusters and instances

Also, in Figure 5.2 how the solution time of the model is affected with increased number of instances and clusters can be observed.

When we observe the above experimental findings, we see that the size of the required memory stays the same as K increases and the solution time of the model increases linearly.

Chapter 6

THE COMPARISON OF THE MODELS AND INTERPRETATIONS DERIVED FROM THE DATA SET

In this section of the thesis, first we compare the results of the proposed clustering algorithm, *MINLP-SWGD* model and *K-Means* algorithm according to the $D_{\max}$, AT and $SWGD$ performance measures. Next, we give some interpretations derived from the solutions of the stated algorithms and the *SWGD* model, and conclude the section by showing that the proposed algorithm and the nonlinear model is better than the *K-Means* algorithm in terms of interpretability of the solutions which indicates that the proposed models and algorithm in this thesis yield a successful segmentation of the data.

6.1 Comparison of the Proposed Algorithm, *MINLP-SWGD* Model and *K-Means* Algorithm According to the $D_{\max}$, AT and $SWGD$ Performance Measures

The aim of this section is to observe and analyze the value of the studied clustering performance measures $D_{\max}$, AT and $SWGD$ achieved by the proposed algorithm, *MINLP-SWGD* model and *K-Means* algorithm. The experiments performed on 100, 200 and 300 instance data sets are reported with this purpose. The previously reported experiments are given to observe the improvements gained by the proposed algorithm, the proposed seed finding algorithms and the proposed preprocessing rules. In this section, the experiments are reported to analyze the solution quality of the models and algorithms in terms of the measures $D_{\max}$, AT and *SWGD* achieved by the proposed algorithm, *MINLP-SWGD* model and *K-Means* algorithm.

In Tables 6.1, 6.2 and 6.3, Dia_k denotes the diameter of the cluster k and N_k denotes the number of instances assigned to cluster k.

From above tables, one can observe that according to the $D_{\max}$ performance measure, the best solutions are occurred with the proposed algorithm. In some experiments *SWGD* model result in better $D_{\max}$ values than the *K-Means* algorithm does and in some *K-Means* algorithm is superior to the *SWGD* model. This observation is not surprising since the objective of the proposed algorithm is to minimize the $D_{\max}$ value, and problems are solved to a near-optimal solution.

Similarly, the *SWGD* model achieved the best *SWGD* values when compared to other two algorithms although it has been solved to local optimality. Except the 2-cluster solutions, the proposed algorithm results in better *SWGD* values than the *K-Means* algorithm. In 2-cluster case, the difference between the *SWGD* values between the two algorithms is at most 1%.

Next when we observe the AT values achieved, we see that *K-Means* algorithm results in the most promising results. One may not expect this finding. When we further investigate the results, this fact is due to the number of instances assigned to the clusters. In some cases, *K-Means* algorithm partitions the data set into big and small clusters. A widely discussed issue with the minimization of the AT measure is its sensitivity to the size of the resulting clusters. As indicated in Section 3.2 of the study, AT measure is a normalized measure and its value decreases as the size of the corresponding cluster increases. Thus, big clusters generated with the *K-Means* algorithm lead to smaller AT values.

Another important observation based on the below experiments is the *SWGD* model partitions the data set into clusters with almost equal number of instances. Again, it is a widely discussed issue of the model in clustering studies but no experimental findings were reported previously. Similarly, *K-Means* algorithm may generate clusters with equal number of instances in some cases due to initially determined cluster centers. On the other hand, in some cases it may produce clusters with different cardinalities.

To sum up, all these facts and findings show us the difficulty and subjectivity of the clustering analysis. With mathematical and statistical models, one can solve the problem but each clustering measure aims to perform a different criterion and although the models seem quite similar they may result in different solutions. The hardest part of the clustering is the analysis of the solutions. Especially in business applications, the interpretation of the resulting clusters constitutes an important step. Even in some cases, a very successful clustering scheme in terms of numerical findings may not result in a high quality clustering. This fact may both depend on the data or the applied performance measure. Thus, in the next section of the thesis we give some interpretations derived from the solutions of the proposed algorithm, *SWGD* model and *K-Means* algorithm in order to observe their solution quality on the Digiturk problem.

Proposed Algorithm

Exp. Number	**N**	**K**	**Dia$_1$**	**Dia$_2$**	**Dia$_3$**	**Dia$_4$**	**Dia$_5$**	**N$_1$**	**N$_2$**	**N$_3$**	**N$_4$**	**N$_5$**	***D_{max}***	***AT***	***SWGD***
82	100	2	10.68	10.77				44	56				**10.80**	12.44	15546
83	100	3	8.83	8.90	8.90			33	28	39			**8.90**	17.61	95349
84	100	4	8.37	8.43	8.37	8.50		19	31	30	20		**8.50**	22.31	6948.5
85	100	5	8.37	8.40	7.75	7.81	8.37	15	30	17	18	20	**8.40**	27.37	5468.5

SWGD Model

Exp. Number	**N**	**K**	**Dia$_1$**	**Dia$_2$**	**Dia$_3$**	**Dia$_4$**	**Dia$_5$**	**N$_1$**	**N$_2$**	**N$_3$**	**N$_4$**	**N$_5$**	***D_{max}***	***AT***	***SWGD***
86	100	2	10.54	11.87				50	50				11.87	**12.37**	**15152.25**
87	100	3	8.89	9.49	9.38			34	33	33			9.49	17.44	**9388.02**
88	100	4	7.55	8.37	8.66	9.38		26	25	25	24		9.38	22.24	**6649.29**
89	100	5	7.55	7.94	8.43	9.38	8.49	24	20	19	19	18	9.38	27.24	**5155.61**

K-Means Algorithm

Exp. Number	**N**	**K**	**Dia$_1$**	**Dia$_2$**	**Dia$_3$**	**Dia$_4$**	**Dia$_5$**	**N$_1$**	**N$_2$**	**N$_3$**	**N$_4$**	**N$_5$**	***D_{max}***	***AT***	***SWGD***
90	100	2	10.80	11.18				55	45				11.18	12.48	15315
91	100	3	8.94	8.49	9.85			47	17	36			9.85	**17.35**	10295
92	100	4	8.94	3.46	8.89	9.90		30	5	34	31		9.90	**19.91**	8301.3
93	100	5	8.37	9.00	7.81	8.43	7.14	28	19	6	31	6	9.00	**25.96**	6196.8

Table 6.1: The comparison of the $D_{\max}$, AT and *SWGD* values achieved by the proposed algorithm, *MINLP-SWGD* model and *K-Means* algorithm on the 100 instance data set

Proposed Algorithm

Exp. Number	N	K	Dia$_1$	Dia$_2$	Dia$_3$	Dia$_4$	Dia$_5$	N$_1$	N$_2$	N$_3$	N$_4$	N$_5$	D_{max}	*AT*	*SWGD*
94	200	2	10.68	10.80				88	112				**10.80**	12.39	62537
95	200	3	8.89	8.83	8.90			79	65	56			**8.90**	17.41	38471
96	200	4	8.42	8.50	8.37	8.50		38	64	56	42		**8.50**	21.97	27872
97	200	5	7.87	8.40	8.40	7.81	8.37	39	59	36	30	36	**8.40**	**26.72**	21887

SWGD Model

Exp. Number	N	K	Dia$_1$	Dia$_2$	Dia$_3$	Dia$_4$	Dia$_5$	N$_1$	N$_2$	N$_3$	N$_4$	N$_5$	D_{max}	*AT*	*SWGD*
98	200	2	10.54	11.87				100	100				11.87	**12.31**	**60916.59**
99	200	3	8.89	9.49	10.58			67	66	67			10.58	17.26	**37749.49**
100	200	4	7.55	8.37	8.49	9.38		54	50	48	48		9.38	21.91	**26725.59**
101	200	5	7.21	9.06	9.64	8.12	8.43	44	38	38	40	40	9.64	26.94	**20855.44**

K-Means Algorithm

Exp. Number	N	K	Dia$_1$	Dia$_2$	Dia$_3$	Dia$_4$	Dia$_5$	N$_1$	N$_2$	N$_3$	N$_4$	N$_5$	D_{max}	*AT*	*SWGD*
102	200	2	11,23	10,77				91	109				11,23	12,38	63993
103	200	3	11,23	9,38	7,55			95	49	56			11,23	**17,04**	42079
104	200	4	8,89	3,46	9,49	9,90		65	10	62	63		9,90	**19,37**	33307
105	200	5	8,89	8,12	7,81	8,89	8,72	54	46	34	33	33	8,89	26,92	22822

Table 6.2: The comparison of the $D_{\max}$, AT and *SWGD* values achieved by the proposed algorithm, *MINLP-SWGD* model and *K-Means* algorithm on the 200 instance data set

Proposed Algorithm

Exp. Number	N	K	Dia_1	Dia_2	Dia_3	Dia_4	Dia_5	N_1	N_2	N_3	N_4	N_5	D_{max}	*AT*	*SWGD*
106	300	2	10.68	10.80				132	168				**10.80**	12.37	140940
107	300	3	8.83	8.89	8.90			97	119	84			**8.90**	17.34	86785
108	300	4	8.50	8.37	8.49	8.43		96	83	63	58		**8.50**	21.84	62690
109	300	5	8.40	7.81	8.40	8.31	8.25	47	51	91	52	59	**8.40**	26.56	49785

SWGD Model

Exp. Number	N	K	Dia_1	Dia_2	Dia_3	Dia_4	Dia_5	N_1	N_2	N_3	N_4	N_5	D_{max}	*AT*	*SWGD*
110	300	2	10.54	11.87				151	149				11.87	**12.29**	**137259.99**
111	300	3	8.89	9.49	10.58			101	99	100			10.58	17.18	**84896.94**
112	300	4	7.55	8.37	8.66	9.38		81	74	72	73		9.38	21.78	**60198.25**
113	300	5	7.28	9.49	7.94	8.72	8.49	69	58	59	57	57	9.49	26.66	**46913.18**

K-Means Algorithm

Exp. Number	N	K	Dia_1	Dia_2	Dia_3	Dia_4	Dia_5	N_1	N_2	N_3	N_4	N_5	D_{max}	*AT*	*SWGD*
114	300	2	11.23	10.77				134	166				11.23	12.37	140330
115	300	3	11.23	8.89	8.89			140	99	61			11.23	**17.13**	95842
116	300	4	8.89	9.90	9.33	7.35		91	111	74	24		9.90	**20.91**	72516
117	300	5	9.33	7.35	8.25	8.89	8.37	70	24	61	87	58	9.33	**25.57**	52217

Table 6.3: The comparison of the $D_{\max}$, AT and *SWGD* values achieved by the proposed algorithm, *MINLP-SWGD* model and *K-Means* algorithm on the 300 instance data set

6.2 Interpretation of the Results of the Proposed Algorithm, the *SWGD* Model and the *K-Means* Algorithm

In this part of the thesis, we compare the results of the proposed algorithm, the *SWGD* model and the *K-Means* algorithm. For this purpose, we used 300 instances data set clustered into 5 classes by three approaches.

Naturally, a realistic interpretation of the results can be made only in cooperation with the true owners of the task. We derive our interpretations based on numeric findings from the partitioning of the data set. We use a hypothesis testing model to observe the numerical attribute significance since the real data we used in our experiments is in numeric form.

As previously stated, the common goal of all clustering methods is to maximize the similarities within the clusters and at the same time to maximize the dissimilarities between the clusters. We set up our hypothesis testing on the maximization of the dissimilarities between the resulting clusters. Therefore, we test if the formed clusters by the methods studied in this thesis are significantly different from each other or not.

To achieve this first we state our *null hypothesis* which states a negative point of view in the formation of different clusters. It specifies the existence of no significant difference between the clusters. Once the experiments are performed, we have the assignment pattern of the instances according to the three different methods; we test if the results show a significant difference between the clusters. Based on the model of the [7], we apply the below model to test for a significant difference between the clusters.

$$T^d{}_{pr} = \frac{\left| x^d{}_p - x^d{}_r \right|}{\sqrt{v^d{}_p / n_p + v^d{}_r / n_r}} \qquad (6.1)$$

where

- p and r denote the cluster p and r such that $p = 1,..,K$, $r = 1,..,K$ and $p \neq r$ where K is the number of clusters
- d denotes the attribute d such that $d = 1,..,m$ where m is the number of attributes

- $T^{d}{}_{pr}$ is the test statistic for between the clusters p and r according to the attribute d
- $x^{d}{}_{p}$ and $x^{d}{}_{r}$ cluster means for the independent clusters for the attribute d
- $v^{d}{}_{p}$ and $v^{d}{}_{r}$ are variance scores for respective means for the attribute d
- n_p and n_r are corresponding cluster sizes

The numerator is the absolute difference between the two means of the clusters p and r according to the attribute d. We calculate the standard error as the square root of the sum of the two variance scores of the associated cluster means d as we assume that the generated clusters are independent from each other. This difference in the numerator is divided by the standard error to test if there is a significant difference between the clusters according to the corresponding attribute. We test if the $T^{d}{}_{pr}$ value for the corresponding clusters and attribute is greater than or equal to 1.96 to be 95% confident that the difference between two means is not by chance. We use 1.96 as the cutoff value since we assume that the distribution of means and differences are normal.

The hypothesis test states that if any pair of cluster comparisons shows a significant difference, the corresponding attribute can be considered significant in clustering.

We calculate the $T^{d}{}_{pr}$ values based on mean, variance and population of clusters and observe if the formed clusters are significantly different from each other. If the answer of this test is positive, then we aim to label the clusters according to their significant differentiating features. Details can be seen in Appendix A.

We observe that the clusters formed by the proposed algorithm in this thesis are more interpretable than the clusters formed by the *K-Means* algorithm. According to the results of our algorithm, we are able to identify clusters of standard package users, sport fanatics and a group of people who are willing to spend their money in interactive events such as TV games. Standard package owners are long-lived customers who are at the same time sensitive to price changes since the amount of payment is a concern for the people belonging to this group. They switch from one package to another seldomly. For the attribute which denotes the frequency of changes between packages, the sport fanatics are the leaders. Since their main concern is the sport package, they do not change their packages. Sport fanatics cluster can be classified as a subgroup of standard package users but this cluster's main difference is its dependency on a certain period of time which is the period of the football league season. Another cluster consists of people who are interested in pay-per-view services, interactive events, TV games, etc. This group of people pay more, use more but also switch between the packages more. They benefit from various features offered to them and thus this group can be seen as the potential revenue generating cluster. Statistical tests indicate that the differentiation in attribute values is significant.

Next, we analyze the interpretability of the results of the proposed *NLP-SWGD* model. One of the stated limitations of the proposed model in the literature is that the formed clusters by this model have approximately equal size. Although, the resulting clusters seem to be equal in size, we are able to derive some interpretations from the solution. Similar to the interpretations given for the proposed algorithm, we observe the clusters of standard package subscribers, sport fanatics and high-spending customers. In this solution, again the standard package subscribers have the same interesting feature. They are stable customers of the company but also they have sensitivity to an increase in price. In general, they do not benefit from interactive events.

When we look at the results of the *K-Means* algorithm, we can not easily interpret the clusters obtained. Moreover, the $T^{d}{}_{pr}$ values calculated for the clusters of our algorithm are more promising for the ones calculated for the *K-Means* clusters. Based on our intuitive observations, our algorithm leads to a better solution than *K-Means* algorithm. However, we note that *K-Means* might deliver different clusters when different initial points are used and it may be possible to obtain interpretable clusters as well. Still, the fact that our algorithm and the *SWGD* model do not require trial and error experimentation constitutes an advantage.

Chapter 7

CONCLUSION AND SUGGESTIONS FOR FUTURE RESEARCH

The objective of this thesis is to develop a mathematical programming based clustering approach applicable on large data sets. Two mathematical programming clustering models and a heuristic clustering algorithm are proposed with this purpose. The proposed models, the components of the proposed clustering algorithm are applied on an illustrative data set and a real data set to analyze their performance and efficiency.

The proposed first mathematical programming model *MIP-Diameter* forms clusters by minimizing the maximum diameter of the generated clusters. First, we give the *MINLP* formulation of the model and then we reduce it to a mixed integer linear (*MIP*) formulation by linearization of the nonlinear constraint existing in the model without increasing the number of constraints and variables. The proposed model is nonhierarchical in the sense that the number of the clusters is assumed to be known a priori.

In order to overcome the issues related with the computational difficulty of the *MIP-Diameter* model, a graph-theoretic approach based on maximal independent set is utilized to find initial cluster seeds. The aim of the seed finding approach is to determine a set of leader data points from the data set in order to improve the solution time of the proposed model without compromising from the quality and accuracy of the resulting solution. Originally, the maximal independent set algorithm is a random algorithm. We observe the possible outcomes of the model initialized by *SF*1 algorithm and show that giving seeds to the *MIP-Diameter* model reduces the solution time drastically. Next, in order to improve the quality of the seeds and the robustness of the proposed seed finding methodology, we propose an alternative seed finding algorithm. This alternative *SF*2 algorithm selects the members of the maximal independent set according to the measure of the popularity of nodes. The term popularity refers to the number of neighbor nodes a data point has over the parametric sparse graph. We analyze the solutions of two seed finding algorithms and observe their positive effect on the model. In order to further improve the solution time of the proposed *MIP-Diameter* model, an analytical preprocessing methodology is proposed. This preprocessing methodology is an extension of the seed finding algorithms. It is a heuristic approach to determine the inclusion of some instances to clusters initially and also exclusion of some instances from clusters according to a determined cutoff value to determine the relative closeness of data points to each cluster seed. The experiments on illustrative and real data sets are also reported for this methodology to analyze the positive impact of these rules on the solution quality of the proposed model and algorithm.

Furthermore, as stated previously, in order to improve the solution quality of the proposed model *MIP-Diameter*, a reassignment algorithm is proposed. The reassignment algorithm is developed to reassign distorted data points existing in the solution of *MIP-Diameter* model which may be possibly assigned in a better way. The proposed seed finding algorithm, the *MIP-Diameter* model and the reassignment algorithm constitute the proposed clustering algorithm. The various experimental results of the algorithm are reported to analyze its efficiency and performance. The results of the algorithm are compared with the results of the well-known *K-Means* algorithm. The solution of the proposed methodology is more interpretable than the solution of *K-Means* algorithm applied over the same real data set.

In Chapter 5 of the thesis, a second mathematical programming based model *MINLP-SWGD* is given. It is a mixed integer nonlinear programming model with the objective function of the minimization of sum of within-group distances. The objective function of the model is nonconvex due to its bilinear structure and therefore the solution of the model to the global optimality is a challenging subject. We show that every basic feasible solution of the relaxation of the model is integer. As a result, the solution of the continuous relaxation of the model results in integer Solutions. Therefore, we solve the corresponding continuous nonconvex model and obtain a local optimal solution. We solve the model on various real data sets and observe that the run times are in the order of seconds for problems with 500 instances partitioned in 5 clusters. This indicates the potential to solve larger data sets. Also, we give interpretations that show a meaningful segmentation analysis is achievable with the proposed clustering model.

The topics that remain to be explored in the future include improving the reassignment algorithm and investigation of further methods to solve the *SWGD* model to global optimality.

APPENDIX A

Proposed Algorithm *SWGD Model* *K-Means Algorithm*

T1

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	2.46	2.58	2.50	2.76	0	5.01	0.62	1.18	2.06	0	3.80	0.10	0.68	9.14
2	2.46	0	0.16	4.46	0.37	5.01	0	5.19	6.26	6.75	3.80	0	3.30	2.80	14.64
3	2.58	0.16	0	4.74	0.55	0.62	5.19	0	0.46	1.34	0.10	3.30	0	0.69	7.50
4	2.50	4.46	4.74	0	4.66	1.18	6.26	0.46	0	0.98	0.68	2.80	0.69	0	9.21
5	2.76	0.37	0.55	4.66	0	2.06	6.75	1.34	0.98	0	9.14	14.64	7.50	9.21	0

T2

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	2.34	9.14	1.52	0.16	0	1.74	0.47	9.63	0.98	0	6.84	3.53	2.01	7.69
2	2.34	0	5.81	0.42	2.20	1.74	0	1.36	6.11	0.98	6.84	0	1.53	4.54	16.67
3	9.14	5.82	0	5.22	8.99	0.47	1.36	0	9.50	0.51	3.53	1.53	0	1.88	9.77
4	1.52	0.42	5.22	0	1.40	9.62	6.11	9.50	0	9.46	2.01	4.54	1.88	0	9.80
5	0.16	2.20	8.99	1.40	0	0.98	0.98	0.51	9.46	0	7.69	16.67	9.77	9.86	0

T3

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	0.51	0.90	2.50	3.66	0	4.36	2.44	1.58	4.76	0	0.53	1.89	4.17	2.78
2	0.51	0	1.18	2.79	3.83	4.36	0	2.72	4.78	2.01	0.53	0	2.28	4.39	3.02
3	0.90	1.18	0	1.18	2.73	2.44	2.72	0	3.23	3.86	1.89	2.28	0	2.86	1.39
4	2.50	2.79	1.18	0	1.90	1.58	4.78	3.23	0	4.97	4.17	4.39	2.86	0	1.40
5	3.66	3.83	2.73	1.90	0	4.76	2.01	3.86	4.97	0	2.78	3.02	1.39	1.40	0

T5

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	2.77	5.11	2.27	9.97	0	2.73	2.79	3.95	6.14	0	2.83	4.76	4.98	3.60
2	2.77	0	1.74	0.13	12.16	2.73	0	0.26	0.61	8.83	2.83	0	1.98	7.46	5.56
3	5.11	1.74	0	1.64	16.76	2.79	0.26	0	0.23	8.39	4.76	1.98	0	9.11	6.88
4	2.27	0.13	1.64	0	10.22	3.95	0.61	0.23	0	11.46	4.98	7.46	9.11	0	0.39
5	9.97	12.16	16.76	10.22	0	6.14	8.83	8.39	11.46	0	3.60	5.56	6.88	0.39	0

T6

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	15.9	20.0	2.24	20.0	0	1.48	10.7	22.1	22.1	0	22.1	0.11	22.1	0.58
2	15.9	0	2.15	10.1	2.15	1.48	0	8.8	13.0	13.0	22.1	0	11.5	0	5.67
3	20.0	2.15	0	12.1	0	10.7	8.89	0	3.86	3.86	0.11	11.5	0	11.5	0.47
4	2.24	10.1	12.1	0	12.1	22.1	13.0	3.86	0	0	22.1	0	11.5	0	5.67
5	20.0	2.15	0	12.1	0	22.1	13.0	3.86	0	0	0.58	5.67	0.47	5.67	0

T7

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	8.92	14.21	14.68	3.14	0	13.43	13.79	14.69	6.05	0	12.78	12.81	5.10	5.43
2	8.92	0	4.01	4.73	5.30	13.43	0	1.38	0	5.46	12.78	0	1.95	6.06	3.73
3	14.21	4.01	0	0.83	9.73	13.79	1.38	0	1.44	6.48	12.81	1.95	0	7.21	5.03
4	14.68	4.73	0.83	0	10.33	14.69	0	1.44	0	5.74	5.10	6.06	7.21	0	1.25
5	3.14	5.30	9.73	10.33	0	6.05	5.46	6.48	5.74	0	5.43	3.73	5.03	1.25	0

T8

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	9.53	12.73	13.18	3.39	0	6.20	11.74	9.44	3.74	0	11.29	10.64	5.05	5.87
2	9.53	0	2.29	2.16	3.96	6.20	0	4.47	2.45	1.73	11.29	0	0.47	3.60	2.69
3	12.73	2.29	0	0.23	6.00	11.74	4.47	0	2.18	5.87	10.64	0.40	0	3.24	2.35
4	13.18	2.16	0.23	0	5.98	9.44	2.45	2.18	0	4.04	5.05	3.60	3.24	0	0.73
5	3.39	3.96	6.00	5.98	0	3.74	1.73	5.87	4.04	0	5.87	2.69	2.35	0.73	0

T9

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	14.05	0.56	6.47	3.58	0	5.05	19.37	3.41	3.32	0	11.60	8.23	3.16	14.93
2	14.05	0	12.71	3.25	17.66	5.05	0	6.76	1.94	7.06	11.60	0	0.65	13.70	3.04
3	0.56	12.71	0	6.48	2.69	19.37	6.76	0	10.94	20.11	8.23	0.61	0	9.81	1.50
4	6.47	3.25	6.48	0	9.27	3.41	1.94	10.94	0	5.82	3.10	13.70	9.81	0	16.87
5	3.58	17.66	2.69	9.27	0	3.32	7.06	20.11	5.82	0	14.93	3.04	1.50	16.87	0

T11

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	4.87	3.88	2.37	3.24	0	1.71	4.46	5.26	1.64	0	4.99	3.64	0.11	1.55
2	4.87	0	0.12	1.94	8.09	1.71	0	2.66	3.48	3.28	4.99	0	0.75	4.95	6.32
3	3.88	0.12	0	1.54	6.33	4.46	2.66	0	0.88	6.05	3.64	0.75	0	3.67	4.87
4	2.37	1.94	1.56	0	5.08	5.26	3.48	0.88	0	6.79	0.16	4.95	3.67	0	1.32
5	3.24	8.09	6.39	5.03	0	1.64	3.28	6.05	6.79	0	1.55	6.32	4.86	1.32	0

T12

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	1.36	3.86	2.56	2.82	0	3.98	3.01	4.83	0	0	1.79	2.79	0.94	2.91
2	1.36	0	4.46	3.36	1.28	3.98	0	1.11	1.73	3.98	1.79	0	1.14	0.95	3.54
3	3.86	4.46	0	1.64	4.95	3.01	1.11	0	2.68	3.01	2.79	1.13	0	2.05	4.19
4	2.56	3.36	1.64	0	4.05	4.83	1.73	2.68	0	4.83	0.94	0.95	2.05	0	3.25
5	2.82	1.28	4.95	4.05	0	0	3.98	3.01	4.83	0	2.91	3.54	4.19	3.24	0

T13

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	0.54	4.48	2.50	1.01	0	2.82	0.53	5.39	1.04	0	3.49	3.36	0.56	4.49
2	0.54	0	3.55	1.78	1.33	2.82	0	3.14	2.46	3.87	3.49	0	0.50	2.69	7.37
3	4.48	3.55	0	1.65	5.00	0.53	3.14	0	5.56	0.40	3.36	0.50	0	2.75	6.26
4	2.50	1.78	1.65	0	3.11	5.39	2.46	5.56	0	6.50	0.56	2.69	2.75	0	4.36
5	1.01	1.33	5.00	3.11	0	1.04	3.87	0.40	6.50	0	4.49	7.37	6.26	4.36	0

T14

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	10.76	4.99	7.39	3.79	0	7.02	10.19	9.46	4.20	0	9.85	5.68	3.23	7.01
2	10.76	0	4.56	1.02	5.44	7.02	0	1.42	0.60	2.93	9.85	0	1.50	6.19	0.23
3	4.99	4.56	0	2.72	0.95	10.19	1.42	0	1.03	4.99	5.68	1.50	0	3.05	1.35
4	7.39	1.02	2.72	0	3.52	9.46	0.60	1.03	0	4.18	3.23	6.19	3.05	0	4.49
5	3.79	5.44	0.95	3.52	0	4.20	2.93	4.99	4.18	0	7.01	0.23	1.35	4.49	0

T15

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	14.5	25.2	8.75	28.1	0	5.91	14.28	24.15	29.39	0	19.09	6.03	25.19	3.54
2	14.5	0	5.39	3.40	8.67	5.91	0	6.04	10.75	15.42	19.09	0	6.99	7.17	9.02
3	25.2	5.39	0	8.65	4.03	14.28	6.04	0	4.08	9.30	6.03	6.99	0	11.63	1.88
4	8.75	3.40	8.65	0	11.5	24.15	10.75	4.08	0	6.51	25.19	7.15	11.63	0	13.53
5	28.1	8.67	4.03	11.5	0	29.39	15.42	9.30	6.51	0	3.54	9.08	1.88	13.53	0

T16

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	6.86	7.65	10.03	1.94	0	8.11	11.12	7.16	0.45	0	8.46	9.61	1.06	1.27
2	6.86	0	0.83	3.11	6.67	8.11	0	2.18	1.97	5.11	8.46	0	2.96	6.98	4.73
3	7.65	0.83	0	2.26	7.33	11.12	2.10	0	4.50	7.17	9.61	2.96	0	8.58	6.68
4	10.03	3.11	2.26	0	9.31	7.16	1.97	4.39	0	3.78	1.06	6.98	8.58	0	1.79
5	1.94	6.67	7.33	9.31	0	0.45	5.11	7.17	3.78	0	1.27	4.73	6.68	1.79	0

T17

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	14.6	21.6	2.31	23.5	0	0	9.97	20.1	21.9	0	15.3	0.60	20.0	2.51
2	14.6	0	6.00	11.9	5.55	0	0	10.4	21.1	23.2	15.3	0	15.6	4.14	10.3
3	21.6	6.00	0	18.5	1.05	9.97	10.4	0	9.42	10.2	0.60	15.6	0	20.1	2.99
4	2.31	11.9	18.5	0	19.7	20.1	21.1	9.42	0	0.08	20.0	4.12	20.1	0	13.8
5	23.5	5.55	1.05	19.7	0	21.9	23.2	10.2	0.08	0	2.51	10.3	2.99	13.8	0

T18

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	3.01	4.53	1.83	1.09	0	0.05	0.79	3.39	1.76	0	3.37	0.48	0.07	3.16
2	3.01	0	0.91	0.59	4.25	0.05	0	0.78	3.08	1.47	3.37	0	2.02	3.72	0.51
3	4.53	0.91	0	1.39	6.30	0.79	0.78	0	2.19	2.46	0.48	2.02	0	0.46	2.16
4	1.83	0.59	1.39	0	2.69	3.39	3.08	2.19	0	6.17	0.07	3.72	0.46	0	3.33
5	1.09	4.25	6.30	2.69	0	1.76	1.47	2.46	6.17	0	3.16	0.51	2.16	3.33	0

T19

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	2.37	4.32	1.85	0.19	0	1.82	1.18	3.83	0.31	0	3.88	1.66	2.09	1.66
2	2.37	0	1.78	3.42	2.52	1.82	0	2.75	5.22	2.67	3.88	0	5.26	1.85	5.26
3	4.32	1.78	0	5.35	4.48	1.18	2.75	0	2.67	0.99	1.66	5.26	0	3.59	0
4	1.85	3.42	5.35	0	1.78	3.83	5.22	2.67	0	3.77	2.09	1.85	3.59	0	3.59
5	0.19	2.52	4.47	1.78	0	0.31	2.67	0.99	3.77	0	1.66	5.26	0	3.59	0

T20

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	3.51	8.89	5.33	0.69	0	3.98	2.41	8.55	1.35	0	6.49	6.14	3.64	0.18
2	3.51	0	5.05	1.84	2.45	3.98	0	2.07	4.32	2.14	6.49	0	0.33	2.48	6.03
3	8.89	5.05	0	3.10	7.50	2.41	2.07	0	6.68	0.40	6.14	0.33	0	2.63	5.79
4	5.33	1.84	3.10	0	4.21	8.55	4.32	6.68	0	6.33	3.64	2.48	2.63	0	3.30
5	0.69	2.45	7.50	4.21	0	1.35	2.14	0.40	6.33	0	0.18	6.03	5.79	3.30	0

T21

	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	0.44	0.23	2.14	2.67	0	1.12	3.70	1.58	1.58	0	1.46	3.72	1.90	1.90
2	0.44	0	0.60	1.71	2.46	1.12	0	2.71	2.44	2.44	1.46	0	2.80	2.83	2.83
3	0.23	0.60	0	2.19	1.80	3.70	2.71	0	4.69	4.69	3.72	2.80	0	4.36	4.36
4	2.14	1.71	2.19	0	3.51	1.58	2.44	4.69	0	0	1.90	2.83	4.36	0	0
5	2.67	2.46	1.80	3.51	0	1.58	2.44	4.69	0	0	1.90	2.83	4.36	0	0

	T22														
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	1.01	0.85	3.47	2.42	0	1.30	0.74	0	3.52	0	1.09	2.17	2.55	1.31
2	1.01	0	0.19	2.61	2.82	1.30	0	0.56	1.28	3.55	1.09	0	1.50	3.33	0.63
3	0.85	0.19	0	2.82	2.83	0.74	0.56	0	0.72	3.33	2.17	1.50	0	3.39	0.74
4	3.47	2.61	2.82	0	4.62	0	1.28	0.79	0	3.34	2.55	3.33	3.39	0	2.61
5	2.42	2.82	2.83	4.62	0	3.52	3.55	3.33	3.34	0	1.31	0.63	0.74	2.61	0

	T23														
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	1.32	1.47	3.79	1.71	0	3.59	1.39	1.59	2.53	0	1.06	2.47	1.13	5.13
2	1.32	0	0.05	4.70	2.78	3.59	0	2.15	5.16	1.03	1.06	0	3.49	2.22	6.55
3	1.47	0.05	0	5.12	3.00	1.39	2.15	0	2.96	1.12	2.47	3.49	0	1.41	1.86
4	3.79	4.70	5.12	0	1.66	1.59	5.16	2.96	0	4.10	1.13	2.22	1.41	0	3.80
5	1.71	2.78	3.00	1.66	0	2.53	1.03	1.12	4.10	0	5.13	6.55	1.86	3.80	0

BIBLIOGRAPHY

[1] H.R. Nemati, and C.D. Barko, Enhancing Enterprise Decisions Through Organizational Data Mining, Journal of Computer Information Systems, Summer (2002), 21-28.

[2] R. Grossman, S. Kasif, R. Moore, D. Rocke, and J. Ullman, Data Mining Research: Opportunities and Challenges A Report of three NSF Workshops on Mining Large, Massive, and Distributed Data, Downloadable from the address: http://www.rgrossman.com/dl/misc-001.pdf, NCDM report, (1999).

[3] U. Fayyad, S. Chaudhuri and P. Bradley, Data Mining and Its Role in Database Systems, VLBD 2000 Tutorial, (2000).

[4] C. Rygielski, J.C Wang. and D.C. Yen, Data mining techniques for customer relationship management, Technology in Society, 24 (2002), 483-502.

[5] H. Hwang, T. Jung and E. Suh, An LTV model and customer segmentation based on customer value: a case study on the wireless telecommunication industry, Expert Systems with Applications, 26 (2004), 181-188.

[6] B. Padmanabhan, and A. Tuzhilin, On the Use of Optimization for Data Mining: Theoretical Interactions and eCRM Opportunities, Management Science, 49 (2003), 1327-1343.

[7] R.J. Roiger, and M.W. Geatz, Data Mining A Tutorial-Based Primer, Addison Wesley, San Francisco, USA, (2003).

[8] B.S. Everitt, S. Landau and M. Leese, Cluster Analysis, Arnold, Replika Press, Kundli, India, (2001).

[9] P.S. Bradley, U.M. Fayyad, and O.L., Mangasarian, Mathematical Programming for Data Mining: Formulations and Challenges, Journal on Computing, 11 (1999), 217-238.

[10] T. Zhang, R. Ramakrishnan, and M. Livny, BIRCH: A New Data Clustering Algorithm and Its Applications, Data Mining and Knowledge Discovery, 1 (1997), 141-182.

[11] S. Sharma, Applied Multivariate Techniques, John Wiley & Sons, USA, (1996).

[12] A. Likas, N. Vlassis and J.J. Verbeek, The global K-Means clustering algorithm, Pattern Recognition, 36 (2003), 451-461.

[13] W.D. Fisher, On Grouping for Maximum Homogeneity, Journal of the American Statistical Association, 53 (1958), 789-798.S.Z.

[14] S.Z. Selim and M.A. Ismail, K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality, IEEE Transactions on Pattern Analysis and Machine Intelligence, 6 (1984), 81-86.

[15] E. Forgy, Cluster Analysis of Multivariate Data: Efficiency Versus Interpretability of Classifications, Biometrics 21, 3 (1965), 768.

[16] P.S. Bradley, O.L. Mangasarian and W.N. Street, Clustering via Concave Minimization, Advances in neural information processing systems, 9 (1997), 368-374.

[17] G. P. Babu and M. N. Murty, A near-optimal initial seed value selection in K-means algorithm using a genetic algorithm, Pattern Recognition Letters, 14 (1993), 763-769.

[18] D. Fisher, Knowledge Acquisition via Incremental Conceptual Clustering, Machine Learning, 2 (1987), 139-172.

[19] R. Ng and J. Han, CLARANS: A Method for Clustering Objects for Spatial Data Mining, IEEE Trans. Knowledge and Data Engineering , 14 (2002), 1003-1016.

[20] I.H. Witten and E. Frank, Data mining: Practical machine learning tools and techniques with Java implementation, Morgan Kaufmann, San Francisco, CA, (2000).

[21] W.L.G. Koontz, P.M. Narendra and K. Fukunaga, A Branch and Bound Clustering Algorithm, IEEE Transactions on Computers, C-24(9) (1975), 908-915.

[22] H.D. Vinod, Integer Programming and the Theory of Grouping, Journal of the American Statistical Association, 64 (1969), 506-519.

[23] M.R. Rao, Cluster Analysis and Mathematical Programming, Journal of the American Statistical Association, 66 (1971), 622-626.

[24] M.J. Brusco, An enhanced branch-and-bound algorithm for a partitioning problem, British Journal of Mathematical and Statistical Psychology, 56 (2003), 83-92.

[25] G. Klein, and J.E. Aronson, Optimal Clustering: A model and method, Naval Research Logistics, 38 (1991), 447-461.

[26] C. E. Banfield and L.C. Bassil, A transfer algorithm for nonhierarchical classification, Applied Statistics, 26 (1977), 206-210.

[27] G. Diehr, Evaluation of A Branch and Bound Algorithm for Clustering, SIAM Journal of Sci. Stat. Comp., 6 (1985), 268-284.

[28] D.S. Hochbaum and D.B. Shmoys, A Unified Approach to Approximation Algorithms for Bottleneck Problems, Journal of the Association for Computing Machinery, 33 (1986), 533-550.

[29] OPL Studio, Version 3.6.1, ILOG, Inc.

[30] Matlab Version 7.0, MathWorks, Inc.

[31] M.A. Duran and I.E. Grossmann, An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs, Mathematical Programming, 36 (1986), 307-339.

[32] O.K. Gupta and A. Ravindran, Branch and Bound Experiments in Convex Nonlinear Integer Programming, Management Science, 31 (1985), 1533-1546.

[33] T. Westerlund and F. Petersson, A Cutting Plane Method for Solving Convex MINLP Problems, Computers Chemical Engineering, 19 (1995), 131-136.

[34] A.M. Geoffrion, A Generalized Benders Decomposition, Journal of Optimization Theory and Applications, 10 (1972), 237-260.

[35] GAMS IDE, Version 2.0.30.1, Gams Development Corporation.

[36] G.R. Kocis and I.E. Grossman, Computational Experience with DICOPT: Solving MINLP Problems in Process Systems Engineering, Computers Chemical Engineering, 13 (1989), 307-315.

[37] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, Network Flows: Theory, Algorithms and Applications, Prentice Hall, New Jersey, (1993).

VITA

Burcu Sağlam was born in Denizli, Turkey, on August 29, 1981. She graduted from Denizli Anatolian High School in 1999. She received her B.S. degree in Industrial Engineering from İstanbul Technical University (ITU), Istanbul, in 2003. Same year, she joined the M.S. program in Industrial Engineering at Koç University as a research and teaching assistant. Next year, Ms. Sağlam will be a Ph.D. candidate at Ecole Polytechnique Federale de Lausanne (EPFL).