AITA
ARTIFICIAL INTELLIGENCE THEORY AND APPLICATIONS

■ RESEARCH ARTICLE

Efficient and adaptive operator selection in swarm intelligence using machine learning approaches

Mehmet Emin Aydın [a†], **Rafet Durgut** [b]

[a] Computer Science and Creative Technologies, University of the West of England, Bristol, United Kingdom
[b] Engineering and Natural Sciences, Bandırma Onyedi Eylül University, Balıkesir, Türkiye
† mehmet.aydin@uwe.ac.uk, corresponding author

RECEIVED DECEMBER 4, 2023
ACCEPTED JANUARY 10, 2024

CITATION Aydın, M., Durgut, R. (2024). Efficient and adaptive operator selection in swarm intelligence using machine learning approaches. *Artificial Intelligence Theory and Applications, 4(1)*, 22-32.

Abstract

Problem solving has been one of renown artificial intelligence fields attracting research for decades. Swarm intelligence is recognised as the family of the state-of-art approaches in problem solving gained much research attention for the enduring problems. The main challenge appears to be is the speed of algorithmic approximation where many approaches were proposed to accelerate approximation avoiding local optima. Recent research demonstrates that inefficiencies in search procedures can be side-stepped using the experiences gained while search is undergoing utilising machine learning approaches. Machine learning turned to be popular to let approach problems on experience basis. Reinforcement learning becomes a success-proven approach for online learning, especial when training data is not available upfront. In this paper, we overview the usefulness of machine learning and reinforcement learning in performance improvement of artificial bee colony algorithms in solving combinatorial optimisation problems. Furthermore, we demonstrate how supervised and reinforcement learning approaches facilitate swarm intelligence algorithms to gain experience for immediate and later use to build capable and powerful operator selection schemes, which help improve efficiency of swarm intelligence problem solvers.

Keywords: adaptive operator selection; machine learning; reinforcement learning; artificial bee colony; set union knapsack problem

1. Introduction

Optimisation involves very hard engineering problems to find out the best solutions within a reasonable time frame. Artificial intelligence offers convenient approaches to alternate the classical problem-solving approaches. Evolutionary computation and swarm intelligence bring many opportunities to researchers and practitioners' attention to handle such hard problems, particularly those problems classified as NP-Hard and NP-Complete problems [1]. However, due to the characteristics of the problems space, difficulties emerge while conducting exploration across the solution space. The difficulties are known as when to intensify search locally and when to diversify towards different regions for better solutions. This is recognised as "Exploration versus Exploitation" (EvE) issue [2]. The dilemma of EvE is applied to all search algorithms including swarm intelligence algorithms such as particle swarm optimisation, artificial bee colonies, and ant colony optimisation variants.

A reasonable success in EvE actions goes through use of different approaches harmonised into problem solving process. However, problem solving requires domain and problem-specific information in order to decide how and when to intensify the search and how and when to diversify it. Research has never stopped to investigate for better problem-solving approaches with lower time complexity. Use of multiple search strategies as well as multiple operators is known an important stream as part of this effort. An important research question in this regard is that how to select an operator / strategy to tackle the temporal state of the problem in the best way and capacity. Is there any generic approach applicable to all types of the problems?

The search issues mentioned above lead to another difficulty on the way, which is how to reach a general approach applicable to every kind of problems. No-Free-Lunch theory [3] has originated from research efforts in this regard which confirms how difficult to come up with an approach applicable to as many problem types as possible. On the other hand, there is a classical dream of artificial intelligence to attain a general problem solver [4], [5].

Recent studies paid attention on the way to develop more generic approaches to reasonably offered solutions for combinatorial optimisation problems [6]–[9]. The author of [10] has surveyed online and off-line machine learning algorithms in efficiency studies of metaheuristics in a wider context, and paid attention to the use of machine learning in building operator selection schemes. Feature analysis has been conducted in the study reported by [9] in order to gauge the impact and predictiveness of each feature identified to characterise the states of operator selection problem. The analysis is carried out with supervised machine learning approaches in that work with a comparative approach. More specific studies reported, especially on utilisation of reinforcement learning approaches for improvement of metaheuristic efficiency [10], [11]. We, the authors of this article, have not come across to any work – to the best of our knowledge – overviewing the use of supervised and reinforcement learning approaches in building efficient adaptive operator selection schemes so as to utilise in swarm intelligence algorithms for boosting performance and efficiency, especially with a generalisation point of view. The aim of this paper is to overview the recent attempts and evaluate them with respect to the level of generalisation.

The rest of this paper is organised as follows: Section 2 introduces how swarm intelligence algorithms are enhanced and made efficient with machine learning referring to all key references and state of the art works in the field, Section 3 summarises the result set for proof-of-concept with relevant discussions and Section 4 concludes the article.

2. Materials and Methods

The main aim of this article is to overview the recent works to develop more generalisable approaches to solve combinatorial optimisation problems, and possibly continuous problems, too. A problem is defined and described with models (e.g., mathematical or simulation models) with which the inter-relations of its components are represented with variables and parameters. Then, the modelled problem requires a solution which best fits in use. The optimisation field has been populated with a variety of heuristic-based approaches alongside traditional search and problem-solving approaches. Among these, swarm intelligence has proven significant success in the field and achieved to be recognised as a state-of the-art paradigm. The following subsections will introduce how recent research has been shaped up around these concepts of the paradigm merged with frontline machine learning principles and approaches.

2.1 Main search algorithm

The optimisation framework always imposes a search algorithm with which the better and the best results fitting to the circumstances is explored. Recent studies pay more attention on heuristic solutions, which are derived from meta-heuristic frameworks such as evolutionary algorithms and swarm intelligence algorithms. Artificial Bee Colony (ABC) optimisation is one of recently developed, success proven, renown swarm intelligence algorithms [12]. The main motivation behind ABC is to imitate the social behaviour of honeybees in nectar search mimicking the way the colonies collaborate to achieve the goals and targets. Search algorithms arrange move from one solution to another via different types of operators, but each operator comes up with limits in approximation, which is believed to be eased with use of other complimentary alternatives. This enforces use of multiple operators /neighbourhood functions by the algorithms [13]. It is believed that ABC is one of bound-free algorithms in approximation due to that there is not much functional constraints systematically impose search instrumented by ABC. Its variants extended with multiple operators have been used for solving many combinatorial [6], [14] and functional [15] optimisation problems, a number of different variants are proposed for efficiency in problem solving.

Another recent swarm intelligence algorithm is called Crow Search Algorithm (CSA), which is recently devised mimicking the social behaviour of crows in handling issues. The algorithm has recently been implemented with multi search strategies [16]. Likewise, particle swarm optimisation [17] and evolutionary, (e.g., differential evolution [18]), algorithms have also been implemented with multiple search strategies, and or operators to diversify search and achieve high efficiency [19], [20]. Grey wolf and whale optimisation algorithm combined with reinforcement learning in [21] and multi-armed bandits [22] used for optimiser selection among Harris Hawks optimiser, differential evolution and whale optimisation algorithm.

2.2 Adaptive operator selection

Operators are neighbourhood functions devised to manage moving from one problem state to another while searching for the best solution with optimisation algorithms. Many algorithms have been designed to use single operator at the beginning, but, adopted utilising multiple operators through the whole process subject to a selection rule or scheme imposed [1]. Obviously, not all schemes proposed are adaptive or systematic. However, recently, especially multi-objective optimisation algorithms have started using multiple operators with adaptive schemes [15, 23, 25].

Operator selection requires choosing an operator from a pool of operators, which is set up to let change the state of the problem in-hand to another state. This is done through the representation vector. The states are represented with sets of numbers; either binary, integer, or real numbers. Evolutionary computation has introduced the concepts of phenotype and genotype [13] with which the problem states are represented in a humanly expression and then into numbers to process and compute, then re-converted back to readable expressions by human. Genotypes are operated with existing move functions, i.e. neighbourhood functions applying the process built in the selected operator.

Operator selection stands out as an issue to tackle with an efficient way. It can be either random or applying a systematic rule. For instance, genetic algorithms imply probabilistic selection of crossover and mutation operators, while variable neighbourhood search requires periodic change of operators so as to regularly condense and relieve the search. On the other hand, there is a sound stream of operator selection studies propose adaptive approaches to impose heuristic intelligence into the selection process for efficiency purposes [18]. Adaptive operator selection with credit assignment and multi-armed bandit approaches has been studied by Fialho as reported in [27]. In general, the logic of how operator selection mechanics work is shown in Figure 1 in which an operator is selected by "Selection Scheme" from "Operator Pool" based on the merits gained and applied to the problem state under consideration, then "Evaluate"d for how productive it was. A "Credit Assignment" mechanism picks up the evaluation results and calculates a credit level to supply to the chosen operator's merit record for its selectability in the future noting that the popularity of the operators is managed based on merits/credits gained.

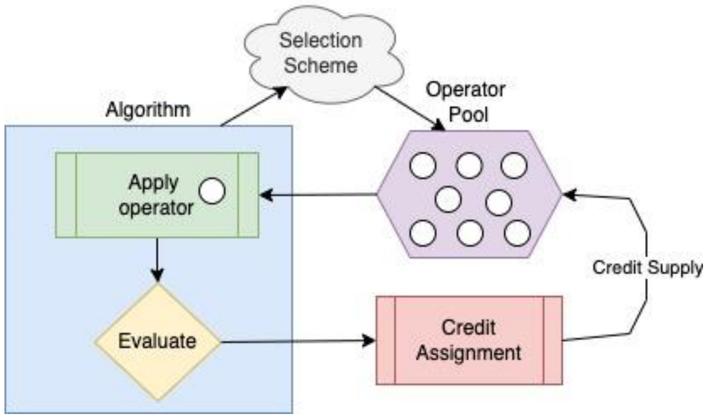

Figure 1. A typical cycle of operator selection mechanism

One of the main concerns of operator selection process is the rule or the scheme to implement for which of the operator from the pool is to be selected for the best move. As mentioned before, the easiest is to do a random selection or a pre-ordered selection, but this would not consider the current circumstances of the search environment and the parametric levels. This fact imposes adaptive approaches. The state-of-the-art approaches are known to be either stochastic or credit-based approaches. As thoroughly discussed in [8] and [12], adaptive approaches include "probability matching" and "adaptive pursuit" are heavily based on credit assignment, while there are a number of other criteria proposed such as fitness-based, diversity-based ones. Many other studies including [15, 18] have been developed on the basis of multi-armed bandit problems.

There are a number of renown operators have been studied and proposed over decades since heuristic optimisation emerged. Operators include random "*flip-flop*", "*swap*" and "*inverse*" are to assist change a single digit on the representation vector. If it is a binary representation, the operators will change a single bit only to offer a move from current state. There are logic-based operators with which more than one digit of the vector is revised. *AND, OR, NOT* and *XOR* operators have been used to set up simple and complex logic operations to propose new solutions to move to while searching. Furthermore, various genetic operators borrowed from evolutionary algorithms have been implemented and similarities and distance-based operations brought to effect, too. More details can be found in [6, 10, 13].

Recently, machine learning is instrumented to build dynamically functioning credit and reward-based approaches, which enforce to learn from gained experiences either online [6] or off-line [7, 29] to act on the basis of changing environmental circumstances of search process.

2.3 Machine learning for adaptive operator selection

Utilisation of various machine learning techniques into metaheuristics involves many aspects of the search process in order to improve search efficiency in various respects. The main idea behind the use of machine learning in problem solving is to optimise data-driven models to work out with unseen data, which facilitates great opportunities to achieve highly efficient systems. As indicated before, a trend of utilisation from machine learning is observed and ongoing to improve metaheuristic and swarm intelligence optimisation [7, 8, 23, 24, 26]. However, the aim of this paper is to look into the major steps recently taken towards generalisation of problem solving. It is paramount to note that previously developed adaptive operator selection approaches did not take the problem state on board while proposing selection of a particular operator from the pool. The following steps take the problem state on board while making decision to select an operator.

1. The first approach taken on board is the binary representation of the problem states and use of binary operators in managing moves from one state to a neighbouring problem state. The reason behind this step is that any representation approach can be converted into binary and be operated with binary operations regardless of any domain knowledge requirement. This approach has been widely implemented and its efficiency is proven through the development of evolutionary algorithms, especially at the early time. Several recent works have been published using the virtue of binarisation including [6–10]. However, this approach brings the major restriction of scalability, which does not help extensive use of gained experiences. This is due to the fact that every problem state has a specific size of input data and does not support the problems with a different size, which undermines the usability of gained experience over previous problem-solving activities.

2. Transfer learning is a new concept brought forward as part of recent machine learning studies, especially offered by deep learning works. Deep learning offers pre-trained huge neural networks with which any unseen data can be easily handled adapting the pre-existing models into the data and the domain knowledge. However, this remains an important open research issue to enhance the performances further and solve the problems more realistically in a satisfactory level. The idea is to investigate if pre-trained agents can utilise the past experience in solving a completely new problem instance [7]. The study reported in [8] implemented a binarisation based approach to achieve transfer learning built up via reinforcement learning – Q learning – supported with hard-c means clustering algorithm. The learning was conducted across the problem instances in the same size. The results are promising, but the gained experience cannot be applied to solving the problem instances of different sizes.

3. Feature-based problem state representation is another promising approach for generalisation which is brought forward by [28], [29], where prominent analysis is conducted to identify the most relevant features to be selected for this purpose. The problems are better represented with a vector of feature set. The analysis conducted has taken fitness landscape information harvested over the individuals

and the populations and used supervised learning approaches. It sounds very promising to handle problems represented with the features in order to train the models with reinforcement learning and other active learning approaches.

2.4 Reinforcement learning-based adaptive selection scheme

Reinforcement learning is one of hotspot machine learning subjects that attracts so much attention by researchers in the field as well as other disciplines for application purposes. It helps agents /systems learn actively – "active learning" – which facilitates agent training while delivering the tasks. Optimisation and search process is a very dynamically changing environment, which is influenced significantly by the actions taken in the previous stages. It means that an operator selected will change the direction of search and other relevant circumstances hence the next steps will produce taking the output of previous steps into account as the follow-up steps. This works in a sequential way in which one operator is selected and activated, then another is selected accordingly based on the changed circumstances and repeats until the completion. Ultimately, the problem turns into a dynamically built sequence of operators.

The logic embedded in mechanism shown in **Hata! Başvuru kaynağı bulunamadı.** has b een expanded into **Hata! Başvuru kaynağı bulunamadı.** in order to demonstrate how reinforcement is devised and instrumented to let the agent learn and produce credit to the corresponding operator selected and applied. Apparently, "Evaluate" component is the one where this algorithm is embedded. Given that input ***x*** is presented to the "Operator Selection" unit and merged with the output ***o*** generated before and passed to "Cluster Update" unit to label operator ***a*** selected. Once done, operator ***a*** generates a new ***o*** and evaluates the quality of solution with ***F(x,a)***. Meanwhile, ***x*** and ***o*** are passed to the "Reward Generation" unit to produce ***r*** and forwarded to "Cluster Update" to adapt with the new knowledge. This repeats throughout the complete search process.

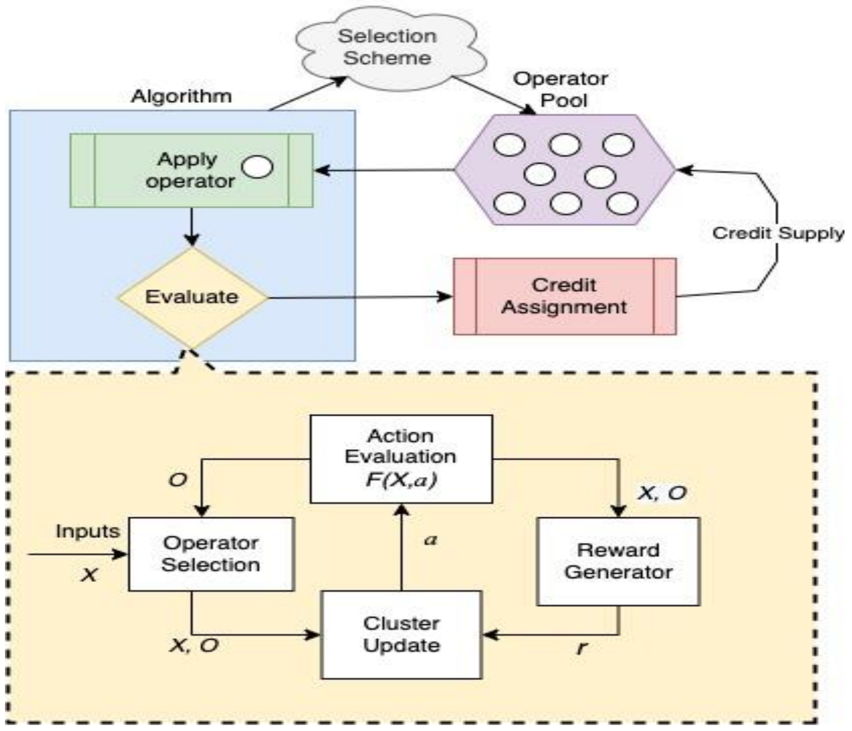

Figure 2. Evaluate function expanded with reinforcement learning components.

3. Results and Discussions

The approaches introduced in the previous subsections have been tested for efficiency with a particular NP-Hard combinatorial problem, namely Set Union Knapsack problem (SUKP). The experimental results are collected running them on a cloud-based high performance compute cluster provided by TUBITAK ULAKBIM. First of all, an ABC framework has been used as a swarm intelligence algorithm with which not much technical constraints are necessarily imposed. A pool of 5 operators is embedded into the ABC implementation, and four variants have been set up to test the above-mentioned ideas to see how helpful the approaches. Similar benchmark instances of SUKP have been solved binary ABC as reported in [30].

Table 2 presents the experimental results of four ABC variants, comparatively, where each is furnished with an embedded pool of operators orchestrated by an operator selection scheme; one of them imposes uniformly randomly selected operators, while the other three variants are bespoken with a reinforcement learning algorithm devised with Q learning and Hard-c-Means algorithms. All four algorithms are compared with respect to mean, standard deviation and the best results (i.e., max value) with respect to the quality of solution measured with fitness function. The tabulated statistics (metrics) have been calculated over 30 trails (runs).

Table 1. Set union knapsack problem (SUKP) benchmark instances.

Problem	*M*	*N*	*W*	*Y*
PI1	400	385	0.15	0.85
PI2	500	485	0.10	0.75
PI3	100	100	0.15	0.85
PI4	400	385	0.10	0.75
PI5	100	100	0.10	0.75
PI6	200	200	0.10	0.75
PI7	200	200	0.15	0.85
PI8	185	200	0.10	0.75
PI9	300	300	0.15	0.85
PI10	200	185	0.15	0.85

Table 2. Comparative results provided by four variants of ABC algorithm for solving 10 instances of SUKP.

	Random			RLABC			RLABC-FL			RLABC-TL		
Problem	**Max**	**Mean**	**STD**	**Max**	**Mean**	**STD**	**Max**	**Mean**	**STD**	**Max**	**Mean**	**STD**
PI1	10168	9997.7	188.3	10168	10027.1	145.2	10168	10055.1	138.0	10175	10123.9	84.4
PI2	11326	11076.9	147.5	11427	11188.1	140.5	11426	11118.6	151.0	11490	11196.1	134.7
PI3	13407	13205.3	206.9	13402	13271.7	100.2	13407	13222.7	149.3	13405	13283.0	67.7
PI4	10852	10512.2	179.2	10877	10647.3	101.1	10994	10616.7	180.0	10831	10626.4	90.5
PI5	13963	13822.4	74.5	14044	13949.0	85.1	14044	13850.6	79.5	14044	13943.2	86.4
PI6	12257	11716.6	255.5	12211	11833.3	178.2	12350	11792.0	253.9	12328	11944.3	201.9
PI7	11800	11491.2	204.6	12019	11652.0	163.0	11821	11550.1	257.2	11821	11627.4	201.4
PI8	13463	13097.6	228.6	13402	13271.7	100.2	13392	13141.2	174.8	13405	13283.0	67.7
PI9	10724	10592.2	177.4	11410	10679.6	176.8	10735	10618.9	118.8	11054	10759.6	144.6
PI10	13671	13230.3	144.7	13609	13352.2	130.0	13671	13376.1	191.4	13609	13399.0	99.7

RLABC is an ABC variant uses binary representation of the problem states and operates the solutions with selecting one operator from a tool of three different operators, training the decision-making agent with reinforcement learning (RL) algorithm – mentioned above – when and how to select each of the operator given the problem state and search circumstances [6]. RLABC-TL uses the same representation and set of operators for transfer learning across different runs of the problem instances utilising the same RL algorithm, but transferring the gained experiences learned previously [7], [8]. On the other hand, RLABC-FL uses a feature-based problems representation, trains the agent with the same RL algorithm to select one of the operators from the pool more efficiently noting that the set of operators in the pool are different from the other three variants. It is open to transfer learning, too.

The results tabulated in Table 2 have been ranked with Wilcoxon sign test, accordingly, to find out the best and the worst performing algorithms. The ranks are presented in Table 3 with respect to 2 metrics; “mean” and the “max” values, where the rank spans from 1 to 4 indicating that 1 is the best and 4 is the worst. The overall performance by each algorithm is averaged at the bottom of the table, where RLABC-TL overperforms the rest in both measures, and the runner up is RLABC in “mean” – with 1.3 versus 1.9 – and RLABC-FL in “max” values – with 1.7 versus 1.9. It is important to note that RLABC-FL may not be properly comparable due to the fact that it uses a different set of operators in the pool. Nevertheless, RLABC-FL demonstrates a clear potential in comparisons of the right-hand-side of the table.

Table 3**.** Comparative results with respect to the ranks collated from both means and maximum results by each of the algorithms.

	Comparisons with Mean in rank				**Comparisons with Max in rank**			
Problem	**Random**	**RLABC**	**RLABC-FL**	**RLABC-TL**	**Random**	**RLABC**	**RLABC-FL**	**RLABC-TL**
PI1	4	3	2	1	2	2	2	1
PI2	4	2	3	1	4	2	3	1
PI3	4	2	3	1	1	4	1	2
PI4	4	1	3	2	3	2	1	3
PI5	4	1	3	2	4	1	1	1
PI6	4	2	3	1	3	4	1	2
PI7	4	1	3	2	4	1	2	2
PI8	4	2	3	1	1	3	4	1
PI9	4	2	3	1	4	1	3	2
PI10	4	3	2	1	1	3	1	2
Mean:	4	1.9	2.8	1.3	2.7	2.3	1.9	1.7

The results collected from the experimentation of RLABC-FL are looked at to realise how contributing is each of the features in the learning process. Figure 1 and Figure 2 show the learning progress of the operator selection agent on which operator has been selected and activated across the whole span of iterations. Apparently, each figure is a multi-plot including 12 plots representing the 12 features found more effective, where each cell – of both figures – indicates the learning progression of the agent by the means of a particular feature. Here, Figure 1 displays the scattered data collected in the first run (i.e. 1st trail of experimentation) per features indicating the selected operators across iterations, while Figure 2 plots the scattered data taken from the last run in the same way to realise the differences in between the approximations through the features. This characterises how good the agent has learned from the first to the last trail / run. All features seem stable except 7th and 11th features, which look not discriminative, sufficiently.

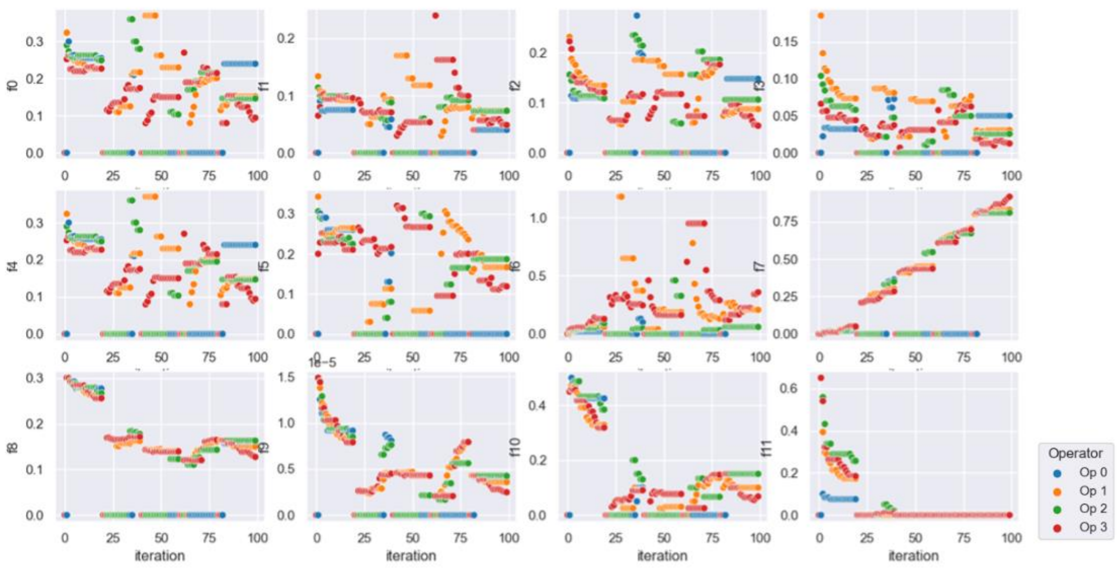

Figure 1. The predictiveness of the features in the first run.

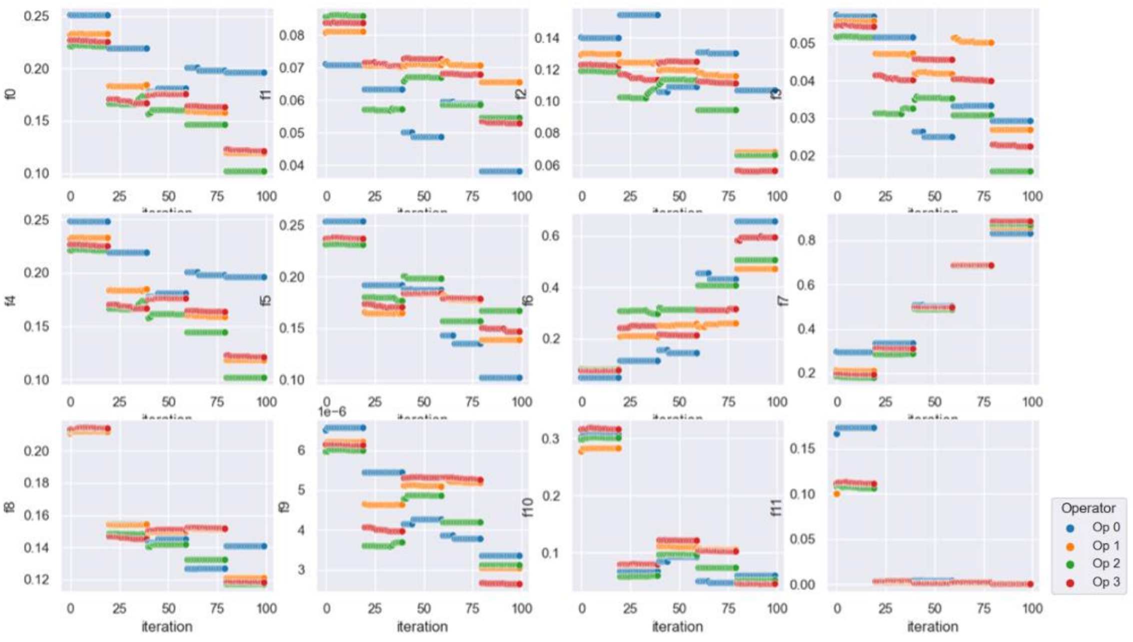

Figure 2. The predictiveness of the features in the last run.

4. Conclusion

This article overviews and discusses use of machine learning, especially reinforcement learning, to improve the efficiency of metaheuristic-based search algorithms with a generalisation point of view. Machine learning has recently been vastly used to model data and help handle real-world problems in a data-driven approach. It offers ways to facilitate domain-awareness in handling the problems. Especially, optimisation problems are normally solved with non-guided search algorithms, which do not use domain knowledge. But with use of machine learning, especially reinforcement learning, the optimisation algorithms can be furnished with data-driven facilities to tackle the problems with domain awareness.

The article focuses on variations of reinforcement learning and data-driven approaches to improve swarm intelligence algorithms, particularly artificial bee colony (ABC) variants. ABC variants are instrumented with adaptive operator selection scheme built with reinforcement learning to solve set union knapsack problem as one of prominent NP-Hard combinatorial optimisation problems. Three variants have been compared initially with a random operator selection scheme and next with one another. The results suggest that reinforcement learning seems promising for building adaptive operators selection scheme, particularly, transfer learning looks a bright way out for this purpose, which needs to be further studied and investigated. Once accomplished, a substantial generalisation would be achieved.

This area of study needs to be extended towards multi-objective optimisation domain with the view that each objective would impose a particular interest in gaining experience and knowledge. Subsequently, there might appear conflicts among the sources of reinforcements, which requires to be resolved. Various aspects of the subject have been tackled so far, but more outstanding issues and aspects need to be studied.

References

[1] G. J. Woeginger, ‘Exact Algorithms for NP-Hard Problems: A Survey’, in *Combinatorial Optimization — Eureka, You Shrink!: Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers*, G. and R. G. Jünger Michael and Reinelt, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 185–207. doi: 10.1007/3-540-36478-1_17.

[2] M. Črepinšek, S.-H. Liu, and M. Mernik, ‘Exploration and exploitation in evolutionary algorithms: A survey’, *ACM computing surveys (CSUR)*, vol. 45, no. 3, pp. 1–33, 2013.

[3] D. H. Wolpert and W. G. Macready, ‘No free lunch theorems for optimization’, *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, Apr. 1997, doi: 10.1109/4235.585893.

[4] K.-P. L. Proctor Robert W. and Vu, ‘Human Information Processing’, in *Encyclopedia of the Sciences of Learning*, N. M. Seel, Ed., Boston, MA: Springer US, 2012, pp. 1458–1460. doi: 10.1007/978-1-4419-1428-6_722.

[5] D. B. Leake, ‘Problem Solving and Reasoning: Case-based’, *International Encyclopedia of the Social & Behavioral Sciences*, pp. 12117–12120, 2001, doi: 10.1016/B0-08-043076-7/00545-3.

[6] R. Durgut and M. E. Aydin, ‘Adaptive binary artificial bee colony algorithm’, *Appl Soft Comput*, vol. 101, p. 107054, Mar. 2021, doi: 10.1016/J.ASOC.2020.107054.

[7] R. Durgut, M. E. Aydin, and A. Rakib, ‘Transfer Learning for Operator Selection: A Reinforcement Learning Approach’, *Algorithms*, vol. 15, no. 1, Jan. 2022, doi: 10.3390/A15010024.

[8] R. Durgut, M. E. Aydin, and I. Atli, ‘Adaptive operator selection with reinforcement learning’, *Inf Sci (N Y)*, vol. 581, pp. 773–790, Dec. 2021, doi: 10.1016/J.INS.2021.10.025.

[9] E.-G. Talbi, ‘Machine Learning into Metaheuristics: A Survey and Taxonomy’, *ACM Comput. Surv.*, vol. 54, no. 6, Jul. 2021, doi: 10.1145/3459664.

[10] M. Tessari and G. Iacca, ‘Reinforcement Learning Based Adaptive Metaheuristics’, in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, in GECCO ’22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 1854–1861. doi: 10.1145/3520304.3533983.

[11] T. Wauters, K. Verbeeck, P. de Causmaecker, and G. vanden Berghe, ‘Boosting metaheuristic search using reinforcement learning’, in *Hybrid metaheuristics*, Springer, 2013, pp. 433–452.

[12] J. C. Bansal, H. Sharma, and S. S. Jadon, ‘Artificial bee colony algorithm: a survey’, *International Journal of Advanced Intelligence Paradigms*, vol. 5, no. 1–2, pp. 123–159, Jan. 2013, doi: 10.1504/IJAIP.2013.054681.

[13] C. Ozturk, E. Hancer, and D. Karaboga, ‘A novel binary artificial bee colony algorithm based on genetic operators’, *Inf Sci (N Y)*, vol. 297, pp. 154–170, Mar. 2015, doi: 10.1016/J.INS.2014.10.060.

[14] R. DURGUT and M. AYDİN, ‘Çok boyutlu sırt çantası problemi için adaptif ikili yapay arı kolonisi algoritması (AİYAK)’, *Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi*, May 2021, doi: 10.17341/gazimmfd.804858.

[15] M. Düğenci and M. E. Aydin, ‘A honeybees-inspired heuristic algorithm for numerical optimisation’, *Neural Comput Appl*, vol. 32, no. 16, pp. 12311–12325, Aug. 2020, doi: 10.1007/s00521-019-04533-x.

[16] R. Durgut and M. E. Aydin, ‘Multi Strategy Search with Crow Search Algorithm’, in *Optimisation Algorithms and Swarm Intelligence*, Prof. N. Vakhania, Ed., Rijeka: IntechOpen, 2022. doi: 10.5772/intechopen.102862.

[17] Z. Cui *et al.*, ‘Hybrid many-objective particle swarm optimization algorithm for green coal production problem’, *Inf Sci (N Y)*, vol. 518, pp. 256–271, May 2020, doi: 10.1016/j.ins.2020.01.018.

[18] D. Kizilay, M. F. Tasgetiren, H. Oztop, L. Kandiller, and P. N. Suganthan, ‘A Differential Evolution Algorithm with Q-Learning for Solving Engineering Design Problems’, in *2020 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Jul. 2020, pp. 1–8. doi: 10.1109/CEC48606.2020.9185743.

[19] W. xiang Wang, K. shun Li, X. zhen Tao, and F. hui Gu, ‘An improved MOEA/D algorithm with an adaptive evolutionary strategy’, *Inf Sci (N Y)*, vol. 539, pp. 1–15, Oct. 2020, doi: 10.1016/J.INS.2020.05.082.

[20] Q. Lin *et al.*, ‘Adaptive composite operator selection and parameter control for multiobjective evolutionary algorithm’, *Inf Sci (N Y)*, vol. 339, pp. 332–352, Apr. 2016, doi: 10.1016/J.INS.2015.12.022.

[21] C. Qu, W. Gai, M. Zhong, and J. Zhang, ‘A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning’, *Appl Soft Comput*, vol. 89, p. 106099, Apr. 2020, doi: 10.1016/J.ASOC.2020.106099.

[22] K. Li, A. Fialho, S. Kwong, and Q. Zhang, ‘Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition’, *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 114–130, 2013.

[23] M. A. Al-Betar, I. A. Doush, A. T. Khader, and M. A. Awadallah, ‘Novel selection schemes for harmony search’, *Appl Math Comput*, vol. 218, no. 10, pp. 6095–6117, Jan. 2012, doi: 10.1016/J.AMC.2011.11.095.

[24] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, ‘Landscape-based adaptive operator selection mechanism for differential evolution’, *Inf Sci (N Y)*, vol. 418–419, pp. 383–404, Dec. 2017, doi: 10.1016/J.INS.2017.08.028.

[25] W. Khan Mashwani, A. Salhi, O. Yeniay, H. Hussian, and M. A. Jan, ‘Hybrid non-dominated sorting genetic algorithm with adaptive operators selection’, *Appl Soft Comput*, vol. 56, pp. 1–18, Jul. 2017, doi: 10.1016/J.ASOC.2017.01.056.

[26] D. B. Fogel, ‘Phenotypes, genotypes, and operators in evolutionary computation’, in *Proceedings of 1995 IEEE International Conference on Evolutionary Computation*, p. 193.

[27] Á. Fialho, ‘Adaptive Operator Selection for Optimization’, Université Paris Sud - Paris XI, 2010. [Online]. Available: https://tel.archives-ouvertes.fr/tel-00578431

[28] R. Durgut, M. E. Aydin, H. Ihshaish, and R. Abdur, ‘Analysing the Predictivity of Features to Characterise the Search Space’, in *Artificial Neural Networks and Machine Learning – ICANN 2022*, E. Pimenidis and M. E. Aydin, Eds., Bristol: Springer, Sep. 2022, pp. 1–13.

[29] M. E. Aydin, R. Durgut, A. Rakib, and H. Ihshaish, ‘Feature-based search space characterisation for data-driven adaptive operator selection’, *Evolving Systems*, Dec. 2023, doi: 10.1007/s12530-023-09560-7.

[30] Y. He, H. Xie, T. L. Wong, and X. Wang, ‘A novel binary artificial bee colony algorithm for the set-union knapsack problem’, *Future Generation Computer Systems*, vol. 78, pp. 77–86, Jan. 2018, doi: 10.1016/J.FUTURE.2017.05.044.