Chapter 2

Hybrid Metaheuristics for Classification Problems

Nadia Abd-Alsabour

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/65253

Abstract

High accuracy and short amount of time are required for the solutions of many classification problems such as real-world classification problems. Due to the practical importance of many classification problems (such as crime detection), many algorithms have been developed to tackle them. For years, metaheuristics (MHs) have been successfully used for solving classification problems. Recently, hybrid metaheuristics have been successfully used for many real-world optimization problems such as flight scheduling and load balancing in telecommunication networks. This chapter investigates the use of this new interdisciplinary field for classification problems. Moreover, it demonstrates the forms of metaheuristics hybridization as well as designing a new hybrid metaheuristic.

Keywords: metaheuristics, hybrid metaheuristics, classification problems

1. Introduction

Before starting this chapter, let us know the trip that led to the appearance of hybrid metaheuristics. Traditionally, rigorous approaches (that are based on hypotheses, characterizations, deductions, and experiments) were used for solving many optimization problems.

However, in order to find possible good solutions for new complex optimization problems, researchers went toward the use of heuristics. Heuristics are rules of thumb, trails and error, common sense, etc. Many of these heuristics strategies are often independent of the undertaken optimization problems and share common aspects. This introduced the term metaheuristics which refers to general techniques that are not specific to a particular problem [1]. Metaheuristics are approximate algorithms, and each of them has its own historical background [2–4]. A metaheuristic is a set of algorithmic concepts used for defining heuristic methods that can

be applied to a variety of optimization problems with relatively few modifications in order to adapt them to particular optimization problems [5, 6].

Metaheuristics have successfully found high-quality solutions for a wide spectrum of NP hard optimization problems [1], that is, they are hard to be solved. This means the needed time to solve an instance of these optimization problems grows exponentially with the instance size in the worst case. These optimization problems are so complex as there is no known algorithm that can solve them in polynomial time. They still have to be solved in a huge number of practical settings. Therefore, a large number of optimization algorithms were proposed to tackle them [5, 6].

Of great importance for the success of designing a new metaheuristic is considering that this metaheuristic will have to explore the search space effectively and efficiently. The search process should be intelligent in order to intensively explore areas of the search space that have high-quality solutions and to move to unexplored areas. This is called intensification and diversification, respectively. Intensification is the exploitation of the gathered information by the metaheuristic at a given time, while diversification is the exploration of the areas imperfectly taken into account. The use of these two important characteristics of a metaheuristic can lead to getting high-quality solutions. Crucial for the success of a metaheuristic is a well-adjusted balance between these two features. This is to on one side identify quickly search areas with high-quality solutions and on the other side to avoid spending too much time in areas consisting of poor-quality solutions or have been well explored [1, 7–11].

There are many classifications for metaheuristics as follows:

- Nature-inspired metaheuristics [such as ant colony optimization (ACO) algorithms, genetic algorithms (GAs), particle swarm optimization (PSO), and simulated annealing (SA)] vs. nonnature-inspired metaheuristics [such as iterated local search (ILS), and tabu search (TS)]. This is based on the origins of a metaheuristic.

- Memory-based metaheuristics vs. memory-less metaheuristics. This is based on the use of the search history, that is, whether they use memory or not. The use of memory is considered one of the crucial elements of a powerful metaheuristic.

- Population-based metaheuristics vs. single-point metaheuristics. This is based on how many used solutions at any given time by a metaheuristic. Population metaheuristics manipulate a set of solutions (at each iteration) from which the population of the next iteration is produced. Examples are evolutionary algorithms and scatter search, and construction-oriented techniques such as ant colony optimization and the greedy randomized adaptive search procedure. The metaheuristics that deal with only one solution at any given time are called trajectory metaheuristics where the search process describes a trajectory in the search space [1, 2, 4] as shown in **Figure 1** [12].

When they first appeared, pure metaheuristics quickly became state-of-the-art algorithms for many optimization problems as they found high-quality solutions for these optimization problems. This was reported in many specific conferences and workshops. This success had motivated researches toward finding answers to questions such as:

http://dx.doi.org/10.5772/65253

- Why a given metaheuristic is successful?
- Which characteristics of a problem instance should be exploited?
- Which metaheuristic is best for a given optimization problem? [1, 2]

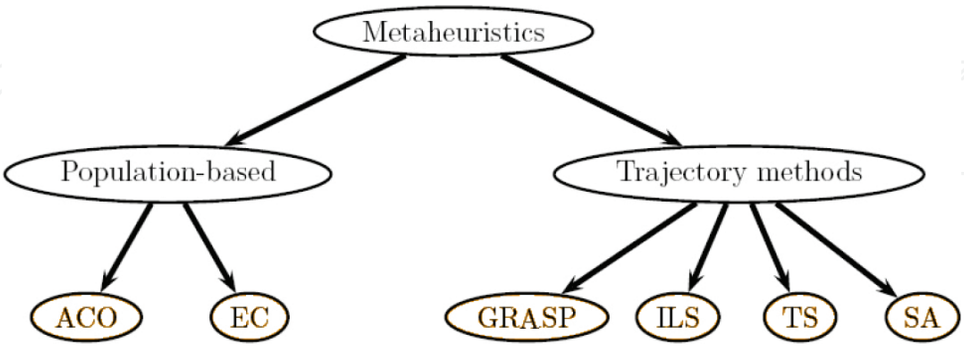

Figure 1. Metaheuristics classification [12].

Despite this success, it became recently evident that the focus on pure metaheuristics is restrictive when tackling particular optimization problems such as real-world and large-scale optimization problems [2]. A skilled combination of a metaheuristic with components from other metaheuristics or with other optimization algorithms such as operations research techniques (mathematical programming), artificial intelligence techniques (constraint programming), or complete algorithms (branch and bound) can lead to getting much better solutions for these optimization problems. This interdisciplinary field is called hybrid metaheuristics which goes beyond the scope of a pure metaheuristic [1]. Over the years, many algorithms that do not purely follow the paradigm of a pure metaheuristic were developed. They combine various algorithmic components originating from different optimization algorithms [2]. This is explained in Section 3.

The rest of this chapter is organized as follows. The following section introduced classification problems. Section 3 explains the main forms of hybridizing metaheuristics. Section 4 demonstrates designing a hybrid metaheuristic. The fifth section demonstrates hybrid metaheuristics for classification problems. The discussion is given in Section 6. The last section concludes this chapter and highlights future work in this area.

2. Classification problems

Classification involves training and testing data which consist of data instances (objects). Each instance in the training set contains one class label (called target, dependent, response, or features) and other features (called attributes, inputs, predictors, or independent features) [13–15]. Classification consists of examining the features of a new object and then assigning it to one of the predefined set of classes. The objects to be classified are generally represented by records in a dataset. The classification task is to build a model that will be applied to unclas-

sified data to classify it, that is, predicting the target values of instances (that are given only the input features) in the testing set [15, 16]. The classification task (determining which of the fixed set of classes an example belongs to) is illustrated in **Figure 2**.

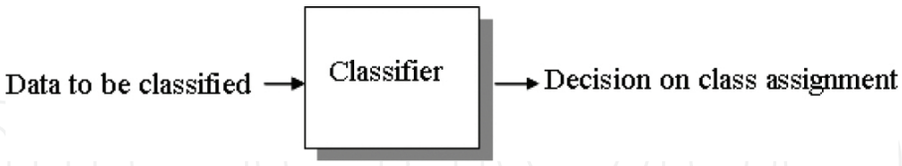

Figure 2. The classification task.

Examples of classification problems are:

- classifying credit applications such as low, medium, or high risky,
- determining whether a customer with a given profile will buy a new computer,
- predicting which of three specific treatments a breast cancer patient should receive,
- determining whether a will was written by the real person or somebody else,
- diagnosing whether a particular illness is present or not,
- choosing particular contents to be displayed on a web page,
- determining which phone numbers correspond to fax machines,
- placing a new student into a particular track based on special needs,
- identifying whether a behavior indicates a possible terrorist threat, and
- spotting fraudulent insurance claims.

In these examples, the classifier is built to predict categorical labels such as "low risky," "medium risky," or "high risky" for the first example; "yes" or "no" for the second example; "treatment A," "treatment B," or "treatment C" for the third example, etc. [16–18].

The accuracy of a classifier refers to how well it can predict the value of the predictor feature for a previous unseen data and how well it captured the dependencies among the input features. Classifier accuracy is the main measure for classification and is widely used. The classifier accuracy goes up when comparing between different classifiers [18–20].

The classifier is considered the basic component of any classification system, and its task is to partition the feature space into class-labeled decision regions (one for each category). Classifiers' performance is sensitive to the choice of the features that are used for constructing those classifiers. This choice affects the accuracy of these classifiers, the time needed for learning, and the number of examples needed for learning. Feature selection (FS) can be seen as an optimization problem that involves searching the space of possible solutions (feature subsets) to identify the optimal one. Many metaheuristics (such as ant colony optimization algorithms, particle swarm optimization, genetic algorithms, simulated annealing, and tabu search) have been used for solving the feature selection problem [20, 21].

http://dx.doi.org/10.5772/65253

Feature selection (deleting a column from a dataset) is one of the main aspects of dimension reduction besides instance reduction (deleting a row from a dataset). This is illustrated in **Figure 3** [18]. Both of these should keep the characteristics of the original input data after excluding some of it.

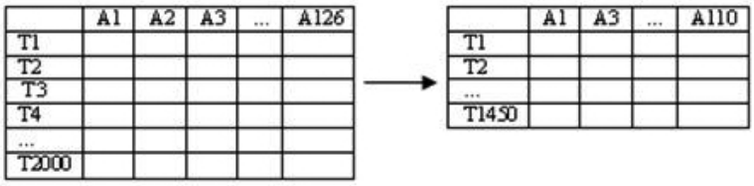

Figure 3. Data reduction [18].

Figure 4 [22] illustrates the revised classification with the use of dimension reduction phase as an intermediate step. In **Figure 4**, dimension reduction is performed first to the given data, and then, the prediction methods are applied to the reduced data.

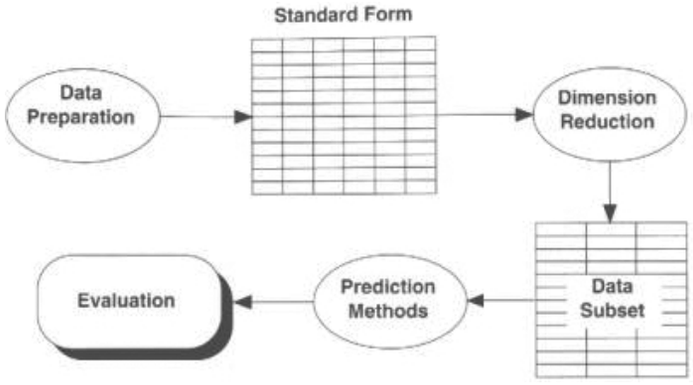

Figure 4. The role of dimension reduction [22].

3. Hybridization of metaheuristics

Although combining different algorithms together dates back to 1980s, in recent years only hybrid metaheuristics have been commonly used. Then, the advantage of combining different algorithms together has been widely recognized [1, 4]. Forms of hybridization can be classified into two categories (as in **Figure 5**): (1) combining components from a metaheuristic into another metaheuristic (examples are: using trajectory methods into population algorithms or using a specific local search method into a more general trajectory algorithm such as iterated local search) and (2) combining metaheuristics with other techniques such as artificial intelligence and operations research (examples are: combining metaheuristics with constraint programming (CP), integer programming (IP), tree-based search methods, data mining techniques, etc.) [1]. The following two subsections explain these types.

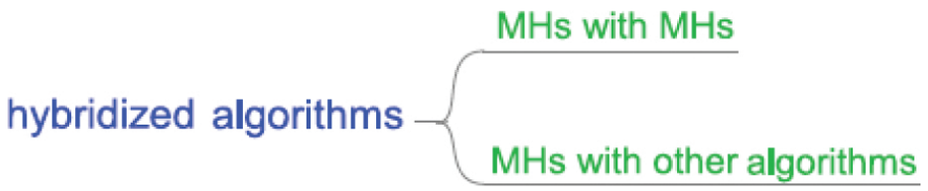

Figure 5. Forms of hybridization [4].

3.1. Hybridizing metaheuristics with metaheuristics

This category represents the beginning of hybridizing metaheuristics. Later, it got widely used especially integrating nature-inspired metaheuristics with local search methods. This is well illustrated in the most common type of this category which is in ant colony optimization algorithms and evolutionary algorithms that often use local search methods in order to refine the generated solutions during the search process. The reason for that is these nature-inspired metaheuristics explore well the search space and identify the regions having high-quality solutions (since they first capture a global picture of the search space and then they successively focus the search toward the promising regions). However, these nature-inspired metaheuristics are not effective in exploiting the accumulated search experiences that can be achieved by adding local search methods into them. Therefore, the resulting hybrid metaheuristic will work as follows: the nature-inspired metaheuristic will identify the promising search areas from which the local search method can then determine quickly the best solutions. Based on the above–mentioned fact, the resulting hybrid metaheuristic combining the strengths of both metaheuristics is often very successful. Apart from this hybridization, there are other hybrids. We mentioned it only here as it is considered the standard way of hybridization [1, 2].

3.2. Hybridizing metaheuristics with other algorithms

There are many possible ways of integration between metaheuristics and other algorithms. For example, metaheuristics and tree search methods can be interleaved or sequentially applied. This can be achieved by using a tree search method for generating a partial solution that a metaheuristic can then complete. Alternatively, a metaheuristic improves a solution generated by a tree search method. Another example is that constraint programming techniques can be used to reduce the search space (or the neighborhoods) that will be explored by a local search method [1, 4].

It should be noted that all of the hybrid metaheuristics mentioned above are integrative combinations in which there is some kind of master algorithm including one or more subordinate components (either embedded or called). Another way of combinations is called either collaborative or cooperative combinations in which the search is performed by different algorithms that exchange information about states, models, entire subproblems, solutions, or search space characteristics. The cooperative search algorithms consist of parallel execution of search algorithms that can be different or instances of the same algorithm working on different models or running with different parameter settings. Therefore, the control strategy in hybrid

http://dx.doi.org/10.5772/65253

metaheuristics can be integrative or collaborative, and the order of executing the combined parts can be sequential, parallel, or interleaved [1, 4, 12]. These are shown in **Figures 6** and **7**.

Figure 6. The control strategy in hybrid metaheuristics [4].

Figure 7. The order of executing the combined algorithms in hybrid metaheuristics [4].

4. Designing a hybrid metaheuristic

The main motivation behind combining various algorithmic ideas from different metaheuristics is to get better performing system that exploits and includes advantages of the combined algorithms [3, 4]. These advantages should be complementary to each other so that the resulting hybrid metaheuristic can benefit from them [2, 3, 23]. The key to achieving high performance in the resulting hybrid metaheuristic (especially when tackling hard optimization problem) is to choose suitable combinations of complementary algorithmic concepts. Therefore, this task of developing a highly effective hybrid metaheuristic is complicated and not easy [3]. The reasons for that are as follows:

1. It requires creative thinking and the exploration of new research directions.
2. Designing and implementing a hybrid metaheuristic involves wide knowledge about algorithms, data structure, programming, and statistics [3].
3. It requires expertise from different optimization areas [2].
4. It should include exploration and intensification capabilities.

According to Blum et al. [2], before starting to develop a hybrid metaheuristic, we should consider whether it is the appropriate choice for the given optimization problem. This can be achieved by answering the following questions:

- What is the optimization objective? Do we need a reasonable good solution? And whether this solution is needed very quickly or not? Or we can sacrifice the computation time in

order to get a very good solution? (these questions in general guide us toward using metaheuristics or complete methods) In this case, when very good solution is needed and it cannot be obtained by the existing complete algorithms in reasonable time, then we need to know the answer of the next question in order to decide on developing a hybrid metaheuristic.

- Is there any existing metaheuristic that can get the required solution for the given optimization problem? If no, can we enhance any of the existing metaheuristics to better suit this optimization problem? If no, then the decision is to develop a hybrid metaheuristic and we will need to know the answer of the following questions.
- Which hybrid metaheuristic will work well for this optimization problem? Unfortunately, till now, there is no answer to this question as it is difficult to set guidelines for developing a well-performing hybrid metaheuristic, but the following can help:
  - Searching the literature carefully for the most successful optimization algorithms for the given optimization problem or for similar optimization problems, and
  - Studying different ways of combining the most promising characteristics of the selected optimization algorithms to be combined [2, 3].
- Identifying special characteristics of the given optimization problem and finding effective ways in order to exploit them [4].

Besides, in order to set guidelines for developing a new hybrid metaheuristic, it is crucial to improve the used research methodology that consists of combining different algorithmic components without identifying the contributions of these components to the performance of the resulting hybrid metaheuristic. The used methodology should consist of theoretical models for the characteristics of the hybrid metaheuristics. It can be experimental such as those used in natural sciences. Moreover, testing and statistical assessment of the obtained results should be included as well [2].

5. Hybrid metaheuristics for classification problems

The first category of using hybrid metaheuristics for classification problems concerns with using a metaheuristic for the feature selection problem besides the used classifier. This is because selecting the most relevant set of input features to use for building the used classifier plays an important role in classification. The most common metaheuristics for the feature selection problem are genetic algorithms, ant colony optimization algorithms, and particle swarm optimization algorithms [24] which are hybridized with the used classifier in each application. This is explained below.

The feature selection problem is used in many applications from choosing the most important social-economic parameters in order to identify who can return a bank loan to dealing with a chemical process and selecting the best set of ingredients. It is used to simplify the datasets by eliminating irrelevant and redundant features without reducing the classification accuracy.

Examples of these applications are: face recognition, speaker recognition, face detection, bioinformatics, web page classification, and text categorization [24, 25].

The idea of using genetic algorithms for solving optimization problems is that they start with a population of individuals each of which represents a solution to the given optimization problem. Initially, the population includes all randomly generated solutions (the first generation of the population). Then, the various genetic operators are applied over the population to produce a new population. Within a population, the goodness (measured by a fitness function) of a solution varies from individual to individual [26].

Genetic algorithms are one of the most common approaches for the feature selection problem. The usual usage is to use them for first selecting the most relevant features (from the given dataset) that will be used for building the used classifier. Examples are the work of Yang and Honavar [27] and Tan et al. [28].

There are other directions for using genetic algorithms for the feature selection problem, for instance, hybridizing the used genetic algorithm with another metaheuristic in order to select the most appropriate feature subset before building the given predictor (such as Oh et al. [29] who embedded local search into the used genetic algorithm). Another example is the work of Salcedo-Sanz et al. [30] who used extra genetic operator in order to fix (in each iteration) the number of features to be chosen out of the available ones.

Similar to the way of using genetic algorithms for the feature selection problem is the use of ant colony optimization algorithms that have been widely used for this optimization problem. Examples are the work of Yang and Honavar [27] and the work of Abd-Alsabour and Randall [31].

There are other ways for using ant colony optimization algorithms for the feature selection problem. An example is the work of Vieira et al. [32] who used two cooperative artificial ant colonies: one for determining the number of features to be selected and the second one for selecting the features based on the cardinality given by the first colony. Another direction is to use ensemble (more than one classifier is built and then is combined to produce a better classification—this is called ensemble techniques [33]) of classifiers to perform the classification besides the used metaheuristic for the feature selection problem.

Another metaheuristic that has also been used for the feature selection problem is particle swarm optimization. Researchers developed variants of PSO in order to be suitable for the feature selection problem such as the work of Chuang et al. [34] who proposed a variant of PSO called complementary PSO (CPSO) with the use of k-nearest neighbor classifier. Another example is Zahran and Kanaan [35] who implemented a binary PSO for feature selection. Also, Jacob and Vishwanath [36] proposed multi-objective PSO that outperformed a multi-objective GA in the same authors' experiments. Moreover, Yan et al. [37] proposed a new discrete PSO algorithm with a multiplicative likeliness enhancement rule for unordered feature selection. Also, Sivakumar and Chandrasekar [38] developed a modified continuous PSO for the feature selection problem with k-nearest neighbor classifier that served as a fitness function for the PSO.

There are other ways for using particle swarm optimization algorithms for the feature selection problem. An example is the work of Wahono and Suryana [39] who used a combination of PSO and a bagging of classifiers (bagging is an ensemble technique where many classifiers are built and the final classification decision is made based on voting of the committee of the combined classifiers. It is used in order to improve the classification accuracy [33]). Another example is the work of Nazir et al. [40] who combined a PSO and a GA to perform together the feature selection.

The classification task involves other subtasks besides the feature selection problem, and many metaheuristics have been used for solving these subtasks, for example, the use of ant colony optimization for designing a classifier ensemble such as the work of Palanisamy and Kanmani [41] who used the main concepts of the proposed ant algorithm in Abd-Alsabour and Randall [31] for designing an ensample of classifiers. Another example is the use of particle swarm optimization algorithms for producing good classification rules such as Kumar [42] who combined a PSO with a GA to produce them and Holden and Freitas [43] who later proposed several modifications to their proposed work in Holden and Freitas [44]. Another example is the work of Revathil and Malathi [45] who proposed a novel simplified swarm optimization algorithm as a rule-based classifier.

6. Discussion

The previous section closely explored the different ways to use hybrid metaheuristics for classification problems. In the light of that, we can come up with the following comments:

1. For solving many applications, using hybrid metaheuristics was crucial to get high-quality solutions especially for real-world applications (such as personnel and machine scheduling, educational timetabling, routing, cutting and packing, and protein alignment). An example for real-world classification problems is the work of Tantar et al. [46], who developed a hybrid metaheuristic (GA and SA) for predicting the protein structure. Examples for other real-world optimization problems are the work of Atkin et al. [47], who proposed a hybrid metaheuristic for runway scheduling at London Heathrow airport and the work of Xu and Qu [48], who used a hybrid metaheuristic to solve routing problems.
2. However, there are other situations where the hybridization was not important for the prediction accuracy. An example is the use of extra metaheuristic (besides the two algorithms used: one for performing feature selection and the classifier) to determine the number of the features to be selected. Similar to this is the use of two instances of a metaheuristic: one to determine the number of the features to be selected and the second one to perform the feature selection. These two scenarios can lead to worse results besides the extra computation cost. The authors should have been avoided using extra metaheuristic or an instance of the used metaheuristic. The reason for that is revealed from the work of Abd-Alsabour et al. [49] who proved that fixing the length of the selected feature subsets can lead to getting worse classification accuracy than not fixing the length of the selected

feature subsets (besides its extra computation). We should avoid selecting too few or too many features than necessary. This is because selecting insufficient features leads to degrading the information content to keep the concept of data. On the other side, if too many features are selected, the classification accuracy will decrease because of the interference of irrelevant features. Subset problems such as the feature selection problem do not have fixed length [49]. Another example is the use of more than one classifier (ensemble methods) rather than using one only. This is because of the extra computational cost, especially that there were already previous similar applications that had been successfully solved using only one classifier (besides the used metaheuristic for getting the best feature subset). This has been evidenced by many authors when they compared their work with the pervious ones and showed that their results were not better than the others. One more example is the use of two metaheuristics for performing the feature selection, while it was already solved using only one metaheuristic. These examples emphasize the fact that sufficient literature search before first hybridizing or adding extra computational steps can avoid extra computation, useless hybridization, or even moving toward a misleading research direction as illustrated in Section 4.

Therefore, choosing the suitable hybrid metaheuristic can achieve the top performance for many optimization problems, but this does not imply that more complex algorithms are always the best choice. This is because of the following disadvantages of the increased complexity:

- The software becomes more difficult to tune and maintain.
- Adaptations in problem specifications are frequently hard to adhere.

Therefore, an important design aim is to keep the proposed algorithm as simple as possible and include extensions only if they will really benefit [4].

Despite the difficulties in developing a new hybrid metaheuristic, it is nontrivial to generalize it, that is, a particular hybrid metaheuristic that works well for a particular optimization problem might not work well for another problem. This means that research on hybrid metaheuristics has gone toward being problem-specific rather than algorithm-oriented as was when promoting a new metaheuristic [1, 2].

7. Conclusions and future work

This chapter addressed the use of hybrid metaheuristics for classification problems. Besides, it demonstrated hybridizing metaheuristics and designing them as well. Moreover, the most common used hybrid metaheuristics for classification problems from literature were presented.

As a research direction, more applications of hybrid metaheuristics for different optimization problems in general and more particularly for real-word classification problems will be considered. Another research direction is to move more toward setting specific methodologies and general guidelines for developing a new hybrid metaheuristic. Moreover, comparisons between hybrid metaheuristics for similar classification problems should be conducted.

Author details

Nadia Abd-Alsabour

Address all correspondence to: nadia.abdalsabour@cu.edu.eg

Cairo University, Cairo, Egypt

References

[1] Blum C, and Roli A. Hybrid Metaheuristics: An Introduction. In: Blum C, Aguilera M, Roli A, and Sampels M, editors. Hybrid Metaheuristics – An Emerging Approach to Optimization. Studies in Computational Intelligence. Springer-Verlag Berlin Heidelberg, Springer; 2008. 114, p.1–30.

[2] Blum C, Puchinger J, Raidl G, and Roli A. Hybrid metaheuristics in combinatorial optimization: A survey. Applied Soft Computing. 2011; 11: 4135–4151.

[3] Blum C, Puchinger J, Raidl G, Roli A. A brief survey on hybrid metaheuristics. In: Filipic B, Silc J, editors. Proceedings of BIOMA 2010 – 4th International Conference on Bio-Inspired Optimization Methods and their Applications, Jozef Stefan Institute, Ljubljana, Slovenia, 2010, pp. 3–18.

[4] Raidl G. Decomposition based hybrid metaheuristics. European Journal of Operational Research. 2015; 244: 66–76

[5] Dorigo M, and Stutzle T. Ant Colony Optimization. MIT Press, Cambridge, MA; 2004.

[6] Blum C. Ant colony optimization: Introduction and recent trends. Physics of Life Reviews. 2005; 2: 353–373.

[7] Blum C, and Roli A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys. 2003; 35 (3): 268–308.

[8] Méndez P. Development of a hybrid metaheuristic for the efficient solution of strategic supply chain management problems: Application to the energy sector. M.Sc. Thesis. Polytechnic University of Catalonia, Catalonia, Spain, 2011

[9] Blum C. ACO Applied to Group Shop Scheduling: A Case Study on Intensification and Diversification. In: Dorigo M, et al., editors. Lecture Notes in Computer Science; 2463: p. 14–27, Springer. 2002.

[10] Randall M, and Tonkes A. Intensification and diversification strategies in ant colony search. Technical Report TR00-02, School of information technology, Bond University, Australia. 2002

[11] Abd-Alsabour N, and Moneim A. Diversification with an ant colony system for the feature selection problem. In Proceeding of 2nd International Conference on Management and Artificial Intelligence. IPEDR; 35: 35–39. IACSIT Press. 2012.

[12] Blum C. Hybrid metaheuristics. BIOMA 2010. Ljubljana, Slovenia.

[13] Hand D, Mannila H, and Smyth P. Principles of Data Mining. Massachusetts Institute of Technology, Cambridge, MA; 2001.

[14] Hastie T, Tibshirani R, and Friedman J. The Elements of Statistical Learning – Data Mining, Inference, and Prediction. Springer Verlag, Springer; 2009.

[15] Xie Y. An introduction to support vector machine. Available from: http://yihoi.name/cv/images/svm_Report_yihui.pdf. 2007. [Accessed: 2016-06-25]

[16] Berry M, and Linoff G. Data Mining Techniques for Marketing, Sales, and Customer Relationship Management. 2nd ed. Wiley, New York; 2004.

[17] Larose D. Discovering Knowledge in Data. John Wiley & Sons; Hoboken, New Jersey; 2005.

[18] Han J, and Kamber M. Data Mining Concepts and Techniques. 1st ed. Morgan Kaufmann Publishers, San Francisco; 2001.

[19] Skillicorn D. Understanding Datasets: Data mining with Matrix Decomposition. Chapman & Hall/CRC, London; 2007.

[20] Yang J, and Honavar V. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems. 1998; 13(2): 44–49.

[21] Duda R.O., and Hart P.E. Pattern Classification and Scene Analysis. 1st ed. John Wiley & Sons, New York; 1973.

[22] Weiss S, and Indurkhya N. Predictive Data Mining, a Practical Guide. Morgan Kaufmann Publishers, San Francisco; 1998.

[23] Al-Duolia F, Rabadia G. Data mining based hybridization of meta-RaPS. Procedia Computer Science. 2014; 36: 301–307

[24] Rokach L, and Maimon O. Data Mining with Decision Trees. World Scientific Publishing, New Jersey; 2008.

[25] Abd-Alsabour N. A review on evolutionary feature selection. 2014 UKSim-AMSS 8th European Modeling Symposium. Italy. 2014. p. 20–26

[26] Shukla A, Tiwari R, and Kala R. Real Life Applications of Soft Computing. CRC Press, Boca Raton; 2010.

[27] Yang J, and Honavar V. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems. 1998; 13(2): 44–49.

[28] Tan K C, Teoh E J, Yu Q, and Goh K.C. A hybrid evolutionary algorithm for attribute selection in data mining. Expert Systems with Applications. 2009; 36: 8616–8630

[29] Oh I, Lee J, and Moon B. Hybrid GAs for feature selection, IEEE Transactions on Pattern Analysis and Machine Intelligence. 2004; 26(11): 1424–1437.

[30] Salcedo-Sanz S, Camps-Valls G, and Pérez-Cruz F. Enhancing genetic feature selection through restricted search and Walsh analysis. IEEE Transactions on Systems, Man, and Cybernetics. Part C. 2004. 34(4):398–406.

[31] Abd-Alsabour N, and Randall M. Feature selection for classification using an ant colony system. In Proceedings of the 6th IEEE International Conference on e–Science Workshops. IEEE Press, 2010; p. 86–91.

[32] Vieira S M, Sousa J MC, and Runkler T A. Two cooperative ant colonies for feature selection using fuzzy models. Expert Systems with Applications; 2010. 37:2714–2723.

[33] Lui B. Web Data Mining. Springer; 2010.

[34] Chuang L, Jhang H, and Yang C. Feature selection using complementary PSO for DNA micro-array data. In Proceedings of the International Multi-Conference of Engineers and Computer Scientists, Hong Kong; 2013; 1. p. 291–294.

[35] Zahran B M, and Kanaan G. Text feature selection using particle swarm optimization algorithm. World Applied Sciences Journal. 2009; 7: pp. 69–74.

[36] Jacob M, and Vishwanath N. Multi-objective evolutionary PSO algorithm for matching surgically altered face images. International Journal of Emerging Trends in Engineering and Development. 2014; 2(4): 640–648.

[37] Yan Y, Kamath G, and Osadciw L. Feature selection optimized by discrete particle swarm optimization for face recognition. In Proceedings of Optics and Photonics in Global Homeland Security and Biometric Technology for Human Identification. SPIE. 2009; 7306.

[38] Sivakumar S, and Chandrasekar C. Modified PSO based feature selection for classification of lung CT images. International Journal of Computer Science and Information Technologies. 2014; 5(2): 2095–2098.

[39] Wahono R, and Suryana N. Combining PSO based feature selection and bagging technique for software defect prediction. International Journal of Software Engineering and Its Applications. 2013; 7(5):153–166.

[40] Nazir M, Majid-Mirza A, and Ali-Khan S. PSO-GA based optimized feature selection using facial and clothing information for gender classification. Journal of Applied Research and Technology. 2014; 12:145–152.

[41] Palanisamy S, and Kanmani S. Artificial Bee Colony Approach for Optimizing Feature Selection. International Journal of Computer Science Issues. 2012; 9(3):432–438.

[42] Kumar K. Intrusion Detection system for malicious traffic by using PSO-GA algorithm. IJCSET. 2013; 3(6):236–238.

[43] Holden N, and Freitas A A. A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data. In Proceedings of 2005 IEEE Swarm Intelligence Symposium (SIS-05). 2005; p. 100–107.

[44] Holden N, and Freitas A A. A hybrid PSO/ACO algorithm for classification. GECCO'07, July 7–11, 2007, London, United Kingdom.

[45] Revathi1 S, and Malathi A. Network intrusion detection using hybrid simplified swarm optimization technique. International Journal of P2P Network Trends and Technology. 2013; 3(8): 375–379.

[46] Tantar A, Melab N, Talbi E. A grid-based genetic algorithm combined with an adaptive simulated annealing for protein structure prediction. Soft Computing. 2008; 12:1185–1198

[47] Atkin J, Burke E, Greenwood J, Reeson D. Hybrid metaheuristics to aid runway scheduling at London Heathrow Airport. Transportation Science. 2007; 41(1):90–106.

[48] Xu Y, Qu R. Solving Multi-objective multicast routing problems by evolutionary multi-objective simulated annealing algorithms with variable neighborhoods. Journal of the Operational Research Society. 2011; 62:313–325.

[49] Abd-Alsabour N, Randall M, and Lewis A. Investigating the effect of fixing the subset length using ant colony optimization algorithms for feature subset selection problems. In Proceedings of the 13th International Conference on Parallel and Distributed Computing: Applications and Technologies. Beijing, China. December 2012. IEEE Press. p. 733–738.

IntechOpen