Open Mathematics **Open Access**

Research Article

Laura Calvet, Jésica de Armas, David Masip, and Angel A. Juan*

Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs

DOI 10.1515/math-2017-0029
Received September 7, 2016; accepted January 5, 2017.

Abstract: This paper reviews the existing literature on the combination of metaheuristics with machine learning methods and then introduces the concept of learnheuristics, a novel type of hybrid algorithms. Learnheuristics can be used to solve combinatorial optimization problems with dynamic inputs (COPDIs). In these COPDIs, the problem inputs (elements either located in the objective function or in the constraints set) are not fixed in advance as usual. On the contrary, they might vary in a predictable (non-random) way as the solution is partially built according to some heuristic-based iterative process. For instance, a consumer's willingness to spend on a specific product might change as the availability of this product decreases and its price rises. Thus, these inputs might take different values depending on the current solution configuration. These variations in the inputs might require from a coordination between the learning mechanism and the metaheuristic algorithm: at each iteration, the learning method updates the inputs model used by the metaheuristic.

Keywords: Hybrid algorithms, Combinatorial optimization, Metaheuristics, Machine learning, Dynamic inputs

MSC:

1 Introduction

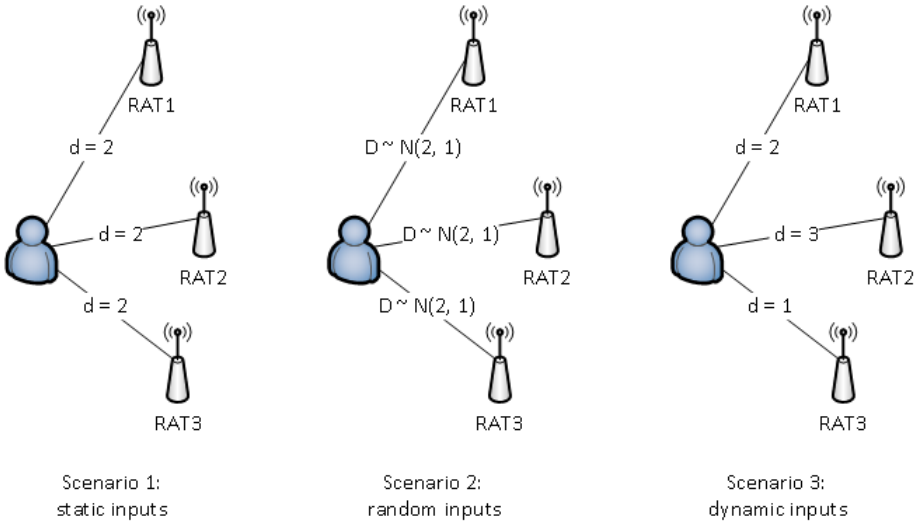
Operations Research (OR) is a well-established field with a huge and active research community. One of its main goals is to support decision-making processes in complex scenarios, i.e., providing optimal (or near-optimal) solutions to combinatorial optimization problems (COPs) defined by a given objective function and a set of realistic constraints. The number of applications is immense, e.g.: transportation and logistics, finance, production, telecommunication systems, etc. A noticeable part of the efforts developed by the OR community has focused on developing exact methods to find optimal solutions to a wide range of COPs. When dealing with NP-hard COPs, this usually requires simplifying somewhat the model and/or addressing only small- and medium-sized instances to avoid incurring in prohibitive computing times. Another noticeable part of the efforts has been invested in developing heuristic and metaheuristic approaches that cannot guarantee optimality of the provided solutions but are usually more powerful in terms of the size of the instances they can solve in reasonable computing times [1]. Additionally, these approximated methods are quite flexible, which makes them suitable for tackling more realistic and rich models. While heuristics are simple and fast procedures based on the specific COP being addressed, metaheuristics are general templates that can be easily adapted to a huge variety of COPs.

Laura Calvet: Dept. of Computer Science – IN3, Open University of Catalonia, Castelldefels, Spain, E-mail: lcalvetl@uoc.edu
Jésica de Armas: Dept. of Computer Science – IN3, Open University of Catalonia, Castelldefels, Spain, E-mail: jde_armasa@uoc.edu
David Masip: Dept. of Computer Science – IN3, Open University of Catalonia, Castelldefels, Spain, E-mail: dmasipr@uoc.edu
***Corresponding Author: Angel A. Juan:** Dept. of Computer Science – IN3, Open University of Catalonia, Castelldefels, Spain, E-mail: ajuanp@uoc.edu

The OR community shows a growing interest in coping with increasingly challenging COPs, such as stochastic COPs (in which some of the problem inputs are random variables) and dynamic COPs (in which some of the problem inputs evolve over time). This might be due to several factors, including: *(i)* the rich characteristics of real-life problems frequently faced by modern companies in sectors such as logistics and transportation [2]; *(ii)* the technological development; *(iii)* the availability of vast amounts of Internet-based data; and *(iv)* a shift to a more data-driven culture. During the last years, hybrid approaches have been extensively employed due to their success when dealing with realistic problems, among others: those combining different metaheuristics [3], matheuristics (i.e., metaheuristics combined with mathematical programming) [4], and simheuristics (i.e., metaheuristics combined with simulation) [5].

The hybridization of metaheuristics with machine learning techniques is an emerging research field in the OR community. In this context, the main contributions of this paper are: *(i)* providing a survey on the existing works combining metaheuristics with machine learning techniques, as well as a classification of the most relevant ones; and *(ii)* proposing a novel 'learnheuristic' framework, combining a heuristic-based constructive procedure with machine learning, to deal with a special kind of COPs with dynamic inputs (COPDIs). In these COPDIs, the inputs are deterministic (i.e., non-stochastic) but, instead of being fixed in advance, they vary according to the structure of the solution (i.e., they change as the solution is being constructed following a heuristic-based iterative process). In this sense, these COPDIs represent an extension of the classical deterministic COPs in which all inputs are given in advance and are immutable. An example of such a COPDI is given next for illustrative purposes. Suppose there is a set of heterogeneous radio access technologies (RATs) that provide pay-per-use services to a group of users. Each user has to be assigned to just one RAT, and each RAT can serve only a limited number of users. Being a pay-per-use service, the goal here is to maximize the total benefit, which depends on the customers' demands. Several scenarios may be described based on the nature of the customers' demands (Figure 1): *(i)* they are deterministic, static (do not change over time), and can be computed or accurately estimated; *(ii)* they contain some degree of uncertainty but can be modeled as random variables or using fuzzy techniques; and *(iii)* they are dynamic in the sense that they depend on the solution characteristics (e.g., the number of users connected to the same RAT, which has an effect on the service quality and, therefore, on the customers' demands of that service).

Fig. 1. Different scenarios according to the nature of the inputs.



While the first case corresponds to a classical deterministic COP, the second case introduces a level of uncertainty that usually requires the use of stochastic programming, simulation-optimization, or fuzzy methods. In this paper we focus on the third case, and propose the use of learnheuristic algorithms, in which the learning mechanism updates the input values as the solution is iteratively constructed using the heuristic logic [6]. Notice, however, that not all the metaheuristics rely on constructive procedures to generate new solutions. Thus, for instance, evolutionary algorithms

or scatter search algorithms typically generate new solutions by simply combining already existing solutions, which might have been generated at random.

The rest of the paper is structured as follows: Section 2 provides a brief introduction to metaheuristics and machine learning, and proposes a classification of hybrid works combining both methodologies. Section 3 presents an overview of works in which machine learning techniques have been used to enhance the performance of metaheuristics, while Section 4 reviews publications in which metaheuristics have been used to improve machine learning methods. Section 5 provides a formal description of the COPDIs we aim to solve and explains the main ideas behind our learnheuristics solving framework. Section 6 discusses potential applications of learnheuristics in different fields. Section 7 provides a numerical experiment that illustrates the use of learnheuristics in a vehicle routing problem with dynamic demands. Finally, Section 8 summarizes the main conclusions and identifies some future research lines.

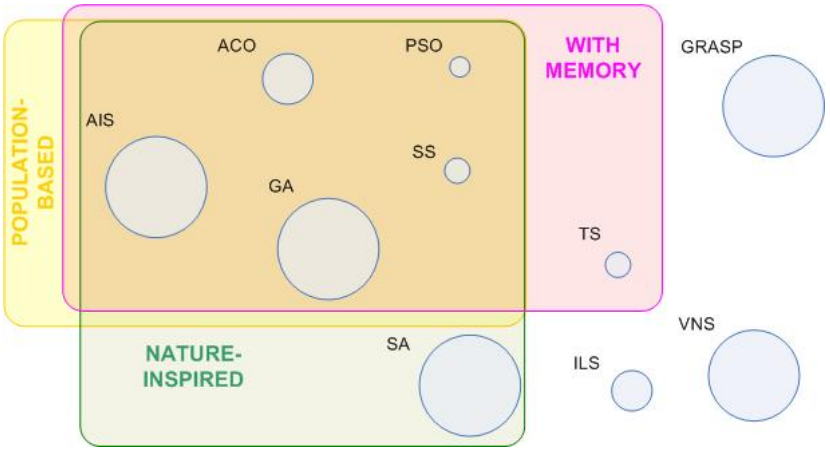
2 Metaheuristics and machine learning

2.1 Definitions and evolution of the number of works

Metaheuristics represent a heterogeneous family of algorithms designed to solve a high number of complex COPs without having to deeply adapt them to each problem. They do not guarantee optimal solutions, but may provide near-optimal ones in a reasonable amount of computing time. A number of them are nature-inspired, include stochastic components, and have several parameters that must be fine-tuned and may interact [7]. Figure 2 includes some of the most popular metaheuristics (first works are cited): ant colony optimization (ACO) [8], artificial immune systems (AIS) [9], genetic algorithms (GA) [10], greedy randomized adaptive search procedure (GRASP) [11], iterated local search (ILS) [12], particle swarm optimization (PSO) [13], scatter search (SS) [14], simulated annealing (SA) [15], tabu search (TS) [16] and variable neighborhood search (VNS) [17]. They are grouped according to the following criteria: *(i)* single-solution versus population-based metaheuristics (SMs and PMs, respectively); *(ii)* whether they use memory; and *(iii)* whether they are nature-inspired. The success of the first implementations of metaheuristics aroused the interest of journals in new versions of these methods, which increased the number of authors exploring this topic. Unfortunately, some publications add only marginal contributions to the already existing frameworks [18]. As stated in [19], the effectiveness of a given metaheuristic depends upon its ability to adapt to a particular instance problem, avoid entrapment at local optima, and exploit the structure of the problem. In addition, the authors highlight the potential benefit of restart procedures, controlled randomization, efficient data structures, and pre-processing. A few fields where they are commonly applied are: logistics and transportation, telecommunications, production and scheduling, bioinformatics, finance, smart cities, cryptology, and nutrition, among many others. The reader interested in a complete review of metaheuristics is referred to [20].

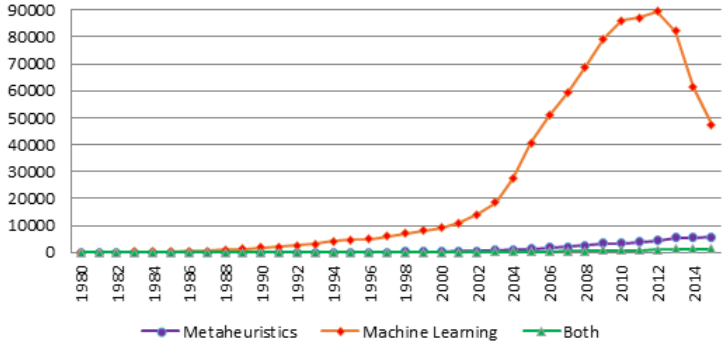
Machine learning is a subfield of computer science and artificial intelligence that encompasses a number of algorithms which learn from a dataset composed of examples or observations and are capable of making predictions [21, 22]. There are three styles of learning: supervised, unsupervised, and semi-supervised. The first relies on a set of procedures for function approximation. Based on a database of labelled samples, the goal is to predict a response variable (or output) from the explanatory variables (or inputs). Supervised methods are used in the following tasks: regression, classification, dimension reduction, time-series prediction, and reinforcement learning. In contrast, unsupervised learning does not include any response variable, and attempts to find compact descriptions of the data. The main tasks are: anomaly detection, dimension reduction, time-series modeling, and latent variable models. Finally, semi-supervised learning is similar to supervised learning but, in this case, not all the examples have associated an output value. Semi-supervised methods are very useful in problems where large amounts of unlabelled samples are available, and only a few of them can be manually labelled. Typical examples are visual object recognition, where milions of untagged images are publicly available, or natural language processing. The most popular applications of machine learning techniques include search engines, robotics, computer vision, finance, bioinformatics, finance, insurance, etc.

Fig. 2. Main metaheuristics grouped by different criteria. Circles' size is proportional to the number of Google Scholar indexed articles, from 2006 to 2015, that include the complete name of the specific metaheuristic and "metaheuristics" or "heuristics" in the article (March 15, 2016).



According to data from Google Scholar, both fields may be considered young (Figure 3). Although the use of machine learning techniques is much more extended, metaheuristics are more employed in the context of COPs. An example of machine learning applied to solve these problems is the work developed in neural networks for solving COPs, mainly in vehicle routing problems [23, 24].

Fig. 3. Evolution of the number of works in Google Scholar (March 15, 2016). The number of works from the fields of metaheuristics and statistics or data mining were 1880 and 785 in 2015, respectively.



2.2 Reviews on the combination of metaheuristics and machine learning

The existing literature analyzing the hybridization of metaheuristics and machine learning may be mainly divided into two groups: works where machine learning is employed to enhance metaheuristics, and those in which metaheuristics are used to improve the performance of machine learning techniques.

Regarding the first group, there are several works providing overviews from different points of view. For instance, the emergence of hybrid metaheuristics is studied in [3], which includes the combination of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; or *(iv)* machine learning. The author proposes a general two-level classification. In this sense, it is possible to distinguish between low-level hybridizations, in which a given internal function of a metaheuristic is replaced by another optimization method, and high-level hybridizations, where the different optimization methods are self-contained. In a second phase, these

algorithms can be further classified into relay or teamwork hybridization. While in the former the techniques are applied one after another (each using the output of the previous as its input), the latter represents cooperative optimization models. In [25], authors describe applications of data mining techniques to help metaheuristics. Finally, a survey on the integration of machine learning in evolutionary computation can be found in [26]. The work presented in [27] gathers the synergies between OR and data mining, remarking the growing importance of multi-objective approaches. The authors highlight three benefits of employing data mining in OR: *(i)* increasing the quality of the results of OR algorithms; *(ii)* speeding up OR algorithms; and *(iii)* selecting an OR algorithm based on instance properties.

Our work builds on the classification in [25] and extends it by proposing more categories and analyzing a higher number of works. In our view, the classification in [25] is more suitable for works where machine learning is employed to enhance metaheuristics than the one presented in [3], which was designed to be more general and to include other hybridizations. In particular, works are classified into specifically-located hybridizations (where machine learning is applied in a specific procedure) and global hybridizations (in which machine learning has a higher effect on the metaheuristic design). As part of the first group, the following categories are defined: parameter fine-tuning, initialization, evaluation, population management, operators, and local search. On the other hand, the second one includes: reduction of search space, algorithm selection, hyperheuristics, cooperative strategies, and new types of metaheuristics.

Similarly, there are a few reviews on works where metaheuristics are used to improve the performance of machine learning techniques. For instance, [28] focuses on two evolutionary algorithms (EAs), namely GAs and genetic programming (GP), and discusses their application to discovery of classification rules, clustering, attribute selection and attribute construction. [27] analyzes the role of OR in data mining discussing the relevance of exact methods, heuristics and metaheuristics in supervised classification, unsupervised classification, rule mining and feature selection. More recently, [29] provides an overview of the use of optimization in Big Data, focusing on metaheuristics. The book introduces the role of metaheuristics in clustering, association rules, classification, and feature selection in classification. It also includes a chapter listing all available frameworks for metaheuristics, data mining, and the combination of both. Building on these reviews, we arrange the literature works into the following categories: classification, regression, clustering, and rule mining.

Figure 4 shows the classification scheme we use. Some relevant and representative works, both considering machine learning for enhancing metaheuristics and metaheuristics in machine learning, are reviewed in Sections 3 and 4, respectively.

3 Using machine learning for enhancing metaheuristics

In order to improve clarity, the review on how machine learning techniques have been used to enhance metaheuristics has been divided into two sub-sections: the first one analyzes local-level hybridizations while the second one discusses global-level hybridizations. Each of these, in turn, have been classified by the corresponding topic.

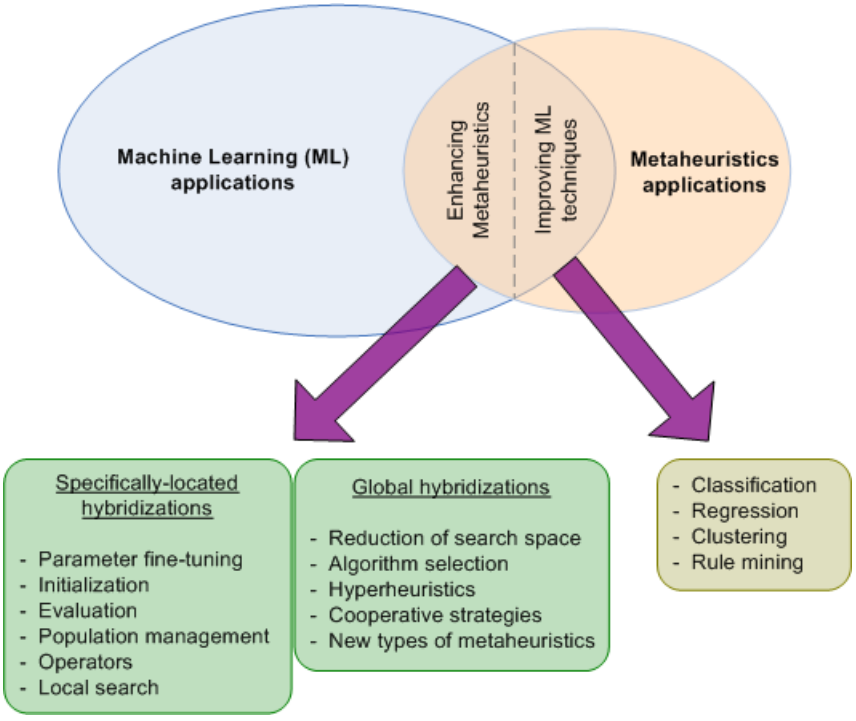
3.1 Specifically-located hybridizations

The ***fine-tuning of metaheuristic parameters*** is known to have a significant effect on the algorithm performance. However, this issue is not always properly addressed and many researchers still continue selecting parameter values by performing exhaustive testing or copying values recommended for similar instances or problems.

Basically, there are three approaches:

1. *Parameter control strategies* [30] apply a dynamic fine-tuning of the parameters by controlling and adapting the parameter values during the solving of an instance. The main types of control are: *(i)* deterministic, which modifies the parameter values by some deterministic rule; and *(ii)* adaptive, which employs feedback from the

Fig. 4. Classification of works combining metaheuristics and machine learning.



search. For instance, there are works relying on fuzzy logic [31], support vector machine (SVM) [32], and linear and SVM regression [33].

2. *Parameter tuning strategies* assume that the algorithms are robust enough to provide good results for a set of instances of the same problem with a fixed set of parameter values. Frequently, researchers focus on a subset of the instances and analyze their fitness landscapes. Popular techniques are: response surface [34], logistic regression [35], and tree-based regression [36].
3. *Instance-specific parameter tuning strategies* present characteristics from the previous approaches. While the parameter values are constant as in the second approach, they are specific for each instance as in the first. These strategies employ a learning mechanism able to return recommended sets of parameter values given a number of instance features. Techniques employed are: Bayesian networks [37], case-based reasoning (CBR) [38], fuzzy logic [39], linear regression [40], and neural networks [41].

A highly popular approach related to the first category is known as reactive search [42]. It proposes the integration of sub-symbolic machine learning techniques into heuristics, in order to allow the algorithm for self-tuning.

Typically, metaheuristics generate their ***initial solutions*** randomly, using design of experiments [43], or via a fast heuristic. There are also works employing machine learning techniques. For instance, some of them apply CBR to initialize GAs [44–46], while others explore the use of Hopfield neural networks [47]. In [48] the authors suggest using the Q-learning algorithm in the constructive phase of a GRASP and a reactive GRASP metaheuristics. In this line, the hybridization of data mining and the GRASP metaheuristic is discussed in [49].

In real-life applications it is common to find objective functions and constraints that are computationally expensive to ***evaluate*** [50, 51]. In these cases, it is required to build an approximation model to assess solutions employing polynomial regression [52], neural networks [53–55], SVM [56], Markov fitness models [57], kriging [58] or radial basis functions [59], for example. Some authors combine their use with that of real objective functions [60, 61]. An interesting survey of model approximation in evolutionary computation may be found in [62]. Another option to reduce evaluation costs is to evaluate only representative solutions. Following this idea, in [63] the authors apply fuzzy clustering. Similarly, in [64] the authors suggest using clustering techniques and neural networks ensembles.

Regarding ***population management***, many authors attempt to extract information from solutions already visited and employ it to build new ones, aiming to explore more promising search spaces. A number of works rely on the Apriori algorithm (to identify interesting subsolutions) [65–68] or on CBR [69]. Another important issue in PMs is the population diversity, since maintaining it may lead to better performances. The most common technique for promoting diversity is clustering analysis. In [70], for instance, individuals in a GA are separated in different sub-populations based on their features and only those in the same cluster compete for survival. The selection operator is applied independently to each cluster. In contrast, in [71] the authors allow interactions among sub-populations of an evolutionary strategy when selecting candidates for recombination. Other works relying on clustering analysis, but in the context of multi-objective metaheuristics, are [72] and [73].

The search of a metaheuristic may be improved by introducing knowledge in ***operators*** such as mutation or crossover operators in PMs. For example, [74] propose a coevolutionary GA that incorporates an extraction mechanism to be employed in the crossover. Two classification algorithms are tested: C4.5 and CN2. In [75], the authors design a class of evolutionary computation processes called learnable evolution model (LEM), which uses symbolic learning methods to create rules that explain why certain individuals are superior to others. These rules are then employed to create new populations by avoiding past failures, using recommendations or generating variants. In [76], this class is extended to address multi-objective problems seeking rules to identify why some individuals dominate others.

Some machine learning techniques have been used as ***local searches***. For instance, [77] employ a multi-objective EA combined with an inverse neural network. This neural network is a local search aiming to discover better individuals from previous generations. In particular, it is trained considering parameters and criteria as inputs and outputs, respectively. First, the criteria obtained from individuals of the present generation are slightly modified. Then, the parameters for the new individuals are obtained using the neural network in a reverse way. The authors test their approach on a set of benchmark bi-objective functions. A similar approach is suggested in [54] to be applied to an aircraft control system design application.

3.2 Global hybridizations

A few works have attempted to ***reduce the search space*** in order to make more effective and efficient searches. Machine learning techniques used are: clustering techniques [78–81], neural networks [82, 83] and principal component analysis [84].

The ***algorithm selection problem*** (ASP) aims to predict the algorithm from a portfolio that will perform best, employing a given set of instance features. The framework for this problem was initially proposed by [85], where it was applied to partial differential equation solvers. More recently, [86] presents it in the context of optimization algorithms. There are four basic elements in the framework: *(i)* the problem space P represents the set of problem instances; *(ii)* the feature space F includes instance characteristics; *(iii)* the algorithm space A is the portfolio of available algorithms; and *(iv)* the performance space Y is the mapping of each algorithm to the performance metrics. Accordingly, the ASP can be stated as follows [87]: given a problem instance $x \in P$ with feature vector $f(x) \in F$, the ASP searches the selection mapping $S(f(x))$ into algorithm space A such that the selected algorithm $\alpha \in A$ maximizes the performance mapping $y(\alpha, x) \in Y$. Thus, for instance, [88] develops a methodology to predict the performance of metaheuristics and acquire insights into the relation between search space characteristics of an instance and algorithm performance. The author tests the ILS and the robust TS metaheuristics, as well as the max-min ant system for solving the quadratic assignment problem. A neural network implementing genetic adaptive smoothing parameter selection is trained to predict which algorithm will perform best. Also, in [89], an approach is designed to select the best optimization method for solving a given travelling salesman problem (TSP) instance. Initially, 14 TSP properties and the performance values obtained with each metaheuristic analyzed (GRASP, TS, SA and GA) are stored. Then, a rank of metaheuristics is determined by using a multi-layer perceptron network. Several network architectures are assessed. In [87], the authors construct a methodology to compare the strengths and weaknesses of a set of optimization algorithms. First, the instance space is generated. This step includes selecting a subset of features to obtain a two-dimensional instance space (for a better visualization) and provide a good separation of easy and hard instances. Afterwards, classification techniques are used to identify the regions where

an algorithm performs well or poorly. Finally, an analysis of the algorithmic power is performed considering the size and location of each algorithm footprint. The experiment is carried out with 8 algorithms for solving the graph coloring problem.

According to [90], ***hyperheuristics*** may be described as search methods or learning mechanisms for selecting or generating heuristics to solve computational search problems. Typically, these methods do not aim to obtain better results than problem-specific metaheuristics, but to be able to automate the design of heuristic methods and/or deal with a wide range of problems. The authors propose a basic classification, which takes into account the following dimensions: *(i)* the nature of the heuristic search space (either heuristic selection or generation); and *(ii)* the feedback, since hyperheuristics may learn (following online or offline learning strategies) or not. Whereas online learning refers to methods that learn during the solving of a problem instance, offline learning methods try to extract information from a set of training instances to be applied for solving new instances. A comprehensive survey on hyper-heuristics may be found in [91]. In [92], the authors explore the potential of associative classifiers in a hyperheuristic approach for solving the training scheduling problem. The classifiers have to choose the low-level heuristic (which represents a given local search neighborhood) to employ at each step while constructing a solution. Reinforcement learning is highly popular in methodologies selecting heuristics employing an online learning strategy (e.g., see [93]). In this case, each heuristic has associated a score that determines the probability of being selected. These scores are updated according to the intermediate solutions obtained with each heuristic. There are also a number of works employing regression techniques. For instance, related to the evaluation category, [94] suggest employing neural networks and logistic regression to predict objective function values of solutions in a hyperheuristic search. It is also worth mentioning the approach described in [95], where CBR is used for selecting heuristics when addressing course and exam timetabling problems. In [96], the authors address a constraint satisfaction problem, where the order in which the variables are selected affects the complexity of the search. The authors present a hyperheuristic based on a logistic regression model that decides which variable ordering heuristic should be applied given the features of an instance at different steps of the search. In [97] an apprenticeship learning hyperheuristic is proposed for vehicle routing. Taking a state-of-the-art hyperheuristic as an expert, the authors follow a learning approach that yields various classifiers, which capture different actions that the expert performs during the search. While this approach relies on a C4.5 algorithm, in [98] it is improved by using a multilayer perceptron. Another approach is presented in [99], where a tensor-based online learning selection hyperheuristic is designed for nurse rostering. The proposed approach consists of the consecutive iteration of four stages: during the first and second stage, two tensors are constructed considering different heuristic selection and move acceptance methods; at the end of the second stage, each tensor is subjected to factorization and, using the information of both tensors, the heuristic space is partitioned; the third is a parameter control phase for the heuristics; and the final stage performs the search switching between heuristics periodically, using appropriate heuristic parameter values.

During the last decades, a new trend in optimization has emerged as a consequence of the technological development based on ***cooperative strategies***. It consists in combining several algorithms/agents to produce a hybrid strategy in which they cooperate in parallel or sequentially. Communication among them can be either many-to-many (direct) or memory-based (indirect). Agents may share partial or complete solutions and models, among others. It is broadly accepted that strategies based on agents with unrestricted access to shared information may experiment premature convergence. Commonly, there is an agent that coordinates the search of the others, organizing the communication. This strategy attempts to develop a robust methodology that provides high-quality solutions by exploiting the specific advantages of each algorithm. For example, [100] develop a centralized hybrid metaheuristic cooperative strategy, where knowledge is incorporated into the coordinator agent through fuzzy rules. These rules have been defined from a knowledge extraction process applied to the results obtained by each metaheuristic. Then, the coordinator agent collects information and sends orders to each solver agent that will affect its search behaviour (such as re-initiate the search with a specific initial solution, or change the parameter values). The strategy is tested on the knapsack problem, employing a TS, a SA, and a GA. In [101], the author describes an approach to deal with the permutation flow shop scheduling problem (PFSP), where a set of agents run in parallel, each applying a given metaheuristic. The best solutions are stored and employed to form a tensor. This tensor is factorized and the pattern obtained is sent to all agents, which will use it to try to build better solutions. In [102], a cooperative strategy relying on different metaheuristic / local search combinations is put forward. The architecture makes use of two types of agents: the launcher and the metaheuristic agent. The launcher conducts the following tasks: queue

instances, configure other agents and gather solutions. Metaheuristic agents execute one of the metaheuristic / local search heuristic combinations. Each of them continuously adapts itself according to a cooperation protocol based on reinforcement learning and pattern matching. This proposal is tested on the PFSP and the capacitated vehicle routing problem.

There are several ***new metaheuristics*** based on learning procedures. Most rely on the fact that a set of pseudo-optimal solutions may be considered a sample drawn from an unknown probability distribution. This distribution may be estimated by employing a selected set of promising solutions and used to generate new solutions. A review of these metaheuristics, called estimation of distribution algorithms (EDAs), can be found in [103]. Typically, authors employ these algorithms using fixed-length strings over a finite alphabet to represent solutions. They may be classified into three groups depending on whether no interactions are considered, or only pairwise or multiple ones. From the first group, the most popular are: the population-based incremental learning (PBIL) [104], the compact genetic algorithm (cGA) [105], and the univariate marginal distribution algorithm (UMDA) [106]. Some well-known algorithms assuming only pairwise interactions are: the mutual-information-maximizing input clustering (MIMIC) [107] algorithm and the bivariate marginal distribution algorithm (BMDA) [108]. Considering multiple interactions, there are: the extended compact genetic algorithm (ECGA) [109], the factorized distribution algorithm (FDA) [110], and the Bayesian optimization algorithm (BOA) [111]. These metaheuristics have been employed in a wide range of fields such as routing [112, 113], scheduling [114], and nutrition [115].

4 Using metaheuristics to improve machine learning

Metaheuristics have been extensively employed to improve machine learning tasks. Briefly, we review some of the most successful approaches in the supervised learning topic, both in classification and regression, and in the unsupervised learning topic, including clustering and rule mining.

Classification is a popular problem in supervised learning, consisting in identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known. In this context, metaheuristics have been mainly applied for feature selection, feature extraction and parameter fine-tuning. In [116] authors suggest that the bags of visual words algorithm could be improved when non linear combinations of weighted features obtained with GP are considered. The approach has successfully been applied to the object recognition field, learning both the weights of each visual word (feature) and the non linear combination of them. Similar approaches have been presented for large feature sets. Thus, [117] employ GAs to select discriminant features applied to intrusion detection using a decision trees classifier. Also studying classification trees, [118] implements a GA to generate a set of diverse trees, each with a large explanatory power. In [119], the authors present a multi-classification algorithm relying on multi-layer perceptron neural network models. In order to obtain high levels of sensitivity and accuracy (which may be conflicting measures), a Pareto-based multi-objective optimization methodology based on a memetic EA is proposed. In [120], the use of GAs is proposed to simultaneously optimize the parameters of the SVM algorithm and perform a feature selection process. In [121], a GA performs feature selection on electroencephalogram signals (EEG) applying non linear classifiers (SVM and neural networks). Working on cancer diagnosis, [122] selects gene by applying a GA combined with SVM. The approach focuses on sensitivity, specificity and number of genes. A comparison between approaches with different criteria and one relying on the k-means algorithm is put forward. In [123] different metaheuristics are evaluated also in the context of feature selection. The authors implement a TS, a memetic algorithm, a GRASP and a GA, and compare their effectiveness with sequential forward feature selection. Different niching GAs for feature selection are proposed in [124], which are applied to the design of fuzzy rule-based classification systems. In [125] two multi-objective PSO algorithms are designed for feature selection, which aim to maximize the classification performance while minimizing the number of features. They are compared against several conventional methods and three well-known multi-objective EAs using the k-nearest neighbor algorithm as classifier. In [126], the author employs a GA, a TS and an ACO for parameter fine-tuning of a single classifier and classifiers ensemble optimization, working with SVM.

Regression aims to estimate the relationships among a response variable and one or more explanatory variables, and has a wide range of applications. Typically, the use of machine learning is related to the training of complex regression models. Neuroevolution is an emergent field which employs EAs to train neural networks. Thus, [127] provides a literature review focusing on elements evolved: connection weights, architectures, learning rules, and input features. In [128], the authors develop the neuroevolution of augmenting topologies (NEAT) method, which evolves topologies and weights at the same time. They claim that its efficiency resides in: *(i)* employing a principled method of crossover of different topologies; *(ii)* protecting structural innovation using speciation; and *(iii)* incrementally growing from minimal structure. More recently, [129] has shown the benefits of optimizing each neuron's transfer function, creating heterogeneous networks. In a similar approach, [130] present a methodology to find the best architecture of a neural network using metaheuristics. The authors tested the following ones: generalized extremal optimization, VNS, SA, and canonical GA.

Clustering refers to grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups. This vague definition encompasses a high number of models. Centroid models are based on an NP-hard optimization problem (thus, only approximated solving methods such as metaheuristics may be employed). In [131] differential evolution algorithms are applied to clustering problems (single and multi-objective). In [132] a TS metaheuristic is implemented to computationally deal with the non convex optimization problem of the unsupervised clustering of data samples. Similarly, [133] use ACO to cluster objects, obtaining faster results in terms of the number of needed operations (i.e., objective functions evaluations). Other authors use GAs for the same task [134–136]. In [137], a PSO metaheuristic is applied to the clustering problem, improving the results obtained using a TS and classic unsupervised learning methods. In [138], the authors also use a PSO metaheuristic to automatically cluster data from students in a learning management system (LMS) to adapt teaching resources to specific students' needs. Gene clustering is performed in [139], where a comparative study is presented based on the following metaheuristics: GA, PSO, cuckoo search and levy flight cuckoo search. More recently, [140] present a GRASP metaheuristic for biclustering (i.e., considering both genes and conditions) of gene expression data. A validation is completed with different synthetic datasets. The reader can find more details in the applications of metaheuristics to unsupervised learning in the surveys [141, 142].

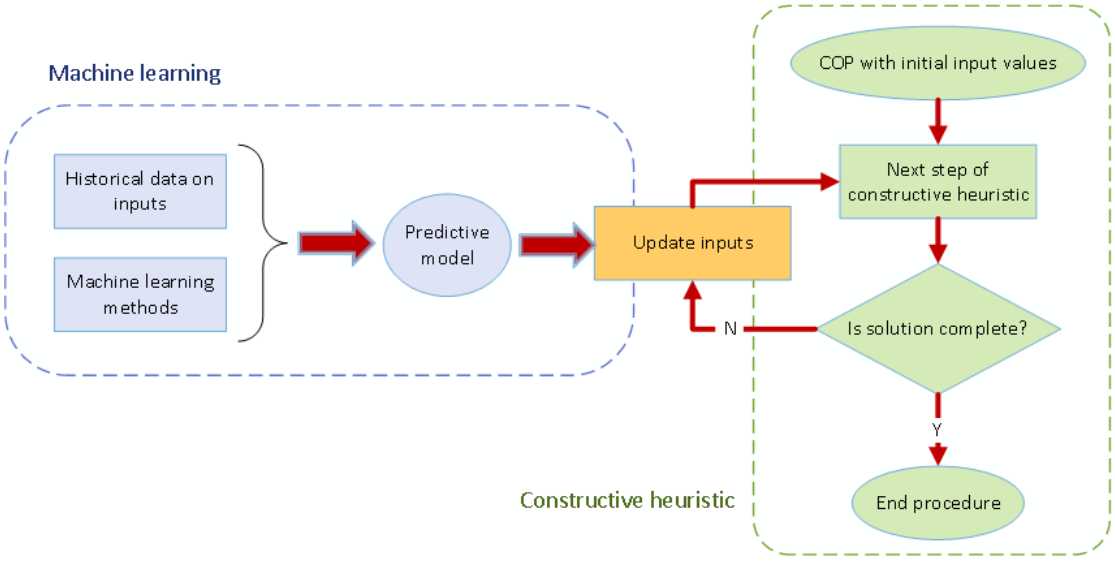
Rule mining gathers methods for discovering relevant relations between variables in large databases. For example, a hybrid approach is presented in [143] for discovering small-disjunct rules combining a decision tree algorithm (C4.5) and a GA. While the first is employed for large-disjuncts rules, the metaheuristic works on small ones. This hybrid approach achieves better predictive accuracy. In the book [144], in addition to present data mining tasks and paradigms, the author describes the application of GAs and GP for rule discovery, and EAs for generating fuzzy rules. After modeling association rules discovery as an optimization problem, [145] explore the use of a GA to obtain associations between genes from DNA microarray data. Noticing that most approaches tend to seek only frequent rules, [146] propose a multi-objective approach combining a GA and exact methods to discover interesting rules in large search spaces. A public micro-array database is employed to carry out computational experiments. A multi-objective metaheuristic approach is proposed in [147] to create rules and build a Pareto front considering the sensitivity and specificity criteria. A GRASP with path-relinking is implemented.

5 Our learnheuristic framework for solving COPDIs

After reviewing in Sections 3 and 4 works where machine learning may enhance metaheuristics or metaheuristics may improve machine learning, this section presents our learnheuristic framework. It integrates both fields, making it possible to address a specific kind of COP.

As previously described, our learnheuristic framework aims at solving COPs in which the model inputs (either located in the objective function or in the set of constraints) are not fixed in advance. Instead, these inputs (e.g., customers' demands, serving times, etc.) might vary in a predictable way according to the current status of the

Fig. 5. Basic scheme of a learnheuristic framework.



partially-built solution at each iteration of the constructive heuristic. More formally, these problems might be represented as follows:

$$\begin{array}{l} \text{Min} \quad C(s, I_{OF}(s)) \text{ or, alternatively,} \\ \text{Max} \quad B(s, I_{OF}(s)) \end{array} \tag{1}$$

$$\text{subject to: } Q_j(s, I_C(s)) \leq r_j \quad \forall j \in J \tag{2}$$

$$s \in S \tag{3}$$

where: *(i)* S refers to a discrete space of possible solutions s; *(ii)* $C(s)$ represents a cost function (alternatively, $B(s)$ represents a benefits function); *(iii)* $I_{OF}(s)$ and $I_C(s)$ refer to inputs in the objective function or the constraints, respectively; and *(iv)* Equations 2 represent a set of constraints. Thus, the aim of this type of problems is to minimize a function of costs (or, alternatively, maximize a function of benefits) subject to a number of constraints. The novel characteristic is that inputs in the objective function and/or the constraints may depend on the solution structure, which makes them to be dynamic as the partially-built solution evolves, and not fixed in advance. This is a basic case of dynamic inputs, which could be easily extended to deal with multi-objective and/or stochastic problems [148].

In order to deal with these COPDIs, we propose the use of a learnheuristic framework as explained next in detail. Figure 5 shows the basic scheme of this approach. Initially, historical data on different system states (e.g., different assignments of users to RATs) and their associated inputs (e.g., users' demands observed for the corresponding assignments) are employed to generate machine-learning predictive models (e.g., regression models, neural network models, etc.). Then, these predictive models are iteratively used during the heuristic-based constructive process in order to obtain updated estimates of the problem inputs (e.g., users' demands) as the structure of the solution (e.g., users-to-RAT assignment map) varies. Eventually, once the construction process is finished, a complete solution is generated. Without the use of the learning mechanism, the heuristic-based construction process will not take into account the variations in the inputs due to changes in the solution structure, which will lead to sub-optimal solutions.

Pseudo-code 1 contains a more detailed description of the basic learnheuristic framework. Notice that the main loop iterates over a list of elements that are provided by the constructive heuristic (e.g., next user-to-RAT assignment). At each iteration, the algorithm evaluates the current status of the partially-built solution, makes use of the predictive model to update the problem inputs according to this status, and follows the heuristic logic to take another solution-building step based on the new problem inputs. This basic scheme could be extended in different ways, e.g.: *(i)* by employing an online approach, where new inputs are used to update and improve the predictive

model; and *(ii)* by using an "assembled" or blending approach, in which several models are built and then combined in order to generate the inputs estimates.

```
Learnheuristics(historicalData, inputs)
% historicalData: historical data on different system states and their associated inputs
% inputs: problem instance
  model ←buildPredictiveModel(historicalData)
  sol ← empty
  while (sol is not completely built) do % iterative learning-heuristic process
      inputs ←updateInputs(model, inputs, sol)
      sol ← nextHeuristicStep(inputs, sol)
  end while
  return sol
```

Pseudo-code 1. Basic scheme of learnheuristic algorithms.

As any other heuristic procedure, the aforementioned learnheuristic approach can be integrated into a more complex metaheuristic framework. For instance, it can be easily integrated into multi-start, GRASP, or ILS frameworks. In order to do so, the learnheuristic algorithm may be combined with biased-randomization strategies as the ones proposed in [149], which allow for generating a number of high quality solutions from a deterministic heuristic (i.e., one that does not include any random behavior, thus providing always the same solution, as opposed to a randomized heuristic). Pseudo-code 2 gives an example of how this integration could be made in the case of a simple multi-start framework.

```
Multi-start(historicalData, inputs, distribution, maxTime)
% distribution: probability distribution and parameters for the biased-randomization process
% maxTime: maximum computing time allowed
  initInputs ← inputs % copy of initial inputs
  elapsedTime ← 0
  initTime ← currentTime
  bestSol ← biasedRandLearnheuristic(historicalData, inputs, distribution)
  inputs ← initInputs % reset inputs
  while (elapsedTime ≤ maxTime) do
      newSol ← biasedRandLearnheuristic(historicalData, inputs, distribution)
      newSol ← localSearch(newSol)
      if (cost(newSol) ≤ cost(bestSol)) then
          bestSol ← newSol
      end if
      inputs ← initInputs % reset inputs
      elapsedTime ← currentTime−initTime
  end while
  return bestSol
```

Pseudo-code 2. Multi-start metaheuristic integrating a learnheuristic algorithm with biased-randomization.

6 Potential applications in different fields

This section provides a series of examples, belonging to different optimization areas, in which the use of learnheuristics might facilitate the solving process of more realistic and rich models.

- ***Transportation:*** In the transportation area and, in particular, in vehicle and arc routing problems, inputs such as the customers' demands might be dynamic in the sense that they might depend upon the delivery time and whether or not certain time-windows are satisfied. It is a function of the solution structure, e.g., the order in which the customers are visited, the number and type of vehicles employed, etc. Similarly, the traveling times, which affect the distribution cost, might also be dynamic and dependent on the solution structure, specially in large cities where traffic jams occur frequently.
- ***Logistics:*** As discussed in [6], the assignment of customers to certain distribution centers might have a significant effect on the customers' willingness to spend (i.e., on their demands). Therefore, in realistic facility location problems and similar ones, modelers might have to face dynamic inputs influenced by the shape of the solution (i.e., which facilities are open and how customers are assigned to them).
- ***Production:*** In scheduling problems, for instance, processing times of jobs into machines might not be fixed but, instead, they may be a function of the order in which they are processed by the machine (e.g., due to 'fatigue' issues or to breaks). A similar situation can happen in project scheduling, where some working teams might be more efficient than others and assigning them to a given sub-project could cause the delay of others.
- ***Finance:*** In problems such as portfolio optimization, the covariance matrix that measures the risk associated with each pair of assets could also be a function of the current portfolio structure (i.e., which other assets are already included and which percentage of investment has been assigned to each of them). Likewise, the expected return for each asset might depend on the current composition of the portfolio. This dynamic behavior of the inputs can be extended to different risk-management problems which include some sort of portfolio optimization.

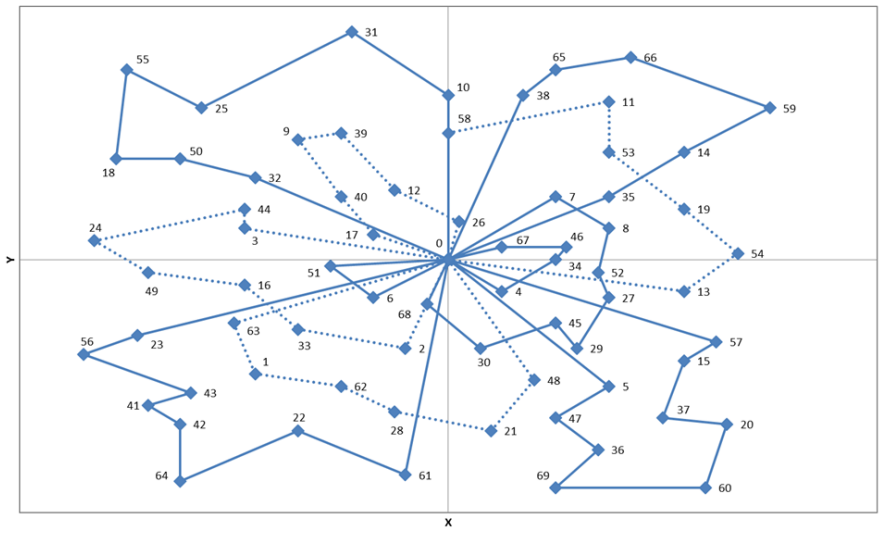
7 A numerical experiment

This section describes a simple numerical experiment that illustrates the use of a learnheuristic approach. We consider a vehicle routing problem in which each customer's demand will depend on the order in which the customer is visited. For each customer, its initial demand value is an upper-bound of the real demand. In other words, this value will be valid only if the customer is visited by a vehicle as the first stop in its route. Then, as the position in which the customer is visited increases, the customer's demand will be reduced (i.e., higher service times imply lower demands). Therefore, if we use a constructive heuristic to solve the vehicle routing problem considering the initial demands as fixed inputs, the solution will be overestimating the real demands. This, in turn, will lead to higher costs, since the number of routes employed to satisfy the real demands will be higher than necessary. Likewise, vehicles will be carrying more load than strictly required. On the contrary, if we are able to forecast the real customers' demands based on their position inside a route, then each route might be able to cover additional customers and the total distance-based costs will be reduced.

In order to compare both cases, the Clarke and Wright constructive heuristic [150] has been applied to a random instance belonging to the well known benchmarks for the vehicle routing problem, particularly to the instance P-n70-k10 (http://neo.lcc.uma.es/vrp/wp-content/data/instances/Augerat/P-VRP.zip). This instance consists of 12 vehicles with the same capacity – 135 units – and 70 customers. The customers are scattered in an area with x axis ranging from -34 to 30, and y axis ranging from -36 to 36. The depot is located at coordinates (0,0). Initial customers demands range from 5 to 37.

On the one hand, we have considered fixed demands, i.e., the original demands provided by the instance are used to obtain the solution through the heuristic in the standard way. On the other hand, we have created a predictive model to calculate dynamic demands in order to apply a learnheuristic algorithm following the scheme in Pseudo-code 1.

Fig. 6. Routes obtained considering fixed demands.



In this case, for illustrative purposes, the following linear regression model has been considered:

$$d = \max\{k_1 \cdot d_0,\ d_0 - k_2 \cdot d_0 \cdot (p-1)\} \tag{4}$$

where d is the predicted demand of a given customer, d_0 is the initial demand of the same customer, k_1 and $k_2 \in (0, 1)$, and p is the position order in the route of the aforementioned customer. In particular, we have applied $k_1 = 0.20$ and $k_2 = 0.05$. The regression model aims at predicting a customer's demand taking into account the position in which the customer is served in the route, so that the demand decreases as the position increases or until a certain demand lower-bound is reached. Once we have the model, the heuristic starts the loop building routes until a solution is obtained. Thus, each time the heuristic performs a step, incorporating a new customer in a route or moving a customer from one route to another, the customer's demand is predicted and updated according to its new position in the corresponding route. As mentioned before, the total demand in a route is limited by the capacity of the vehicle. Therefore, this prediction affects the next steps that can be performed.

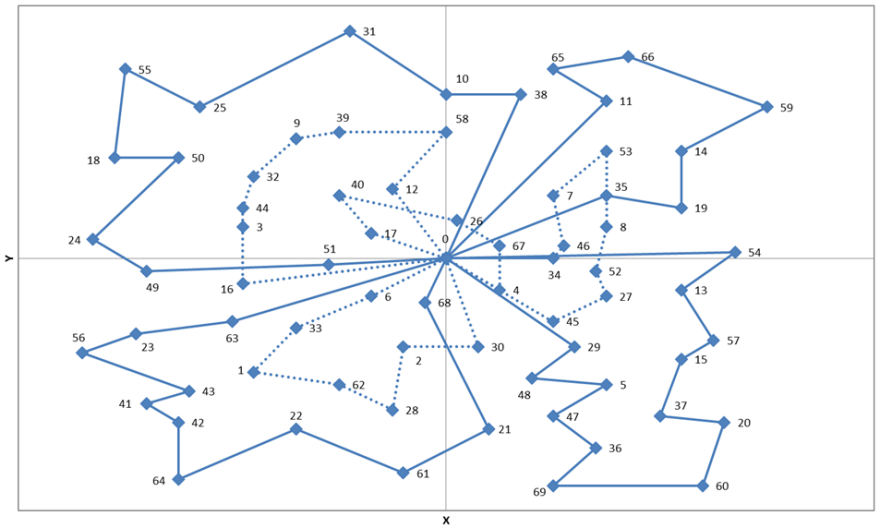
Accordingly, we have performed two experiments using these two versions of the Clarke and Wright heuristic, both in its standard way and using the learnheuristic framework. When fixed demands are considered, the best solution the constructive heuristic is able to obtain has an associated cost of 896.86, and it involves 11 routes (see Figure 6). However, if demands are predicted taking into account the delivery order, the same heuristic obtains a solution with 8 routes and a cost of 791.26 (see Figure 7). Therefore, the savings might be noticeable when dynamic demands are considered.

8 Conclusions and future research

Real-life problems faced by companies are becoming increasingly complex. This can be due, among other factors, to the existence of more competitive markets as well as to larger and more interconnected supply chain systems. On the other hand, the technological development of the last few decades allows the implementation of more powerful and faster algorithms, and the analysis of huge amounts of diverse types of data. As a consequence, hybrid approaches for addressing hard combinatorial optimization problems (COPs) are highly popular.

This paper has focused on the combination of metaheuristics and machine learning. An overview of the different approaches and a general classification have been provided. We have presented a specific type of realistic COP which requires this hybridization. In particular, these problems are characterized by inputs (located either in the objective function or the set of constraints) that are not fixed in advance, but may vary according to the solution characteristics. Then we propose a new solving approach, learnheuristic algorithms, to cope with these dynamic

Fig. 7. Routes obtained considering dynamic demands.



optimization problems. This approach relies on machine learning techniques to learn the relationships between inputs and solution characteristics from historical data, and a constructive heuristic (which may be embedded in a metaheuristic algorithm), to build a high quality solution using predictions. Different extensions of this approach can be considered, e.g.: *(i) online version*, in which information regarding new inputs can be used to improve the predictive modeland *(ii) blended version*, in which predictions from several models are averaged, not necessarily giving the same weight to each of them. Finally, some potential applications to a variety of fields have been also pointed out.

From the work developed, we foresee several open research lines that could be explored: *(i)* implement the methodologies proposed to specific fields/problems testing several techniques of machine learning; *(ii)* extend the methodology to stochastic and/or multi-objective optimization problems; and *(iii)* study the use of distributed and parallel computing paradigms to allow for real-time decision-making.

Acknowledgement: This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and FEDER (TRA2013-48180-C3-P, DPI2013-44461-P, TIN2015-66951-C2-2-R), and the Catalan Government (2014-CTP-00001). Likewise, we want to thank the support of the UOC doctoral programme.

References

[1] Talbi, E.G., Metaheuristics: from design to implementation, Wiley Publishing, 2009, ISBN 0470278587, 9780470278581.
[2] Caceres-Cruz, J., Arias, P., Guimarans, D., Riera, D., Juan, A.A., Rich vehicle routing problem: survey, ACM Comput Surv, 2014, 47(2),1–28.
[3] Talbi, E.G., Combining metaheuristics with mathematical programming, constraint programming and machine learning, 4OR, 2013, 11(2),101–150.
[4] Maniezzo, V., Stützle, T., Vo, S., Matheuristics: hybridizing metaheuristics and mathematical programming, Springer Publishing Company, Incorporated, 1st ed., 2009, ISBN 144191305X, 9781441913050.
[5] Juan, A.A., Faulin, J., Grasman, S.E., Rabe, M., Figueira, G., A review of simheuristics: extending metaheuristics to deal with stochastic combinatorial optimization problems, Operations Research Perspectives, 2015, 2,62–72.
[6] Calvet, L., Ferrer, A., Gomes, M.I., Juan, A.A., Masip, D., Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation, Comput Ind Eng, 2016, 94(C),93–104.
[7] Boussaïd, I., Lepagnot, J., Siarry, P., A survey on optimization metaheuristics, Inf Sci, 2013, 237,82–117.
[8] Dorigo, M., Optimization, learning and natural algorithms, Ph.D. thesis, Politecnico di Milano, Italy, 1992.
[9] Farmer, J.D., Packard, N.H., Perelson, A.S., The immune system, adaptation, and machine learning, Phys D, 1986, 2(1-3),187–204.

[10] Holland, J.H., Outline for a logical theory of adaptive systems, Journal of the ACM, 1962, 3(9),297–314.
[11] Feo, T.A., Resende, M.G.C., A probabilistic heuristic for a computationally difficult set covering problem, Oper Res Lett, 1989, 8(2),67–71.
[12] Martin, O., Otto, S.W., Felten, E.W., Large-step Markov Chains for the TSP incorporating local search heuristics, Oper Res Lett, 1992, 11(4),219–224.
[13] Kennedy, J., Eberhart, R.C., Particle swarm optimization, In: Proceedings of the IEEE International Conference on Neural Networks. 1995, p. 1942–1948.
[14] Glover, F., Heuristics for integer programming using surrogate constraints, Decision Sci, 1977, 8(1),156–166.
[15] Kirkpatrick, S., Optimization by simulated annealing: quantitative studies, Journal of statistical physics, 1984, 34(5-6),975–986.
[16] Glover, F., Future paths for integer programming and links to artificial intelligence, Comput Oper Res, 1986, 13(5),533–549.
[17] Mladenovic, N., A variable neighborhood algorithm: a new metaheuristic for combinatorial optimization, Abstracts of papers presented at Optimization Days, 1995, ,112.
[18] Sörensen, K., Metaheuristics—the metaphor exposed, International Transactions in Operational Research, 2015, 22(1),3–18.
[19] Feo, T.A., Resende, M.G.C., Greedy randomized adaptive search procedures, J Global Optim, 1995, 6(2),109–133.
[20] Gendreau, M., Potvin, J.Y., Handbook of metaheuristics, Springer Publishing Company, Incorporated, 2nd ed., 2010, ISBN 1441916636, 9781441916631.
[21] Barber, D., Bayesian reasoning and machine learning, New York, NY, USA: Cambridge University Press, 2012, ISBN 0521518148, 9780521518147.
[22] Lantz, B., Machine learning with R, Packt Publishing, 2013, ISBN 1782162143, 9781782162148.
[23] Potvin, J.Y., Smith, K.A., Artificial neural networks for combinatorial optimization, Boston, MA: Springer US, 2003, p. 429–455.
[24] Smith, K.A., Neural networks for combinatorial optimization: a review of more than a decade of research, INFORMS J on Computing, 1999, 11(1),15–34.
[25] Jourdan, L., Dhaenens, C., Talbi, E.G., Using datamining techniques to help metaheuristics: a short survey, In: International Workshop on Hybrid Metaheuristics. Gran Canaria, Spain: Springer Berlin Heidelberg, 2006, p. 57–69.
[26] Zhang, J., Zhang, Z.h., Lin, Y., Chen, N., Gong, Y.j., Zhong, J.h., et al., Evolutionary computation meets machine learning: a survey, Comp Intell Mag, 2011, 6(4),68–75.
[27] Corne, D., Dhaenens, C., Jourdan, L., Synergies between operations research and data mining: the emerging use of multi-objective approaches, Eur J Oper Res, 2012, 221(3),469–479.
[28] Freitas, A., A review of evolutionary algorithms for data mining, In: Soft computing for knowledge discovery and data mining. Springer, 2008, p. 79–111.
[29] Dhaenens, C., Jourdan, L., Metaheuristics for big data, Wiley, 2016, ISBN 9781119347606.
[30] De Jong, K., Parameter setting in EAs: a 30 year perspective, In: Lobo, F., Lima, C., Michalewicz, Z., editors. Parameter setting in evolutionary algorithms. Springer, 2007, p. 1–18.
[31] Jeong, S.J., Kim, K.S., Lee, Y.H., The efficient search method of simulated annealing using fuzzy logic controller, Expert Syst Appl, 2009, 36(3),7099–7103.
[32] Zennaki, M., Ech-Cherif, A., A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial optimization problems, Journal of Applied Sciences, 2010, 10(18),1991–2000.
[33] Lessmann, S., Caserta, M., Arango, I.M., Tuning metaheuristics: a data mining based approach for particle swarm optimization, Expert Syst Appl, 2011, 38(10),12826 – 12838.
[34] Gunawan, A., Lau, H.C., Wong, E., Real-world parameter tuning using factorial design with parameter decomposition, New York, NY: Springer New York, ISBN 978-1-4614-6322-1, 2013, p. 37–59.
[35] Ramos, I.C.O., Goldbarg, M.C., Goldbarg, E.G., Neto, A.D.D., Logistic regression for parameter tuning on an evolutionary algorithm, In: IEEE Congress on Evolutionary Computation. Edinburgh, Scotland: IEEE, 2005, p. 1061–1068.
[36] Bartz-Beielstein, T., Parsopoulos, K.E., Vrahatis, M.N., Design and analysis of optimization algorithms using computational statistics, Applied Numerical Analysis & Computational Mathematics, 2004, 1(2),413–433.
[37] Pavón, R., Díaz, F., Laza, R., Luzón, M.V., Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study, Expert Syst Appl, 2009, 36,3407–3420.
[38] Pereira, I., Madureira, A., de Moura Oliveira, P.B., Abraham, A., Tuning meta-heuristics using multi-agent learning in a scheduling system, In: Transactions on Computational Science XXI. Springer Berlin Heidelberg, 2013, p. 190–210.
[39] Ries, J., Beullens, P., Salt, D., Instance-specific multi-objective parameter tuning based on fuzzy logic, Eur J Oper Res, 2012, 218(2),305–315.
[40] Caserta, M., Rico, E.Q., A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times, Computers & OR, 2009, 36(2),530–548.
[41] Dobslaw, F., A parameter tuning framework for metaheuristics based on design of experiments and artificial neural networks, World Academy of Science, Engineering and Technology, 2010, 64,213–216.
[42] Battiti, R., Brunato, M., Reactive search optimization: learning while optimizing, In: Gendreau, M., Potvin, J.Y., editors. Handbook of Metaheuristics. Springer, 2010, p. 543–571.
[43] Leung, Y.W., Wang, Y., An orthogonal genetic algorithm with quantization for global numerical optimization, Trans Evol Comp, 2001, 5(1),41–53.

[44] Ramsey, C.L., Grefenstette, J.J., Case-based initialization of genetic algorithms, In: Proceedings of the 5th International Conference on Genetic Algorithms. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., ISBN 1-55860-299-2, 1993, p. 84–91.
[45] Louis, S.J., McDonnell, J., Learning with case-injected genetic algorithms, Trans Evol Comp, 2004, 8(4),316–328.
[46] Li, Z.q., Zhang, H.l., Zheng, J.h., Dong, M.j., Xie, Y.f., Tian, Z.j., Heuristic evolutionary approach for weighted circles layout, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, p. 324–331.
[47] Yalcinoz, T., Altun, H., Power economic dispatch using a hybrid genetic algorithm, IEEE Power Engineering Review, 2001, 21(3),59–60.
[48] De Lima, F.C., De Melo, J.D., Neto, A.D.D., Using the Q-learning algorithm in the constructive phase of the GRASP and reactive GRASP metaheuristics, ISBN 9781424418213, 2008, p. 4169–4176.
[49] Santos, L.F., Martins, S.L., Plastino, A., Applications of the DM-GRASP heuristic: a survey, International Transactions in Operational Research, 2008, 15(4),387–416.
[50] Lim, D., Jin, Y., Ong, Y.S., Sendhoff, B., Generalizing surrogate-assisted evolutionary computation, Trans Evol Comp, 2010, 14(3),329–355.
[51] Tenne, Y., Goh, C.K., Computational intelligence in expensive optimization problems, vol. 2, Springer Science & Business Media, 2010.
[52] Zhou, Z., Ong, Y.S., Nguyen, M.H., Lim, D., A study on polynomial regression and Gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm, In: IEEE Congress on Evolutionary Computation, vol. 3. IEEE, 2005, p. 2832–2839.
[53] Hunger, J., Huttner, G., Optimization and analysis of force field parameters by combination of genetic algorithms and neural networks, J Comput Chem, 1999, 20(4),455–471.
[54] Adra, S.F., Hamody, A.I., Griffin, I., Fleming, P.J., A hybrid multi-objective evolutionary algorithm using an inverse neural network for aircraft control system design., In: Congress on Evolutionary Computation. IEEE, 2005, p. 1–8.
[55] Pathak, B.K., Srivastava, S., Srivastava, K., Neural network embedded multiobjective genetic algorithm to solve non-linear time-cost tradeoff problems of project scheduling, Journal of scientific and industrial research, 2008, 67(2),124–131.
[56] Yang, S., Liu, Q.H., Lu, J., Ho, S.L., Ni, G., Ni, P., et al., Application of support vector machines to accelerate the solution speed of metaheuristic algorithms, IEEE T Magn, 2009, 45(3),1502–1505.
[57] Brownlee, A.E., Regnier-Coudert, O., McCall, J.A., Massie, S., Using a Markov network as a surrogate fitness function in a genetic algorithm, In: IEEE Congress on Evolutionary Computation. IEEE, 2010, p. 1–8.
[58] Díaz-Manríquez, A., Toscano-Pulido, G., Gómez-Flores, W., On the selection of surrogate models in evolutionary optimization algorithms, In: 2011 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2011, p. 2155–2162.
[59] Regis, R.G., Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions, IEEE Transactions on Evolutionary Computation, 2014, 18(3),326–347.
[60] Rasheed, K., Hirsh, H., Informed operators: speeding up genetic-algorithm-based design optimization using reduced models, In: Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation. Morgan Kaufmann Publishers Inc., 2000, p. 628–635.
[61] Zhou, A., Zhang, Q., A surrogate-assisted evolutionary algorithm for minimax optimization, In: IEEE Congress on Evolutionary Computation. IEEE, 2010, p. 1–7.
[62] Jin, Y., A comprehensive survey of fitness approximation in evolutionary computation, Soft Comput, 2005, 9(1),3–12.
[63] Yoo, S.H., Cho, S.B., Partially evaluated genetic algorithm based on fuzzy c-means algorithm, In: International Conference on Parallel Problem Solving from Nature. Springer, 2004, p. 440–449.
[64] Jin, Y., Sendhoff, B., Reducing fitness evaluations using clustering techniques and neural network ensembles, In: Genetic and evolutionary computation conference. Springer, 2004, p. 688–699.
[65] Dalboni, F.L., Ochi, L.S., Drummond, L.M.A., On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem, In: International Network Optimization Conference. 2003, p. 182–188.
[66] Santos, L., Ribeiro, M.H., Plastino, A., Martins, S.L., A hybrid GRASP with data mining for the maximum diversity problem, In: International Workshop on Hybrid Metaheuristics. Springer, 2005, p. 116–127.
[67] Ribeiro, M.H., Plastino, A., Martins, S.L., Hybridization of GRASP metaheuristic with data mining techniques, Journal of Mathematical Modelling and Algorithms, 2006, 5(1),23–41.
[68] Santos, H.G., Ochi, L.S., Marinho, E.H., Drummond, L.M.d.A., Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem, Neurocomputing, 2006, 70(1),70–77.
[69] Louis, S.J., Genetic learning from experience, In: IEEE Congress on Evolutionary Computation, vol. 3. IEEE, 2003, p. 2118–2125.
[70] Streichert, F., Stein, G., Ulmer, H., Zell, A., A clustering based niching method for evolutionary algorithms, In: Genetic and Evolutionary Computation Conference. Springer, 2003, p. 644–645.
[71] Aichholzer, O., Aurenhammer, F., Brandstatter, B., Ebner, T., Krasser, H., Magele, C., et al., Evolution strategy and hierarchical clustering, IEEE T Magn, 2002, 38(2),1041–1044.
[72] Pulido, G.T., Coello, C.A.C., Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer, In: Genetic and Evolutionary Computation Conference. Springer, 2004, p. 225–237.

[73] Park, S.Y., Lee, J.J., Improvement of a multi-objective differential evolution using clustering algorithm, In: 2009 IEEE International Symposium on Industrial Electronics. IEEE, 2009, p. 1213–1217.
[74] Handa, H., Baba, M., Horiuchi, T., Katai, O., A novel hybrid framework of coevolutionary GA and machine learning, International Journal of Computational Intelligence and Applications, 2002, 2(01),33–52.
[75] Michalski, R.S., Learnable evolution model: evolutionary processes guided by machine learning, Mach Learn, 2000, 38(1-2),9–40.
[76] Jourdan, L., Corne, D., Savic, D., Walters, G., Preliminary investigation of the learnable evolution model for faster/better multiobjective water systems design, In: International Conference on Evolutionary Multi-Criterion Optimization. Springer, 2005, p. 841–855.
[77] Gaspar-Cunha, A., Vieira, A.S., A hybrid multi-objective evolutionary algorithm using an inverse neural network, 2004, p. 25–30.
[78] Hu, X.B., Huang, X.Y., Solving TSP with characteristic of clustering by ant colony algorithm, Acta Simulata Systematica Sinica, 2004, 12,014.
[79] Senjyu, T., Saber, A.Y., Miyagi, T., Shimabukuro, K., Urasaki, N., Funabashi, T., Fast technique for unit commitment by genetic algorithm based on unit clustering, HJEEE Proceedings-Generation, Transmission and Distribution, 2005, 152(5),705–713.
[80] Barreto, S., Ferreira, C., Paixao, J., Santos, B.S., Using clustering analysis in a capacitated location-routing problem, Eur J Oper Res, 2007, 179(3),968 – 977.
[81] Adibi, M.A., Shahrabi, J., A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem, Int J Adv Manuf Tech, 2013, 70(9),1955–1961.
[82] Lee, C., Gen, M., Tsujimura, Y., Reliability optimization design using a hybridized genetic algorithm with a neural-network technique, IEICE T Fund Electr, 2002, 85(2),432–446.
[83] Marim, L.R., Lemes, M.R., Dal Pino Jr., A., Neural-network-assisted genetic algorithm applied to silicon clusters, Phys Rev A, 2003, 67(3).
[84] Auger, A., Hansen, N., Performance evaluation of an advanced local search evolutionary algorithm, In: 2005 IEEE congress on evolutionary computation, vol. 2. IEEE, 2005, p. 1777–1784.
[85] Rice, J.R., The algorithm selection problem, Adv Comput, 1976, 15,65–118.
[86] Smith-Miles, K.A., Cross-disciplinary perspectives on meta-learning for algorithm selection, ACM Computing Surveys, 2009, 41(1),6.
[87] Smith-Miles, K., Baatar, D., Wreford, B., Lewis, R., Towards objective measures of algorithm performance across instance space, Comput Oper Res, 2014, 45,12–24.
[88] Smith-Miles, K., Towards insightful algorithm selection for optimisation using meta-learning concepts, In: WCCI 2008: IEEE World Congress on Computational Intelligence. IEEE, 2008, p. 4118–4124.
[89] Kanda, J.Y., de Carvalho, A.C.P.L.F., Hruschka, E.R., Soares, C., Using meta-learning to recommend meta-heuristics for the traveling salesman problem, In: Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on, vol. 1. IEEE, 2011, p. 346–351.
[90] Burke, E.K., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Woodward, J.R., A classification of hyper-heuristic approaches, In: Handbook of metaheuristics. Springer, 2010, p. 449–468.
[91] Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., et al., Hyper-heuristics: a survey of the state of the art, Eur J Oper Res, 2013, 64(12),1695–1724.
[92] Thabtah, F., Cowling, P., Mining the data from a hyperheuristic approach using associative classification, Expert Syst Appl, 2008, 34(2),1093–1101.
[93] Berberoğlu, A., Uyar, A.Ş., A hyper-heuristic approach for the unit commitment problem, In: European Conference on the Applications of Evolutionary Computation. Springer, 2010, p. 121–130.
[94] Li, J., Burke, E.K., Qu, R., Integrating neural networks and logistic regression to underpin hyper-heuristic search, Knowl-based Syst, 2011, 24(2),322–330.
[95] Burke, E.K., Petrovic, S., Qu, R., Case-based heuristic selection for timetabling problems, J Sched, 2006, 9(2),115–132.
[96] Ortiz-Bayliss, J.C., Terashima-Marín, H., Conant-Pablos, S.E., A supervised learning approach to construct hyper-heuristics for constraint satisfaction, In: Mexican Conference on Pattern Recognition. Springer, 2013, p. 284–293.
[97] Asta, S., Ozcan, E., An apprenticeship learning hyper-heuristic for vehicle routing in HyFlex, In: 2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS). 2014, p. 65–72.
[98] Tyasnurita, R., Ozcan, E., Asta, S., John, R., Improving performance of a hyper-heuristic using a multilayer perceptron for vehicle routing, In: 15th Annual Workshop on Computational Intelligence. Lancaster, UK: Springer, 2015, .
[99] Asta, S., Ozcan, E., Curtois, T., A tensor based hyper-heuristic for nurse rostering, Knowl-based Syst, 2016, 98,185–199.
[100] Cadenas, J.M., Garrido, M.C., Muñoz, E., Using machine learning in a cooperative hybrid parallel strategy of metaheuristics, Inform Sciences, 2009, 179(19),3255–3267.
[101] Asta, S., Machine learning for improving heuristic optimisation, Ph.D. thesis, The University of Nottingham, UK, 2015.
[102] Martin, S., Ouelhadj, D., Beullens, P., Ozcan, E., Juan, A.A., Burke, E., A multi-agent based cooperative approach to scheduling and routing, Eur J Oper Res, 2016, 254(1),169–178.

[103] Pelikan, M., Goldberg, D.E., Lobo, F.G., A survey of optimization by building and using probabilistic models, Comput Optim Appl, 2002, 21(1),5–20.
[104] Baluja, S., Population-based incremental learning. A method for integrating genetic search based function optimization and competitive learning, Tech. Rep., DTIC Document, 1994.
[105] Harik, G.R., Linkage learning via probabilistic modeling in the ECGA, Tech. Rep. 99010, Illinois Genetic Algorithms Laboratory, 1999.
[106] Mühlenbein, H., Paass, G., From recombination of genes to the estimation of distributions I. Binary parameters, In: International Conference on Parallel Problem Solving from Nature. Springer, 1996, p. 178–187.
[107] De Bonet, J.S., Isbell, C.L., Viola, P., MIMIC: finding optima by estimating probability densities, In: Advances in neural information processing systems. Morgan Kaufmann publishers, 1997, p. 424–430.
[108] Pelikan, M., Mühlenbein, H., The bivariate marginal distribution algorithm, In: Advances in Soft Computing. Springer, 1999, p. 521–535.
[109] Harik, G.R., Lobo, F.G., Goldberg, D.E., The compact genetic algorithm, IEEE T Evolut Comput, 1999, 3(4),287–297.
[110] Mühlenbein, H., Mahnig, T., Rodriguez, A.O., Schemata, distributions and graphical models in evolutionary optimization, J Heuristics, 1999, 5(2),215–247.
[111] Pelikan, M., Goldberg, D.E., Cantu-Paz, E., Linkage problem, distribution estimation, and Bayesian networks, Evol Comput, 2000, 8(3),311–340.
[112] Euchi, J., Hybrid estimation of distribution algorithm for a multiple trips fixed fleet vehicle routing problems with time windows, International Journal of Operational Research, 2014, 21(4),433.
[113] Wang, J., Tang, K., Lozano, J., Yao, X., Estimation of distribution algorithm with stochastic local search for uncertain capacitated arc routing problems, IEEE T Evolut Comput, 2015, 20(c),1–1.
[114] Ceberio, J., Irurozki, E., Mendiburu, A., Lozano, J.A., A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems, Pattern Recogn, 2012, 1(1),103–117.
[115] Gumustekin, S., Senel, T., Cengiz, M.A., A comparative study on Bayesian optimization algorithm for nutrition problem, J Food Nutr Res, 2014, 2(12),952–958.
[116] Escalante, H.J., Ponce-López, V., Escalera, S., Baró, X., Morales-Reyes, A., Martínez-Carranza, J., Evolving weighting schemes for the bag of visual words, Neural Comput Appl, 2016, ,1–15.
[117] Stein, G., Chen, B., Wu, A.S., Hua, K.A., Decision tree classifier for network intrusion detection with GA-based feature selection, In: Proceedings of the 43rd annual Southeast regional conference-Volume 2. ACM, 2005, p. 136–141.
[118] Sörensen, K., Janssens, G.K., Data mining with genetic algorithms on binary trees, Eur J Oper Res, 2003, 151(2),253–264.
[119] Fernández Caballero, J.C., Martinez, F.J., Hervas, C., Gutierrez, P.A., Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks, IEEE T Neural Network, 2010, 21(5),750–770.
[120] Huang, C.L., Wang, C.J., A GA-based feature selection and parameters optimization for support vector machines, Expert Syst Appl, 2006, 31,231–240.
[121] Garrett, D., Peterson, D.A., Anderson, C.W., Thaut, M.H., Comparison of linear, nonlinear, and feature selection methods for EEG signal classification, IEEE T Neur Sys Reh, 2003, 11(2),141–144.
[122] García-Nieto, J., Alba, E., Jourdan, L., Talbi, E., Sensitivity and specificity based multiobjective approach for feature selection: application to cancer diagnosis, Inform Process Lett, 2009, 109(16),887–896.
[123] Yusta, S.C., Different metaheuristic strategies to solve the feature selection problem, Pattern Recogn Lett, 2009, 30(5),525–534.
[124] Aguilera, J.J., Chica, M., del Jesus, M.J., Herrera, F., Niching genetic feature selection algorithms applied to the design of fuzzy rule-based classification systems, In: 2007 IEEE International Fuzzy Systems Conference. IEEE, 2007, p. 1–6.
[125] Xue, B., Cervante, L., Shang, L., Zhang, M., A particle swarm optimisation based multi-objective filter approach to feature selection for classification, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, p. 673–685.
[126] Candelieri, A., A hyper-solution framework for classification problems via metaheuristic approaches, 4OR, 2011, 9(4),425–428.
[127] Yao, X., Evolving artificial neural networks, Proceedings of the IEEE, 1999, 87(9),1423–1447.
[128] Stanley, K.O., Miikkulainen, R., Evolving neural networks through augmenting topologies, Evol Comput, 2002, 10(2),99–127.
[129] Turner, A.J., Miller, J.F., NeuroEvolution: evolving heterogeneous artificial neural networks, Evolutionary Intelligence, 2014, 7(3),135–154.
[130] Carvalho, A.R., Ramos, F.M., Chaves, A.A., Metaheuristics for the feedforward artificial neural network (ANN) architecture optimization problem, Neural Comput Appl, 2011, 20(8),1273–1284.
[131] Das, S., Abraham, A., Konar, A., Metaheuristic pattern clustering–an overview, In: Metaheuristic clustering. Springer, 2009, p. 1–62.
[132] Selim, S.Z., Alsultan, K., A simulated annealing algorithm for the clustering problem, Pattern Recogn, 1991, 24(10),1003–1008.
[133] Shelokar, P.S., Jayaraman, V.K., Kulkarni, B.D., An ant colony approach for clustering, Anal Chim Acta, 2004, 509(2),187–195.
[134] De Jong, K.A., Spears, W.M., Gordon, D.F., Using genetic algorithms for concept learning, In: Genetic algorithms for machine learning. Springer, 1993, p. 5–32.
[135] Chiou, Y.C., Lan, L.W., Genetic clustering algorithms, Eur J Oper Res, 2001, 135(2),413–427.
[136] Garai, G., Chaudhuri, B.B., A novel genetic algorithm for automatic clustering, Pattern Recogn Lett, 2004, 25(2),173–187.
[137] Marinakis, Y., Marinaki, M., Matsatsinis, N., A stochastic nature inspired metaheuristic for clustering analysis, International Journal of Business Intelligence and Data Mining, 2008, 3(1),30–44.

[138] Govindarajan, K., Somasundaram, T.S., Kumar, V.S., Particle swarm optimization (PSO)-based clustering for improving the quality of learning using cloud computing, In: 2013 IEEE 13th International Conference on Advanced Learning Technologies. IEEE, 2013, p. 495–497.
[139] Banu, P.K.N., Andrews, S., Gene clustering using metaheuristic optimization algorithms, International Journal of Applied Metaheuristic Computing, 2015, 6(4),14–38.
[140] Ferone, D., Facchiano, A., Marabotti, A., Festa, P., A new GRASP metaheuristic for biclustering of gene expression data, PeerJ PrePrints, 2016, .
[141] Hruschka, E.R., Campello, R.J., Freitas, A., A survey of evolutionary algorithms for clustering, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2009, 39(2),133–155.
[142] Kurada, R.R., Pavan, K.K., Rao, A., A preliminary survey on optimized multiobjective metaheuristic methods for data clustering using evolutionary approaches, International Journal of Computer Science & Information Technology, 2013, 5.
[143] Carvalho, D.R., Freitas, A.A., A genetic algorithm for discovering small disjunct rules in data mining, Appl Soft Comput, 2002, 2(2),75–88.
[144] Freitas, A., Data mining and knowledge discovery with evolutionary algorithms, Advances in Evolutionary Computation, 2002, 105,819–845.
[145] Khabzaoui, M., Dhaenens, C., Talbi, E.G., A multicriteria genetic algorithm to analyze DNA microarray data, In: Cec2004: Proceedings of the 2004 Congress on Evolutionary Computation, Vols 1 and 2. 2004, p. 1874–1881.
[146] Khabzaoui, M., Dhaenens, C., Talbi, E.G., Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery, RAIRO Operations Research, 2008, 42(1),69–83.
[147] Ishida, C.Y., Pozo, A., Goldbarg, E., Goldbarg, M., Multiobjective optimization and rule learning: subselection algorithm or meta-heuristic algorithm?, In: Innovative applications in data mining. Springer, 2009, p. 47–70.
[148] Yang, S., Jiang, Y., Nguyen, T.T., Metaheuristics for dynamic combinatorial optimization problems, IMA Journal of Management Mathematics, 2013, 24(4),451–480.
[149] Juan, A.A., Faulin, J., Jorba, J., Riera, D., Masip, D., Barrios, B., On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics, J Oper Res Soc, 2011, 62(6),1085–1097.
[150] Clarke, G., Wright, J.W., Scheduling of vehicles from a central depot to a number of delivery points, Oper Res, 1964, 12(4),568–581.