A Survey of Multiobjective Evolutionary Algorithms for Data Mining: Part I

Anirban Mukhopadhyay, *Senior Member, IEEE,* Ujjwal Maulik, *Senior Member, IEEE,*
Sanghamitra Bandyopadhyay, *Senior Member, IEEE,* and Carlos Artemio Coello Coello, *Fellow, IEEE*

***Abstract*—The aim of any data mining technique is to build an efficient predictive or descriptive model of a large amount of data. Applications of evolutionary algorithms have been found to be particularly useful for automatic processing of large quantities of raw noisy data for optimal parameter setting and to discover significant and meaningful information. Many real-life data mining problems involve multiple conflicting measures of performance, or objectives, which need to be optimized simultaneously. Under this context, multiobjective evolutionary algorithms are gradually finding more and more applications in the domain of data mining since the beginning of the last decade. In this two-part paper, we have made a comprehensive survey on the recent developments of multiobjective evolutionary algorithms for data mining problems. In this paper, Part I, some basic concepts related to multiobjective optimization and data mining are provided. Subsequently, various multiobjective evolutionary approaches for two major data mining tasks, namely feature selection and classification, are surveyed. In Part II of this paper, we have surveyed different multiobjective evolutionary algorithms for clustering, association rule mining, and several other data mining tasks, and provided a general discussion on the scopes for future research in this domain.**

***Index Terms*—Classification, feature selection, multiobjective evolutionary algorithms, Pareto optimality.**

I. Introduction

DATA MINING involves discovering novel, interesting, and potentially useful patterns from large data sets. The objective of any data mining process is to build an efficient predictive or descriptive model of a large amount of data that not only best fits or explains it, but is also able to generalize to new data. It is very important to optimize the model parameters for successful applications of any data mining approach. Often such problems, due to their complex nature, cannot be solved using standard mathematical techniques. Moreover, due to the large size of the input data, the problems sometimes become intractable. Therefore, designing efficient deterministic algorithms is often not feasible. Applications of evolutionary algorithms, with their inherent parallel architecture, have been found to be potentially useful for automatic processing of large amounts of raw noisy data for optimal parameter setting and to discover significant and meaningful information [2], [3].

Traditionally, evolutionary algorithms (EAs) [4] were used to solve single objective problems. However, many real-life problems have multiple conflicting performance measures or objectives, which must be optimized simultaneously to achieve a tradeoff. Optimum performance in one objective often results in unacceptably low performance in one or more of the other objectives, creating the necessity for a compromise to be reached [2]. This facet of multiobjective optimization is highly applicable in the data mining domain. For example, in association rule mining, a rule may be evaluated in terms of both its support and confidence, while a clustering solution may be evaluated in terms of several conflicting measures of cluster validity. Such problems thus have a natural multiobjective characteristic, the goal being to simultaneously optimize all the conflicting objectives. A number of EAs have been proposed in the literature for solving multiobjective optimization (MOO) problems [5], [6]. Unlike single-objective EAs, where a single optimum solution is generated in the final generation, the definition of optimality is not straightforward for the multiobjective case due to the presence of multiple objective functions. In MOO, the final generation yields a set of nondominated solutions, none of which can be improved on any one objective without degrading it in at least one other [5], [6]. Multiobjective evolutionary algorithms (MOEAs) [5], [6] have become increasingly popular in the domain of data mining over the last few years. Typical data mining tasks include feature selection, classification, clustering/biclustering, association rule mining, deviation detection, etc. A variety of MOEAs for solving such data mining tasks can be found in the literature. However, no previous effort has been made to review such methods in a systematic way.

Motivated by this, in this two-part paper, we attempt to make a comprehensive survey of the important recent developments of MOEAs for solving data mining problems. This survey focuses on the primary data mining tasks, namely feature selection, classification, clustering, and association rule mining, since most of the multiobjective algorithms that are applied to data mining have dealt with these tasks. In this paper, we discuss the basic concepts of multiobjective optimization and MOEAs, followed by fundamentals of data mining tasks

Manuscript received November 3, 2013; accepted November 3, 2013. Date of publication November 8, 2013; date of current version January 27, 2014. This work was supported by the Indo-Mexico Grant DST/INT/MEX/RPO-04/2008 from the Department of Science and Technology, India.

A. Mukhopadhyay is with the Department of Computer Science and Engineering, University of Kalyani, Kalyani 741235, India (e-mail: anirban@klyuniv.ac.in).

U. Maulik is with the Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, India (e-mail: umaulik@cse.jdvu.ac.in).

S. Bandyopadhyay is with the Machine Intelligence Unit, Indian Statistical Institute, Kolkata 700108, India (e-mail: sanghami@isical.ac.in).

C. A. Coello Coello is with CINVESTAV-IPN, Departamento de Computación (Evolutionary Computation Group), Mexico City, Mexico (e-mail: ccoello@cs.cinvestav.mx).

Digital Object Identifier 10.1109/TEVC.2013.2290086

and motivation for applying MOEAs for solving these data mining tasks. Subsequently, we review different MOEAs used for feature selection and classification tasks of data mining. In Part II of this paper [1], different MOEAs used for clustering, association rule mining, and other data mining tasks are surveyed followed by a discussion on the future scope of research.

II. Multiobjective Optimization

In this section, some basic concepts of MOO are first introduced. Then, an overview of available MOEAs is provided.

A. Concepts of Multiobjective Optimization

In many real-world situations, there may be several objectives that must be optimized simultaneously in order to solve a certain problem. This is in contrast to the problems tackled by conventional EAs, which involve optimization of just a single criterion. The main difficulty in considering multiobjective optimization is that there is no accepted definition of optimum in this case, and therefore it is difficult to compare one solution with another one. In general, these problems admit multiple solutions, each of which is considered acceptable and equivalent when the relative importance of the objectives is unknown. The best solution is subjective and depends on the need of the designer or decision maker [2], [5], [6].

We are interested in the multiobjective optimization problem (MOP), which can be stated as follows[1] [5], [6]:

$$\text{minimize } \vec{F}(\vec{x}) := \left[f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})\right] \tag{1}$$

subject to

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, m \tag{2}$$

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{3}$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables, $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, ..., k$ are the objective functions and $g_i, h_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, ..., m$, $j = 1, ..., p$ are the constraint functions of the problem.

To describe the concept of optimality in which we are interested, we will introduce next a few definitions.

Definition 1: A vector $\vec{u} = (u_1, \ldots, u_k)$ is said to dominate (in a Pareto sense) another vector $\vec{v} = (v_1, \ldots, v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if $\vec{u}$ is partially less than $\vec{v}$, i.e., $\forall i \in \{1, \ldots, k\}$, $u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.

Definition 2: A solution $\vec{x} \in \mathcal{F}$ (where $\mathcal{F}$ is the feasible region, in which the constraints are satisfied) is said to be Pareto optimal with respect to $\mathcal{F}$ if and only if (iff) there is no $\vec{x'} \in \mathcal{F}$ for which $\vec{v} = \vec{F}(\vec{x'}) = (f_1(\vec{x'}), \ldots, f_k(\vec{x'}))$ dominates $\vec{u} = \vec{F}(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x}))$.

Definition 3: For a given MOP, $F(\vec{x})$, the Pareto optimal set $\mathcal{P}^*$ is defined as

$$\mathcal{P}^* := \{\vec{x} \in \mathcal{F} \mid \neg\exists\, \vec{x'} \in \mathcal{F} \;\; \vec{F}(\vec{x'}) \preceq \vec{F}(\vec{x})\}. \tag{4}$$

[1]Without loss of generality, we will assume only minimization problems.

Definition 4: For a given MOP, $\vec{F}(\vec{x})$, and Pareto Optimal Set $\mathcal{P}^*$ the Pareto front $\mathcal{PF}^*$ is defined as

$$\mathcal{PF}^* := \{\vec{u} = \vec{F}(\vec{x}) \mid \vec{x} \in \mathcal{P}^*\}. \tag{5}$$

We thus wish to determine the Pareto optimal set from the set $\mathcal{F}$ of all the decision variable vectors that satisfy (2) and (3). Note however that in practice, not all the Pareto optimal set is normally desirable (e.g., it may not be desirable to have different solutions that map to the same values in objective function space) or achievable.

B. Multiobjective Evolutionary Algorithms

Traditional search and optimization methods such as gradient-based methods are difficult to extend to the multiobjective case because their basic design precludes the consideration of multiple solutions. In contrast, population-based methods such as evolutionary algorithms are well-suited for handling such situations. There are different approaches for solving multiobjective optimization problems [5], [6].

MOEAs have evolved over several years, starting from traditional aggregating approaches to the elitist Pareto-based approaches and, more recently, to the indicator-based algorithms. In the aggregating approaches, multiple objective functions are combined into a single scalar value using weights, and the resulting single-objective function is then optimized using conventional evolutionary algorithms. In population-based non-Pareto approaches such as the vector evaluated genetic algorithm [7], a special selection operator is used and a number of subpopulations are generated by applying proportional selection based on each objective function in turn. Among the Pareto-based approaches, multiple objective GA [8], niched Pareto GA (NPGA) [9], and nondominated sorting GA (NSGA) [10] are the most representative nonelitist MOEAs. Although these techniques take into account the concept of Pareto optimality in their selection mechanism, they do not incorporate elitism and, therefore, they cannot guarantee that the nondominated solutions obtained during the search are preserved. In the late 1990s, a number of elitist models of Pareto-based multiobjective evolutionary algorithms were proposed. The most representative elitist MOEAs include strength Pareto evolutionary algorithm (SPEA) [11] and SPEA2 [12], Pareto archived evolutionary strategy (PAES) [13], Pareto envelope-based selection algorithm (PESA) [14] and PESA-II [15], and nondominated sorting genetic algorithm-II (NSGA-II) [16]. Most of the recent applications of MOEAs for data mining problems have used one of these Pareto-based elitist approaches as their underlying optimization strategy. A more recent trend regarding the design of MOEAs is to adopt a selection mechanism based on some performance measure. For example, the indicator-based evolutionary algorithm [17] is intended to be adapted to the user's preferences by formalizing such preferences in terms of continuous generalizations of the dominance relation. Since then, other indicator-based approaches, such as the S metric selection evolutionary multiobjective optimization algorithm (SMS-EMOA) [18] (which is based on the hypervolume [19]) have also been proposed. The main advantage of indicator-based MOEAs such as SMS-EMOA is that they seem to

scale better in the presence of many objectives (four or more). However, approaches based on the hypervolume are very computationally expensive. Since we do not review any application of an indicator-based MOEA in data mining, these approaches are not discussed further in this paper, and they are mentioned only for the sake of completeness.

III. Data Mining Fundamentals

Data mining involves discovering interesting and potentially useful patterns of different types, such as associations, summaries, rules, changes, outliers, and significant structures. Commonly, data mining and knowledge discovery are treated as synonymous, although some scientists consider data mining to be an integral step in the knowledge discovery process. In general, data mining techniques comprise three components [20]: a model, a preference criterion, and a search algorithm. Association rule mining, classification, clustering, regression, sequence and link analysis, and dependency modeling are some of the most common functions in current data mining techniques. Model representation determines both the flexibility of the model for representing the underlying data and the interpretability of the model in human terms.

Data mining tasks can broadly be classified into two categories: predictive or supervised and descriptive or unsupervised [3], [21]. The predictive techniques learn from the current data in order to make predictions about the behavior of new datasets. On the other hand, the descriptive techniques provide a summary of the data. The most commonly used tasks in the domain of data mining include feature selection, classification, regression, clustering, association rule mining, deviation detection, etc. In this paper, we have mainly focused on the four tasks, i.e., feature selection, classification, clustering, and association rule mining. This is because most of the multiobjective algorithms applied to data mining have dealt with these tasks. MOEAs have thoroughly been applied in these four primary fields of data mining. These are briefly described in Part I (feature selection and classification) and Part II [1] (clustering and association rule mining) of this paper. However, for the sake of completeness, other data mining tasks, where MOEAs have found applications are also discussed in Part II of this paper.

A. Feature Selection

Feature selection problem deals with selection of an optimum relevant set of features or attributes that are necessary for the recognition process (classification or clustering). It helps reduce the dimensionality of the measurement space. The goal of feature selection is mainly threefold. First, it is practically and computationally difficult to work with all the features if the number of features is too large. Second, many of the given features may be noisy, redundant, and irrelevant to the classification or clustering task at hand. Finally, it is a problem when the number of features becomes much larger than the number of input data points [2]. For such cases, reduction in dimensionality is required to permit meaningful data analysis [3]. Feature selection facilitates the use of easily computable algorithms for efficient classification or clustering.

In general, the feature selection problem (Ω, P) can formally be defined as an optimization problem: determine the feature set F^* for which

$$P(F^*) = \min_{F \in \Omega} P(F, X) \tag{6}$$

where Ω is the set of possible feature subsets, F refers to a feature subset, and $P : \Omega \times \psi \rightarrow (R)$ denotes a criterion to measure the quality of a feature subset with respect to its utility in classifying/clustering the set of points $X \in \psi$. The elements of X, which are vectors in d-dimensional space, are projected into the subspace of dimension $d_F = |F| \leq d$ defined by F. P is used to judge the quality of this subspace.

Feature selection can be either supervised or unsupervised. For the supervised case, the actual class labels of the data points are known. In filter approaches for supervised feature selection, features are selected based on their discriminatory power with regard to the target classes. In wrapper approaches for supervised feature selection, the utility of F is usually measured in terms of the performance of a classifier by comparing the class labels predicted by the classifier for feature space F with the actual class labels. For the unsupervised case, actual class labels are not available. Hence, in filter approaches, features are selected based on the distribution of their values across the set of point vectors available. In wrapper-based unsupervised feature selection, the utility of a feature subset F is generally computed in terms of the performance of a clustering algorithm when applied to the input dataset in the feature space F.

B. Classification

The problem of classification is basically one of partitioning the feature space into regions, one region for each category of inputs [22]. Thus, it attempts to assign every data point in the entire feature space to one of the possible (say, K) classes. Classifiers are usually, but not always, designed with labeled data, in which case these problems are sometimes referred to as supervised classification (where the parameters of a classifier function are learned). Supervised classifiers assume that a set of training data is available. The training dataset consists of a set of instances that are properly labeled with the correct class labels. A learning algorithm then generates a model that attempts to minimize the prediction error on the training instances as much as possible, and also generalize as far as possible to new data.

The problem of supervised classification can formally be stated as follows. Given an unknown function $g : \mathcal{X} \rightarrow \mathcal{Y}$ (the ground truth) that maps input instances $\mathbf{x} \in \mathcal{X}$ to output class labels $y \in \mathcal{Y}$, and a training dataset $\mathbf{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ which is assumed to represent accurate examples of the mapping g, produce a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that approximates the correct mapping g as closely as possible. The learning algorithms help in identifying the class boundaries in the training set as correctly as possible by minimizing the training error.

Various classification algorithms are available in the literature. Some common examples of the supervised pattern classification techniques are the nearest neighbor (NN) rule, the Bayes maximum likelihood classifier, support vector machines (SVM), and neural networks [3], [22], [23]. A number

of applications of evolutionary algorithms for classification purposes can also be found in the literature [22].

C. Clustering

Clustering [24] is an important unsupervised classification technique where a set of patterns, usually vectors in a multidimensional space, are grouped into clusters in such a way that patterns in the same cluster are similar in some sense and patterns in different clusters are dissimilar in the same sense.

Clustering in a d-dimensional Euclidean space $\mathbb{R}^d$ is the process of partitioning a given set of n points into a number, say K, of groups (or clusters) $\{C_1, C_2, \ldots, C_K\}$ based on some similarity/dissimilarity metric. The value of K may or may not be known *a priori*. The main objective of any clustering technique is to produce a $K \times n$ partition matrix $U(X)$ of the given dataset X consisting of n patterns, $X = \{x_1, x_2, \ldots, x_n\}$. The partition matrix may be represented as $U = [u_{kj}]$, $k = 1, \ldots, K$ and $j = 1, \ldots, n$, where u_{kj} is the membership of pattern x_j to cluster C_k. In the case of hard or crisp partitioning

$$u_{kj} = \begin{cases} 1 & \text{if } x_j \in C_k \\ 0 & \text{if } x_j \notin C_k. \end{cases} \tag{7}$$

On the other hand, for probabilistic fuzzy partitioning of the data, the following conditions hold on U (representing nondegenerate clustering):

$$\forall k \in \{1, 2, \ldots, K\} \quad 0 < \sum_{j=1}^{n} u_{kj} < n \tag{8}$$

$$\forall j \in \{1, 2, \ldots, n\} \quad \sum_{k=1}^{K} u_{kj} = 1 \tag{9}$$

and

$$\sum_{k=1}^{K} \sum_{j=1}^{n} u_{kj} = n. \tag{10}$$

Several clustering methods are available in the literature. These can be broadly categorized into hierarchical (agglomerative and divisional), partitional (K-means, fuzzy C-means, etc.), and density-based (density-based spatial clustering of applications with noise, clustering for large applications, etc.), clustering [2], [3]. Evolutionary algorithms have also widely been used for clustering [25].

D. Association Rule Mining

The principle of association rule mining (ARM) [26] lies in the market basket or transaction data analysis. Association analysis is the discovery of rules showing attribute–value associations that occur frequently. Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n items and X be an itemset where $X \subset I$. A k-itemset is a set of k items. Let $T = \{(t_1, X_1), (t_2, X_2), \ldots, (t_m, X_m)\}$ be a set of m transactions, where t_i and X_i, $i = 1, 2, \ldots, m$, are the transaction identifier and the associated itemset, respectively. The cover of an itemset X in T is defined as follows:

$$cover(X, T) = \{t_i | (t_i, X_i) \in T, X \subset X_i\}. \tag{11}$$

The support of an itemset X in T is

$$support(X, T) = |cover(X, T)| \tag{12}$$

and the frequency of an itemset is

$$frequency(X, T) = \frac{support(X, T)}{|T|}. \tag{13}$$

Thus, support of an itemset X is the number of transactions where all the items in X appear in each transaction. The frequency of an itemset is the probability of its occurrence in a transaction in T. An itemset is called frequent if its support in T is greater than some threshold min_sup. The collection of frequent itemsets with respect to a minimum support min_sup in T, denoted by $\mathcal{F}(T, min\_sup)$, is defined as

$$\mathcal{F}(T, min\_sup) = \{X \subset I, support(X, T) > min\_sup\}. \tag{14}$$

The objective of ARM is to find all rules of the form $X \Rightarrow Y$, $X \bigcap Y = \emptyset$ with probability $c\%$, indicating that if itemset X occurs in a transaction, the itemset Y also occurs with probability $c\%$. X and Y are called the antecedent and consequent of the rule, respectively. Support of a rule denotes the percentage of transactions in T that contains both X and Y. This is taken to be the probability $P(X \bigcup Y)$. An association rule (AR) is called frequent if its support exceeds a minimum value min_sup.

The confidence of a rule $X \Rightarrow Y$ in T denotes the percentage of the transactions in T containing X that also contains Y. It is taken to be the conditional probability $P(X|Y)$. In other words

$$confidence(X \Rightarrow Y, T) = \frac{support(X \bigcup Y, T)}{support(X, T)}. \tag{15}$$

A rule is called confident if its confidence value exceeds a threshold min_conf. Formally, the ARM problem can be defined as follows. Find the set of all rules R of the form $X \Rightarrow Y$ such that

$$\begin{aligned} R = \quad & \{X \Rightarrow Y | X, Y \subset I, X \bigcap Y = \emptyset, \\ & X \bigcup Y \subseteq \mathcal{F}(T, min\_sup), \\ & confidence(X \Rightarrow Y, T) > min\_conf\}. \end{aligned} \tag{16}$$

Generally, the ARM process consists of the following two steps [26].

1) Find all frequent itemsets.
2) Generate strong ARs from the frequent itemsets.

The number of itemsets grows exponentially with the number of items $|I|$. A commonly used algorithm for generating frequent itemsets is the *a priori* algorithm [26], [27]. This is based on the concept of downward closure property which states that if even one subset of an itemset X is not frequent, then X cannot be frequent. It starts from all itemsets of size one, and proceeds in a recursive fashion. If any itemset X is not frequent, then that branch of the tree is pruned, since any possible superset of X can never be frequent.

E. Why to Use Multiobjective Data Mining

The most important question in data mining problems is how to evaluate a candidate model, and, obviously, this question depends on the type of data mining task at hand. For example, a feature selection model may be evaluated based on its performance in correctly classifying the dataset, whereas a clustering model can be evaluated based on some cluster

validity index. Thus, most of the data mining problems can be thought of as optimization problems, where the aim is to evolve a candidate model that optimizes certain performance criteria. However, the majority of data mining problems have multiple criteria to be optimized. For example, a feature selection problem may try to maximize the classification accuracy, while minimizing the size of the feature subset. Similarly, a rule mining problem may optimize several rule interestingness measures such as support, confidence, comprehensibility, and lift [28] at the same time. Similar cases may arise for a clustering problem also where one tries to optimize several cluster validity indices simultaneously to obtain robust and improved clustering, because no single validity index performs well for all types of datasets [2]. Hence, most of the data mining problems are multiobjective in nature. Therefore, it is natural to pose data mining problems as multiobjective ones. For this reason, over the past decade, several researchers have applied MOEAs for different data mining problems.

An MOEA provides a set of nondominated solutions, which the user can compare (it is important to keep in mind that the set of nondominated solutions represents the best possible tradeoffs among the objectives). Then, a single solution from this set can be chosen, based on the user's preferences. There are several possible schemes for selecting a single solution. For example, one can generate a consensus solution that shares the knowledge contained in all the nondominated solutions. This method has been successfully used in clustering [29] and classifier ensemble [30] problems. Some other approaches for choosing the final solution from the nondominated front are discussed in this survey in the subsequent sections. It is worth noting, however, that for some problems, all the nondominated solutions are considered final solutions without having to choose a single solution from the set. For example, in the problem of association rule mining [31] or biclustering [32], all the nondominated solutions, representing rules and biclusters, respectively, are considered the final solution set.

Due to the above reasons, MOEAs have been popularly used for data mining problems. In this two-part paper, we have surveyed a number of different MOEAs techniques applied to data mining problems mainly focusing on encoding techniques, objective functions, evolutionary operators, and final solution selection strategies. In this part of this paper, we have reviewed different MOEAs used for feature selection and classification problems. In Fig. 1, we have outlined the different MOEA-based feature selection and classification approaches reviewed in this part along with their corresponding references. In subsequent sections, these approaches are reviewed in detail.

IV. MOEAs for Feature Selection

The feature selection problem can easily be posed as an optimization problem where the goal is to select a subset of features for which some feature-subset evaluation criterion is optimized. Therefore, genetic and other evolutionary algorithms have been widely used for the feature selection problem [33], [34]. Evolutionary algorithms for feature selection techniques mostly take a wrapper approach where a subset of features are encoded in a chromosome and a feature evaluation criterion is used as the fitness function. The feature subsets are evaluated based on how well the selected features classify (for the supervised case) or cluster (for the unsupervised case) the dataset. However, evaluation of selected features by a single criterion does not work equally well for all datasets. Therefore, the need of simultaneously optimizing multiple such criteria arose. Multiobjective feature selection helps improve the robustness of the feature selection methods. In the recent past, a number of MOEAs, both in supervised and unsupervised domains, have been proposed. Next, we review these methods.

A. Underlying MOEAs

Several MOEAs have been used as the underlying optimization tool for a number of different feature selection algorithms. In [35] and [36], NPGA has been adopted. An elitist version of NPGA, called ENPGA, has been employed in [37]. NSGA has been adopted in [38]–[41]. In [42]–[44], NSGA-II has been used as the MOO tool. In [45], a reduced Pareto set genetic algorithm (elitist) (RPSGAe) [46] was employed as the optimization method. In RPSGAe, a clustering algorithm is applied to reduce the size of the Pareto optimal set. In [47], an evolutionary local search algorithm (ELSA) was used. ELSA works on each objective separately [48]. In [49], PESA-II was used for multiobjective feature selection.

B. Chromosome Representation

The first and foremost step toward solving a feature selection problem using MOEAs is to encode a possible feature subset in the form of a chromosome. Almost all the MOEA-based feature selection algorithms have used a binary chromosome to encode a feature subset. The length of each chromosome is taken as d, where d is the total number of features. Each position of the chromosome can take either a 1 or a 0 value. If position i has value 1, then feature i is considered to be a part of the selected feature subset. If position i has a value of 0, then the corresponding feature is ignored. For example, if $d = 10$, then the chromosome 0010110101 encodes the feature subset $\{3, 5, 6, 8, 10\}$. This approach has been used in both supervised [35], [38], [42], [45], [50] and unsupervised [39], [49], [51] feature selection techniques using MOEAs. In the case of unsupervised methods, the goodness of the candidate feature subset encoded in a chromosome is measured in terms of the performance of some clustering algorithm on the projected subspace of the input dataset. As the number of clusters has significant impact on the performance of a clustering algorithm, it is also encoded in the chromosome in [39] and [49].

One possible criticism of binary encoding could be that the chromosome lengths for binary encoding may be very large if the dimension of the input dataset is large. The use of binary encoding may lead to longer decoding times and slower convergence, but has the advantage of being a universal encoding, which can be used to represent any sort of decision variables. The use of alternative encodings, such as integers, would lead to a reduced chromosome length, but requires special operators for recombination and mutation, and may also require additional mechanisms (for example, remotion of duplicate individuals).

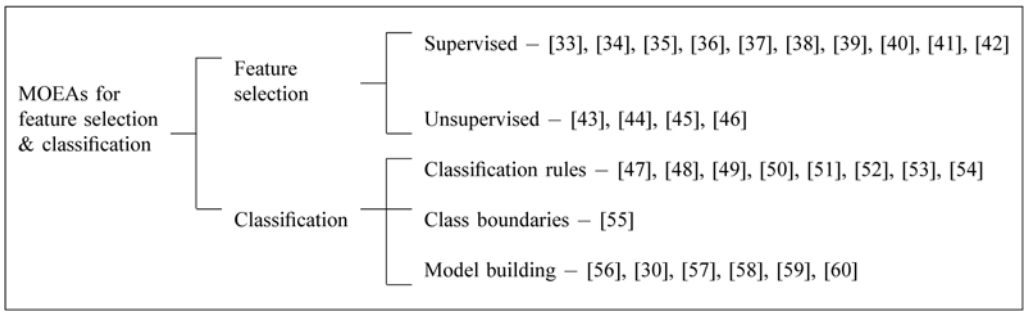

Fig. 1. MOEAs for feature selection and classification tasks surveyed in Part I.

C. Objective Functions

Most of the multiobjective evolutionary feature selection approaches in current use have addressed the supervised problem, i.e., when the class labels of the training objects are known. However, there are a few unsupervised approaches as well. Here, we briefly discuss the different objective functions that have been simultaneously optimized by different multiobjective feature selection algorithms for both the supervised and unsupervised cases.

1) *Supervised Case:* Supervised approaches assume the existence of a training set of objects for which actual class labels are known. Therefore, in these cases, usually some classification algorithms are used to measure the goodness of the selected feature subset based on how well they can classify the training examples using a certain classifier. The performance of the feature subset is evaluated using some classification performance metrics. One of the pioneering works in this regard is [35], where the two objective functions correspond to the misclassification rate and the number of features. Both objective functions are minimized with the expectation of reducing the misclassification rate as much as possible, with a minimum set of features. Two classifiers, namely probabilistic neural network and multilayer perceptron (MLP), are used to compute the misclassification rate. The misclassification rate for a subset of features is computed by randomly breaking the input training set into three subsets. The three subsets are used for training, validation, and testing, respectively. Several such cycles are performed and the average misclassification rate is computed. In a similar approach, the authors have computed the same objective values based on the generalized regression neural network classifier [36]. The authors have also used a 1-NN classifier in [37] for optimizing three objective functions, namely number of features, sensitivity, and specificity of the classification result. In [45], an SVM classifier with radial basis function kernel has been used for evaluating the encoded feature subset. Several objective functions regarding classification performance, such as accuracy, false positive rate, false negative rate, precision, recall, F-measure, and number of features, are considered and different combinations are tested. The metrics are computed based on ten-fold cross-validation on the training set. In [50], three objective functions have been optimized. First is the minimization of misclassification rate, second is the minimization of imbalance in class sizes, and third is the minimization of number of features. Seventy percent of the training patterns are used to train a feature correlation-based classifier (GS-classifier) [52], [53], and the above measures are computed on the remaining 30%. In [54], the two objective functions to be minimized are the misclassification error and the size of the tree using a C4.5 decision tree classifier built using the selected feature subset. Two objectives, the number of features and the misclassification rate (using a neural network classifier), have also been minimized in [38] and [40]. In [55], these two criteria are optimized, but they have used a logistic regression and SVM classifier. To find out the most relevant and nonredundant feature subset, in [42], the two objective functions adopted were the minimization of correlation among the selected features and the maximization of correlation among the selected features and class labels of training patterns. In [43], different combinations of feature evaluation criteria, such as the number of inconsistent pattern pairs, feature versus class correlation, Laplacian score, representation entropy, and interclass and interclass distances have been used as the objective functions. They have used a filtering approach for computing these criteria independent of a classifier.

Although various approaches have been proposed in the literature, it is worth noting that feature selection results heavily depend on the chosen classification algorithm that is used as the wrapper. Moreover, the number of objective functions and their choice play an important role in the selection of the final feature subset. It would be therefore a nice idea to perform a comparative study of the performance of the proposed techniques based on some benchmark datasets. To the authors' best knowledge, no comparative study of this sort has been reported so far in the specialized literature.

2) *Unsupervised Case:* There have been a few works related to the development of evolutionary algorithms for multiobjective unsupervised feature selection as well. For the unsupervised case, the algorithms do not assume the existence of true class labels, and, therefore, there is no training set. For this case, usually, a clustering algorithm is used to evaluate a feature subset on the basis of how well these features are able to identify the clustering structure of the dataset. In this regard, some cluster validity index [56] is used to evaluate the goodness of the clustering solution generated by the feature subset. One of the first studies in this direction was done in [51]. K-means and expectation maximization clustering are used to evaluate a feature subset encoded in a

chromosome. Different clustering objectives, all of which are a function of the number of clusters, the number of features, the intraclass similarity, and the interclass dissimilarity, are simultaneously optimized. The MOEA adopted in this case is the ELSA. In a similar work, Morita *et al.* [39] also used K-means as the wrapper clustering algorithm, but they minimized two objective functions, namely, the number of features and the cluster validity index DB [57]. Handl and Knowles [49] examined different combinations of objective functions in the context of multiobjective evolutionary feature selection. One objective was taken as the number of selected features, which is optimized along with one of the DB, normalized DB or silhouette indices (wrapper approach), or with an entropy measure (filter approach). The K-means algorithm is used as the wrapper clustering algorithm. In [44], again, the number of features and the DB index/normalized DB index are simultaneously optimized.

For the unsupervised case, the output of the algorithms heavily depends (as in the supervised case) on the choice of the objective functions. Moreover, the choice of the clustering algorithm as a wrapper evaluator of the candidate feature subset also plays a major role in deciding the final feature subset. In [49], a preliminary effort has been made to compare the performance of different combinations of objective functions for multiobjective unsupervised feature selection. However, a more detailed effort considering different combinations of clustering algorithms and objective functions could be more beneficial to researchers.

D. Evolutionary Operators

The evolutionary operators, crossover and mutation, are used to produce the population of the following generation in an MOEA. Since the MOEAs reviewed in this paper use binary encoding, this explains that single-point and uniform crossover had been the most popular choices in such references. Single-point crossover has been adopted in [38]–[43] and [54], whereas uniform crossover has been employed in [44] and [49]. A few exceptions are also noticed. In [36], a two-point crossover operator is employed. In [35], [37], and [51], a commonality-based crossover operation is used. This operator takes two agents, a parent a and a random mate, and then it scans each bit of both agents. Whenever the bits are different, one of the two bits is randomly chosen to obtain the corresponding offspring bit. Thus, the mate contributes only to obtain the bit string of the offspring, in which all the common features of the parents are inherited. In all the above works, a standard bit-flip mutation operator has been adopted.

It is to be noted that most of the MOEAs reported in the papers reviewed depend on standard crossover and mutation operators, without having to rely on more sophisticated operators. In the only references in which a nonstandard operator is adopted (i.e., the commonality-based crossover previously indicated, which is adopted in [35], [37], and [51]), no comparisons are provided with respect to standard crossover operators, and, therefore, it is not possible to judge the actual improvements achieved by its use.

E. Obtaining the Final Solution

As stated before, MOEAs produce a set of nondominated solutions in the final generation. Each of these solutions encodes a possible feature subset. None of the nondominated solutions can be said to be better than the others. However, for practical reasons, it is necessary to select a single solution from the final nondominated set. In many of the multiobjective feature selection approaches that were reviewed for this survey, this issue has been properly addressed.

1) *Supervised Case:* For the supervised case, identification of the final solution is a relatively easy task, since a labeled training set, which can guide this selection, is available. In [54], an internal cross-validation approach is adopted to select the final feature subset. Each nondominated solution is evaluated by k-fold cross-validation and the solutions, which stay nondominated at each fold, are returned as the selected promising solutions. However, this internal cross-validation is computationally expensive. Therefore, they have used this only in the final generation, instead of employing it at every generation. Moreover, it is to be noted that this method results in multiple final solutions, from which the user must choose one subjectively. In [50], a simple aggregation of the objective function values is used as the criterion for selecting the final solution from the nondominated front. This is one of the simplest methods, and is computationally inexpensive. However, the final selection depends on the aggregation function used. The authors have not clearly explained the aggregation function used to find the raw fitness of the individuals. However, aggregation of the fitness values to decide the final solution may raise the question on the requirement of generating the complete Pareto front, because one can optimize the aggregated fitness function directly. In [38], a validation dataset is used for measuring the performance of each nondominated solution on independent data. The solution that provides the best performance on the validation set is chosen as the final solution. This is done with the expectation of selecting the solution with the best generalization power to an unknown validation dataset. In [42], a combination of the objective functions, feature correlation, and feature versus class correlation, called relative overall correlation, is used to select the final feature subset from the nondominated front. However, the authors have not explained well why this relative overall correlation is not directly used for optimization. In [43], a compromise programming-based approach is used to select the final solution from the nondominated front. As the authors have not explained the method in detail, it is difficult to discuss its merits or possible drawbacks any further.

2) *Unsupervised Case:* For the unsupervised case, the selection of the final solution is more difficult since no labeled data is available. Morita *et al.* [39] computed the classification accuracy using the feature subset encoded in each nondominated solution, and then selected the solution providing maximum classification accuracy. Therefore, it is evident that here the authors used a supervised technique for choosing the final solution. This approach is not obviously acceptable when class labels of the samples are unknown. In [49], a knee-based approach is adopted. Here, an MOEA

is applied first on the original dataset in order to obtain a nondominated front. Thereafter, the same algorithm is applied on a random dataset that is created with the same bound of the original dataset. The resulting nondominated front is called the control front. Finally, the authors compare the solutions of the actual front with those of the control fronts for each feature, and the solution that is most distant from its corresponding control solution is identified as the final knee solution. This approach is able to identify the best tradeoff solution from the nondominated front. However, applying the feature selection algorithm on random datasets to generate the control fronts is time consuming. Hence, this method is computationally expensive. In another knee-based approach [44], the authors first sort the nondominated solutions on the basis of one of the objectives, and then they find the slope of each solution by computing the ratio of differences of the first objective function to the difference of the second objective function in two consecutive solutions. Next, they select the solution for which the change of slope is maximum. This approach is very straightforward in comparison with the control front generation, and this is also more computationally efficient. However, it may not be easy to extend this method if the number of objective functions is more than two.

Although a number of different approaches have been proposed for selecting the final solution from the nondominated front, surprisingly, none of them utilized the information contained in the complete front effectively. Due to the nature of multiobjective optimization problems, all the solutions of the final front share some information of the underlying dataset. But, none of the solution selection methods has tried to combine this information from all the nondominated solutions through some kind of ensemble. This approach might be, indeed, effective to exploit knowledge from the solutions obtained. In addition, to the authors' best knowledge, a systematic comparison of methods used for selecting the final solution from the Pareto front is also missing in the literature.

F. Relative Comparison and Applications

For the purposes of getting an overview of all the MOEA-based methods that we have reviewed for feature selection, we provide in Table I a comparative description of them in terms of the underlying MOO tool, type of approach (supervised/unsupervised), encoding strategy, objective functions, evolutionary operators, and method for obtaining the final solution. We have arranged the approaches in ascending order of the publication year, and for better understanding, we distinguish between the supervised and the unsupervised cases. The names of the algorithms are mentioned if they are found in the corresponding publications. It is evident from the table that a variety of MOEAs have been used to design feature selection techniques. It is also worth noting that a systematic comparison of all the MOEA-based feature selection algorithms is missing. However, in different publications, the authors have compared their proposed approaches with different existing nonevolutionary algorithms, or with different modifications of the proposed approaches. Some of the possible reasons for not comparing different approaches may be unavailability of code/software and limited information for reproducing existing results.

Interestingly, in different works, the authors have used their proposed MOEA-based feature selection technique in various real-life applications. For example, in [45], the authors have applied their method in cardiac single proton emission computed tomography diagnosis. In [37], the proposed evolutionary multiobjective feature selection technique has been used for industrial machinery fault diagnosis. Multiobjective feature selection has also been applied in bioinformatics. For example, Liu and Iba [50] have applied their method in selecting informative genes from microarray gene expression data. In [38], multiobjective feature selection has been employed for handwritten digit string recognition. Another interesting application, namely, bankruptcy prediction, has been addressed in [55] using multiobjective evolutionary feature selection. This discussion shows that MOEA-based feature selection approaches have potential to be used for a wide variety of real-life problems.

V. MOEAs for Classification

MOEAs have been widely used for classification. There are mainly three different approaches. The most commonly studied approach is the use of MOEAs for evolving a good set of classification rules. The second approach is to employ MOEAs to define the class boundaries (hyperplanes) in the training data. The final approach is to use MOEAs for training and to model the construction of well known classifiers such as neural networks and decision tree classifiers. Here, we review some representative algorithms adopted in these three types of approaches.

A. Evolving Classification Rules

A classification rule can be represented as an if-then rule of the form *If* <*condition*> *Then* <*class*>. The <*condition*> represents the antecedent of the rule, which usually denotes a set of attribute-value pairs combined with an *and* operator. For example, a classification rule may be as follows: if height > 6 feet and weight > 70 kg then class=Big. It is to be noted that these attribute-value pairs are generated on categorical values. Therefore, if some attribute consists of continuous values, it must be first discretized to make it categorical before using the attribute in a classification rule. The objective of any rule-based classification system is to identify a good set of classification rules that can properly represent the training dataset, i.e., that provides a good classification performance on the training data.

1) *Underlying MOO Algorithms:* In the majority of the MOEA-based classification approaches, NSGA-II has been adopted as the underlying MOO algorithm for optimization. For example, NSGA-II has been used in a series of works by Ishibuchi *et al.* [58]–[62] for fuzzy classification rule mining. NSGA-II has also been employed in fuzzy classification rule discovery in [63]–[65]. In [66], a nonfuzzy categorical classification rule mining algorithm is proposed using an elitist multiobjective genetic algorithm (EMOGA). Another multiobjective fuzzy classification rule mining approach has

TABLE I
COMPARISON OF DIFFERENT MOEAS FOR FEATURE SELECTION

Algorithm	Underlying MOO tool	Type	Encoding	Wrapper/ filter	Objective functions	Evolutionary operators	Final solution from non-dominated front
Emmanouilidis et. al. [33], 2000	NPGA	Supervised	Binary (Features)	PNN, MLP claasifiers	Misclassification rate, number of selected features	Commonality crossover, bit-flip mutation	None
Emmanouilidis et. al. [34], 2001	NPGA	Supervised	Binary (Features)	GRNN classifier	RMSE, number of selected features	Two-point crossover, bit-flip mutation	None
Gaspar-Cunha [35], 2001	RPSGAe	Supervised	Binary (Features)	SVM classifier	Different combinations of Classification accuracy, FPR, FNR, F-measure, number of selected features	Not mentioned	None
Emmanouilidis [36], 2002 (EMOFS)	ENPGA	Supervised	Binary (Features)	1-NN classifier	Sensitivity, specificity number of selected features	Commonality crossover, bit-flip mutation	None
Pappa et. al. [37], 2002 (MOFSS)	Non-standard	Supervised	Binary (Features)	C 4.5 classifier	Misclassification rate, size of C4.5 classification tree	Bit-swapping crossover, bit-flip mutation	Internal cross-validation
Liu and Iba [38], 2002	NPGA variant	Supervised	Binary (Features)	GS classifier	Misclassification rate, class imbalance number of selected features	Not mentioned	Aggregated objective values (raw fitness)
Oliveira and Sabourin [39], 2003	NSGA	Supervised	Binary (Features)	MLP classifier	Misclassification rate, number of selected features	One-point crossover, bit-flip mutation	Validation database
Wang and Huang [40], 2009	NSGA-II	Supervised	Binary (Features)	Filter approach (no classifier)	Correlation among features, feature vs. class correlation	One-point crossover, bit-flip mutation	Based on relative overall correlation
Mendes et. al. [41], 2010	RPSGA	Supervised	Binary (Features)	LR and SVM classifiers	Classification accuracy, number of features	Not mentioned	None
Venkatadri and Rao [42], 2010	NSGA-II	Supervised	Binary (Features)	Filter approach (no classifier)	Different combinations of number of inconsistent pattern pairs, feature vs. class correlation, Laplacian score, representation entropy, intra- and inter-class distance	One-point crossover, bit-flip mutation	Compromise programming
Kim et. al. [43], 2002 (ELSA/KM, ELSA/EM)	ELSA	Unsupervised	Binary (Features)	K-means & EM clustering	Function of number of features, function of number of clusters, intra- and inter-cluster similarities	Commonality crossover, bit-flip mutation	None
Morita et. al. [44], 2003	NSGA	Unsupervised	Binary (features + no. of clusters)	K-means clustering	DB index, number of selected features	One-point crossover, bit-flip mutation	Classification accuracy (supervised)
Handl and Knowles [45], 2006	PESA-II	Unsupervised	Binary (features + no. of clusters), (only features in filter approach)	K-means clustering and Filter approach	Combinations of number of selected features, and one of DB, normalized DB, silhouette and entropy index	Uniform crossover, bit-flip mutation	Select knee solution using control front
Mierswa and Wurst [46], 2006	NSGA-II	Unsupervised	Binary (Features)	K-means clustering	Number of selected features and DB / normalized DB	Uniform crossover, bit-flip mutation	Select knee solution based on slope

been presented in [67], where PAES has been utilized for optimizing the evolved rules.

2) *Chromosome Representation:* For evolving classification rules using MOEAs, one has to first encode the classification rules in the chromosomes. There are mainly two approaches in this regard. The first one is the Pittsburgh approach, in which a set of rules is encoded in a single chromosome. The second one is the Michigan approach, where each chromosome encodes one rule. From the classification point of view, the Pittsburgh approach is more useful since here each chromosome represents a complete classifier system. The number of rules encoded in a chromosome can either be fixed or variable. Therefore, from the final nondominated solutions, one particular solution must be picked up to represent a complete classification system. Most of the MOEAs available for building rule-based classifiers use this approach [68]. On the other hand, the Michigan approach is usually much easier and less complex, because each chromosome encodes only one rule. Thus, in this approach, the final solution is composed of the full nondominated set of classification rules [68].

Most of the multiobjective evolutionary classification rule-based systems are focused on evolving fuzzy classification rules. Let us assume that the training set contains m patterns of the form $x_p = \{x_{p1}, x_{p2}, \ldots, x_{pn}\}$, $p = 1, \ldots, m$, i.e., each pattern is n-dimensional. There are M classes. A fuzzy If-Then classification rule takes the form [58]

$$R_q = If\ x_1\ is\ A_{q1}\ and\ \ldots\ and\ x_n\ is\ A_{qn}$$
$$Then\ Class = C_q\ with\ CF_q$$

where R_q is the qth fuzzy rule, $x = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, A_{qi} is the antecedent fuzzy set of the ith attribute, C_q denotes the consequent class, and CF_q is the weight (certainty grade) of the rule [58]. The antecedent fuzzy sets A_{qi} can either represent a linguistic value of the corresponding attribute, or a don't care condition. When the antecedent part of the fuzzy rule is provided, the consequent class and the rule weight are obtained in a heuristic manner from compatible training patterns.

The Pittsburgh encoding strategy for classification rule generation can be broadly divided into two categories. One is rule selection and the other is rule learning. In the first strategy, the aim is to select a subset of predefined rules and, in the second, the objective is to learn the rules through evolutionary algorithms. While the first strategy has been adopted in [58]–[60], the second strategy is employed in [63]–[65]. In [67], a combination of these approaches is proposed.

Ishibuchi *et al.* [58]–[60] have studied the use of MOEAs for fuzzy classification rule extraction in a series of works. In all these works, the authors used the Pittsburgh approach. Here, the authors first generate a set of rules that is constructed using some heuristic rule generator. Thereafter, an MOEA is used to select a suitable set of rules for classification. The length of a chromosome is equal to the number of rules in the rule-based system. A chromosome encodes a subset of rules represented by 1 bits of the chromosome. The rules corresponding to the 0 bits are ignored (i.e., a binary encoding is adopted). Hence, in this approach, chromosomes can encode a variable number of predefined rules. One possible disadvantage of this approach is that as the set of rules is predefined, the job of the MOEA is only to select a subset of rules, but it cannot manipulate the individual rules by changing the fuzzy membership parameters. Moreover, if the set of rules is very large, the length of the chromosome and thus the search space becomes very large too. However, the

advantage is that due to the use of binary encoding and a fixed chromosome length, standard crossover and mutation operators can be adopted.

Another alternative approach for the Pittsburgh representation is to directly encode the rules (i.e., attributes and fuzzy sets) into the chromosomes. This approach is adopted in [63], where each chromosome directly encodes rules by encoding the attributes and the corresponding fuzzy set parameters for each rule. Real-valued encoding is used here for this purpose. There are three parts in each chromosome. The first part encodes the relevant attributes of the rules' antecedents and their fuzzy sets. The second part encodes the parameters of the fuzzy sets. Finally, the third part encodes the consequent classes, as the number of attributes can vary in different rules, and also the number of rules may vary in different chromosomes; hence, the chromosome lengths are variable. To start with a good initial population, the initial population is partially filled up by fuzzy classification rules obtained through a decision tree built using the C4.5 algorithm. The remaining population is filled up by randomly replacing some parameters of the fuzzy classifier. This encoding method is very flexible but crossover and mutation operators need to handle variable-length chromosomes.

Another Pittsburgh encoding strategy encodes a set of rules and the fuzzy partitions (granularity levels) corresponding to each attribute of the rules in the chromosomes. Here, the granularity levels are predefined and are represented by a set of integers. This strategy has been adopted in [64] and [65]. In these strategies, although the rule-base is not predefined, the granularity levels of the fuzzy partitions for each attribute are predefined. Therefore, this approach is less flexible than the strategy adopted in [63].

In [67], an effort has been made to combine the benefits of the rule selection and the rule learning strategies. As the rule selection strategy deals with a simple encoding and a smaller search space but has less flexibility, and the rule learning strategy deals with higher flexibility but has a larger search space, here the authors proposed an encoding strategy that combines both approaches. In this technique, called rule and condition selection, the authors used a mixed (integer + real-value) encoding. The integer values represent the selected rules and granularity levels, whereas the real values encode the parameters of the fuzzy sets corresponding to the granularity levels. Thus, this approach provides a good tradeoff between flexibility and search space size.

In [66], a rule-based classifier is proposed based on EMOGA, where the authors employed a Michigan encoding approach, i.e., one rule in one chromosome. In this approach, the authors generated nonfuzzy categorical rules and used a discrete integer encoding to represent the rule attributes and their corresponding categorical values. However, the Michigan approach of encoding for classification is less appropriate for classification problems since they usually produce a set of good rules, which may not be a good set of rules.

3) *Objective Functions:* In different works, different sets of objective functions have been considered to be optimized. The general notion is to achieve a tradeoff between accuracy and complexity of the candidate rule set. In [58]–[60], the authors used three objective functions to be simultaneously optimized. These are the classification accuracy, number of fuzzy rules, and number of antecedent conditions. The first one is maximized whereas the last two are simultaneously minimized. Hence, the objective is to obtain the minimum number of short rules that maximize the prediction accuracy. A similar set of objectives has also been optimized in [63], where classification accuracy is just replaced by misclassification errors to transform this into a minimization problem. In [66], where a Michigan approach is employed, the objective functions chosen are predictive accuracy, comprehensibility, and interestingness. Comprehensibility is a function of the number of attributes in the rule that is to be minimized, whereas interestingness in defined using information gained to quantify how interesting is the rule. Predictive accuracy is computed using the objects covered by that rule. Ducange *et al.* [64] addressed the imbalanced class problem (binary classification) and thus maximized sensitivity and specificity while minimizing rule length. In [65], an imbalanced class problem is also considered and the two objectives are to maximize the area under the ROC curve and minimize the sum of granularity levels of all the selected attributes. Thus, as per the second objective function, a rule with n attributes of granularity m is treated the same way as a rule with p attributes of granularity q, provided $m \times n = p \times q$. In [67], the two objectives are the accuracy and the total number of attributes (conditions). Hence, it is evident from this discussion that the authors have posed the problem as the optimization of accuracy and complexity using various objective functions.

4) *Evolutionary Operators:* In [58] and [59], where the authors used a binary encoding of fixed length chromosomes, standard uniform crossover and bit-flip mutation have been employed. Narukawa *et al.* [60] introduced a new operator to remove overlapping rules. Using this, they ensured that the solutions in the initial and subsequent populations are different in the objective space. The parents for uniform crossover are chosen to be similar as this has been shown to provide better performance [69]. Moreover, the authors also proposed using the extreme solutions as one of the parents, whereas the other parent is chosen using binary tournament. In addition, bit-flip mutation with a biased mutation probability is proposed where a larger probability is assigned to the mutation from 1 to 0 than that from 0 to 1. This is done to introduce a bias to search for a lower number of rules. The authors have shown that the modified operators lead to improved solutions in comparison with the standard operators used in [58] and [59].

In [66], a hybrid crossover operator combining one-point and uniform crossover is proposed. Here, the idea is to combine the positional and distributional biases of one-point and uniform crossover. However, the authors did not perform any sort of comparison with respect to the use of any of these crossover operators in an independent way. For mutation, as the chromosome encodes categorical values, the authors perform random replacement of categorical values and random insertion/deletion of attributes.

For real-valued chromosomes in [63], simulated binary crossover (SBX) [70] and standard polynomial mutation operators are employed. SBX has been shown to be effective

in [70] for real-valued chromosomes. In [64], one-point crossover is employed and for mutation, random addition/deletion/modification of rules is performed. In [65], a mixed encoding (binary + integer) is used. Here, they adopted standard one-point crossover. In the binary part, standard bit-flip mutation is employed, whereas a random increase/decrease of integer values (representing granularity levels) is performed. Antonelli *et al.* [67] employed a mixed encoding (integer + real-value) also. For the integer part (rules and conditions), one-point crossover is employed. On the other hand, in the real-valued part (fuzzy set parameters), BLX-α (α = 0.5) crossover [71] is used. For mutation, both random replacement and complement mutation are used.

5) *Obtaining the Final Solution:* As stated earlier, an approach based on Pittsburgh encoding needs to choose a particular solution, i.e., a rule set from the set of nondominated solutions provided by the MOEA. In different works, authors have proposed using different metrics to choose the final solution from the nondominated front. In [58]–[60], the authors have used classification accuracy as the metric to choose the final solution. Note that classification accuracy has also been used as one of the objective functions in these works. The authors used classification accuracy over the complexity criteria (number of rules and rule length) as the final decision objective. In [65] and [67], the area under curve (AUC), which has been used as one of the objective functions, is adopted to select the final solution. Moreover, here, the authors also reported the solutions with minimum value of AUC, and intermediate value of AUC.

Pulkkinen and Koivisto [63] proposed a method for reducing the number of nondominated solutions from which the final solution is to be picked up. They only kept those solutions that were present in at least 50% of all generations during the execution of the MOEA. Thus, this method does not necessarily provide a single solution, but usually a smaller set of solutions than the complete nondominated set.

Intuitively, it is more practical to use an independent metric, which is not adopted for optimization, to select the final solution from the nondominated set. In [64], such an approach is presented, where the authors first generated a 3-D nondominated front (sensitivity, specificity, rule-length) using a MOEA. Then, each point in the nondominated front is projected to the ROC plane (true positive rate versus false positive rate) and the area under the ROC convex hull (AUCH) is computed. The solution providing the best value for AUCH is then chosen. This is an interesting approach but, maybe, time consuming, because of the computation of AUCH. However, this method is effective when dealing with two-class imbalanced class problems. However, the technique may not be extended in a straightforward way for multiclass problems.

B. Evolving Class Boundaries

Another promising approach for using MOEAs for classification problems is to evolve appropriate class boundaries that can successfully distinguish the various classes [72]. Usually, the boundaries between different classes are nonlinear. Any nonlinear surface can be approximated by using a number of hyperplanes. Therefore, the classification problem can be viewed as that of searching for a number of linear surfaces that can appropriately model the class boundaries, while providing a minimum number of misclassified data points. In [72], this problem has been posed as a multiobjective optimization problem, where the three objectives are: to minimize the number of misclassified patterns and the number of hyperplanes and to maximize the classification accuracy. This ensures that overfitting/overlearning is avoided, while classes of smaller size are not ignored during training. Binary chromosomes of variable length are used to encode the parameters of a variable number of hyperplanes. A constrained elitist version of NSGA-II (CEMOGA) has been used as the underlying MOO tool in this case. The final solution is selected based on an aggregation function defined by combining the objective functions. The performance of the CEMOGA classifier has been compared with that of NSGA-II and PAES-based classifiers in a similar framework. A comparison was done with other state-of-art classifiers as well. Although the approach was a novel and promising one, this paper was not extended after the first attempt reported in [72].

C. Model Building of Standard Classifiers

There exist several approaches that use MOEAs for model building or training of standard classifiers such as artificial neural networks (ANNs), SVMs, and decision trees. This section discusses some of these methods.

1) *Underlying MOEAs:* The underlying MOEAs used for training and model building of standard classifiers are as follows. NSGA-II is found to be the most commonly used approach. For example, NSGA-II is used for model building and training for SVMs [73], [74], ANNs [75], [76], and decision trees [77]. Besides this, SPEA2 has also been used in [75] and [77] for optimizing ANNs and decision tree classifiers. In [30], a single-front genetic algorithm (SFGA) was used for designing the MG-Prop algorithm that is adopted for optimizing the parameters of an MLP.

2) *Chromosome Representation:* Most papers dealing with model building of standard classifiers adopt binary encoding. In [73], the chromosomes encode a feature subset (binary vector) and also the parameters of an SVM kernel. In this case, a Gaussian kernel is used and the problem of feature selection along with SVM parameter learning is considered. Mierswa [74] proposed an MOEA, called evoSVM, in which an SVM's parameters are encoded using a vector of real numbers.

In [30], an MOEA (MG-Prop) is adopted for optimizing the topology and performance of an MLP. Here, the authors encode the topology and weights of the candidate MLP using real numbers. Cuéllar *et al.* [75] encode the topology and weights of a dynamic recurrent neural network (RNN) (the Elman RNN) classifier adopting a mixed encoding (binary + integer + real-values). The integer part encodes the number of hidden neurons, a binary mask is used to represent the active and inactive connections, and the real-valued part encodes the connection weights. Thus, this encoding provides a flexible way of representing an RNN. In [76], the network topology is considered to be fixed *a priori*. Hence, the real-valued chromosomes only encode the network weights and biases.

In [77], an MOEA has been proposed for optimizing oblique decision trees. Here, each chromosome contains real-parameter values that represent the coefficients of a hyperplane that splits the dataset to form a node of the decision tree. The decision tree is represented as a binary tree of chromosomes.

3) *Objective Functions:* The different objective functions that have been used in different studies usually represent various classification performance metrics. In [73], three objective functions are used to minimize the false positive rate, the false negative rate, and the number of support vectors. The number of support vectors is minimized to reduce the complexity of the model. In the study on SVM learning presented in [74], the authors maximized the margin between the hyperplanes while minimizing the training error. These two objective functions help control the overfitting of the SVM model.

For the optimization of an MLP, Castillo *et al.* [30] minimized both the number of false positives (Type-I error) and the number of false negatives (Type-II error). Note that, here, the authors did not include any complexity objective. However, due to the use of these two objective functions instead of using accuracy, the algorithm is capable of properly handling the imbalanced class problem. Cuéllar *et al.* [75] used both accuracy and complexity as the objective functions. Here, the authors minimized the output error while minimizing the number of hidden units and the number of connections in order to reduce the complexity of the RNN model. Lahoz and Mateo [76] divided the training set into different subsets and the classification accuracies in different subsets all of which were maximized.

Pangilinan and Janssens [77] preferred to optimize the accuracy and complexity of the candidate decision trees. The authors maximized in this case the classification accuracy, while minimizing the size of the decision tree. In general, it may be said that considering only classification performance as the objective function may provide a complex classifier. But, if both the classification performance and the complexity of the model are taken into account, the algorithm may produce solutions that provide a good tradeoff between accuracy and complexity of the models.

4) *Evolutionary Operators:* A variety of crossover and mutation operators have been adopted in different studies. Suttorp and Igel [73] did not mention explicitly what type of crossover and mutation operators were adopted. However, as they used NSGA-II for optimizing, it is expected that they adopted its standard operators for binary encoding. Castillo *et al.* [30] used multipoint crossover for real-valued chromosomes. The mutation operators are of two types. The first consists of random modifications of connection weights and the second one consists of a random addition/deletion of hidden layer neurons.

Cuéllar *et al.* [75] used Wright's heuristic crossover [78] for real-valued chromosomes. In this encoding technique, the parent networks may have different sets of connections. Therefore, if a network weight is not contained in both parent networks, it is extended directly to the largest child. Wright's heuristic crossover has been shown to perform better than standard crossover operations in [78]. The mutation is performed by random modifications of the weights (real-values) and connections (binary values).

In [74], where the chromosomes encode α_i (Lagrange's multipliers) parameters of an SVM, the authors employed a hybrid mutation approach. In this mutation, they check for each α_i with probability $1/n$ (n=number of α_is) if the value should be mutated at all. If $\alpha_i > 0$, then its new value is set to 0. If $\alpha_i = 0$, then it is set to a random value between 0 and C (the SVM generalization parameter). However, the authors did not demonstrate the advantages of this mutation operator over a standard mutation operation. The crossover operator used in this paper is not explicitly mentioned.

In [76], a BLX-α crossover with $\alpha = 0.5$ is employed. Here, the authors used a nonuniform mutation operator for real-valued chromosomes [5]. This means that the mutation rate is decreased from one generation to the next one. In this mutation, the probability that the amount of mutation will go to 0 at the next generation is increased. This favors diversity at the early stages of the evolutionary process and increases the selection pressure toward the end. In [77], this mutation scheme is also adopted. In this paper, the authors adopted standard arithmetic crossover.

5) *Obtaining the Final Solution:* As each chromosome of the nondominated set encodes a possible classifier model, it is necessary to obtain one final solution from this set. However, in [73], [75] and [77], the authors did not address this issue. They reported results based only on the nondominated set produced. Among the other works, two main approaches have been noticed. One is to use the nondominated classifiers as an ensemble classifier system, and another is to use some metric to choose one particular solution from the set. In [30] and [76] the first approach was taken, i.e., designing an ensemble classifier system using the nondominated solutions. In [30], three different ensemble techniques were studied. The first one is a simple majority voting among the classifiers. The second method predicts the class considering the largest activation among all the outputs of the networks. The third approach computes the average outputs for all the networks. Notably, the third approach does not provide a particular classification result, but only the average accuracy of all the nondominated classifiers. In [76], two different ensemble methods are compared (majority voting and simple averaging).

In [74], instead of an ensemble among the nondominated solutions, the authors chose one of them as the final classifier based on their performance on a hold-out validation set. A hold-out validation set is kept aside from the training set and is not used during the evolutionary process. Finally, the classification accuracy on this hold-out set is computed for each nondominated solution. The solution giving the maximum accuracy value for this hold-out set is finally selected.

Due to the nature of multiobjective optimization problems, each of the generated nondominated solutions (classifiers in this case) shares some information about the input training set. Therefore, it is more intuitive to combine the information contained in these classifiers by means of some ensemble. However, it would be interesting to compare the performance of individual nondominated classifiers with the ensemble results on some unknown test dataset to judge their robustness.

TABLE II
COMPARISON OF DIFFERENT MOEAS FOR CLASSIFICATION

Algorithm	Underlying MOO tool	Type	Encoding	Objective functions	Evolutionary operators	Final solution from non-dominated front
Ishibuchi and Nojima [47], [48], 2005	NSGA-II	Fuzzy rule-based	Binary (Pittsburgh)	Accuracy, number of fuzzy rules, number of antecedent conditions	Uniform crossover, bit-flip mutation	Based on accuracy
Ishibuchi and Nojima [49], 2005	NSGA-II	Fuzzy rule-based	Binary (Piitsburgh)	Accuracy, number of fuzzy rules, number of antecedent conditions	Overlapping rule removal, uniform crossover between parents, bit-flip mutation (with biased probabilities)	Based on accuracy
Dehuri et. al. [50], 2008 (EMOGA)	EMOGA	Non-fuzzy rule-based	Discrete (Michigan)	Predictive accuracy, comprehensibility, interestingness	Hybrid (one-point + uniform) crossover, random value replacement mutation, insert/delete condition	None
Pulkkinen and Koivisto [51], 2008	NSGA-II	Fuzzy rule-based	Real-valued (Pittsburgh)	Misclassification eroor, number of fuzzy rules, total rule length	Simulated binary crossover, polynomial mutation	Solutions that were present in more than 50% of all generations
Ducange et. al. [52], 2010	NSGA-II	Fuzzy rule-based	Integer (Pittsburgh)	Sensitivity, specificity, total rule length	One-point crossover, mutations by adding rules, deleting rules, changing rule conditions	Based on Area Under Convex Hull (AUCH)
Villar et. al. [53], 2011 (MGA-FS-GL)	NSGA-II	Fuzzy rule-based	Mixed (binary + integer) (Pittsburgh)	Area Under Curve (AUC), sum of granularity levels (complexity)	One-point crossover, bit-flip mutation (binary part), increase/decrease (integer part)	Based on AUC, complexity, intermediate value
Antonelli et. al. [54], 2012 (PAES-RCS)	M-PAES	Fuzzy rule-based	Mixed (integer + real-valued) (Pittsburgh)	Accuracy, number of conditions (complexity)	One-point and BLX-α crossover, random replacement and complement mutation	Based on AUC, complexity, intermediate value
Bandyopadhyay et. al. [55], 2004 (CEMOGA-classifier)	CEMOGA	Class boundary	Binary (variable length) (Hyperplane parameters)	Number of misclassified points, number of hyperplane, classification accuracy	One-point crossover, bit-change mutation	Based on aggregation function defined by combinations of objectives
Suttorp and Igel [56], 2006	NSGA-II	Optimizing standard classifier (model building)	Binary (SVM parameters)	False positive rate, false negative rate, number of support vectors	Standard NSGA-II operators	None
Castillo et. al. [30], 2006 (MG-Prop)	SFGA	Optimizing standard classifier (model building)	Real-valued (Topology and weights of MLP)	number of false positives, number of false negatives	Multi-point crossover, weight modification mutation, addition/deletion hidden neurons	Ensemble of non-dominated classifiers
Cuéllar et. al. [57], 2007	NSGA-II, SPEA2	Optimizing standard classifier (model building)	Mixed (binary + integer + real) (Topology and weights of RNN)	Output error, number of hidden units, number of connections	Wright's heuristic crossover, structure and weight modification mutation	None
Mierswa [58], 2007	NSGA-II	Optimizing standard classifier (model building)	Real-valued (SVM parameters)	Hyperplane margin, training error	Crossover not mentioned, hybrid mutation	Based on error in hold-out set
Lahoz and Mateo [59], 2008	NSGA-II	Optimizing standard classifier (model building)	Real-valued (weights of ANN)	Classification errors in different subsets of training patterns	BLX-0.5 crossover, non-uniform mutation	Ensemble of non-dominated classifiers
Pangilinan et. al. [60], 2011	NSGA-II, SPEA2	Optimizing standard classifier (model building)	Real-valued (Coefficients of hyperplanes corresponding to decision tree nodes	Classification accuracy, decision tree size	Arithmetic crossover, non-uniform mutation	None

To the authors' best knowledge, such a comparative study is not available yet in the literature.

D. Relative Comparison and Applications

To facilitate the comparative study of the proposed MOEAs for classification problems, we have summarized all the methods discussed in this section in Table II. The methods have been categorized as rule-based approaches, hyperplane optimization approaches, and model building of standard classifiers. Under each category, we have arranged the methods in increasing order of publication times to illustrate how the methods have evolved over time. As with the feature selection case, here, we have also characterized the algorithms with respect to the underlying MOEA, the encoding policy, the objective functions, the evolutionary operators, and the technique for choosing the final solution from the nondominated front. The rule-based classification approaches are classified into fuzzy and nonfuzzy, and all of the methods in this category address the fuzzy classification rule generation except for one. These fuzzy rule-based classifiers use a Pittsburgh encoding strategy and a variety of objective functions and evolutionary operators. The methods for selecting the final solution from the non-dominated front also vary in different algorithms as discussed before. There is only one approach in the second category, i.e., hyperplane optimization. This approach was promising but did not mature after the first attempt. In the third category, as can be seen from the table, MOEAs have been used for model building of different standard classifiers such as SVM, ANN, and decision trees. In all the categories, NSGA-II has been found to be the most commonly adopted MOEA, but some authors have also reported the use of other algorithms such as SPEA2, PAES, EMOGA, CEMOGA, and SFGA. Also, there are no comparative studies of MOEA-based classification methods in a systematic way.

MOEA-based classifiers have been applied in various real-life application domains. Suttorp and Igel [73] used an MOEA-optimized SVM for pedestrian detection in infrared images for driver assistance systems. This real-world task has strict real-time constraints that require highly optimized classifiers and a reasonable tradeoff between sensitivity and specificity. The authors demonstrated the effectiveness of MOEAs in optimizing SVMs for this purpose. Castillo *et al.* [30] generated an MOEA-based classifier ensemble for breast cancer classification and bankruptcy prediction problems. In other works, the researchers have applied their proposed techniques in classifying different real-life datasets available in the University of California at Irvine machine learning repository (http://archive.ics.uci.edu/ml/). There are, however, several possible applications that are not available in the literature yet. For example, the classifying samples in microarray

gene expression data into different classes such as benign and tumor, and classifying satellite and medical images.

VI. Conclusion

As most data mining tasks need the optimization of model parameters along with multiple performance criteria, multiobjective optimization is the natural choice for dealing with such tasks. Over the years, MOEAs have become very popular within the data mining community because of their flexibility in representing a data mining problem with relative ease. Therefore, over the past decade, a number of MOEAs have been proposed for solving a variety of data mining problems. In Part I of this two-part paper, we have introduced some basic concepts related to multiobjective optimization, as well as the fundamentals of data mining followed by a description of the main motivations for multiobjective data mining. Then, different MOEAs that have been used to solve two major data mining tasks, namely, feature selection and classification have been discussed with a special focus on issues such as chromosome encoding, evolutionary operators, the type of objective function used for optimization, and selection of final solution from the nondominated set. In Part II [1], MOEAs employed for other data mining tasks, such as clustering, association rule mining, ensemble learning, biclustering, are reviewed followed by a discussion on the future scope of research in this field.

References

[1] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, and C. A. Coello Coello, "A survey of multiobjective evolutionary algorithms for data mining: Part II," *IEEE Trans. Evolut. Comput.*, to be published.

[2] U. Maulik, S. Bandyopadhyay, and A. Mukhopadhyay, *Multiobjective Genetic Algorithms for Clustering—Applications in Data Mining and Bioinformatics*. Berlin, Germany: Springer, 2011.

[3] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann, 2000.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. New York, NY, USA: Addison-Wesley, 1989.

[5] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. London, U.K.: Wiley, 2001.

[6] C. A. Coello Coello, G. B. Lamont, and D. A. van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems* (Genetic and Evolutionary Computation), 2nd ed. Berlin/Heidelberg, Germany: Springer, 2007.

[7] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. 1st Int. Conf. Genet. Algorithms Their Appl.*, 1985, pp. 93–100.

[8] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in *Proc. 5th Int. Conf. Genet. Algorithms*, 1993, pp. 416–423.

[9] J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched Pareto genetic algorithm," Univ. Illinois at Urbana-Champaign, Urbana, IL, USA, Tech. Rep. IlliGAl Rep. 93005, 1993.

[10] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, 1994.

[11] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[12] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Proc. EUROGEN*, 2001, pp. 95–100.

[13] J. D. Knowles and D. W. Corne, "The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation," in *Proc. IEEE Cong. Evol. Comput.*, 1999, pp. 98–105.

[14] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto envelope-based selection algorithm for multiobjective optimization," in *Proc. Conf. PPSN-VI*, 2000, pp. 839–848.

[15] D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates, "PESA-II: Region-based selection in evolutionary multiobjective optimization," in *Proc. GECCO*, 2001, pp. 283–290.

[16] K. Deb, A. Pratap, S. Agrawal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[17] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. PPSN VIII*, vol. 3242. Sep. 2004, pp. 832–842.

[18] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *Eur. J. Operat. Res.*, vol. 181, no. 3, pp. 1653–1669, Sep. 2007.

[19] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications," Ph.D. dissertation, Comput. Eng. Networks Lab., (TIK), Swiss Federal Instit. Technol. (ETH), Zurich, Switzerland, Nov. 1999.

[20] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds., *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA, USA: MIT Press, 1996.

[21] S. Bandyopadhyay, U. Maulik, L. B. Holder, and D. J. Cook, *Advanced Methods for Knowledge Discovery From Complex Data* (Advanced Information and Knowledge Processing). London, U.K.: Springer-Verlag, 2005.

[22] S. Bandyopadhyay and S. K. Pal, *Classification and Learning Using Genetic Algorithms: Applications in Bioinformatics and Web Intelligence* (Natural Computing Series). Berlin, Germany: Springer, 2007.

[23] V. Vapnik, *Statistical Learning Theory*. New York, NY, USA: Wiley, 1998.

[24] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1988.

[25] U. Maulik and S. Bandyopadhyay, "Genetic algorithm based clustering technique," *Pattern Recognit.*, vol. 33, no. 9, pp. 1455–1465, Sep. 2000.

[26] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 1993, pp. 207–216.

[27] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proc. 20th VLDB*, 1994, pp. 487–499.

[28] P.-N. Tan, V. Kumar, and J. Srivastava, "Selecting the right interestingness measure for association patterns," in *Proc. 8th KDD*, 2002, pp. 32–41.

[29] U. Maulik, A. Mukhopadhyay, and S. Bandyopadhyay, "Combining Pareto-optimal clusters using supervised learning for identifying co-expressed genes," *BMC Bioinform.*, vol. 10, no. 27, doi: 10.1186/1471-2105-10-27, 2009,

[30] P. A. Castillo, M. G. Arenas, J. J. Merelo, V. M. Rivas, and G. Romero, "Multiobjective optimization of ensembles of multilayer perceptrons for pattern classification," in *Proc. 9th PPSN*, 2006, pp. 453–462.

[31] A. Ghosh and B. Nath, "Multi-objective rule mining using genetic algorithms," *Inf. Sci.*, vol. 163, nos. 1–3, pp. 123–133, Jun. 2004.

[32] S. Mitra and H. Banka, "Multiobjective evolutionary biclustering of gene expression data," *Patt. Recogn.*, vol. 39, no. 12, pp. 2464–2477, 2006.

[33] L. Tseng and S. Yang, "Genetic algorithms for clustering, feature selection, and classification," in *Proc. IEEE Int. Conf. Neural Netw.*, 1997, pp. 1612–1616.

[34] M. Karzynski, A. Mateos, J. Herrero, and J. Dopazo, "Using a genetic algorithm and a perceptron for feature selection and supervised class learning in DNA microarray data," *Artif. Intell. Rev.*, vol. 20, nos. 1–2, pp. 39–51, 2003.

[35] C. Emmanouilidis, A. Hunter, and J. MacIntyre, "A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator," in *Proc. IEEE CEC*, 2000, pp. 309–316.

[36] C. Emmanouilidis, A. Hunter, J. Macintype, and C. Cox, "A multiobjective genetic algorithm approach to feature selection in neural and fuzzy modeling," *Evol. Optimiz.*, vol. 3, no. 1, pp. 1–26, 2001.

[37] C. Emmanouilidis, *Evolutionary Multi-Objective Feature Selection and ROC Analysis With Application to Industrial Machinery Fault Diagnosis* (Evolutionary Methods for Optimization, Design and Control). Barcelona, Spain: CIMNE, 2002.

[38] L. E. S. de Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 17, no. 6, pp. 903–929, 2003.

[39] M. Morita, R. Sabourin, F. Bortolozzi, and C. Suen, "Unsupervised feature selection using multi-objective genetic algorithm for handwritten word recognition," in *Proc. ICDAR*, Aug. 2003, pp. 666–670.
[40] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen, "Feature selection for ensembles: A hierarchical multi-objective genetic algorithm approach," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, Aug. 2003, pp. 676–680.
[41] L. S. Oliveira, M. Morita, and R. Sabourin, "Feature selection for ensembles using the multi-objective optimization approach," in *Multi-objective Machine Learning*, Y. Jin, Ed. Berlin, Germany: Springer, *Stud. Comput. Intell.*, vol. 16, pp. 49–74, 2006.
[42] C.-M. Wang and Y.-F. Huang, "Evolutionary-based feature selection approaches with new criteria for data mining: A case study of credit approval data," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5900–5908, Apr. 2009.
[43] M. Venkatadri and K. Srinivasa Rao, "A multiobjective genetic algorithm for feature selection in data mining," *Int. J. Comput. Sci. Inform. Technol.*, vol. 1, no. 5, pp. 443–448, 2010.
[44] I. Mierswa and M. Wurst, "Information preserving multi-objective feature selection for unsupervised learning," in *Proc. GECCO*, 2006, vol. 2. Jul. 2006, pp. 1545–1552.
[45] A. Gaspar-Cunha, "Feature selection using multi-objective evolutionary algorithms: Application to cardiac SPECT diagnosis," in *IWPACBB* (Advances in Soft Computing, vol. 74), M. P. Rocha, F. F. Riverola, H. Shatkay, and J. M. Corchado, Eds. Berlin, Germany: Springer, 2010, pp. 85–92.
[46] A. Gaspar-Cunha, "RPSGAe-reduced Pareto set genetic algorithm: Application to polymer extrusion," in *Metaheuristics for Multiobjective Optimisation*. Berlin, Germany: Springer, 2004, pp. 221–249.
[47] K. Kim, R. B. McKay, and B.-R. Moon, "Multiobjective evolutionary algorithms for dynamic social network clustering," in *Proc. 12th GECCO*, 2010, pp. 1179–1186.
[48] Y.-S. Kim, W. N. Street, and F. Menczer, "An evolutionary multi-objective local selection algorithm for customer targeting," in *Proc. CEC*, 2001, pp. 759–766.
[49] J. Handl and J. D. Knowles, "Feature subset selection in unsupervised learning via multiobjective optimization," *Int. J. Comput. Intell. Res.*, vol. 2, no. 3, pp. 217–238, 2006.
[50] J. Liu and H. Iba, "Selecting informative genes using a multiobjective evolutionary algorithm," in *Proc. CEC*, 2002, pp. 297–302.
[51] Y. Kim, N. W. Street, and F. Menczer, "Evolutionary model selection in unsupervised learning," *Intell. Data Anal.*, vol. 6, no. 6, pp. 531–556, Dec. 2002.
[52] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, *et al.*, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, Oct. 1999.
[53] D. K. Slonim, P. Tamayo, J. P. Mesirov, T. R. Golub, and E. S. Lander, "Class prediction and discovery using gene expression data," in *Proc. 4th RECOMB*, 2000, pp. 263–272.
[54] G. L. Pappa, A. A. Freitas, and C. A. A. Kaestner, "A multiobjective genetic algorithm for attribute selection," in *Proc. 4th RASC*, pp. 116–121, Dec. 2002.
[55] A. Gaspar-Cunha, F. Mendes, J. Duarte, A. Vieira, B. Ribeiro, A. Ribeiro, *et al.*, "Feature selection for bankruptcy prediction: A multi-objective optimization approach," *Int. J. Natural Comput. Res.*, vol. 1, no. 2, pp. 71–91, 2010.
[56] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 12, pp. 1650–1654, Dec. 2002.
[57] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 1, no. 2, pp. 224–227, Apr. 1979.
[58] H. Ishibuchi and Y. Nojima, "Multiobjective formulations of fuzzy rule-based classification system design," in *Proc. EUSFLAT Conf.*, 2005, pp. 285–290.
[59] H. Ishibuchi and Y. Nojima, "Comparison between fuzzy and interval partitions in evolutionary multiobjective design of rule-based classification systems," in *Proc. 14th FUZZ-IEEE*, May 2005, pp. 430–435.
[60] K. Narukawa, Y. Nojima, and H. Ishibuchi, "Modification of evolutionary multiobjective optimization algorithms for multiobjective design of fuzzy rule-based classification systems," in *Proc. 14th Fuzz-IEEE*, May 2005, pp. 809–814.
[61] R. Alcala, Y. Nojima, F. Herrera, and H. Ishibuchi, "Generating single granularity-based fuzzy classification rules for multiobjective genetic fuzzy rule selection," in *Proc. FUZZ-IEEE*, Aug. 2009, pp. 1718–1723.
[62] R. Alcalá, Y. Nojima, F. Herrera, and H. Ishibuchi, "Multiobjective genetic fuzzy rule selection of single granularity-based fuzzy classification rules and its interaction with the lateral tuning of membership functions," *Soft Comput.*, vol. 15, no. 12, pp. 2303–2318, 2011.
[63] P. Pulkkinen and H. Koivisto, "Fuzzy classifier identification using decision tree and multiobjective evolutionary algorithms," *Int. J. Approx. Reasoning*, vol. 48, no. 2, pp. 526–543, Jun. 2008.
[64] P. Ducange, B. Lazzerini, and F. Marcelloni, "Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets," *Soft Comput.*, vol. 14, no. 7, pp. 713–728, May 2010.
[65] P. Villar, A. Fernandez, and F. Herrera, "Studying the behavior of a multiobjective genetic algorithm to design fuzzy rule-based classification systems for imbalanced data-sets," in *Proc. FUZZ-IEEE*, Jun. 2011, pp. 1239–1246.
[66] S. Dehuri, S. Patnaik, A. Ghosh, and R. Mall, "Application of elitist multi-objective genetic algorithm for classification rule generation," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 477–487, Jan. 2008.
[67] M. Antonelli, P. Ducange, and F. Marcelloni, "Multi-objective evolutionary rule and condition selection for designing fuzzy rule-based classifiers," in *Proc. FUZZ-IEEE*, Jun. 2012, pp. 1–7.
[68] S. Srinivasan and S. Ramakrishnan, "Evolutionary multiobjective optimization for rule mining: A review," *Artif. Intell. Rev.*, vol. 36, no. 3, pp. 205–248, Oct. 2011.
[69] H. Ishibuchi and Y. Shibata, "An empirical study on the effect of mating restriction on the search ability of emo algorithms," in *Proc. 2nd EMO*, 2003, pp. 433–447.
[70] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.
[71] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," in *Foundation of Genetic Algorithms 2*, D. L. Whitley, Ed. San Mateo, CA, USA: Morgan Kaufmann, 1993, pp. 187–202.
[72] S. Bandyopadhyay, S. Pal, and B. Aruna, "Multiobjective GAs, quantitative indices, and pattern classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 5, pp. 2088–2099, Oct. 2004.
[73] T. Suttorp and C. Igel, *Multi-Objective Optimization of Support Vector Machines* (Multi-Objective Machine Learning Studies in Computational Intelligence, vol. 16). Berlin, Germany: Springer-Verlag, 2006, pp. 199–220.
[74] I. Mierswa, "Controlling overfitting with multi-objective support vector machines," in *Proc. 9th GECCO*, 2007, pp. 1830–1837.
[75] M. P. Cuéllar, M. Delgado, and M. C. Pegalajar, "Topology optimization and training of recurrent neural networks with Pareto-based multi-objective algorithms: A experimental study," in *Proc. 9th IWANN*, 2007, pp. 359–366.
[76] D. Lahoz and P. Mateo, "Neural network ensembles for classification problems using multiobjective genetic algorithms," in *Proc. 18th ICANN*, 2008, pp. 443–451.
[77] J. M. Pangilinan and G. K. Janssens, "Pareto-optimality of oblique decision trees from evolutionary algorithms," *J. Global Optimization*, vol. 51, no. 2, pp. 301–311, Oct. 2011.
[78] A. H. Wright, "Genetic algorithms for real parameter optimization," in *Foundations of Genetic Algorithms*, G. J. Rawlins, Ed. San Mateo, CA, USA: Morgan Kaufmann, 1991, pp. 205–218.

Anirban Mukhopadhyay (SM'11) received the B.E. degree in computer science and engineering from the National Institute of Technology, Durgapur, India, in 2002, and the M.E. degree in computer science and engineering, and the Ph.D. degree in computer science from Jadavpur University, Kolkata, India, in 2004 and 2009, respectively.

He is currently an Associate Professor and the Head of the Department of Computer Science and Engineering, University of Kalyani, Kalyani, India. From 2009 to 2010, he was a Post-Doctoral Fellow with the University of Heidelberg and German Cancer Research Center, Heidelberg, Germany. He was a Visiting Professor at the I3S Laboratory, University of Nice Sophia-Antipolis, Nice, France, in 2011 and a Visiting Scientist with the University of Goettingen, Goettingen, Germany, with a DAAD scholarship, in 2013. He has co-authored one book and approximately 100 research papers in various international journals and conferences. His current research interests include soft and evolutionary computing, data mining, multiobjective optimization, pattern recognition, bioinformatics, and optical networks.

In 2004, Dr. Mukhopadhyay received the University Gold Medal and the Amitava Dey Memorial Gold Medal from Jadavpur University.

Ujjwal Maulik (SM'05) See page 3 of this issue for Dr. Maulik's biography.

Sanghamitra Bandyopadhyay (SM'05) See page 3 of this issue for Dr. Bundyopadhyay's biography.

Carlos Artemio Coello Coello (F'11) See page 3 of this issue for Dr. Coello Coello's biography