<u>İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY</u>

GRAPH BASED SEQUENCE CLUSTERING THROUGH MULTIOBJECTIVE EVOLUTIONARY ALGORITHMS

M.Sc. Thesis by

Gül Nildem DEMİR

Department :	**Computer Engineering**
Programme:	**Computer Engineering**

JUNE 2008

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

GRAPH BASED SEQUENCE CLUSTERING THROUGH MULTIOBJECTIVE EVOLUTIONARY ALGORITHMS

M.Sc. Thesis by
Gül Nildem DEMİR
(504061526)

Date of submission : **5 May 2008**

Date of defence examination: **10 June 2008**

Supervisor (Chairman): **Asst.Prof.Dr. A. Şima ETANER UYAR**

Members of the Examining Committee **Prof.Dr. Coşkun Sönmez (YTÜ)**

Asst.Prof.Dr. Şule GÜNDÜZ ÖĞÜDÜCÜ (İTÜ)

JUNE 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÇOKAMAÇLI EVRİMSEL ALGORİTMALARLA ÇİZGE TABANLI SIRALI DİZİ DEMETLEME

YÜKSEK LİSANS TEZİ

Gül Nildem DEMİR
(504061526)

Tezin Enstitüye Verildiği Tarih : 5 Mayıs 2008
Tezin Savunulduğu Tarih : 10 Haziran 2008

Tez Danışmanı :	**Yrd. Doç. Dr. A. Şima ETANER UYAR**
Diğer Jüri Üyeleri	**Prof. Dr. Coşkun SÖNMEZ (YTÜ)**
	Yrd. Doç. Dr. Şule GÜNDÜZ ÖĞÜDÜCÜ (İTÜ)

Haziran 2008

ACKNOWLEDGEMENTS

First and foremost, I would like thank to my advisor Asst. Prof. Dr. Şima Etaner. I would not even begin with the M.Sc program if she had not trusted me and offered to work with her in the first place. I would like to express my sincere gratitude to her and Asst. Prof. Dr. Şule Gündüz Öğüdücü for their guidance, patience and understanding during the course of my work.

I would like to thank to Murat Göksedef, not as a coworker but as a friend, for his continous help, support and encouragement.

Finally, I want to express my good wishes to all members of the research laboratory-2 for creating such a nice working environment.

May 2008 **Gül Nildem DEMİR**

CONTENTS

ABBREVIATIONS

CO	: Connectivity
DB	: Davies-Bouldin
DE	: Direct Encoding
EA	: Evolutionary Algorithm
EC	: Evolutionary Computing
DM	: Decision Maker
GS	: Global Silhouette Index
LAR	: Locus-based Adjacency Representation
MMC	: Min-Max Cut
MOCK	: Multiobjective Clustering with Automatic K-determination
MST	: Minimum Spanning Tree
MOEA	: Multiobjective Evolutionary Algorithm
MOP	: Multiobjective Optimization Problem
NSGA-II	: Nondominated Sorting Genetic Algorithm II
OD	: Overall Deviation
PESA-II	: Pareto Enevelope-based Selection Algorithm-II
RI	: Random Initialization
SPEA2	: Strength Pareto Evolutionary Algorithm 2

TABLE LIST

FIGURE LIST

SYMBOLS

μ	: Cluster medoid
I_H	: Hypervolume Indicator
I_ϵ	: Epsilon Indicator
I_{R2}	: R2 Indicator
α	: Significance level

GRAPH BASED SEQUENCE CLUSTERING THROUGH MULTIOBJECTIVE EVOLUTIONARY ALGORITHMS

SUMMARY

Clustering is grouping similar data items in an unlabelled data set. As a result of a meaningful clustering, items within a cluster will be more similar to each other than to the items in other clusters. Clustering can be seen as a data mining technique summarizing the data. Traditional clustering algorithms usually work on data sets given in metric space as multidimensional vectors. However for many types of data such representation would be either expensive or insufficient. For sequences, the order of items in the sequence is very important. Therefore it would be better to describe sequence data as pairwise similarities/dissimilarities, which are calculated based on a similarity metric preserving the structural information of the sequences. Examples of this type of data appears in many domains such as bioinformatics, chemistry, computer vision and Web mining.

It is possible to represent the sequence data through a weighted, undirected graph. Each sequence becomes a vertex of the graph and the pairwise similarities or dissimilarities form the edges connecting the corresponding vertices in the graph. This graph-based representation of the sequence data maps the sequence clustering problem onto graph partitioning problem. To partition the graph into subgraphs properties of a graph are used. However graph partitioning is an NP-hard problem. Evolutionary algorithms (EAs), which are population based search and optimization methods inspired from Darwin's evolutionary theory, are proven to be successful at solving NP-hard problems. The key process in an EA is evolving a population in many generations. A population consists of individuals representing possible solutions of the problem and the individuals are encoded in some way. Principle components of EAs are representation and initialization schemes, a fitness (or objective) function, genetic operators (cross-over and mutation) and termination criteria. Multiobjective evolutionary algorithms (MOEAs) are special cases of EAs, optimizing more than one objective. Since the objective functions in MOEAs usually conflict with each other, a MOEA may return many optimal solutions, none of them better than the others in all of the objectives. The set of these solutions are approximation to the so called Pareto front. Strength Pareto Evolutionary Algorithm 2 (SPEA2), Nondominated Sorting Genetic Algorithm II (NSGA-II) and Pareto Envelope-based Selection Algorithm-II (PESA-II) are successful MOEAs in the literature.

There is one promising MOEA applied to the clustering problem of similarity based data: MultiObjective Clustering with automatic K-determination Around Medoids (MOCK-am). The underlying MOEA of the clustering algorithm is PESA-II. MOCK-am optimizes the two objectives overall deviation (OD) and connectivity (CO). OD measures whether the clustering consists of compact clusters, where the data items are really similar to the their cluster medoids. CO examines whether neighboring items are in the same cluster. Both objectives are conflicting with each other. The individuals are represented by a graph based representation scheme called locus-based adjacency representation. Each individual contains N genes where N is the total number data items. The gene j can take values between 1 and N and the value i of gene j means that there is link between data items i and j and consequently they are in the same cluster. The individuals are initialized partially by a method based on minimum spanning trees and partially by k-medoid algorithm. Genetic

operator set consists of uniform cross-over and restricted nearest-neighbor mutation where an item can only be linked to one of its L nearest neighbors.

In this work we propese a MOEA for graph clustering problem called GRaph-based Sequence Clustering algorithm (GraSC). GraSC is primarily based on SPEA-2 as MOEA. The objectives of GraSC are min-max cut (MMC) and the global silhouette index (GS). MMC aims to maximize the similarity within each subgraph while trying to minimize the similarity between the subgraphs. GS is actually a cluster validation index and can be used to compare the qualities of clustering solutions with different number of clusters. It indicates how well each object has been classified to its assigned cluster as compared to the other possible clusters in the data set. Individuals are directly encoded and initialized randomly. In direct encoding, each individual contains N genes where N is the total number of vertices (nodes) in the graph. Each gene corresponds to a vertex and the value of the gene denotes the cluster number the vertex is placed in. The genetic operator set consists of a heuristic cross-over, standard mutation and a heuristic disband operator.

One advantage of MOCK-am and the proposed graph clustering algorithm is that they do not expect a cluster count parameter beforehand. Unlike other traditional algorithms the user does not need to have an overview of the data before starting the clustering process. Besides, algorithms return many solutions with different cluster counts. The solution which fits the user's needs the best can be selected as the final solution. Both clustering algorithms have different MOEAs, initialization and representation methods, evolutionary operators and objective functions. To see the individual effects of all these genetic components different variations of MOCK-am and GraSC are implemented. In order to select the best MOEA variation for clustering first each variation is run several times. The approximation sets are transformed into real values by using quality indicators. Three different quality indicators are implemented: Hypervolume Indicator I_H, Unary Epsilon Indicator I_ϵ and R2 Indicator I_{R2} from R Indicator Family. After approximation sets are transformed into real values, a standard nonparametric statistical testing method can be applied to examine the statistical significance. Kruskal-Wallis test is chosen to compare multiple variations. This test compares the quality indicator results of variation pairs and shows whether one variation is significantly better than the other based on a significance level α. Following this testing procedure compares only variations with same objective set. Namely at the end there exists two best variations: one variation with overall deviation and connectivity objectives and one variation with min-max cut and the global silhouette index. The next step for analysis of MOEAs for clustering is to determine which algorithm generates the best clustering among these two variations. For this purpose, single solutions are identified from the combined Pareto fronts of multiple approximations sets for best two variations and Davies-Bouldin index measures the quality of both clusterings. Moreover, the data sets are clustered using a deterministic graph clustering algorithm in Cluto package which performs $k - 1$ repeated bisections if the cluster count k is given as input parameter.

For min-max cut and global silhouette index objective set the best variation has been the so called GND variation which consists of NSGA-II as MOEA, direct encoding, MST and k-medoid based initialization method, and default operators of GraSC. The best variation with the overall deviation and connectivity is the MPM variation which consists of PESA-II as MOEA, locus-based adjacency representation, MST and k-medoid based initialization method, and default operators of MOCK-am. For both variations and for all of the data sets single solutions are selected for further clustering evaluation. The solutions are identified based on knee identification on the Pareto front of combined approximation sets. The deterministic graph clustering algorithm in Cluto package is run on the data sets for the same cluster counts of these selected solutions. Both variations outperform the deterministic graph clustering algorithm. There was no significant difference between both variations.

ÇOK AMAÇLI EVRİMSEL ALGORİTMALARLA ÇİZGE TABANLI SIRALI DİZİ DEMETLEME

ÖZET

Demetleme etiketlenmemiş veri kümesi içindeki benzer nesneleri gruplamak olarak tanımlanır. Mantıklı bir demetlemenin sonunda demet içindeki nesneler birbirlerine diğer demetlerdeki nesnelerden daha çok benzer olacaklardır. Bu anlamda demetleme eldeki veriyi özetleyen bir veri madenciliği tekniği olarak görülebilir. Geleneksel demetleme algoritmaları genelde metrik uzayda çok boyutlu vektörler olarak ifade edilen veriler üzerinde işlem yapabilirler. Ancak birçok tip veri üzerinde bu temsil şekli ya çok maaliyetli ya da yetersiz kalacaktır. Sıralı diziler için sıralı dizi içindeki elemanların sıraları çok önemlidir. Bu yüzden sıralı dizilerin ikili benzerlikler olarak tanımlanmaları daha mantıklı olacaktır. İkili benzerlikler, sıralı dizilerin yapısal bilgilerini koruyacak bir metrikle hesaplanabilir. Bu tipte veri örnekleri ile biyoinformatik, kimya, bilgisayarlı görü, web madenciliği gibi birçok alanda karşılaşılmaktadır.

Sıralı dizileri ağırlıklı yönsüz bir çizge ile temsil etmek mümkündür. Böyle bir durumda her sıralı dizi çizgenin bir düğümü, ikili benzerlikler de çizgenin kenarları olurlar. sıralı dizilerin çizge tabanlı bu temsil biçimi sıralı dizi demetleme problemini çizge bölümleme problemine çevirir. Bir çizgenin alt-çizgelere ayrılmasında çizgenin özelliklerinden yararlanılır. Ancak çizge bölümleme NP-zor bir problemdir. Darwin'in evrim teorisinden ilham almış, populasyon tabanlı arama ve optimizasyon yöntemi olan evrimsel algoritmaların (EA) NP-zor problemlerin çözümünde başarılı oldukları kanıtlanmıştır. EA'daki anahtar süreç bir populasyonun birçok nesil boyunca evrim geçirmesidir. Populasyon ise problemin olası çözümlerini temsil eden ve bir şekilde kodlanmış bireylerden oluşur. EA'ların temel bileşenleri temsil ve başlangıç durumuna getirme yöntemleri, uygunluk ya da amaç fonksiyonu, genetik operatörler (çaprazlama ve mutasyon) ve sonlanma kriteridir. Çokamaçlı evrimsel algoritmalar (ÇAEA) EA'aların birden çok amaç fonksiyonunun optimize edildiği özel halleridir. ÇAEA'lardaki amaç fonksiyonlarï£¡ genelde birbirleri ile çeliştiklerinden, bir ÇAEA hiçbiri diğerinden iyi olmayan birden çok optimal çözüm üretebilirler. Tüm bu çözümlerin kümesi Pareto cephesi olarak adlandırılan gerçek optimal çözümler kümesine yakınmasama kümeleridir. Strength Pareto Evolutionary Algorithm 2 (SPEA2), Nondominated Sorting Genetic Algorithm II (NSGA-II) ve Pareto Enevelope-based Selection Algorithm-II (PESA-II) literatürde başarılı olarak kabul edilen ÇAEA'lardır.

İkili benzerlikler halinde ifade edilen verinin demetleme problemine uyarlanmış başarılı bir algoritma mevcuttur: MultiObjective Clustering with automatic K-determination Around Medoids (MOCK-am). Bu algoritma ÇAEA olarak PESA-II'ye dayanmaktadır. MOCK-am'in optimize etmeye çalıştığı iki amaç fonksiyonu genel sapma (GS) ve bağlanırlıktır (BA). GS, demetlemenin yoğun demetlerden oluşup oluşmadığını ölçer. Yoğun demet ise demet içindeki düğümlerin demet merkezine gerçekten yakın olduğu demetlerdir. BA ise komşu düğümlerin aynı demete düşüp düşmediklerine bakar. Her iki amaç fonksiyonu birbiriyle çelişir. Demetleme algoritmasında bireyler konum tabanlı bitişiklik temsili olarak isimlendirilen çizge tabanlı bir yöntemle temsil edilirler. Her birey toplam düğüm sayısı olan N adet gen içerir. j geni 1 ile N değer alabilir ve j geninin i değeri i ve j nesneleri arasındaki bağlantıyı dolayısıyla aynı demette olduklarını gösterir. Bireylerin bir kısmı minimum kapsayan ağaca dayanan bir yönteme, diğer kısmı

ise k-medoid algoritmasına göre başlangıç durumuna getirilirler. Genetik operatörler eş değerli çaprazlama ve sınırlı en yakın komşu mutasyonudur. Sözkonusu mutasyon operatörüne göre bir nesne ancak en yakın L komşusundan birine bağlanabilir.

Bu çalışmada GRaph-based Sequence Clustering algorithm (GraSC) olarak isimlendirilen çizge tabanlı bir ÇAEA önerilmektedir. GraSC'ın ilk olarak ÇAEA olarak SPEA2'ye dayanması tasarlanmıştr. GraSC'ın amaç fonksiyonları minimum-maksimum kesme (MMK) ve global siluet göstergesidir (SG). MMK alt-çizgelerdeki benzerliği enbüyüklemeye çalışırken, alt-çizgeler arasındaki benzerliği enküçüklemeyi hedefler. SG ise aslında bir demet geçerleme göstergesidir ve farklı demet sayılarına sahip demetlemelerin kalitelerinin karşılaştırılmasında kullanılır. Her nesnenin atanmış demetine diğer demetlere nazaran ne kadar iyi uyum sağladığını gösterir. GraSC'ta bireyler doğrudan kodlanmış rasgele başlangıç durumlarına getirilmişlerdir. Doğrudan kodlamada, her birey çizgedeki toplam düğüm sayısı olan N adet gen içerir. Her gen bölümülenecek çizgedeki bir düğüme karşılık düşer ve değeri düğümün atandığı demetin numarasını verir. Genetik operatör kümesi sezgisel bir çaprazlama operatörü, standart mutasyon ve sezgisel dağıtma operatöründen oluşur.

MOCK-am ve önerilen çizge tabanlı demetleme algoritmasının avantajı demet sayısına giriş parametresi olarak ihtiyaç duymamalarıdır. Diğer geleneksel demetleme algoritmalarının aksine kullanıcının demetlemeyi başlatmadan önce veri üzerinde fikir sahibi olmasına gerek yoktur. Ayrıca algoritmalar bir kere çalıştıklarında farklı özelliklere ve demet sayılarına sahip çözümler döndürürler. Kullanıcı kendi ihtiyacı doğrultusunda en uygun çözümü seçebilir. Her iki demetleme algoritmasının farklı ÇAEA'ları, başlangıç durumuna getirme ve temsil yöntemleri, genetik operatörleri ve amaç fonksiyonları vardır. Tüm bu genetik bileşenlerin etkilerini görmek için MOCK-am ve GraSC'ın çeşitli varyasyonları gerçeklenmiştir. Demetleme için en iyi ÇAEA varyasyonunu seçmek için öncelikle her varyasyon çok kez çalıştırılmıştır. Elde edilen yakınsama kümeleri çeşitli kalite göstergeleri ile gerçel sayılara dönüştürülmüştür. Üç farklı kalite göstergesi gerçeklenmiştir: Hiperhacim göstergesi I_H, Epsilon göstergesi I_ϵ ve R gosterge ailesinden R2 göstergesi I_{R2}. Yakınmasama kümeleri gerçel sayılara dönüştürüldükten sonra istatistiksel anlamı araştırmak için, parametrik olmayan istatistiksel test yöntemleri uygulanmıştır. Kruskal-Wallis testi varyasyonları bu şekilde karşılaştırmak için tercih edilmiştir. Sözkonusu test varyasyon çiftlerinin gösterge değerlerini istatistiksel olarak karşılaştırarak, bir vasyasyonun diğerinden bir anlam seviyesi α'ya göre anlamlı olarak daha iyi olup olmadığını gösterir. Bu test yöntemi sadece amaç fonksiyonları aynı olan varyasyonları karşılaştırabilir. Test sonucunda iki en iyi varyasyon bulunacaktır: Genel sapma ve bağlanırlılık amaç fonksiyonlarını kullanan varyasyon ve minimum-maksimum kesme ile global siluet göstergesi kullanan diğer bir varyasyon. Demetleme için ÇAEA'ların analizi için sonraki adım bu iki varyasyondan hangisinin daha kaliteli demetlemeler ürettiğidir. Bu amaç doğrultusunda en iyi iki varyasyon için yakınsama kümeleri birleştirilerek her biri için birer birleştirilmiş Pareto cephesi oluşturulur. Bu Pareto cephelerinden birer optimum çözüm seçilerek, çözümlerin Davies-Bouldin göstergesi ile kaliteleri ölçülür. Ayrıca karşılaştırma amaçlı olarak veri kümeleri Cluto algoritma paketine ait gerekirci bir çizge demetleme algoritması demetlenirler. Sözkonu algoritma demet sayısı k olarak belirlendiği takdirde tekrarlı olarak $k-1$ ikiye bölme yapar.

Minimum-maksimum kesme ile global siluet göstergesi amaç fonksiyon kümesi için en iyi varyasyon GND olarak isimlendirilmiş, ÇAEA olarak NSGA-II'ye dayanan, doğrudan kodlama, MST ve k-medoid tabanlı başlangıç durumuna getirme yöntemi, GraSC'ın operatör kümesini içeren algoritma olmuştur. Genel sapma ve bağlanırlılık amaç fonksiyon kümesi içinse en iyi varyasyon MPM olarak isimlendirilmiş, ÇAEA olarak PESA-II'ye dayanan, konum tabanlı bitişiklik temsili, MST ve k-medoid tabanlı başlangıç durumuna getirme yöntemi, MOCK-am'nin operatör kümesini içeren algoritma olmuştur. Her iki varyasyon ve tüm veri kümeleri için birer demetleme çözüme ileri demetleme değerlendirilmesi için seçilmiştir. Çözümler birleştirilmiş Pareto cephesine ait grafiklerden 'diz' belirleme yöntemine göre seçilmişlerdir. Cluto algoritma paketindeki gerekirci çizge demetle algoritması da seçilen çözümlerin demet sayıları için çalıştırılmıştır. Her iki ÇAEA varyasyonu gerekirci algoritmadan daha başarılı sonuç vermiştir. Ancak kendi aralarında anlamlı bir fark tespit edilememiştir.

1. INTRODUCTION

1.1 Problem Definition

Clustering is the partitioning of data items in a data set into groups, where items in each group are more similar to each other than items is other groups. For the clustering problem, a conventional way is to represent the items as multidimensional vectors where each dimension corresponds to a feature of the data [1]. However multidimensional vector representation is not suitable for every data set. For example data items in form of sequences can be transformed into d-dimensional binary vectors where d is total number items in the data set. With a huge d, this representation will be very expensive. Moreover, transforming sequences into numerical vectors will cause to lose the structural information like order of the items in the sequences. In that case it is more convenient to describe the data as pairwise similarities/dissimilarities. The similarities are measured prior to clustering by a metric taking the structural information of the data items into account. Examples of this type of data appears in many domains such as bioinformatics, chemistry, computer vision and Web mining.

One of the possibilities to represent the sequence data is through weighted, undirected graphs G. Each sequence becomes a vertex of the graph and the pairwise similarities or dissimilarities form the edges connecting the corresponding vertices in the graph. As a result of this graph-based representation of the sequence data, the sequence clustering problem is mapped onto graph partitioning problem. Properties of a graph can then be used to cluster sequences by constructing a set of subgraphs from G.

1.2 Overview of the Work

The graph partitioning problem is a NP-hard problem. Thus, studies mainly focus on developing heuristics to generate approximations of the optimal solution rather than to find the optimal solution itself. Evolutionary Algorithms (EAs) [2] are population based search and optimization methods inspired from Darwin's evolutionary theory. The main property of EAs is evolving a population which consists of candidate solutions of the problem. The evolutionary process is based on an objective function determining the quality of solutions.

It has been shown that EAs are good at producing approximate solutions for NP-hard problems. This work deals with the graph clustering problem and introduces a graph-based sequence clustering approach through multiobjective evolutionary algorithms (MOEAs), which is a subset of EAs with multiple objective functions. The resulting algorithm is named as GRaph based Sequence Clustering algorithm (GraSC).

A successful MOEA, MultiObjective Clustering with automatic K-determination Around Medoids (MOCK-am), already exists in literature for the sequence clustering of data given as pairwise similarities. One main difference between MOCK-am and GraSC is that MOCK-am does not treat the problem as graph partitioning. Moreover both approaches consist of different MOEA components which are strongly related with the performance of the algorithm. The components are the objective functions to be optimized, genetic representation, initialization method and genetic operators. To see the individual effects of these components, multiple variations of the algorithms are implemented by interchanging the components between the two algorithms. Through a statistical comparison procedure of the outcomes of these variations it will be possible to identify the best variations for the sequence clustering. In this study the MOEA variations are run multiple times and the results of the MOEAs, called approximations sets, are collected. The approximation sets are transformed into real numbers through quality indicators which are used in performance assessment of multiobjective optimizers. A nonparametric statistical test works on these indicator values and examines the statistical significance. As a result of the statistical testing stage two MOEA variations are identified, one variation for one objective set. The final decision on the performance of these MOEA variations is made through a cluster validity index called Davies-Bouldin index [3]. The cluster validity index can be applied to a single clustering solution. However the outcome of a MOEA, the approximation set, consists of many solutions corresponding to the different trade-offs of the objectives. Therefore a solution selection method based on the characteristics of the plot of the approximation set identifies the most “interesting solution” in an approximation set. The selected solutions become the inputs of Davies-Bouldin index and according to the value of the index the best variation is found.

1.3 Contributions of the Thesis

Traditional clustering algorithms work only with metric data given as multidimensional vectors. However it is more convenient to express the sequence data as pairwise similarities/dissimilarities to keep the structural information of the sequences. This work focuses on clustering of sequence data which is available in many domains.

Many existing algorithms require the cluster count as an input of the algorithm. In order to specify a cluster count, the user has to have an overview of the data before starting the clustering process. The proposed approach returns many solutions with different characteristics and cluster counts in a single run without the contribution of the user. The user can either select the clustering solution which fits to her/his needs the best or let the algorithm decide on an optimal solution automatically.

The performance of the proposed graph-based method has been tested on real-world data sets and in some data set it has been more successful than the existing sequence clustering algorithm. The results are promising and encourages for further experimental evaluation.

1.4 Organization of the Thesis

This thesis is organized as seven chapters. Chapter 2 summarizes the clustering problem and explores the sequence clustering in great detail. Since the proposed sequence clustering algorithms are based on MOEAs, Chapter 3 presents EAs and MOEAs. The sequence clustering algorithms including the new graph based clustering GraSC and the contestant algorithm MOCK-am are explained in Chapter 4 in great detail. GraSC and MOCK-am have different MOEAs, initialization and representation methods, evolutionary operators and objective functions. To see the individual effects of all these evolutionary components different variations of MOCK-am and GraSC are implemented. The details of the experimental evaluation and analysis procedure are described in Chapter 5. In Chapter 6 the results of the experiments are given based on three real-world data sets. Finally, the last chapter provides a summary and conclusion of the work, identifying the best MOEA variation for the clustering of similarity based data.

2. CLUSTERING

Clustering is simply defined as finding groups of similar items (patterns) in a data set. It is categorized as an unsupervised learning method. Thus, the input items are not labeled and the labels of each item, namely the clusters, are derived from the data itself. As a result of a meaningful clustering, items within a cluster will be more similar to each other than to the items in other clusters. According to [4], clustering process consists of the following steps:

1. Pattern representation
2. Definition of the data proximity measure
3. Clustering
4. Data abstraction (if necessary)
5. Assessment of output (if necessary)

Pattern representation involves the description of the data to be clustered. If the data is described by many attributes, the best way for its representation is examined. Pattern proximity between data item pairs is measured by a function. The simplest function would be the Euclidian distance if the patterns are represented as points in the metric space. Data abstraction is the modeling of the data set in terms of clusters. A sample abstraction can be using cluster centers or medoids. Different clustering methods generate different clusterings. Assessment of the output is an important aspect of the whole clustering process since different clusterings has to be compared for performance analysis.

Clustering algorithms can be classified in several ways. In the traditional taxonomy one clustering algorithm can be hierarchical, partitional or density based. A hierarchical clustering can be either agglomerative or divisive. An agglomerative method combines single items into small clusters and small clusters into bigger ones. A divisive algorithm starts from a single cluster containing all data items and divides the cluster into smaller ones until a criterion is satisfied. A partitional clustering method decomposes the data into clusters directly by optimizing a criterion. In density based approaches clusters are regarded as regions in the space in which the items are dense, and which are separated by regions of low items density (noise). The regions may have any kind of shape.

2.1 Graph Clustering Problem

For the clustering problem, a conventional way is to represent the items as multidimensional vectors where each dimension corresponds to a feature of the data [1]. However multidimensional vector representation is not suitable for every data set. For example data items in form of sequences can be transformed into d-dimensional binary vectors where d is total number items in the data set. With a huge d, this representation will be very expensive. Moreover, transforming sequences into numerical vectors will cause to lose the structural information like order of the items in the sequences. In that case it is more convenient to describe the data as pairwise similarities/dissimilarities. The similarities are measured prior to clustering by a metric taking the structural information of the data items into account. Examples of this type of data appears in many domains such as bioinformatics, chemistry, computer vision and Web mining.

One of the possibilities to represent the sequence data is through weighted, undirected graphs G. Each sequence becomes a vertex of the graph and the pairwise similarities or dissimilarities form the edges connecting the corresponding vertices in the graph. As a result of this graph-based representation of the sequence data, the sequence clustering problem is mapped onto graph partitioning problem. Properties of a graph can then be used to cluster sequences by constructing a set of subgraphs from G. In a valid partitioning of a graph the edges between partitions will have low weights and the weights of the edges in a partition will be higher.

2.1.1 Definitions and Notation

A graph G is defined as an ordered pair $G = (V, E)$ where V is a set of vertices (nodes) and $|V| = n$, E is a set of edges between the vertices and $|E| = m$. In a weighted graph a positive value is assigned to each edge and the weight of the graph is the sum of all edge weights. In an undirected graph the edges from vertices v_i to v_j and from v_j to v_i are equal and named as edge $e(v_i, v_j)$ with weight $w_{ij} = w_{ji}$. The adjacency (or similarity/weight) matrix W is an nxn square matrix, where w_{ij} is the weight between nodes v_i and v_j. For an undirected graph the adjacency matrix is symmetric.

The degree of a vertex is the number of edge points to that vertex. The degree matrix D is an nxn matrix where d_{ij} is the degree of v_i. All its entries other then diagonal elements are zero.

A cut (A, B) in graph G partitions the vertices V into two subsets A and B where $A \cup B = V$ and $A \cap B = \emptyset$. In a weighted graph, the size of a cut is the sum of weights of edges crossing the cut. If the size of the cut is minimum, it is called *min-cut*.

2.1.2 Common Graph Clustering Algorithms

Spectral Clustering

Spectral clustering algorithms, whose key concepts are introduced by Fiedler in 1973 [5], use algebraic properties of graphs. The basis of spectral clustering is the Laplacian of the graph adjacency matrix (or similarity matrix). The Laplacian L of a graph is $D - W$. L is symmetric and has nonnegative real valued eigenvalues with the smallest eigenvalue 0. The Laplacian, its eigenvectors and eigenvalues describe many properties of a graph. If the adjacency matrix W is given for n objects the common spectral clustering algorithm for k-clustering a graph, called unnormalized spectral clustering, operates as follows:

1. Calculate Laplacian L
2. Compute first k eigenvectors of L $e_1, e_2, \ldots, e_k$
3. Generate a matrix E, where its columns are eigenvectors $e_1, e_2, \ldots, e_k$
4. Let y_i be a vector containing the ith row of E, cluster the vectors y_i $i = 1, 2, \ldots, n$ with k-means algorithm into k clusters
5. Return clusters

In the algorithm above, the representation of the objects are transformed using algebraic properties of the corresponding graph and then a standard clustering technique is applied.

Spectral clustering can also be implemented as an approximation to the graph partitioning using graph cuts. The simplest way to partition a graph is solving the min-cut problem. However it tends to favor partitioning into subgraphs where a subgraph can be very small compared to the others. Various objective functions are introduced, such as ratio-cut [6], normalized-cut [7], and min-max cut [8] to overcome the issues related with the unbalanced partitions. The mentioned cuts are defined by the following formulas:

- Min-cut attempts to minimize:
  $MinCut(A_1, \ldots, A_k) = \sum_{i=1}^{k} cut(A_i, \bar{A}_i)$
- In ratio-cut the size of a subset A of a graph is measured by its number of vertices:
  $RatioCut(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{|A_i|}$

- In normalized-cut the size of a subset A of a graph is measured by the weights of its edges Vol(A):
  $NCut(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{Vol(A_i)}$
- Min-max cut is similar to normalized-cut, but the denominator contains the intra-cluster similarity instead of the sum of intra-cluster similarity and the cut:
  $$MinMaxCut(G) = \sum_{m=1}^{k} \frac{cut(G_m, G \setminus G_m)}{\sum_{v_i v_j \in G_m} E(v_i, v_j)}$$

Markov Clustering

Markov Clustering (MCL) is a graph clustering algorithm based on simulation of flow in graphs. According to MCL if there is a cluster in a graph, a random walk tends to visit many of its vertices. A random walk starts from a vertex and visits a random neighbor vertex with probability proportional edge weight between them. The random walk is implemented by changing a matrix of transition probabilities. In MCL two operations are applied alternatively until the transition matrix converges:

- Expansion is taking the eth power of the matrix, making e steps of the random walk in the current transition matrix. It models the spreading out of flow.
- Inflation is taking the rth of all entries in the matrix. It models the contraction of flow, becomes thicker in regions of higher current and thinner in regions of lower current.

At the end, multiple expansion and inflation operations result in the separation of the graph into different partitions. The clustering is identified by detecting the connected components in the final matrix.

3. MULTIOBJECTIVE EVOLUTIONARY ALGORITHMS

3.1 Evolutionary Algorithms

Evolutionary Algorithms (EAs) [2] are population based search and optimization methods inspired from Darwin's evolutionary theory. The origin of EAs goes back to fifties [9]. During sixties, three independent researches regarding the implementation of the Darwinian principles for automated problem solving were in progress: Evolutionary Programming by Fogel [10], Genetic Algorithm by Holland [11], Evolution Strategies by Rechenberg and Schwefel [12]. In the beginning of the ninties, these different works with the common idea united under the category of Evolutionary Computing (EC).

The common property of EAs is evolving a population in many generations. Each individual in the population represents a possible solution of the problem. The solution is coded into the genes of an individual. At each generation individuals are evaluated based on a fitness function which is an estimate of the solution quality and "fitter" individuals are selected to form the mating pool. A set of genetic operators (recombination and/or mutation) are applied to the selected individuals to create the offspring population. The offspring may replace the parent population according to their fitness values. This evolutionary process is repeated until a predetermined criteria is satisfied. The main flow of a simple EA is given in Algorithm 3.1. More detailed information abouts EAs can be found in [2].

Algorithm 1 The basic generational evolutionary algorithm.

```
1: Initialize population
2: while stopping criteria not met do
3:     Evaluate population
4:     Select mating pool
5:     Apply reproduction
6:     Create new population
7: end while
```

3.1.1 Components of EAs

Representation: The first step in solving a problem with EA is to specify a representation scheme for individuals. Some of the standard representations are strings, real-valued vectors, trees.

Fitness Function: Fitness function is a quantitive measure evaluating the optimality of the solutions in the population. Improvement of individuals at each generation is strongly related with the fitness function. Individuals with higher fitness values are more likely to be selected and therefore their chance to pass their genetic information to the next generations is high. The fitness function is determined according the goal of the algorithm and nature of the problem.

Initialization: The EA starts with the initialization of the population. A common way for initialization is doing random sampling in the search space. It is also possible to embed some known good solutions into the initial population.

Selection: In the selection step fitter individuals are selected for reproduction (mating). Standard selection techniques are the following:

- Roulette Wheel Selection: Individuals are selected randomly but with a probability proportional their fitness values. Fitter individual have better chance to be selected.
- Tournament Selection: Some individuals are chosen randomly and the fittest one is selected for crossover.

Crossover: The crossover operator combines the genetic information of two or more parents to create offspring. The underlying idea is to breed fitter children than their parents by exchanging data between parents. Several ways exists to perform crossover on string representations:

- One-point Crossover: One random crossover point on the chromosomes is selected and the genetic data of the parents after that point is exchanged between parents.
- Two-point Crossover: Two random points on the chromosomes are selected and the genetic data of the parents between these points is exchanged between parents.
- Uniform Crossover: The genes of the parent chromosomes are exchanged with a given probability.

For genes as real-valued number crossover can be in form of linear combination as long as the genetic information of both parent is passed to the offspring.

Mutation: Mutation changes the individuals randomly. For instance if individuals are represented as a real-valued vectors, mutation will increment/decrement the values of the genes from a probability distribution like Gaussian. The mutation operation aims at preserving the diversity in the population.

Parameters: An EA usually consists of many parameters to be set such as the population size, recombination and mutation rates, termination condition. Assigning a wrong value even to a single parameter may effect the results of the EA very badly. For example a high mutation rate may force the EA to act randomly. One conventional way is to tune the parameters manually by trying different possibilities. Other parameter setting techniques include statistical methods and using another EA to evolve parameters.

3.2 Multiobjective Optimization

If two or more objectives need to be optimized simultaneously, the problem is then called a multiobjective optimization problem (MOP). Single-objective optimization problems may have unique solutions. Since the objective functions in MOPs usually conflict with each other, a multiobjective optimization may return many optimal solutions, none of them better than the others in all of the objectives. The notion “optimality” in MOP is proposed by Francis Ysidro Edgeworth in 1881 and generalized by Vilfredo Pareto in 1896 and called since then *Pareto optimality*. In a MOP the decision maker (DM) usually selects solutions from Pareto optimal solutions, which correspond to different trade-offs of the objectives. Namely optimizing a vector of multiple objectives corresponds to finding a solution whose all objective function values are acceptable for a DM[13].

3.2.1 Definitions and Notation

A vector of *decision variables* represents a decision in a MOP. The quantities of the these variables are determined in the optimization problem. The vector x of n decision variables is written as:

$x = [x_1, x_2, x_3, ...x_\mathrm{n}]^\mathrm{T}$

where T is the transposition operator.

Objective functions $f_1(x), f_2(x), \ldots, f_k(x)$ are computable functions of decision variables, evaluating the quality of solutions in a MOP. The objective functions can be expressed as a k-dimensional vector $f(x)$, where k is the number of objectives:

$f(x) = [f_1(x), f_2(x), \ldots, f_k(x)]^\mathrm{T}$

A general MOP can be defined as:

$'minimize(maximize)'z = F(x)$, with $x \in \Omega$

where Ω is the decision space, namely the set of all possible solutions of the problem.

A solution $x \in \Omega$ is called Pareto optimal with respect to Ω if there is no $x' \in \Omega$ for which $F(x') = (f_1(x'), f_2(x'), \ldots, f_k(x'))$ dominates $F(x) = (f_1(x), f_2(x), \ldots, f_k(x))$. For a maximization problem, $u = F(x) = (u_1, u_2, \ldots, u_k)$ dominates $v = F'(x) = (v_1, v_2, \ldots, v_k)$ ($u \preceq v$) means that u is partially better than v, i.e. $\forall i \in 1, \ldots, k, u_i \geq v_i \wedge \exists i \in 1, 2, \ldots, k : u_i > v_i$.

For a MOP the Pareto optimal set is defined as:

$P* := \{x \in \Omega \mid \neg\exists x' \in \Omega F(x') \preceq F(x)\}$

The vectors of the Pareto optimal set are called *nondominated* and form the Pareto front when plotted in the objective space. A selected vector from this set is an acceptable solution or decision variable for the MOP. For a given MOP, $F(x)$ and Pareto optimal set $P*$, the *Pareto Front* $PF*$ (PF_{true}) is defined as:

$PF* := u = F(x) | x \in P*$

It is not possible to detect all points on $PF*$. The conventional approach is to determine many points of Ω, evaluate their $f(\Omega)$, extract the nondominated points and than construct the Pareto front.

3.3 Multiobjective Evolutionary Algorithms

EAs which are used to solve the MOPs are expressed as multiobjective evolutionary algorithms (MOEA). MOEAs can approximate the true Pareto front and find several Pareto optimal solutions in a single run. The main difference between a single-objective EA and a MOEA is in the fitness evaluation stage of the algorithm. In single objective case, the selection is carried out based on single objective values. However in MOEAs a transformation of objective vectors into scalars is necessary. The first MOEA is introduced by David Schaffer in mid-1980s. David Goldberg proposed the Pareto-based fitness assignment [14] to solve the problems in Schaffer's work [15]. Since then, MOEA has become an interesting research area in computer science.

MOEAs have introduced new definitions to the MOP terminology. At generation t of a MOEA a current set of solutions is named as $P_{current}(t)$. Some MOEAs keep a second population to store "good" solutions denoted as $P_{known}(t)$. $P_{known}(t)$ contains Pareto optimal solutions found until generation t. The corresponding Pareto fronts of $P_{current}$ and

P_{known} are $PF_{current}$ and PF_{known}. The true Pareto front P_{true} is not known and it is implicitly defined by the MOP functions.

3.3.1 MOEA Concepts

The MOEA approaches can be grouped in three categories according to the decision making process: A Priori Techniques, Progressive Techniques and A Posteriori Techniques. In *A Priori techniques* a DM defines the MOP objective relative importance before the search. Usually weights are assigned to the objectives and an aggregated sum technique is applied to solve the problem as a single-objective case. *Progressive techniques* require interaction between the DM and the algorithm. An interactive process might be very difficult if the nature of the problem is unknown. The decision making takes places during the search. *A Posteriori techniques* try to find the true Pareto front P_{true} by spreading the search as much as possible and using the Pareto dominance in the selection stage of the EA. The decision making process comes after the completition of the search.

MOEAs have four major goals to achieve [16]:

1. Preserve nondominated points with $PF_{current} \rightarrow PF_{known}$
2. Progress PF_{known} towards PF_{true}
3. Maintain diversity of points on the Pareto front PF_{known}
4. Return a limited number of Pareto optimal solutions from PF_{known} to the DM

The most important issues in a Pareto based MOEA approach are dominance based ranking and diversity preservation. The dominance operator compares two solutions to check whether one dominates the other. According to this dominance results, solutions in a population can be ranked using an approtiate dominance definition given below:

- Dominance Rank: The number of individuals which dominate an individual
- Dominance Count: The number of individuals which are dominated by an individual
- Dominance Depth: The front where an individual belongs to after sorting the population.

A MOEA should approximate to the known Pareto front as much as possible and the solution points should be distributed uniformly. To maintain such a diversity the techniques include:

- Weight Vector Approach: To spread the points, a vector set in the objective space is used. To introduce new directions to the search, the weights are changed.

- Fitness Sharing/Niching Approach: The solutions are assigned to certain niches in the objective space. The size of a niche is controlled through a parameter σ_{share} which is the maximum number of solutions in a niche. The fitness of the solutions in populated niches are worse than in incrowded niches. It is aimed to move solutions from most populated niches to the least populated ones in the search space.
- Crowding/Clustering Approach: The solutions are selected according to region crowdedness metric similar to fitness sharing technique.

For a MOEA several populations are defined. The $P_{current}$ is the nondominated solutions of the current generation and P_{known} is an archive population storing the nondominated solutions of all generations so far. The P_{known} can be seen as a MOEA necessity and kept as a separate population.

Finally the flow of a generic Pareto based MOEA can be given as in Algorithm 3.3.1

Algorithm 2 The generic MOEA

```
 1: Initialize populations $P$ and $P_{archive}$
 2: Evaluate $P$
 3: Assign ranks based on Pareto dominance
 4: Compute niche count
 5: Assign shared fitness or crowding
 6: while stopping criteria not met do
 7:     Select mating pool $P^i$
 8:     Apply reproduction
 9:     Create child population $P^{ii}$
10:     Evaluate child population $P^{ii}$
11:     Rank $P^i \cup P^{ii} = P^{iii}$ based on Pareto dominance
12:     Compute niche count
13:     Assign shared fitness or crowding
14:     Reduce $P^{iii}$ to $P$
15:     Copy $P^{iii}$ to $P_{archive}$ based on Pareto dominance
16: end while
```

According to the above pattern which is followed by the most MOEAs, first a population of individuals is initialized. A generational loop is executed with evolutionary operators and ranking of individuals while storing the nondominated solutions in a separate archieve population. The difference between MOEAs lies in the design of specific operators. More detailed information can be found in [16].

3.3.2 Successful MOEA Examples

As stated in “No Free Lunch” theorem [17], there is no single best MOEA valid for every type of MOP. However some of the MOEAs are proven to perform better than other algorithms: Strength Pareto Evolutionary Algorithm 2 (SPEA2), Nondominated Sorting Genetic Algorithm II (NSGA-II) and Pareto Envelope-based Selection Algorithm-II (PESA-II).

SPEA2

The original *SPEA* is introduced by Eckart Zitzler and Lothar Thiele [18]. SPEA uses a regular population P and an archive population $\bar{P}$ which keeps the nondominated solutions of the previous generation. At each generation, the nondominated individuals are copied to the archive and the dominated individuals are removed in return. If the archive exceeds its limit size, a truncation operator based on clustering deletes some individuals. For each individual i in the archive a strength value $S(i)$ is calculated. Strength is the number of population members dominated by ith archive member divided by the population size plus one. $S(i)$ becomes the fitness value $F(i)$ of individual i in the archive. For individuals in the standard population, the fitness values are calculated by using strength values in the archive: Fitness of member j is the summation of all strength values of archive members i which dominate or equal to j. In the mating selection step, parents are selected both from standard population and the archive by binary tournament selection based on their fitness values. After recombination and mutation the current population is replaced by the offspring. The performance results of SPEA are well, but some weak points exist in the algorithm:

- Fitness assignment: If archive contains only one individual, the fitness values in the population will be equal without taking the dominance relationships between the individuals into account.
- Density estimation: If many individuals in the population indifferent, namely do not dominate each other, density information has to be used.
- Archive truncation: The truncation operator uses clustering to remove nondominated solutions. However it may lose outer solutions which are needed for diversity maintenance.

The SPEA2 [19] is developed to overcome the issues above. To prevent the case of equal fitness assignments, SPEA2 takes both dominated and dominating individuals into account. This time, for individuals both in population P_t and in archive $\bar{P}_t$ strength values $S(i)$ are computed. Based on the $S(i)$ raw fitness values $R(i)$ are calculated as follows:

$R(i) = \sum S(i)_{j \in P_t + \bar{P}_t j \succ i}$

Namely, raw fitness $R(i)$ of an individual i is the summation of the strength values of individuals dominating it. Although this is a better fitness assignment mechanism than the one in SPEA, it can fail when many individuals don't dominate each other. To diffentiate individuals with same raw fitness, the density information is also inserted into the fitness calculation stage. For density estimation, for each individual i the distances to other individuals j in P and $\bar{P}$ are computed and sorted in increasing order. The kth distance (kth nearest neighbour) denoted as ${\sigma_i}^k$ is the value sought. k is usually taken as the square root of the population size, $k = \sqrt{N + \bar{N}}$ where N is the population size and $\bar{N}$ is the archive size. The density $D(i)$ of an individual is given as:

$D(i) = \frac{1}{{\sigma_i}^k + 2}$

And the final fitness $F(i)$ of individual i is the sum of its raw fitness and density: $F(i) = R(i) + D(i)$

At each generation the nondominated individuals both from P_t and $\bar{P}_t$ which have a fitness value lower than one are copied to the archive of the next generation $\bar{P}_{t+1}$. If the size of $\bar{P}_{t+1}$ is exactly $\bar{N}$, the environmental selection stage is completed. If archive is not full, the best dominated $\bar{N}- \mid \bar{P}_{t+1} \mid$ individuals in P_t and $\bar{P}_t$ are copied to archive of the next generation. If the archive is overfilled, the improved archive truncation operator removes individuals until $\bar{N} =\mid \bar{P}_{t+1} \mid$. An individual i from $\bar{P}_{t+1}$ is removed if it has the minimum distance to another individual. The flow of SPEA2 is given in Algorithm 3.3.2.

Algorithm 3 SPEA2

```
 1: Randomly initialize population $P_0$ and create empty archive population $\bar{P}_0$
 2: **while** *while max no of generations not reached* **do**
 3:     Calculate fitness values of individuals in $P_t$ and $\bar{P}_t$
 4:     Copy nondominated individuals in $P_t$ and $\bar{P}_t$ to $\bar{P}_{t+1}$
 5:     Select parents from $\bar{P}_{t+1}$ based on binary tournament selection
 6:     Recombination and mutation
 7:     Evaluate children
 8:     Place children in $P_{t+1}$
 9: **end while**
10: Return nondominated individuals in $\bar{P}_{t+1}$
```

NSGA-II

The NSGA is proposed by Srinivas and Deb in [20]. It is based on classification of individuals and generating several levels of classification. Prior to selection, all nondominated individuals are grouped and assigned a dummy fitness value. These are separated from the population and from the remaining individuals another nondominated group is created with another shared fitness value. This classification process (nondominating sorting) continues until all individuals belong to a certain nondomination level. All individuals in one group have identical fitness values (nondomination rank), but this value gets greater in lower layers. The selection is proportional to the fitness value assigned to a layer. The first layer has the greatest fitness and individuals in this layer are more likely to reproduce. For several years this algorithm is considered as successful, but had three major problems:

- High computational complexity of the nondominating sorting: The complexity of the sorting is $O(MN^3)$ where N is the population size and M is the number of objectives. It is very unefficient to apply this algorithm to big populations.
- Lack of elitism: Good solutions can be lost during the reproduction.
- Specifying fitness sharing parameter σ_{share}

In NSGA-II[21], the sorting algorithm is replaced by a fast nondominated sorting algorithm with $O(MN^2)$ complexity. The population is sorted and ranked based on nondomination. Apart from NSGA, a crowding distance is calculated for each individual. Crowding distance measures the density of solutions surrounding a particular solution and its usage maintains the diversity in the population. To calculate the crowding distance of a solution, its nearest neighbours along all objectives are identified. These solutions form a cuboid in the objective space. The crowding distance is the average side length of the cuboid. The selection is based both on nondomination rank and crowding distance (crowded comparison operator). If two solutions have different nondomination ranks, the one with the lower rank is preferred. Otherwise the solution in a less crowded region is selected. The selected individuals undergo to the evolutionary process and offspring population is created. To ensure elitism, the current population and the offspring population are combined and sorted based on nondomination. The nondominated fronts are moved to population of the next generation until the population is full. If the last front does not fill completly to the population, its sorted according to the crowding comparison operator and the best ones fill the next population.

The crowding distance measure replaces the shared fitness assignment in NSGA. It preserves the diversity in the population and does not require to tune a parameter. The elitism based on nondomination rank and crowding distance ensures that good solutions are saved in the next generation. The flow of the algorithm is given below:

Algorithm 4 NSGA-II

```
1: Randomly initialize population $P_0$
2: Calculate fitness values of individuals in $P_0$
3: Nondominated sorting on $P_0$
4: Binary tournament selection from $P_0$ based on nondomination rank
5: Generate child population $Q_0$
6: Recombination and mutation
7: while while max no of generations not reached do
8:     Generate $R_t = P_t \cup Q_t$
9:     Nondominated sorting on $R_t$
10:    Copy individuals from nondominated fronts to $P_{t+1}$
11:    Binary tournament selection from $P_{t+1}$ based on crowding comparison
12:    Generate child population $Q_{t+1}$
13:    Recombination and mutation
14: end while
15: Return
```

PESA-II

The PESA is firstly introduced by Corne, Knowles and Orates in [22]. In PESA there are two populations: a smaller internal population (IP) and a larger external population (EP). The EP contains good solutions which form an approximation to the Pareto front. The IP consists of candidate solutions for the EP. At the initial state the EP is empty. In each generation the nondominated individuals of the IP are copied to EP. A nondominated individual in IP can enter EP if it is also not dominated in EP. After it enters, the individuals dominated by it are removed from EP. As an application of the crowding measure, the objective space is divided implicitly into hyper-boxes (niches) in EP. Every solution in the EP belongs to a certain hyper-box and has a squeeze factor indicating the number of solutions in the particular hyper-box. The selection is based on this squeeze factor: Two individuals are selected randomly from EP and the one with the smaller squeeze factor is the winner of the tournament. The squeeze factor is also used in the update of the EP. If the gets full during the copy of nondominated IP members, the individual in EP with highest squeezing factor is removed. The selected individuals then undergo to the recombination and mutation and form the IP of the next generation.

To maintain a better diversity PESA is upgraded to PESA-II [23]. In PESA-II the selection mechanism is different than in PESA: Instead of selecting random individuals for tournament, first a populated hyper-box and than an individual from this hyper-box is selected randomly. In the EP update stage, a nondominated individual in IP can enter to a full EP if it belongs to a less crowded niche than some other solution in EP. Afterwards, the solution in the more crowded niche is replaced by this solution. The outline of the algorithm is given below:

Algorithm 5 PESA-II

```
 1: Set IP and EP to the empty set
 2: Initialize and evaluate IP
 3: Update EP according to the crowding strategy
 4: while max no of generations not reached do
 5:     Select individuals from EP
 6:     Recombination and mutation
 7:     Evaluate children
 8:     Empty IP and fill IP with children
 9:     Update EP according to the crowding strategy
10: end while
11: Return EP
```

4. MULTIOBJECTIVE EVOLUTIONARY CLUSTERING

Traditional clustering algorithms mainly focus on optimizing one objective for the clustering problem. However using a single objective may work only on certain data sets successfully. For example the simple k-means (or k-medoid) algorithm is good at finding clusters with spherical structures and spatially well-separated, but fails if the clusters are in form of spirals or the cluster centers/medoids are close to each other. To identify clusters of different structures optimization of multiple objective is a necessity. As suggested in [24, 25] clustering criterion can be grouped under three category based on the kind of objective to be optimized:

1. Cluster compactness as a measure to keep the intra-cluster variance small
2. Cluster connectedness as a measure to put neighboring data items into the same cluster
3. Spatial separation as a balancing factor

MOEAs can be adapted to the multiobjective clustering problem. First step in such an adaptation would be to specify the multiple objectives. To reveal different cluster structures the objectives must be conflicting. As the next step a suitable MOEA is selected. After the representation scheme and initialization method of the solutions is decided, evolutionary operators which can work properly on the representation scheme are specified.

Although there are many works on single objective EAs, the number of studies on multiobjective evolutionary clustering is very limited. One of the most successful works on clustering through MOEA is MultiObjective Clustering with automatic K-determination (MOCK)[24, 25] which needs data items represented as vectors. There exists another version of MOCK, called as MOCK around medoids (MOCK-am)[26], capable of working with data given as pairwise similarities. This study is analyzed in the following sections in detail. In [27] a graph based multiobjective evolutionary approach is used to cluster sequence data. The two objectives are combined using an aggregated sum approach. This dissertation improves this work by replacing the aggregated sum method with a successful MOEA. Therefore the components of this algorithm are also subject to a deeper explanation in the next section.

An multiobjective approach based on NPGA (Niched Pareto Genetic Algorithm) [28] is proposed in [29] and it uses a graph based representation scheme called as restricted linkage encoding (LL encoding). In this representation, each individual g contains N genes where N is the total number data items. The gene j can take values between j and N and the value i of gene j means that there is link between data items i and j and consequently they are in the same cluster. Two genes cannot have identical values except the ending node whose value is its own index. The objective functions are total within cluster variation and cluster number. Both are to be minimized. The individuals are randomly initialized and than checked in terms of LL encoding constraints. The evolutionary operators consist of one point crossover and a grafting mutation which splits the cluster. In [30], the author improves his work by clustering the data in two stages. In the initial stage, the data is divided randomly into disjoint subsets and each subset is clustered separately. The initial population of the MOEA in the second stage is initialized using these solutions.

MOKGA [31] is another clustering algorithm similar to the work in [29] where the same two objectives and the MOEA are used. The representation is direct encoding where the gene j can take values between 1 and k and the value i of gene j means that jth item is in cluster i. The population is initialized randomly. One point crossover and standard mutation are the evolutionary operators. In the mutation operator an item is more likely to be assigned to another cluster whose center is closer to the item. The algorithm has an additional operator called k-means operator which assigns all items to the closest clusters for faster convergence.

MOCLE (Multi-Objective Clustering Ensemble) [32], is another algorithm where clustering is performed in two stages. At the initialization stage, the initial solutions are generated by different algorithms, thus they correspond to clusterings with different characteristics and cluster counts. The objective functions are overall deviation and connectivity as in [24] and both are to be minimized. The crossover operator select two parents through binary tournament and then construct a bipartite graph with them. The resulting graph is partitioned and the solution becomes the child. The MOEA of the algorithm is SPEA[18] which is an earlier version of SPEA2 [19].

4.1 A Graph based Sequence Clustering Algorithm

In this section a graph based sequence clustering algorithm (GraSC) through MOEA is described. The algorithm is an extension of the work by Etaner-Uyar and Gündüz-Öğüdücü [27]. As mentioned in the first section in graph based clustering of sequences the data is given as pairwise similarities. The sequences correspond to nodes and pairwise similarities become the edge weights between the nodes in the graph. The original algorithm combines the two objectives *Min-max cut* and *the global silhouette index* using an aggregated sum approach where the fitness f of an individual is expressed as:

$$f = w_1 * MMC + w_2 * GS$$

where w_1 and w_2 are weights assigned to the objectives *Min-max cut* and *the global silhouette index* respectively. This approach requires to determine objective weights for each data set. To solve the graph clustering problem a standard steady-state EA with duplicate elimination is used. The flow of the algorithm is given in algorithm 6

Algorithm 6 Steady-state EA for graph clustering

```
 1: Randomly initialize population $P_0$
 2: Calculate fitness values of individuals in $P_0$
 3: while while max no of generations not reached do
 4:    Binary tournament selection from $P_t$
 5:    Create child through heuristic crossover
 6:    Mutate child
 7:    if child is not duplicate then
 8:       Apply heuristic disband
 9:       Replace child with the worst individual in $P_t$
10:    end if
11: end while
12: Return
```

The weighted sum approach on a steady state EA is replaced originally with the SPEA2 as MOEA while keeping the representation, initialization, evolutionary operators and objective functions. The components of the new clustering algorithm are given in the following subsections.

4.1.1 Objective Functions

Min-max cut and *the global silhouette index* are the objectives to be optimized. The first objective, the *min-max cut* [8] function, given in Eq. **4.1** aims to maximize the similarity within each subgraph while trying to minimize the similarity between the subgraphs.

$$MinMaxCut(G) = \sum_{m=1}^{k} \frac{cut(G_m, G \setminus G_m)}{\sum_{v_i v_j \in G_m} E(v_i, v_j)} \tag{4.1}$$

In this equation $cut(G_m, G \setminus G_m)$ is the sum of edge weights between the vertices in G_m and in the rest of the graph $G \setminus G_m$. $E(v_i, v_j)$ gives the weight of the edge between the nodes v_i and v_j. The edge weights can be thought of as pairwise similarities between data items. This objective is to be minimized.

The second objective is *the global silhouette index* (GS) [34]. In the Silhouette validation technique, for each node a silhouette width as in Eq. **4.2**, average silhouette width for each cluster as in Eq. **4.3** and the global silhouette value for the clustering as in Eq. **4.4** are calculated. GS is a cluster validation index and can be used to compare the qualities of clustering solutions with different number of clusters. It indicates how well each object has been classified to its assigned cluster as compared to the other possible clusters in the data set. The measurement is in terms of the average Euclidean distance of an object to its own cluster, compared to the average Euclidean distance to the objects in each of the other clusters. It ranges from 1 to 1 and is to be maximized. If the result is close to 1, this means that the objects are well clustered.

$$s(v_i) = \frac{b_i - a_i}{max(b_i - a_i)} \tag{4.2}$$

where a_i is the average dissimilarity between $v_i \in C_j$ and other vertices in C_j, b_i is the minimum average dissimilarity between v_i and other clusters. The dissimilarity values are computed as $(1 - E(v_i, v_j))$. A silhouette index S_j is assigned to each cluster C_j as in Eq. **4.3**.

$$S_j = \frac{\sum_{i=1}^{n_j} s(v_i)}{n_j} \tag{4.3}$$

where n_j is the number of vertices in cluster C_j. The final formula of GS is as in Eq. **4.4** for a k-clustering of the data set.

$$GS = \frac{\sum_{j=1}^{k} S_j}{k} \tag{4.4}$$

To apply the MOEA, both of the objectives are taken as maximization. So the $MinMaxCut$ value is converted to:

$$MMC = \frac{1}{1 + MinMaxCut} \tag{4.5}$$

MMC will be maximum if all vertices are in the same cluster. GS will be maximum if each vertex is a separate cluster. Thus it conflicts with MMC.

4.1.2 Representation and Initialization

The representation scheme of GraSC is the group number encoding method which is a form of *Direct Encoding* (DE). Each individual g contains N genes where N is the total number of vertices (nodes) in the graph. Each gene corresponds to a vertex and the value of the gene denotes the cluster number the vertex is placed in. Assuming $N = 7$, a sample individual encoding a 3-clustering is the following: $[1233211]$. According to this encoding the nodes $(1, 6, 7)$ are in cluster 1, the nodes $(2, 5)$ are in cluster 2 and $(3, 4)$ are in cluster 3. In DE, the cluster numbers are not important. The thing that really matters is which nodes are in the same cluster. For instance the individual $[1322311]$ is the same as the individual $[2133122]$. Although the genotypes are different both phenotypes are the same and encode the same 3-clustering. For a k-clustering of N nodes the total number of genotypes is k^N. As a result of this representation the size of the search space increases. To overcome this problem, a post-processing step is added after the initialization. After an individual is initialized, it is scanned from left to right and the nodes in the first cluster are numbered as 1, nodes in second cluster 2, and so on.

At the initialization phase random initialization (RI) method is used by assigning a value between 0 and a constant $maxCluster$ to each gene.

4.1.3 Evolutionary Operators

Crossover Operator

Using a standard crossover operator with DE will not make any sense because cluster numbers have different meanings in different individuals. If parents are combined with a standard crossover technique, the child will not contain partial information of its parents. For example two individuals are given to perform uniform crossover are $[1221313]$ and $[1123244]$. A possible child might be $[1121343]$ whose clustering is completely different than its parents. Therefore a new heuristic crossover operator, to be used with DE is introduced in [27]. According to this operator, first all vertices are marked as uncovered. At each step one uncovered vertex and one of the parents are selected randomly. The cluster containing the vertex in the chosen individual is identified and the uncovered vertices in that cluster form a cluster of the child. The uncovered vertices are marked as covered and the process is repeated until all vertices are covered. At the end the child will contain clusters both from its parents. The example of the heuristic crossover operator is given in table **4.1**.

Mutation Operator

A standard mutation operator is used for mutation. According to a mutation rate, the cluster number of each node is replaced by a new number in a given interval $[1, maxCluster]$, where $maxCluster$ is the maximum number clusters in solution. From graph clustering

Table 4.1: Steps of heuristic crossover operator

step	vertex	parent	cluster	covered vertices
1	2	1	(2,3)	2,3
2	4	2	(4)	2,3,4
3	1	2	(1)	1,2,3,4
4	7	1	(5,7)	1,2,3,4,5,7
5	6	1	6	1,2,3,4,5,6,7

perspective, this corresponds to deleting a node from a cluster and putting in some other cluster randomly. After the mutation operator is applied, some clusters may become empty. Therefore the postprocessing step at the initialization step is repeated for each individual as an extension of the mutation.

Heuristic Disband Operator

After the heuristic crossover operator is applied, the child usually has more clusters than both of its parents. If the number of clusters are not reduced, after a certain generation the individuals would have too many clusters, corresponding to unwanted solutions of the problem. To accomplish this a heuristic disband operator is introduced. First the cluster with the minimum intra cluster similarity is identified. Each node in this cluster is moved to its closest cluster in the partitioning. The closest cluster of a node is the cluster with maximum average similarity to that node. Again here, a cluster becomes empty so the postprocessing step is required to correct the cluster numbers.

Since this is the last operator, every individual is scanned whether it contains more clusters than the predefined parameter $maxCluster$ and any cluster contains less items than $minNode$. The global silhouette index is undefined if a cluster contains only one item. Therefore the $minNode$ constant is selected as $minNode > 1$. If the number of items in a cluster is less than $minNode$, for each item in the cluster its closest cluster is found and the item is moved to that cluster. If the cluster count k of the solution is greater than $maxCluster$, $(k - maxCluster)$ clusters with the least intra cluster similarity are identified. For each item in these clusters, their closest clusters are found and the itemse are moved to these clusters.

4.2 Multiobjective Clustering Around Medoids

MultiObjective Clustering with automatic K-determination (MOCK) [24, 25] is a MOEA based on PESA-II for data clustering in metric space. It tries to optimize two conflicting objective functions: *overall deviation* and *connectivity*. A locus-based adjacency representations is implemented where an individual g contains N genes where N is the total number data items. The gene j can take values between j and N and the value i of gene j means that there is link between data items i and j and consequently they are in the same cluster. MOCK uses a special initialization approach where the some of the initial solutions are better according to the overall deviation and the rest are better based on the connectivity measure. The evolutionary operators consist of standard crossover and restricted nearest neighbor. The algorithm involves a final step for the automatic solution selection from the Pareto front and for the determination of the number of clusters in the data set.

The original algorithm is extended to work with data given as pairwise similarities and the modified algorithm is called MOCK-am. In this chapter the details of the MOCK-am are given whose results will be compared to the results of the proposed algorithm in the previous chapter. Representation scheme, initialization method, evolutionary operators are almost same in MOCK and MOCK-am except that the cluster centroid notion is replaced in MOCK-am with cluster medoid which is suitable to work with similarity based data. The details of MOCK-am can be seen below.

4.2.1 Objective Functions

The first objective function *overall deviation* OD, given in Eq. **4.6**, sums the distances of items to their cluster medoid. A cluster medoid μ is the data item closest to the cluster center. The distance function δ is the dissimilarity between the item i and the medoid $\delta(i, \mu_k)$ of its cluster.

$$OD(C) = \sum_{C_k \in C} \sum_{i \in C_k} \delta(i, \mu_k) \qquad \textbf{(4.6)}$$

where C is the set of all clusters and μ_k is the medoid of the cluster C_k. This objective measures whether the clustering consists of compact clusters, where the data items are really similar to the their cluster medoids. OD is to be minimized.

The second objective *connectivity* (CO), given in Eq. **4.7**, examines whether neighboring items are in the same cluster. If an item is not in the same cluster with some of its L nearest neighbors, penalty points accumulate. CO is also to be minimized.

$$CO(C) = \sum_{i=1}^{N} \sum_{j=1}^{L} x_{i,nn_i(j)} \qquad \textbf{(4.7)}$$

where

$$x_{r,s} = \begin{cases} \frac{1}{j} & \text{if } \nexists C_k : r, s \in C_k \\ 0 & \text{otherwise} \end{cases}$$

In this formula, N is the item count, L is a parameter for the nearest neighbor count, $nn_i(j)$ is the j-th nearest neighbor of item i.

4.2.2 Representation and Initialization

The individuals are represented by a graph-based representation scheme: Locus-based adjacency representation (LAR). Each individual g contains N genes where N is the total number data items. The gene j can take values between 1 and N and the value i of gene j means that there is link between data items i and j and consequently they are in the same cluster.

At the initialization step two types of solutions are generated: good solutions according to OD objective and good solutions according to CO objective. Initial solutions with better CO values are generated through minimum spanning trees (MSTs) [35]. MST of a connected, undirected graph is a subgraph with all vertices and forms a tree with minimum weight. First, the complete MST corresponding to the one cluster solution is created according to Prim's algorithm. To have initial solution with different number of clusters longest links in the MST can be removed. However in some datasets this would cause to some problems. If there are many outliers, removing the longest links in the MST will isolate these points. The rest of the clustering will be very similar in many initial solutions. To overcome this issue, a definition of "interestingness" of links is introduced. Removal of interesting links may reveal real clusters. Formally, it is defined as follows: A link between nodes i and j is defined interesting, if neither of the nodes is one its L nearest neighbors. A clustering is called "interesting" if it can be generated by removing only interesting links from the complete MST. To generate initial solutions based the complete MST following steps are needed:

1. The complete MST is generated.
2. I interesting links on the MST are detected.
3. The interesting links are sorted according to their degree of interestingness. The degree d of link between i and j is $min(l, k)$ where i is lth nearest neighbor of j and j is kth nearest neighbor of j

4. The solution $n \in 0, \ldots, min(I, 0.5 * N) - 1$ is initialized by removing n most interesting links on the MST. Thus, it will consist of $n + 1$ clusters.

The number of MST-based solutions in the initial population is $min(I, 0.5 * N)$, the rest of the population will be filled up with good solutions according to OD objective. This is enabled through usage of simple k-means clustering. The k-medoid algorithm is run multiple times for $k = 2, \ldots, N - min(I, 0.5 * N)$ until the other solutions are initialized. Each different k-clustering with k-medoid algorithm corresponds to an initial solution.

4.2.3 Evolutionary Operators

Crossover Operator

Unlike direct encoding, the locus-based adjacency representation enables the usage of uniform crossover operator. The genes at position i of two individuals are exchanged according to predefined crossover rate P_c. Since the values of genes indicate links with other nodes, the child will preserve the link information of both from its parents.

Mutation Operator

Restricted nearest-neighbor mutation is used for mutation operator. According to this mutation, a node can only be linked to one its L nearest neighbors. If the gene will undergo to mutation with mutation rate P_m, a number m f is selected randomly from the interval $[1, L]$. The values of the gene is changed to r which is its mth nearest neighbor.The reason behind that choice is to reduce the size of the search space from N_N to L_N caused by the locus-based adjacency representation. This requires the calculation of L nearest neighbors of all items. However it can be done in the initialization step only once.

5. ANALYSIS OF MOEAs FOR CLUSTERING

MOCK-am and GraSC use different initialization methods, MOEAs, evolutionary operators and objective functions. To see the individual effects of the selected MOEAs, the objective functions and other algorithmic details like initialization and genetic operators, different variations of MOCK-am and GraSC are implemented. Table **5.1** shows the implemented variations. By comparing the different variations with each other it will be possible to identify the best combinations of initialization method, objective set, operator group and MOEA. The purpose of the analysis step is the determination of the best MOEA variation for the clustering problem of sequence data given as pairwise similarities.

For GraSC, default algorithmic details (*gr-def*) denotes direct encoding DE, random initialization RI, the default operator group (*gr-op*) consisting of the heuristic crossover operator and standard mutation. Default algorithmic details for MOCK-am (*mock-def*) are locus-based adjacency representation LAR, initialization based on MST and k-medoid algorithm, the default MOCK operators (*mock-op*) consisting of uniform crossover and restricted nearest neighbour mutation. The MOEAs of the clustering algorithms are replaced by PESA-II, SPEA2 and NSGA-II. The complementary objectives of the algorithms are not separated from each other. Min-max cut and global silhouette index are always maximized together like overall deviation and connectivity are minimized. Namely two objective sets exists: Objective set 1 (OS1) with MMC and GS of GraSC and objective set 2 (OS2) of MOCK-am.

Table 5.1: Implemented algorithm variations

Variation	MOEA	Obj.	Algorithmic Details
GNG	NSGA-II	MMC,GS	gr-def
GPG	PESA-II	MMC,GS	gr-def
GSG	SPEA2	MMC,GS	gr-def
GND	NSGA-II	MMC,GS	DE,MST,gr-op
GPD	PESA-II	MMC,GS	DE,MST,gr-op
GSD	SPEA2	MMC,GS	DE,MST,gr-op
GNM	NSGA-II	MMC,GS	mock-def
GPM	PESA-II	MMC,GS	mock-def
GSM	SPEA2	MMC,GS	mock-def
MNM	NSGA-II	OD,CO	mock-def
MPM	PESA-II	OD,CO	mock-def
MSM	SPEA2	OD,CO	mock-def

5.1 Overview of MOEA Analysis

As mentioned in the first chapter, a multiobjective optimizer returns not only a single solution, but a set of solutions (an approximation to the true Pareto optimal solutions) corresponding to different trade-offs in the objective space. Therefore, the performance assessment procedure of a MOEA is different than a single objective EA. If A and B are two approximation sets generated by two different MOEAs, they can be compared according to Pareto dominance criteria. There are four possible outcomes of this comparison [36]:

1. A is better than B (Every solution in A dominates every solution in B)
2. B is better than A (Every solution in B dominates every solution in A)
3. A and B are incomparable
4. A and B are indifferent

However the Pareto dominance comparison is not a quantative performance measure. If A is better than B, it does not tell how much better is A. Quality indicators, assigning a real value for a given approximation set, are introduced for this purpose. Many quality indicators (unary and binary) exist in nature. In this work the implemented unary indicators are the following:

- Hypervolume Indicator I_H [37]: It calculates the hypervolume of the objective space which is dominated by an approximation set. The objective space must be bounded by a reference point (dominated by all points) lying beyond the approximation sets. I_H is to be maximized.
- Unary Epsilon Indicator I_ϵ [38]: Assume A, B and C are approximation sets and A dominates B. Epsilon indicator is the smallest distance changing B in such a way that it dominates A. If A is selected as a reference set dominating B and C, B and C can be compared according to their I_ϵ based on the reference set A. This value is to be minimized. The I_ϵ of an approximation set B according to a reference set A is defined by the following formula:
  $I_\epsilon(A, B)$ = inf $\forall z^2 \in B \exists z^1 \in A : z^1 \preceq_\epsilon z^2$
  $I_\epsilon(A, B)$ is the minimum factor ϵ such that for any solution in B there is at least one solution in A which is not worse by a factor ϵ in all objectives.
- R2 Indicator I_{R2} from R Indicator Family [39]: It is used to compare approximation sets based on utility functions. A utility function u is a mapping from objective vectors set to real numbers. If the user's preferences are given through a

parameterized utility function u_λ with parameters Λ, these utility functions can be transformed into a quality indicator by the following equation:
$I_{R2} = \frac{\sum u*(\lambda,A)-u*(\lambda,B)}{|\Lambda|}$
where $u*$ is the maximum reachable value by the utility function on an approximation set A and B is a reference set. $I_{R2}(A)$ will be smaller or equal to $I_{R2}(B)$ if A is not a better approximation set than B.

The MOEAs are stochastic procedures; in one run the approximation set A might be better than B, but in another run just the opposite. Therefore, the algorithms must be run several times to prove the outperformance of an algorithm according to the other one. Two approaches exist for analyzing multiple runs of an multiobjective optimizer statistically:

1. Transforming approximation sets into real values by using a quality indicator and applying standard nonparametric statistical testing methods.
2. Transforming approximation sets into emprical attainment functions.

In this work, the former approach which is also the most popular one is preferred. After transforming approximation sets into real numbers using unary quality indicators, a standard nonparametric statistical testing method is used to examine the statistical significance. As suggested in [36] Kruskal-Wallis test is chosen to compare multiple variations. This test compares the quality indicator results of variation pairs and shows whether one variation is significantly better than the other based on a significance level α. According to these relations between variation pairs, variations are ranked. The key point in ranking is only variations with the same objective set (MMC-GS as OS1 or OD-CO as OS2) can be compared to each other. The indicator values of approximation sets belonging to different objective spaces are not comparable.

Based on the results of the statistical tests, best variations for each of the objective sets are identified: one variation using the OS1 and one variation using the OS2. The next step for analysis of MOEAs for clustering is to determine which algorithm generates the best clustering among these two variations. For this purpose, single solutions are identified from the combined Pareto fronts of multiple runs for each variation and a clustering validation index measures the quality of both clusterings. The index verifying the quality of clustering solutions is the Davies-Bouldin (DB) index [3]. It measures the ratio of similarities among items in same cluster and dissimilarities among items in different clusters. The original DB index has been modified in [40] to be used on graphs. The DB index is calculated using

Eq. **5.1**, Eq. **5.2**, Eq. **5.3**, Eq. **5.4**.

$$\Delta(C_i) = \frac{1}{|C_i| * (|C_i| - 1)} \sum_{v_i, v_j \epsilon C_i, v_i \neq v_j} d(v_i, v_j) \tag{5.1}$$

where $\Delta(C_i)$ is the average diameter of cluster C_i, $|C_i|$ denotes the number of vertices in cluster C_i and $d(v_i, v_j)$ is the dissimilarity between the two vertices.

$$\delta(C_i, C_j) = \frac{1}{|C_i| * |C_j|} \sum_{v_i \epsilon C_i, v_j \epsilon C_j} d(v_i, v_j) \tag{5.2}$$

where $\delta(C_i, C_j)$ is the average linkage between the two clusters.

$$DB_j(C_j) = max_{i \neq j} \{ \frac{\Delta(C_i) + \Delta(C_j)}{\delta(C_i, C_j)} \} \tag{5.3}$$

where DB_j is the average similarity between cluster C_j and its most similar one.

$$DB(C) = \frac{1}{k} \sum_{j=1}^{k} DB_j(C_j) \tag{5.4}$$

where $DB(C)$ gives the DB index value of the clustering solution C. A lower value of DB index indicates a good clustering solution.

For comparison purposes, the graph clustering algorithm of the Cluto package[41] is also used. By default the graph clustering algorithm performs $k - 1$ repeated bisections for k-clustering of the graph. In each iteration, a partition is selected for bisection and the partition is splitted into two subgraphs such a way that the clustering optimizes an objective function. The algorithm returns a single solutions. Unlike MOCK-am and GraSC this algorithm requires the cluster count given as input parameter. Therefore Cluto is run for cluster counts of the optimum clusterings found by MOCK-am and GraSC and DB indices of the resulting clusterings are also calculated.

5.2 Optimum Solution Selection

The run of a MOEA is not fully completed when the Pareto front is found. Many problems require the selection of a single solution from the Pareto front according a decision maker's preferences. The optimal solution selection problem is studied in literature [42]. Two works are examined more detailed: [43] and [24]. Both methods depend on “knees” on the Pareto front. If the Pareto front is plotted, some bumps (called knee) may be seen on front. These regions attract more attention than other regions because they mean that a small improvement in one objective results in worsening of other objectives and makes the other Pareto-optimal solutions in the neighborhood less interesting. In [43], the optimum solutions are determined using the angles between the lines which go through a solution point and its four neighbors. The solution with the largest angle measure is selected as the best. In [24], random control distributions are used. For each solution, attainment scores are calculated after the generation of attainment surfaces for the solution and control fronts. The solution with the highest attainment score is selected as the best solution.

In this work, the method described in [43] is implemented for its computational efficiency. The knees on the Pareto front are identified according to an angle-based approach. For each solution x_i four different angles are calculated:

1. α-angle between the lines going through (x_{i-1}, x_i) and (x_i, x_{i+1})
2. β-angle between the lines going through (x_{i-2}, x_i) and (x_i, x_{i+1})
3. γ-angle between the lines going through (x_{i-1}, x_i) and (x_i, x_{i+2})
4. δ-angle between the lines going through (x_{i-2}, x_i) and (x_i, x_{i+2})

The largest angle among these four are assigned to each of the Pareto-optimal solutions. The solution with the largest angle is identified as the "knee" point and returned as the candidate solution of the clustering.

6. EXPERIMENTAL RESULTS

6.1 Data Preparation

To identify the best MOEA variation for clustering, three real-world datasets named as BIDB, CE and SAS are used. These datasets contain user sessions of different web sites. A user session consists of a set of ordered distinct web pages on the web site that the user requests in her/his single visit. Therefore user sessions correspond to sequences in sequence clustering problem. The sources of user sessions in datasets are the following:

- BIDB: Web site of IT Department of Istanbul Technical University (ITU), http://www.bidb.itu.edu.tr/
- CE: Web site Computer Engineering Department of ITU, http://www.ce.itu.edu.tr/
- SAS: Web site of University of Saskatchewan [44]

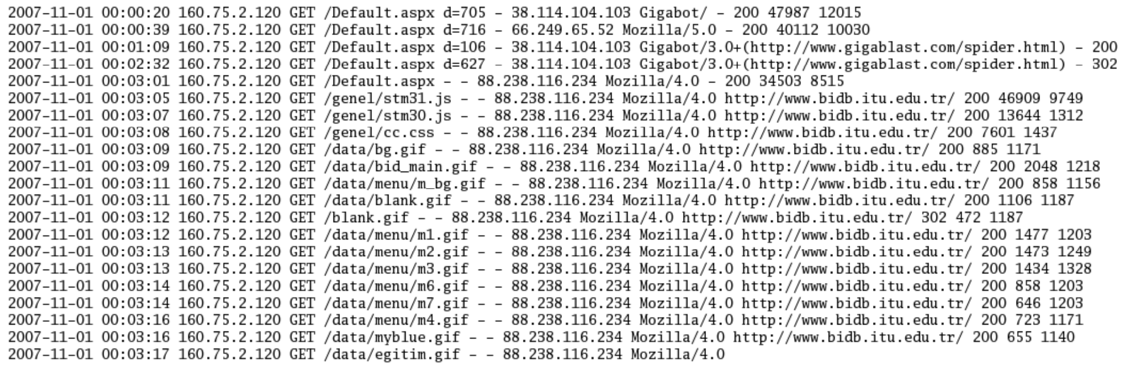
The pairwise similarity matrix of user sessions are obtained from web server logs after a data preparation process which consists of two steps:

1. Data cleaning
2. Pairwise similarity matrix generation

6.1.1 Data Cleaning

For BIDB and CE datasets, the raw web server logs are cleaned and user sessions are extracted using a web log processor software called Polliwog [45]. Polliwog gets as input the unprocessed web server log file. Sample lines of a log file with extended log format can be seen in figure **6.1**:

When a user requests a web page from the web server, she/he does not only get a text file but also many other files in various formats like jpg, gif, png. The log file contains also entries about these file requests. However these files should not appear in the sessions. Polliwog constructs the user session where each user session has a duration (maximum 30 minutes) and contains only pages of allowed text formats such as html, asp, php.



```
2007-11-01 00:00:20 160.75.2.120 GET /Default.aspx d=705 - 38.114.104.103 Gigabot/ - 200 47987 12015
2007-11-01 00:00:39 160.75.2.120 GET /Default.aspx d=716 - 66.249.65.52 Mozilla/5.0 - 200 40112 10030
2007-11-01 00:01:09 160.75.2.120 GET /Default.aspx d=106 - 38.114.104.103 Gigabot/3.0+(http://www.gigablast.com/spider.html) - 200
2007-11-01 00:02:32 160.75.2.120 GET /Default.aspx d=627 - 38.114.104.103 Gigabot/3.0+(http://www.gigablast.com/spider.html) - 302
2007-11-01 00:03:01 160.75.2.120 GET /Default.aspx - - 88.238.116.234 Mozilla/4.0 - 200 34503 8515
2007-11-01 00:03:05 160.75.2.120 GET /genel/stm31.js - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 46909 9749
2007-11-01 00:03:07 160.75.2.120 GET /genel/stm30.js - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 13644 1312
2007-11-01 00:03:08 160.75.2.120 GET /genel/cc.css - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 7601 1437
2007-11-01 00:03:09 160.75.2.120 GET /data/bg.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 885 1171
2007-11-01 00:03:09 160.75.2.120 GET /data/bid_main.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 2048 1218
2007-11-01 00:03:11 160.75.2.120 GET /data/menu/m_bg.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 858 1156
2007-11-01 00:03:11 160.75.2.120 GET /data/blank.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 1106 1187
2007-11-01 00:03:12 160.75.2.120 GET /blank.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 302 472 1187
2007-11-01 00:03:12 160.75.2.120 GET /data/menu/m1.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 1477 1203
2007-11-01 00:03:13 160.75.2.120 GET /data/menu/m2.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 1473 1249
2007-11-01 00:03:13 160.75.2.120 GET /data/menu/m3.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 1434 1328
2007-11-01 00:03:14 160.75.2.120 GET /data/menu/m6.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 858 1203
2007-11-01 00:03:14 160.75.2.120 GET /data/menu/m7.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 646 1203
2007-11-01 00:03:16 160.75.2.120 GET /data/menu/m4.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 723 1171
2007-11-01 00:03:16 160.75.2.120 GET /data/myblue.gif - - 88.238.116.234 Mozilla/4.0 http://www.bidb.itu.edu.tr/ 200 655 1140
2007-11-01 00:03:17 160.75.2.120 GET /data/egitim.gif - - 88.238.116.234 Mozilla/4.0
```

Figure 6.1: Sample Log Lines

The sessions in SAS dataset were already processed, so no further data cleaning was needed.

6.1.2 Similarity Calculation

For all of the three datasets the similarity matrices of sequences (user sessions) are calculated with an algorithm based on FastLSA [46] which is a sequence alignment technique. In the first step of the algorithm a matrix called the score matrix is created and initialized. The matrix contains $K + 1$ columns and $N + 1$ rows where K is the length of sequence #1 and N is the length of sequence #2. Sequence #1 is placed to the top of the matrix and sequence #2 to left. A gap is added to the end of each sequence. The last row of the matrix is filled from left to right is such a way that each cell is the sum of the previous cell and the gap penalty. The last column is filled from bottom to top according to the same principle. The value of $M_{N+1,K+1}$ entry of the score matrix is 0. In the second stage of the algorithm the whole score matrix is filled. The $M_{i,j}$ entry is calculated based on the following formula:

$M_{i,j} = max[M_{i+1,j+1} + Score_{i,j}($ match/mismatch in the diagonal $), M_{i,j+1} + s_g($ gap in sequence #2$), M_{i+1,j} + s_g($gap in sequence #1$)]$

In the final step, the actual alignments which leads to the maximum score is determined. The similarity metric has to parts: Alignment score component and local similarity component. The alignment score s_a measures how similar are two sequences in the region of their overlap. The local similarity component s_l computes the importance of the overlap region. The total similarity $sim(S_i, S_j)$ between two sequences is given by the final formula:

$$sim(S_i, S_j) = s_a(S_i, S_j) * s_l(S_i, S_j) \qquad \textbf{(6.1)}$$

The properties of the graphs generated based on the datasets are given in table **6.1**.

Table 6.1: Data set properties

Name	#nodes	#edges	#pages
BIDB	1238	210214	323
CE	3484	54818	500
SAS	7452	636312	171

The BIDB and CE datasets are used to determine the best two MOEA variations; one variation with OS1 objective set, one variation with OS2 objective set. The experiments with the last dataset selects the best MOEA variation for the clustering problem.

6.1.3 An Illustrative Example

Assume that the Web logs are given like in figure **6.1** and after the data cleaning step 4 user sessions are extracted. The extracted user sessions (S1, S2, S3, S4) are given in table **6.2**. The p_n is the nth page of the user session. In this example the total number of Web pages in the data set (#pages) is 5 and *A, B, C, D, E* are the Web pages. The similarity matrix of these user sessions calculated according to the given metric is in table **6.3** and the resulting graph is illustrated in figure **6.2**

Table 6.2: Sample user sessions

	p_1	p_2	p_3	p_4
S1	A	B	C	
S2	B	D	C	
S3	E	A	B	C
S4	A	D	C	E

Table 6.3: Sample similarity matrix

	S1	S2	S3	S4
S1	1	0.375	0.75	0.625
S2	0.375	1	0..3	0
S3	0.75	0.3	1	0.5
S4	0.625	0	0.5	1

6.2 Parameter Settings

The general parameter settings for GraSC and MOCK-am used in the experiments are given in Table **6.4**. The L value of the restricted nearest neighbour mutation and medoid computation of MOCK-am is set to 10. For PESA-II the EP size is 1000 and the IP size is 10. The resolution of the hypergrid per dimension is 10. For SPEA2 the population size is 100 and the archive size is 40. The NSGA-II population size is also 100. The

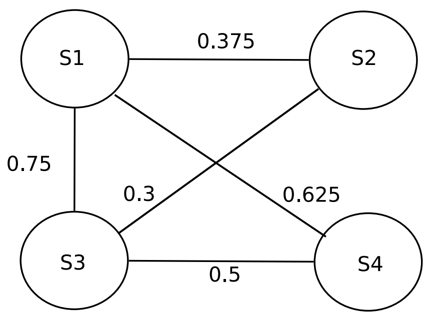


Figure 6.2: Sample Graph

Table 6.4: General settings

Parameter	GraSC	MOCK-am
Num.of Gen.	500	500
Recom. Rate	1	0.7
Mutation Rate	1/N	1/N
MinNode	2	2

settings related to MOCK-am and PESA-II parameters are selected as suggested in [26]. The parameters of GraSC are tuned empirically. The parameters of SPEA2 and NSGA-II are set as their original proposed versions. During determining the population sizes, the run-time of the algorithm is considered. It is obvious that having a big population helps in maintaining a good level of diversity. However, increasing the population size too much would also increase the number of evaluations and consequently the run-time. Thus, population size is determined experimentally, balancing diversity and run-time.

6.3 Experiments

Each variation is run 30 times. For each MOEA all resulting approximation sets are combined and statistical tests are performed. For significance tests, a commonly used significance level, $\alpha = 0.01$, is chosen. In hypothesis testing, the significance level is the criterion for rejecting the null hypothesis. Tables **6.5**, **6.6** and **6.7** show the comparison results of variations with OS1 and OS2 respectively for BIDB and CE datasets. The algorithms given in the tables are sorted in descending order according to the statistical testing procedures. There is no significant difference between the variations in the same row. For the OS1 (MMC-GS objective set) the GND variation, namely the one using NSGA-II as MOEA, GrasC operators and MST based initialization is the only variation which is in the first position for all of the statistical tests. For the OS2 (OD-CO objective set) the standard MOCK-am algorithm, denoted as MPM, has shown the best performance in all of the tests. Therefore these are chosen for further examination of clustering quality.

Table 6.5: Comparison results of BIDB data set for OS1

I_ϵ	I_H	I_{R2}	Rank
GSM/GND	GSM/GND	GPM/GND	GNM/GSM/GND/ GPD/GSD
GPG/GSD	GPG/GSD	GNG/GSD	GPG
GNM	GNM	GSG	GPM
GSG/GPM	GSG	GNM	GSG
GNG	GPM	GSM/GPD	GNG
GPD	GNG	GPG	
	GPD		

Table 6.6: Comparison results of CE dataset for OS1

I_ϵ	I_H	I_{R2}	Rank
GPG/GNM/ GSM/GND	GPG/GNM/ GSM/GND	GNG/GSG/ GPM/GND	GSM/GND/ GPD/GSD
GSD	GSD	GSD	GNM
GSG/GPM	GSG/GPM	GNM/GSM	GNG/GPG/ GSG/GPM
GNG	GNG	GPG	
GPG	GPD	GPD	

Table 6.7: Comparison results of the BIDB&CE datasets for OS2

I_ϵ	I_H	I_{R2}	Rank
MPM	MPM	MPM	MPM/MNM/ MSM
MNM/MSM	MNM/MSM	MNM/MSM	

After the best variations for OS1 and OS2 are identified, the optimum solutions for each of the variations are selected from their corresponding Pareto fronts using the knee identification method mentioned in the previous chapter. For the selected solutions the DB index values are calculated. Table **6.8** shows the DB indices of the clustering solutions for all three datasets.

Table 6.8: DB index results of the best variations

CS	GND	MPM
BIDB	1.00 (3C)	1.40 (4C)
CE	1.16 (3C)	0.61 (9C)
SAS	1.42 (18C)	1.69 (30C)

Table 6.9: DB index results of clusterings using Cluto

Data Set	Number of Clusters				
	3	4	9	18	30
BIDB	1.47	1.45			
CE	1.97		1.94		
SAS				1.78	1.75

According to the DB index values, there is no significant difference between the GND variation with OS1 (MMC-GS objective set) and the MPM variation with OS2 (OD-CO objective set) almost in all cases, however both are superior to Cluto results for the same number of clusters.

7. CONCLUSION AND DISCUSSION

Existing clustering algorithms work usually only data sets given in metric space. However in many domains such as bioinformatics, chemistry, computer vision and Web mining we encounter with sequence data which needs to be represented as pairwise similarity/dissimilarities. In this study, we have introduced the graph-based MOEA GraSC for sequence clustering problem of such data. Another important property of the suggested sequence clustering algorithm is that it does not expect a cluster count parameter beforehand. Unlike other traditional algorithms the user does not need to have an overview of the data before starting the clustering process. Moreover, the algorithm returns many solutions with different characteristics and cluster counts in a single run.

To improve the performance of the original algorithm we created a "component pool" of MOEA components from the proposed approach and a successful MOEA called MOCK-am. Multiple MOEA variations are derived from this component pool. Through statistical tests and a well-known cluster validation index called Davies-Bouldin index the best MOEA variation for clustering is identified. Moreover, the data sets are also clustered using a stochastic graph clustering algorithm in Cluto package which performs $k-1$ repeated bisections. Statistical test can only evaluate variations having the same objective set. The best variation with the min-max cut and global silhouette index is the so called GND variation which consists of NSGA-II as MOEA, direct encoding, MST and k-medoid based initialization method, and default operators of GraSC. The best variation with the overall deviation and connectivity is the MPM variation which consists of PESA-II as MOEA, locus-based adjacency representation, MST and k-medoid based initialization method, and default operators of MOCK-am. The MST and k-medoid based initialization of MOCK-am has been more successful than random initialization. However for some data sets this approach may lead to outlier isolation rather than clustering if there exists small group of nodes dissimilar to many other nodes. It would be more convenient to use MST-based initialization on highly connected graphs.

For both of the algorithms and for all of the data sets single solutions are selected for cluster evaluation. The solutions are identified by detecting the knees on the plot of Pareto front. The stochastic graph clustering algorithm is run on the data sets for cluster counts of these selected solutions. According to the Davies-Bouldin index results there is no significant performance difference for both MOEAs. However they are superior to the other contestant algorithm of Cluto package. To sum up, the proposed algorithm has been able to cluster the test data test successfully without giving the cluster count as an input parameter. In a single run it produces many solutions with different cluster structures. With an simple automatic solution technique based on knee-identification it also returns a single solution which might be interesting for the user.

The future work includes testing the algorithms on different data sets not only to cluster user sessions but also to cluster Web documents and protein sequences. Moreover, the graph structures of the data sets can be examined and a relation between the graph structure and best variation to cluster the graph can be set up. It is also on the agenda to evaluate the performance of the algorithm with more complex and huge graphs where the number of the nodes is more than 10000.

BIBLIOGRAPHY

[1] **Duda, R. O. and Hart, P. E.**, 1973. Pattern Classification and Scene Analysis, Wiley-Interscience Publication, New York.

[2] **Eiben, A. E. and Smith, J. E.**, 2003. Introduction to Evolutionary Computing, Springer, New York.

[3] **Davies, D.L. and Bouldin, D.W.**, 1979. PA Cluster Separation Measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 224-227.

[4] **Jain, A. K., Murty, M. N. and Flynn, P. J.**, 1999. Data clustering: a review, *ACM Comput. Surv.*, **31**, 264-323.

[5] **Fiedler, M.**, 1973. Algebraic connectivity of graphs, *Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach.*, 298-305.

[6] **Cheng, C.K. and Wei, Y. A.**, 1991. An Improved Two-way Partitioning Algorithm with Stable Performance, *IEEE Trans. on Computed Aided Design*, 1502-1511.

[7] **Shi, J. and Malik, J.**, 2000. Normalized Cuts and Image Segmentation, *IEEE Trans. on Patterns Analysis and Machine Intelligence*, 56-76.

[8] **Ding, C.H.Q., Xiaofeng, H., Hongyuan, Z., Ming G. and Simon, H.D.**, 2001. A Min-max Cut Algorithm for Graph Partitioning and Data Clustering, *Proceedings IEEE International Conference on Data Mining*, 107-114.

[9] **Fogel, D. B.**, 1998. Evolutionary Computation: The Fossil Record, Wiley-IEEE Press, New Jersey.

[10] **Fogel, D. B.**, 1995. Evolutionary Computation. Toward a new Philosophy of Machine Intelligence, IEEE Press, New Jersey.

[11] **Holland, J. H.**, 1975. Adaptation in Natural and Artifical Systems, University of Michigan Press, Ann Arbor.

[12] **Rechenberg, I.**, 1973. Evolutionstrategie: Optimierung Technisher Systeme nach Prenzipien des Biologischen Evolution, Fromman-Hozlboog Verlag, Stuttgart.

[13] **Eschenauer, H., Koski, J. and Osyczka, A.**, 2001. Multicriteria Design Optimization, Springer-Verlag, New York.

[14] **Goldberg, D.E.**, 1989. Genetic Algorithms in Search, Optimization, Machine Learning, Addison-Wesley Publishing Company, Massachusetts.

[15] **Schaffer, J.D.**, 1985. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Hillsdale, New Jersey, 93-100.

[16] **Coello, C., Lamont, G. and Veldhuizen, D.** , 2007. Evolutionary Algorithms for Solving Multi-Objective Problems, Springer US, New York.

[17] **Wolpert, D.H. and Macready, W.G.**, 1997. No Free Lunch Theorems for Optimization, *IEEE Transactions On Evolutionary Computation*, 67-82.

[18] **Zitzler, E. and Thiele, L.**, 1999. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach., *IEEE Transactions On Evolutionary Computation*, 257-271.

[19] **Zitzler E., Laumanns E., Thiele L.**, 2001. SPEA2: Improving the Strength Pareto Evolutonary Algorithm., *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, Athens, Greece, 95-100.

[20] **Srinivas, N. and Deb, K.**, 1995. Multiobjective function optimization using nondominated sorting genetic algorithms, *Evolutionary Computing*, 221-248.

[21] **Deb, K., Agrawal, S., Pratab, A. and Meyarivan, T.**, 2000. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, Paris, France, 849-858.

[22] **Corne, D. W., Knowles, J. D. and Oates, M. J.**, 2000. The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, 839-848.

[23] **Corne, D. W., Jerram, N.R., Knowles, J. D. and Oates, M. J.**, 2001. PESA-II: Regionbased Selection in Evolutionary Multiobjective Optimization, *Proceedings of the Genetic and Evolutionary Computation Conference*, 283-290.

[24] **Handl, J. and Knowles, J.**, 2004. Multiobjective Clustering with Automatic Determination of the Number of Clusters, *University of Manchester Technical Report*, **TR-COMPSYSBIO-2004-02**, Manchester, UK.

[25] **Handl, J. and Knowles, J.**, 2007. An Evolutionary Approach to Multiobjective Clustering, *IEEE Transactions On Evolutionary Computation*, 56-76.

[26] **Handl, J. and Knowles, J.**, 2005. Multiobjective Clustering Around Medoids, *Proceedings of the Congress on Evolutionary Computation.*

[27] **Uyar, A. Ş. and Gündüz Öğüdücü, S.**, 2005. A New Graph-Based Evolutionary Approach to Sequence Clustering, *Proceedings of Fourth International Conference of Machine Learning and Applications*, 283-290.

[28] **Horn, J., Nafpliotis, N. and Goldberg D. E.**, 1994. A Niched Pareto Genetic Algorithm for Multiobjective Optimization, *In Proceedings of the Congress on Evolutionary Computation*, 82-87.

[29] **Du, J., Korkmaz, E., Alhajj, R. and Barker, K.**, 2004. Novel Clustering Approach that Employs Genetic Algorithm with New Representation Scheme and Multiple Objectives, *In Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery*, 219-233.

[30] **Korkmaz, E.**, 2006. A Two-Level Clustering Method using Linear Linkage Encoding, *In Proceedings of the Parallel Problem Solving from Nature*, 681-690.

[31] **Du, J., Korkmaz, E., Alhajj, R. and Barker, K.**, 2004. Multi-objective Genetic Algorithm based Clustering approach and its Application to Gene Expression Data, *In Proceedings of the Advances in Information Systems*, 451-461.

[32] **Faceli, K. and de Souto, M. C. P.**, 2006. Multiobjective Clustering Ensemble, *In Proceedings of the Sixth International Conference on Hybrid Intelligent Systems*, 451-461.

[33] **Ding, C.H.Q., Xiaofeng H., Hongyuan, Z., Ming, G. and Simon, H.D.**, 2001. Multiobjective Clustering Ensemble, *In Proceedings of the Sixth International Conference on Hybrid Intelligent Systems*, 451-461.

[34] **Rousseeuw, P.J**, 1987. Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis, *J. Comput. Appl. Math.*, 53-65.

[35] **Wilson, R.J and Watkins, J.J**, 1990. Graphs: An Introductory Approach: A First Course in Discrete Mathematics, John Wiley, New York.

[36] **Knowles, J., Thiele, L. and Zitzler, E.**, 2006. A Tutorial on Performance Assessment of Stochastic Multiobjective Optimizers, *Computer Engineering and Networks Laboratuary Technical Report*, **TIK-Report No 214**, Zurich, Switzerland.

[37] **Zitzler, E. and Thiele, L.**, 2001. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Evolutionary Algorithm, *IEEE Transactions On Evolutionary Computation*, 257-271.

[38] **Zitzler, E., Thiele, L., Laumanns, M. and Foneseca, C. M.**, 2003. Performance Assessment of Multiobjective Optimizers: An Analysis and Review, *IEEE Transactions On Evolutionary Computation*, 117-132.

[39] **Hansen, M.P. and Jaszkiewicz A.**, 1998. Evaluating the Quaility of Approximations to the Nondominated Set, *Institue of Mathematical Modelling Technical Report*, **IMM-REP-1998-7**, Denmark.

[40] **Speer, N., Spieth, C. and Zell, A.**, 2005. Biological Cluster Validity Indices Based on the Gene Ontolog, *Advances in Intelligent Data Anylsis VI: 6th International Symposium on Intelligent Data Analysis*, 429-439.

[41] **Cluto**, http://www-users.cs.umn.edu/ karypis/cluto/index.html.

[42] **Deb, K.**, 2001. Multi-objective optimization using evolutionary algorithms, John Wiley and Sons, New York.

[43] **Branke, J., Deb, K., Dierolf, H. and Osswald, M.**, 2004. Finding Knees in Multi-objective Optimization, *Parallel Problem Solving from Nature*, 722-731.

[44] **The University of Saskatchewan Log**, http://ita.ee.lbl.gov/html/contrib/Sask-HTTP.html.

[45] **Polliwog Java Web-log Processor**, http://polliwog.sourceforge.net.

[46] **Charter, K., Schaeffer, J. and Szafron D.**, 2000. Sequence Alignment Using FastLSA, *International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences*, 429-439.

BIOGRAPHY

Gül Nildem Demir graduated from Istanbul Özel Alman Lisesi Lisesi in 1994. She received her B.Sc. in 2006 from Computer Engineering Department of Istanbul Technical University. She worked as full-time researcher in Web mining project supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) EEEAG project 105E162. During her work on M.Sc. thesis, she published the following related papers:

International Conferences

- **Göksedef, M., Gündüz-Öğüdücü, Ş. and Demir, G.N.**, 2007. A Web Recommender System Based on Ant Colony Optimization, *European Conference on Data Mining*, Lisbon, Portugal, 93-100.
- **Göksedef, M., Gündüz-Öğüdücü, Ş. and Demir, G.N.**, 2007. Effects of Session Representation Models on the Performance of Web Recommender Systems, *Workshop on Data Mining and Business Intelligence*, Istanbul, Turkey, 50-56.
- **Demir, G.N., Uyar, A. Ş. and Gündüz-Öğüdücü, Ş.**, 2007. Graph-based Sequence Clustering through Multiobjective Evolutionary Algorithms for Web Recommender Systems, *Genetic and Evolutionary Computation Conference*, London, UK, 256-266.