Many-Objective Cooperative Co-Evolutionary Feature Selection: A Lexicographic Approach* **

Jesús González[0000−0002−0415−1821], Julio Ortega[0000−0002−2998−220X], Miguel Damas[0000−0003−2599−8076], and Pedro Martín-Smith[0000−0001−9561−4125]

Department of Computer Architecture and Technology, CITIC,
University of Granada, Granada, Spain
{jesusgonzalez,jortega,mdamas,pmartin}@ugr.es
http://atc.ugr.es/

Abstract. This paper presents a new wrapper method able to optimize simultaneously the parameters of the classifier while the size of the subset of features that better describe the input dataset is also being minimized. The search algorithm used for this purpose is based on a co-evolutionary algorithm optimizing several objectives related with different desirable properties for the final solutions, such as its accuracy, its final number of features, and the generalization ability of the classifier. Since these objectives can be sorted according to their priorities, a lexicographic approach has been applied to handle this many-objective problem, which allows the use of a simple evolutionary algorithm to evolve each one of the different sub-populations.

Keywords: Many-objective evolutionary algorithm · Cooperative co-evolutionary algorithm · Lexicographic optimization · Feature selection · Wrapper approach.

1 Motivation

Wrapper methods are intrinsically simple, what has made them quite popular. They basically consist of a classifier, a search algorithm, and way to assess the prediction accuracy of the learning machine in order to guide the search towards good feature subsets [19]. Besides, since wrapper methods select the features subset according to the classifier that will be applied later on to the test set, they usually achieve better accuracy than filter methods, although they are also more computationally expensive [15].

** This work was supported by projects TIN2015-67020-P (Spanish "Ministerio de Economía y Competitividad") and PGC2018-098813-B-C31 (Spanish "Ministerio de Ciencia, Innovación y Universidades"), and by European Regional Development Funds (ERDF).

Regarding the classifier, Support Vector Machines (SVM) have been widely used within wrappers [22]. However, the functioning of this classifier rely on some parameters that must be fine tuned, according to the dataset, in order to achieve a good accuracy. SVM relies on the regularization parameter C and also on the set of parameters that define the type of kernel used. The correct initialization of these parameters is of capital importance, mainly because the final result of the wrapper method will depend on them. The problem here is that the value of these parameters depend on the final dataset defined by the selected features, which is a priori unknown.

Some approaches fix the parameters of the classifier heuristically before the wrapper method is applied. For example, in [22] these parameters were optimized with the whole training set (containing all the features) before the application of the wrapper procedure. However, these parameters might not be optimal for the final feature subset found by the wrapper algorithm. Besides, different values for the parameters could have provided a different feature subset.

Since the parameters of the classifier depend on the final feature subset, and this subset is the result of the search algorithm applied within the wrapper method, this paper proposes that the classifier parameters should be simultaneously optimized while the search algorithm is finding the best subset of features. That is, two interdependent problems should be simultaneously optimized: the parameters of the classifier, which depend on the subset of features used, and the best subset of features, which depend on the classifier used within the wrapper method. Cooperative Co-Evolutionary algorithms (CCEAs) are particularly well suited to this scenario, since they have been designed to evolve different species of solutions simultaneously [27].

On the other hand, some objectives should be taken into account in order to guide the search towards a good couple of classifier and subset of features. First of all, the classification error and generalization capability should be optimized, since the aim of this work is to find the subset of features that best describe the original dataset. Another objective that many approaches take into account is the number of features, that should also be minimized. Lastly, and since the classifier parameters are also going to be optimized, some objectives could also be defined. For example, in the case of the SVM classifier, the minimization of C is preferred, since lower values of C avoid over-fitting and also speed up the training and test processes. Thus, this is a Many-Objective Optimization Problem (MaOP) too, that is, a Multi-Objective Problem (MOP) with more than tree objectives [10].

Evolutionary algorithms (EAs) have also been widely applied to solve MaOPs [20]. However, and although there are several approaches to design Many-Objective EAs (MaOEAs), such as Pareto-based, indicator-based or aggregation-based approaches, all of them have been designed to treat all the objectives equally. That is, all the objectives in the problem have the same level of importance. Although this is the case in most MaOPs, not all the objectives have the same priority in the case of feature selection problems. For example, given two possible solutions for the problem, the one with a lower classification error is preferred, and only

when the classification error for both is similar, the number of features should be considered. Thus, a priority-based scheme should be incorporated into the algorithm to guide the search towards adequate solutions.

Although some attempts have been made to support objective priorities in EAs [9, 29], a much simpler approach is possible if the problem allows to set a different priority level for each objective. This approach, which was initially proposed in 1975 [25], is lexicographic optimization, and problems meeting this restriction are also known as Lexicographic MOPs (LMOPs) [18]. Lexicographic optimizers try to meet all the objectives sequentially. First, the most important objective is optimized. Then, among the solutions meeting this objective, a smaller set of solutions is chosen to satisfy the second objective, and so on until all the objectives have been considered [1]. Although it may seem a quite simple approach, there are relevant LMOPs that have been successfully solved with it, even at the present time, such as the design and optimization of integrated vehicle control systems [17] or the design of autonomous vehicles [28].

There exist many analytical algorithms for LMOPs, but all of them impose restrictions, such as the differentiability and convexity of the objective functions [33]. Thus, for any general problem not meeting these constraints a more robust optimization technique is necessary. On the other hand, Multi-Objective EAs (MOEAs) have been applied successfully to MOPs where analytical approaches have failed [5]. Therefore they can also be used to solve any kind of LMOP, even with not convex and not continuous objectives. This is the approach presented in this paper, a Lexicographic Many-Objective Cooperative Co-Evolutionary Algorithm (LeMaOCCEA) to simultaneously optimize the parameters of the classifier within a wrapper method while the number of features is also being minimized.

The rest of the paper is organized as follows. Section 2 describes the lexicographic relation for MaOEAs, a relation that allows the full ranking of possible solutions for a MaOP where a different level of priority can be defined for each objective. Then, Section 3 describes the wrapper method proposed in this paper, a CCEA based approach using this lexicographic relation. After that, Section 4 presents some experimental results obtained with the proposed approach and compares them with those obtained with other wrapper methods, and finally, Section 5 concludes this work.

2 A lexicographic relation for MaOEAs

Assuming that n_o objectives have been defined for the problem, and that these objectives have been sorted according to their priority, the fitness for any solution for the problem can be expressed as:

$$\boldsymbol{f} = \left[f^0, f^1, \ldots, f^{n_o-1}\right]^T \in \mathbb{R}^{n_o} \tag{1}$$

Given two fitness evaluations, $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$, and a precision threshold t, the lexicographic relations between them, noted as $\prec_l$ and $\preceq_l$, can be defined as:

$$\boldsymbol{f_1} \prec_l \boldsymbol{f_2} \;\Leftrightarrow\; \exists k \in [0, n_o) \cap \mathbb{N} : f_1^k < f_2^k \;\wedge\; |f_1^k - f_2^k| \geq t \;\wedge\; |f_1^i - f_2^i| < t \;\; \forall i < k \tag{2}$$

$$\boldsymbol{f_1} =_l \boldsymbol{f_2} \;\Leftrightarrow\; |f_1^i - f_2^i| < t \;\; \forall i \in [0, n_o) \cap \mathbb{N} \tag{3}$$

$$\boldsymbol{f_1} \preceq_l \boldsymbol{f_2} \;\Leftrightarrow\; \boldsymbol{f_1} \prec_l \boldsymbol{f_2} \;\vee\; \boldsymbol{f_1} =_l \boldsymbol{f_2} \tag{4}$$

As can be seen, this formulation differs from the pure mathematical lexicographic relation because a threshold t has been introduced to let the Decision Maker (DM) decide the precision used to make the comparison. Regarding the behavior of the algorithm using this relation, it is quite similar to classical lexicographic optimization techniques. It optimizes the objectives in order, but with an important difference. Since an EA is used, local optima can now be avoided [24].

The use of this relation allows the full ranking of the solutions of a MaOP, an thus, a simple EA can be used (not a MOEA or a MaOEA) with smaller populations, since the algorithm will now provide only one optimal solution in each population instead of a Pareto set of not comparable solutions.

3 Proposal

This section describes the main components of the LeMaOCCEA wrapper method presented in this paper, a wrapper method able to optimize the parameters of the classifier while the number of features taken into account is also being minimized. Fig. 1 shows its flowchart. The contributions of this paper have been highlighted, indicating also the section that describes each one of them. The remaining steps of the method are taken from the original EA.

3.1 Cooperative co-evolutionary approach

Since the wrapper method is based on an CCEA, potential solutions of the problem will be evolved within populations of different species. In this case, a hybrid single-level and two-level approach is proposed [16]. One species will be used to evolve the parameters of the classifier while the features of the input dataset will also be split into several species. The number of species used to minimize the input features subset is not fixed a priori, and should be chosen for each experiment according to the number of features in the dataset, in order to balance the search spaces of every population.

With respect to the many-objective optimization part of the feature selection problem, the use of the lexicographic relation described above allows the use of a simple EA scheme to evolve each one of the species defined within the CCEA.

Finally, regarding the selection of representatives, the shuffle-and-pair method [26] will be used. This method shuffles the indexes to access individuals in each population and then combines all the individuals having the same index to form and evaluate a complete solution. Since only one evaluation per individual may seem a poor estimation of its fitness, the process is repeated r times, and then the

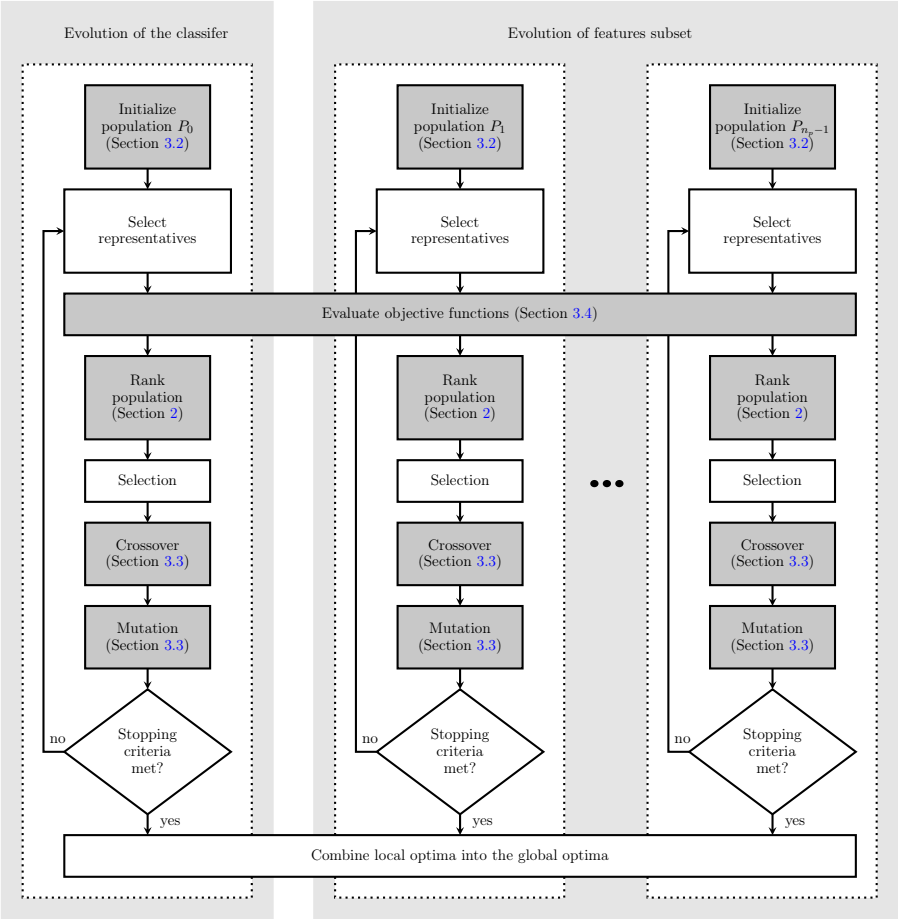

Fig. 1. Flowchart of the proposed wrapper method. The steps that are not highlighted are taken from the original EA

best evaluation for each individual is chosen as its fitness. The only restriction for this method is that all the populations must be of the same size m.

3.2 Species representation

Since the proposed LeMaOCCEA is based on a hybrid single-level and two-level approach, different representations for the species are needed. Specifically, one representation for the parameters of the classifier, that will evolve in one population, and another one for the subsets of features that will be co-evolved in the remaining populations.

Classifier species The SVM parameters will be encoded as a vector of floating point numbers in the first population (P_0).

Features subset species For the features subset species the representation proposed in [12] will be used, that is, all the features will be evenly distributed among the different species, and each sub-population will evolve subsets of feature indexes.

3.3 Breeding operators

Each kind of species will also use its own breeding operators, detailed below.

Breeding operators for the classifier species Given that the species used to evolve the classifier parameters is represented as a vector of real numbers, Simulated Binary Crossover (SBX) [6] and polynomial mutation [7] will be used as breeding operators, since they have been specifically designed to handle real numbers. Both operators rely on a polynomial distribution with a user-defined index parameter ν, which usually is set up to 20 as standard default value.

Breeding operators for the features subset species Regarding the features subset species, the breeding operators have also been adapted from [12], in order that offspring belong to the same species of their parents.

Crossover operator Given a couple of progenitors, I_{j_k} and I_{j_l}, belonging to population P_j, the two offspring, O_{j_k} and O_{j_l}, will be generated as follows. All the common selected features in I_{j_k} and I_{j_l} will also be common in O_{j_k} and O_{j_l}. The remaining selected features coded in I_{j_k} and I_{j_l} will be randomly distributed between O_{j_k} and O_{j_l} in a way that O_{j_k} and O_{j_l} will have the same sizes than I_{j_k} and I_{j_l} respectively. This crossover procedure will always generate valid solutions that meet the constraints stated above.

Mutation operator This operator affects each individual gene or selected feature within an individual separately. If a determined gene is chosen to be mutated, then it may be deleted (reducing the number of selected features within the individual) or randomly altered, being assigned a new value that should be a valid feature index for the problem and should not be repeated within the individual.

Once all the genes have been processed, the mutation operator may also add a new feature index to the individual, increasing the number of selected features, but only if the new feature index is not already selected.

3.4 Lexicographic objectives for the feature selection problem

It seems sensible that the most prioritary objective should be related to the accuracy of the classifier with the reduced features subset. Many works propose the use of the cross-validation error to avoid over-fitting when training classifiers [19]. However, and although this approach has proven successful, it has an important drawback. It is quite computationally expensive, since all the potential solutions tried by the search algorithm must be evaluated several times. This inconvenient affects even more to CCEA approaches, since all the individuals in each sub-population must be re-evaluated every generation because their fitness also depend on the individuals belonging to the remaining sub-populations.

To overcome this problem, a new MOP-based distributed cross-validation approach has been proposed in [12]. For each potential solution (subset of input features) to be evaluated, the original training dataset D is randomly split into two subsets, D_{tr} and D_{val} according to parameter p_{val}, which indicates the percentage of samples used to validate the solution. Although D is split into two different random subsets for each individual evaluation, the division procedure always assures that a percentage p_{val} of samples of each class in D are included in D_{val}. Then, the classifier is trained only with the samples belonging to D_{tr}, and later, two objectives are evaluated: the accuracy obtained by the classifier using D_{tr} and also the accuracy using D_{val}. For the accuracy estimation, the Kappa index [4] is proposed in [12], since it not only takes into account the accuracy of the classifier, but also the per class error distribution.

The following objective in importance should be the size of the final features subset, which should also be minimized, since the problem being solved is a feature selection problem. Finally, the last objective in importance is related to the regularization parameter C used by SVM classifiers. High values for this parameter allows the SVM to use more training examples for the definition of the hyperplane, which may tend to over-fitting, while lower values of C relax this condition. Since the accuracy of the classifier has been included as one of the higher priority objectives, setting the minimization of C as the lower priority objective will guide the algorithm towards solutions with a higher generalization ability without sacrificing the classifier accuracy.

Taking these reflections into account, the following objectives, sorted according to their priority, will be taken into account:

1. Minimize the error rate estimated using D_{val} (SVM is trained with D_{tr}).
2. Minimize the error rate estimated using D_{tr}
3. Minimize the number of features
4. Minimize the regularization parameter C of the SVM

This approach considers the validation accuracy as the most prioritary objective, in order to avoid overfitting. Then the training accuracy is used to untie solutions having similar validation accuracies. For those solutions with similar validation and training accuracies, the number of features will be considered, and finally, if there are sill tied solutions, the regularization parameter of the SVM will be taken into account.

4 Experimentation

This section describes all the details related to the experimentation process carried out to evaluate the proposed LeMaOCCEA wrapper method. Experiments have been performed using the Zoo dataset [11], a well known dataset belonging to the UCI machine learning repository [8]. For all the experiments, each dataset has been randomly split into two sets as proposed in [32]: a training set containing 70% of all samples in each class, and a test set formed by the remaining samples. Adopting the same methodology will provide a fair comparison between the proposed wrapper method and those described in [32].

4.1 Other wrapper alternatives

The results of the proposed LeMaOCCEA wrapper method will be compared to those obtained by the following wrapper methods. All of them use KNN with $k = 5$ to simplify the evaluation process [32].

Linear Forward Selection (LFS): This wrapper method [13] is derived from the classical Sequential Forward Selection (SFS) [30], with the main difference that LFS restricts the number of features that are considered in each step of the forward selection, which can reduce the number of evaluations, optimizing the overall computation time.

Greedy Stepwise Backward Selection (GSBS): It is based on the traditional Sequential Backward Selection (SBS) method [23]. It starts with all the available features and removes sequentially one feature per iteration until the elimination of any remaining feature worsens the accuracy of the classifier [2].

Commonly Used PSO Algorithm (ErFS): This method applies the Particle Swarm Optimization (PSO) search algorithm [14] to minimize the error rate of the classifier. The implementation presented in [32] fixes the inertia weight $w = 0.7298$ and the acceleration constants $c_1 = c_2 = 1.49618$.

PSO With a Two-Stage Fitness Function (2SFS): This wrapper method, also based on the PSO algorithm, divides the evolutionary process into two stages. The first one only minimizes the error rate, whereas the second one also takes into account the number of features in the fitness function [31]. Since the results of this algorithm have also been taken from [32], the parameters of the PSO algorithm are the same than in ErFS.

4.2 Implementation details and parameterization of the proposed LeMaOCCEA method

The implementation of the co-evolutionary algorithm, along with the breeding operators for the population evolving the parameters of the SVM classifier have

Table 1. Parameters for the wrapper method

Parameter	Value
Number of populations (n_p)	5
Population size (m)	150
Number of generations (n)	300
Mutation probability for the feature selection species ($p_{m_{\text{fs}}}$)	0.01
Mutation probability for the SVM parameters species ($p_{m_{\text{svm}}}$)	0.05
Rate of training samples used for validation (p_{val})	0.33
Number of executions of the wrapper method (l)	40
Lexicographic precision threshold (t)	0.001
Co-evolutionary evaluation number of shuffles (r)	2

Table 2. Results obtained by the different wrapper alternatives

Method	Test error rate	# Features
LFS	20.950	8
GSBS	20	7
ErFS	4.500 ± 0.009	9.180
2SFS	4.500 ± 0.009	9.180
LeMaOCCEA	2.832 ± 1.609	4.800 ± 0.791

been taken from ECJ [21], a research Evolutionary Computation (EC) system written in *Java* and developed within the *Evolutionary Computation Laboratory* at the George Mason University, VA, USA. Moreover, for the SVM classifiers LibSVM has been used [3]. The rest of the code has been written by the authors of this work.

Since the LeMaOCCEA wrapper method presented in this paper relies on a CCEA, there are some parameters that must be chosen to make it work. Table 1 shows the values used for the experiments presented in this section. The number of species (populations) has been fixed in a way that the number of features assigned to each population should be 4 or 5.

4.3 Results

Table 2 shows the results obtained by the proposed LeMaOCCEA wrapper method, along with those achieved by the other wrapper methods introduced in Section 4.1. Although at first sight it seems that the proposed wrapper method outperforms the others, a Kruskal-Wallis statistical test has been applied to both, the test error rates and also the number of features of the results provided by each alternative. Table 3 shows the pairwise comparison of the proposed LeMaOCCEA wrapper method with all the other wrapper methods. As can be seen, the proposed wrapper method is able to achieve statistically significant better error rates and also smaller subsets of features than the other alternatives.

Table 3. p-values obtained from multiple pairwise comparison of the test error rate and the number of features achieved by the different wrapper alternatives using the Kruskal-Wallis statistical test

Wrapper 1	Wrapper 2	Test error rate	# Features
LFS	LeMaOCCEA	0.000	0.000
GSBS	LeMaOCCEA	0.000	0.000
ErFS	LeMaOCCEA	0.000	0.000
2SFS	LeMaOCCEA	0.000	0.000

5 Conclusions

This paper has presented a new wrapper approach which hybridized ideas of CCEAs and lexicographic optimization in order to be able to optimize simultaneously the parameters of a SVM classifier and the subset of features that better represent a dataset. The lexicographic approach allows to introduce any number of objectives easily, allowing to solve even MaOPs with a simple EA scheme as the search algorithm for each species. Another advantage of the proposed LeMaOCCEA wrapper method is that it produces only one solution per execution, instead of a set of Pareto optimal solutions, which simplifies the work of the DM.

The proposed LeMaOCCEA wrapper method is also able to obtain statistically significant better results than the rest of wrapper methods it has been compared to, taking into account both the test error rate and also the number of features finally selected.

References

1. Ben-Tal, A.: Characterization of pareto and lexicographic optimal solutions. In: Fandel, G., Gal, T. (eds.) Proceedings of the Third Conference on Multiple Criteria Decision Making Theory and Application. Lecture Notes in Economics and Mathematical System, vol. 177, pp. 1–11. Springer, Berlin, Germany (August 1979). https://doi.org/10.1007/978-3-642-48782-8_1
2. Caruana, R., Freitag, D.: Greedy attribute selection. In: Cohen, W.W., Hirsh, H. (eds.) Proceedings of the Eleventh International Conference on International Conference on Machine Learning, ICML'94. pp. 28–36. Morgan Kaufmann, New Brunswick, NJ, USA (July 1994)
3. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**(3), 27 (2011), software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
4. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement **20**(1), 37–46 (1960). https://doi.org/10.1037/h0026256
5. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley Interscience Series in Systems and Optimization, Wiley, Chichester, NY, USA (2001)
6. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems **9**(2), 115–148 (1995), https://www.complex-systems.com/abstracts/v09_i02_a02/

7. Deb, K., Agrawal, S.: A niched-penalty approach for constraint handling in genetic algorithms. In: Dobnikar, A., Steele, N.C., Pearson, D.W., Albrecht, R.F. (eds.) Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms. pp. 235–243. Springer, Portorož, Slovenia (April 1999). https://doi.org/10.1007/978-3-7091-6384-9_40
8. Dheeru, D., Karra Taniskidou, E.: UCI Machine Learning Repository. School of Information and Computer Sciences, University of California, Irvine, CA, USA (2017), http://archive.ics.uci.edu/ml
9. Drechsler, N., Sülflow, A., Drechsler, R.: Incorporating user preferences in many-objective optimization using relation ϵ-preferred. Natural Computing **14**(3), 469–483 (2015). https://doi.org/10.1007/s11047-014-9422-0
10. Farina, M., Amato, P.: On the optimal solution definition for many-criteria optimization problems. In: Keller, J., Nasraoui, O. (eds.) Proceedings of the 2002 Annual Meeting of the North American Fuzzy Information Processing Society. pp. 233–238. IEEE, New Orleans, LA, USA (June 2002). https://doi.org/10.1109/NAFIPS.2002.1018061
11. Forsyth, R.S.: Zoo Dataset. Mapperley Park, Nottingham, UK (1990), https://archive.ics.uci.edu/ml/datasets/Zoo
12. González, J., Ortega, J., Damas, M., Martín-Smith, P., Gan, J.Q.: A new multi-objective wrapper method for feature selection – accuracy and stability analysis for bci. Neurocomputing **333**(14), 407–418 (2019). https://doi.org/10.1016/j.neucom.2019.01.017
13. Gutlein, M., Frank, E., Hall, M., Karwath, A.: Large-scale attribute selection using wrappers. In: Smith-Miles, K., Keogh, E., Lee, V.C. (eds.) Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining, CIDM'2009. IEEE, Nashville, TN, USA (March 2009). https://doi.org/10.1109/CIDM.2009.4938668
14. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, ICNN'95. vol. 6, pp. 1942–1948. IEEE, Perth, WA, Australia (November 1995). https://doi.org/10.1109/ICNN.1995.488968
15. Khalid, S., Khalil, T., Nasreen, S.: A survey of feature selection and feature extraction techniques in machine learning. In: Arai, K., Mellouk, A. (eds.) Proceedings of the 2014 Science and Information Conference. pp. 372–378. The Science and Information (SAI) Organization, London, UK (August 2014). https://doi.org/10.1109/SAI.2014.6918213
16. Khare, V.R., Yao, X., , Sendhoff, B.: Credit assignment among neurons in co-evolving populations. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.P. (eds.) Proceedings of the 8$^{\text{th}}$ International Conference on Parallel Problem Solving from Nature, PPSN VIII. Lecture Notes in Computer Science, vol. 3242, pp. 882–891. Springer, Berlin, Germany (September 2004). https://doi.org/10.1007/978-3-540-30217-9_89
17. Khosravani, S., Jalali, M., Khajepour, A., Kasaiezadeh, A., Chen, S.K., Litkouhi, B.: Application of lexicographic optimization method to integrated vehicle control systems. IEEE Transactions on Industrial Electronics **65**(12), 9677–9686 (2018). https://doi.org/10.1109/TIE.2018.2821625
18. Klepikova, M.G.: The stability of lexicographic optimization problem. USSR Computational Mathematics and Mathematical Physics **25**(1), 21–29 (1985). https://doi.org/10.1016/0041-5553(85)90037-0

19. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence **97**(1–2), 273–324 (1997). https://doi.org/10.1016/S0004-3702(97)00043-X
20. Li, B., Li, J., Tang, K., Yao, X.: Many-objective evolutionary algorithms: A survey. ACM Computing Surveys **48**(1) (2015). https://doi.org/10.1145/2792984
21. Luke, S., *et al.*: Ecj 26. a java-based evolutionary computation research system, https://cs.gmu.edu/~eclab/projects/ecj/
22. Madonado, S., Weber, R.: A wrapper method for feature selection using support vector machines. Information Sciences **179**(13), 2208–2217 (2009). https://doi.org/10.1016/j.ins.2009.02.014
23. Marill, T., Green, D.: On the effectiveness of receptors in recognition systems. IEEE Transactions on Information Theory **9**(1), 11–17 (1963). https://doi.org/10.1109/TIT.1963.1057810
24. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin, Germany, 3rd edn. (1998)
25. Podinovskii, V.V., Gavrilov, V.M.: Optimization with respect to successive criteria (Optimizatsiya po posledovatel'no primenyaemym kriteriyam). Sovetskoe Radio, Moscow, Russia (1975)
26. Popovici, E., Bucci, A., Wiegand, R.P., De Jong, E.D.: Coevolutionary principles. In: Rozenberg, G., Bäck, T., Kok, J.N. (eds.) Handbook of Natural Computing, pp. 987–1033. Springer, Berlin, Germany (2012). https://doi.org/doi.org/10.1007/978-3-540-92910-9_31
27. Potter, M.A., De Jong, K.A.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evolutionary Computation **8**(1), 1–29 (2000). https://doi.org/10.1162/106365600568086
28. Rasekhipour, Y., Fadakar, I., Khajepour, A.: Autonomous driving motion planning with obstacles prioritization using lexicographic optimization. Control Engineering Practice **77**, 235–246 (2018). https://doi.org/10.1016/j.conengprac.2018.04.014
29. Schmiedle, F., Drechsler, N., Große, D., Drechsler, R.: Priorities in multi-objective optimization for genetic programming. In: Spector, L., Goodman, E.D., Wu, A., Langdon, W.B., Voigt, H.M. (eds.) Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, GECCO'2001. pp. 129–136. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (July 2001), https://dl.acm.org/citation.cfm?id=2955256
30. Whitney, A.W.: A direct method of nonparametric measurement selection. EEE Transactions on Computers **C-20**(9), 1100–1103 (1971). https://doi.org/10.1109/T-C.1971.223410
31. Xue, B., Zhang, M., Browne, W.N.: New fitness functions in binary particle swarm optimisation for feature selection. In: Abbass, H., Essam, D., Sarker, R. (eds.) Proceedings of the 2012 IEEE Congress on Evolutionary Computation, CEC'2012. IEEE, Brisbane, QLD, Australia (June 2012). https://doi.org/10.1109/CEC.2012.6256617
32. Xue, B., Zhang, M., Browne, W.N.: Particle swarm optimization for feature selection in classification: A multi-objective approach. IEEE Transactions on Cybernetics **43**(6), 1656–1671 (2013). https://doi.org/10.1109/TSMCB.2012.2227469
33. Zykina, A.V.: A lexicographic optimization algorithm. Automation and Remote Control **65**(3), 363–368 (2004). https://doi.org/10.1023/B:AURC.0000019366.84601.8e