Archives of Computational Methods in Engineering
https://doi.org/10.1007/s11831-023-09975-0

ORIGINAL PAPER

A Literature Review and Critical Analysis of Metaheuristics Recently Developed

Luis Velasco[1] · **Hector Guerrero**[1] · **Antonio Hospitaler**[2]

Received: 26 December 2022 / Accepted: 2 July 2023

Abstract
Metaheuristic algorithms have applicability in various fields where it is necessary to solve optimization problems. It has been a common practice in this field for several years to propose new algorithms that take inspiration from various natural and physical processes. The exponential increase of new algorithms is a controversial issue that several researchers have criticized. However, their efforts to point out multiple issues involved in these practices have been insufficient since the number of existing metaheuristics continues to increase yearly. To know the current state of this problem, this paper analyzes a sample of 111 recent studies where so-called new, hybrid, or improved optimization algorithms are proposed. Throughout the document, the topics reviewed will be addressed from a general perspective to their specific aspects. Among the study's findings, it is observed that only 43% of the analyzed papers make some mention of the No Free Lunch (NFL) theorem, being this significant result ignored by most of the studies where new algorithms are presented. Of the analyzed studies, 65% present an improved version of some established algorithm, which reveals that the trend is no longer to propose metaheuristics based on new analogies. Additionally, a compilation of solutions found in engineering problems commonly used to verify the performance of state-of-the-art algorithms is presented. To demonstrate that algorithms with a low level of innovation can be erroneously considered as new frameworks for years, the metaheuristics known as Black Widow Optimization and Coral Reef Optimization are analyzed. The study of its components reveals that they do not have any innovation. Instead, they are just deficient mixtures of different evolutionary operators. This result applies by extension to their recently proposed improved versions.

1 Introduction

There is a controversy regarding the ever-increasing number of new metaheuristic algorithms referred to as "innovative" [1]. As indicated by Sörensen et al. [2], many of these metaheuristics do not make real contributions to the research field, having only as theoretical support an ambiguous metaphor with some natural process. Although these types of new metaheuristics continue to be proposed, other types of studies are focused on combining existing algorithms or adding operators to metaheuristics already presented to create "improved" versions. Either of these cases results in the publication of many similar studies whose scientific relevance seems questionable due to the low level of innovation. Notwithstanding these objectionable characteristics and the claim of part of the scientific community [1], such studies continue to be published in internationally renowned scientific journals. A remarkable feature is that these studies constantly announce outstanding performances, which, in turn, may explain the interest they have generated in the scientific community.

Of particular concern is the number of different documented cases where these new metaheuristics are just plagiarisms of other more popular algorithms or simple reformulations of metaheuristics already presented. One such case is pointed out by Weyland [3, 4] who shows that Harmony Search [5], a very popular and widespread metaheuristic, presents an identical formulation to Evolution Strategies, a metaheuristic based on evolutionary processes presented many years earlier [6]. Another example is given by Camacho et al. [7] who indicate that the Intelligent Water Drops Algorithm [8] is a special case of Ant Colony Optimization [9]. Additionally, Camacho et al. [10] analyzed the

✉ Luis Velasco
lvelascoe@iingen.unam.mx

[1] Institute of Engineering, Universidad Nacional Autónoma de México, 04510 Ciudad de México, Mexico

[2] ICITECH, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain

Published online: 22 July 2023

components of Gray Wolf Optimizer [11], Firefly Algorithm [12], and Bat Algorihm [13], finding that these metaheuristics are simple reformulations of Particle Swarm Optimization (PSO) [14]. Another example can be found in Tzanetos and Dounias [15], who reviewed multiple metaheuristics and found that many of them have a structure similar to PSO.

The studies above provide a glimpse of a growing problem in the field of metaheuristics: with so many new algorithms being published every year, how could anyone discern those that are genuinely innovative and valuable to the scientific community from those that are not? Such is the seriousness of the matter that some journals have introduced new and stricter conditions for accepting papers that present new metaheuristics. Journal of Heuristics [16] and ACM Transactions on Evolutionary Learning and Optimization [17] state that they do not accept papers that present algorithms with already known ideas or that only demonstrate superior performance without clearly explaining why the algorithm performs well. Furthermore, Swarm Intelligence [18] asks its editorial body to reduce the number of accepted studies that present innovative metaheuristics, only accepting to the review process those with high-quality standards.

Although the controversy over the relevance and innovation of new metaheuristics has been addressed by different publications [1, 2, 15, 19], the truth is that their number only seems to grow year by year. As a reiterative call to reconsider the validity and relevance of this myriad of new algorithms, this paper studies the characteristics of a sample of metaheuristics proposed in papers released in 2022. Despite being a short observation window, the authors were able to find 111 different algorithms self-described as "new", "novel", "advanced", "enhanced", "improved", or alike, which is an average of two proposed algorithms per week. The main objective of this paper is to conduct a review and a critical analysis of recently proposed metaheuristics. For that purpose, the structure of the paper is organized as follows: In Sect. 2, the ways in which new algorithms were proposed are described. It also describes the most commonly added components and strategies to improve their performance. Section 3 gives a brief abstract of the No Free Lunch (NFL) theorem and mentions the cases where the theorem does not hold. Additionally, the validity of the up-the-wall-game argument for proposing new or improved algorithms is discussed. Section 4 describes the procedures and problems used to evaluate the new algorithms and discusses their validity. In Sect. 5, an analysis of two so-called innovative algorithms (Black Widow and Coral Reef Optimization) is conducted to prove how repeated concepts can be masked by a metaphorical language. Finally, Sects. 6 and 7 discuss the results and offer conclusions of the study.

2 On the Interest and Motivation of New Metaheuristics

A sample of 111 recently published papers proposing new metaheuristics was compiled to make this study. To know the interest that these publications have generated in the scientific community, the citations received by the papers that make up the sample were recorded. The number of received citations was obtained using Google Scholar on December 15, 2022. They are shown in Fig. 1 by a box plot of their distribution and their histogram. In the box plots, the sample values between the 25th and 75th percentiles are represented by a box referred to as the interquartile range, with an intermediate line representing the 50th percentile or median. The lines extending from the box have a range equal to 1.5 times the interquartile range; likewise, all values outside that range are called outliers. It is essential to point out that, to facilitate its interpretation, five outliers are omitted in the box plot, which presented values of 16, 18, 21, 25, 30, 35, 37, 51, and 112 citations.

Figure 1 shows that the citations' distribution is positively skewed because many studies have not received any citations yet. Additionally, 50% of the 111 articles in the sample have received three or fewer citations. This value increases to six citations or less when 75% of the total sample is considered. Given that the articles in the sample are less than a year old, it is noteworthy that several of them have already accumulated a considerable number of citations. A remarkable case

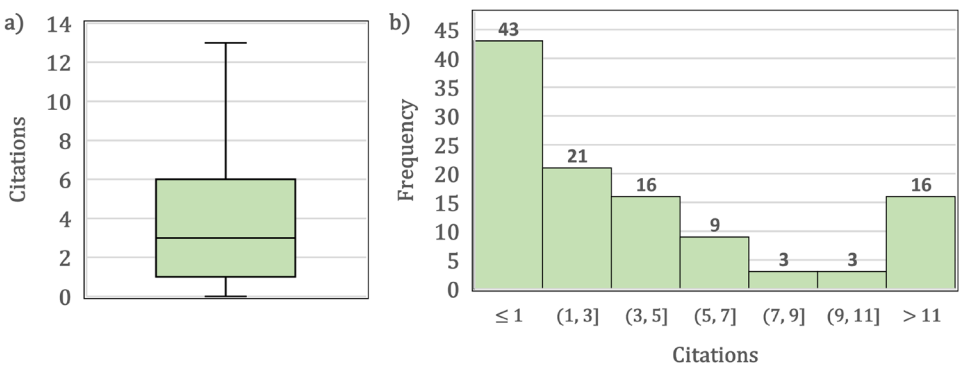

Fig. 1 **a** Box plot; **b** frequency of citations received

is given by the algorithm called Ebola Optimization Search Algorithm [20], which makes a metaphor for the spread of the Ebola virus. This one has received 112 citations at the time of this study. This number of citations gives an idea of the great interest that these studies have generated in the scientific community.

Another aspect addressed during the sample's study was the origin of the proposed algorithm, i.e., whether it was based on a new metaphor, an improved version of an existing one, or a combination of two already established algorithms. The percentages found for each type are presented in Fig. 2. Sörensen [2, 19] highlights a relevant point on this subject: many articles that present new metaheuristics use as theoretical support some metaphor with natural or physical processes that often relate little to optimization. However, it can be seen from Fig. 2 that the trend has now changed. According to data, 65% of the papers present improved versions of existing algorithms. On the other hand, the number of studies that presented a metaheuristic based on a new metaphor or a combination of already established algorithms was 19% and 16%, respectively.

2.1 New, Combined, and Improved Algorithms

In the following subsections, the characteristics of the new, hybrid, and improved algorithms are described in detail. Among the 111 articles consulted, only 21 presented a new algorithm based on metaphors. Figure 3 shows the percentages of the types of metaphors used by the articles in the sample that present new metaheuristics, classified as metaphors based on biology, physics, and human behavior. The largest group, with 62% of the sample, comprises algorithms resembling biological processes; their authors state that they took inspiration from animal [21], plant [22], and even disease [23] behaviors. New algorithms resembling human and physical processes reach 24% and 14%, respectively. In the case of metaphors based on human behavior, some algorithms resemble the migration processes of nomadic tribes [24] or the teaching of the sewing [25] and cooking [26] professions. On the other hand, those that mimic physical processes take inspiration from phenomena such as special relativity [27] or the process of fission and fusion in atomic nuclei [28].

In addition to proposing algorithms based on entirely new metaphors, combining already established metaheuristics is another practice among algorithm designers. For this reason, it was decided to study the most frequently combined algorithms in the sample collected. Only 18 of the 111 articles in the sample proposed an algorithm resulting from combining two different algorithms; this represents 16% of the sample. Figure 4 shows the algorithms combined in the collected sample with their respective percentages of occurrence. The data indicate that 11% of the combined algorithms correspond to the Simulated Annealing and Sine Cosine Algorithms. Kirkpatrick et al. [29] proposed the first of these in 1983, and is one of the best-known metaheuristics today. The second algorithm was proposed by Mirjalili [30] in 2016 and had a formulation based on the sine and cosine functions. The third most employed metaheuristic as a combination element in the sample was Slime Mould Algorithm [31] with 8% of occurrence. It should be noted that those algorithms that had low appearance percentages were grouped in the "other" category.

As shown in Fig. 2, 65% of the new metaheuristics considered in this study are improved versions of others already established. In the literature, it is possible to find multiple operators and strategies employed by the authors to improve

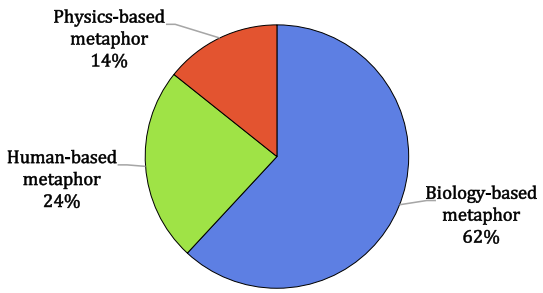

Fig. 3 Metaphor types used in new metaheuristics

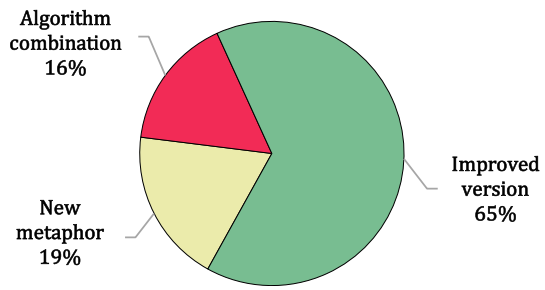

Fig. 2 Percentages of algorithms based on new metaphors, combination of existing algorithms and improved versions

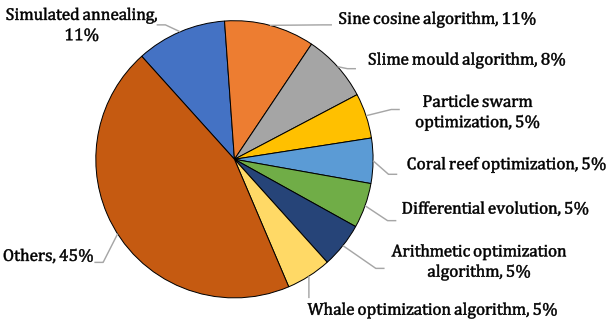

Fig. 4 Combined metaheuristics in the sample

the performance of the algorithms. Those present in the sample collected, with their respective percentages of appearance, are shown in Fig. 5. The most numerous group, with 24% of appearance, is the operator modification, which groups slight changes in the algorithm's formulation. This type of change can be of various kinds, such as changing the algorithm's operation rules or modifying some equation. For example, Xue et al. [32] proposed an improved version of the algorithm called Brainstorm optimization [33], where the strategy for generating new solutions was modified. In the original algorithm, the following equation is used to create new solutions:

$$x_{new}^{i} = x_{old}^{i} + \xi(t) \cdot N(\mu, \sigma^2) \tag{1}$$

where x_{new}^{i} is the new solution to be created, x_{old}^{i} is the solution selected to create a new solution, $N(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ, and $\xi(t)$ is a coefficient that weights the contribution of the normal distribution random variable. In the modified version proposed by Xue et al. [32] it employs any of the following three equations:

$$x_{new}^{i} = x_{old}^{i} + \left(x_{central}^{icluster} - x_{old}^{i}\right) \cdot N(\mu, \sigma^2) \tag{2}$$

$$x_{new}^{i} = x_{old}^{i} + (x_{near} - x_{old}^{i}) \cdot N(\mu, \sigma^2) \tag{3}$$

$$x_{new}^{i} = x_{old}^{i} + (x_{optimal} - x_{old}^{i}) \cdot N(\mu, \sigma^2) \tag{4}$$

Other examples of modifications of similar nature can be found in Li and Xu [34] and Chakraborty et al. [35]. Other frequent modifications to the algorithms are those that consider the addition of opposition-based learning (15%), chaotic maps (12%) and Lévy flight (7%). Opposition-based learning was initially proposed by Tizhoosh [36] in 2005. This strategy is based on the idea that randomly generated solutions tend to have low quality, which causes them to be in the "opposite" position of the problem optima. Therefore, instead of searching in all possible directions for the location of the optimal solution, it is more convenient to search in the "opposite" direction. As Mahdavi et al. [37] point out, multiple versions of this search strategy exist. Another common strategy is using chaotic maps, nonlinear equations exhibiting chaotic behavior. According to Aydemir [38], chaotic maps are beneficial components for optimization algorithms due to their ergodicity and the non-repeatability of chaos. It should be noted that there are dozens of chaotic maps currently in existence. The third most used strategy is the Lévy flight, a random walk considering a Lévy distribution. Viswanathan et al. [39] point out that some animal behaviors seem to follow a Lévy distribution during foraging. It is necessary to remember that foraging is a process where living beings must optimize their resources (energy or time) to avoid starvation. For this reason, it seems that using a Lévy flight strategy is beneficial for optimization algorithms, where solutions must "search" in the configuration space for the optimal solution to the problem. As can be seen, there are other components added to improve the performance of the algorithms, such as restart strategies, Cauchy or Gaussian mutation, or the Nelder-Mead simplex method. Other strategies with a much lower frequency were grouped in "others".

2.2 On Improved Algorithms

Due to the high number of improved algorithms presented recently, this topic is studied more deeply. Two specific cases that draw great attention are the Whale Optimization Algorithm and Slime Mould Optimization due to the high number of "improved" versions they received only in 2022. The whale Optimization Algorithm (WOA) is a population-based algorithm proposed by Mirjalili and Lewis

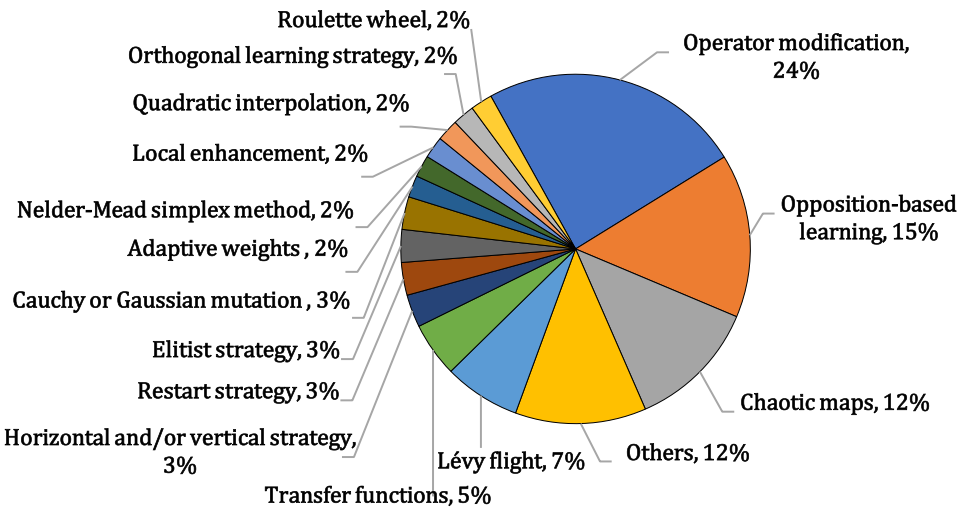

Fig. 5 Commonly added operators and components to improve metaheuristic's performance

[40] based on a metaphor for the hunting behavior of humpback whales. According to its creators, "*WOA algorithm is very competitive compared to the state-of-art meta-heuristic algorithms as well as conventional methods*" [40]. Despite these statements, only in 2022 WOA has been "improved" at least seven times. The list of modifications made to the algorithm is long and usually consists of changes to the algorithm's operators. El-Kenawy et al. [41] combined Sine Cosine Algorithm (SCA) with a modified version of WOA (MWAO) to create Sine Cosine Modified Whale Optimization Algorithm (SCMWOA). Liu and Zhang [42] proposed the Differential Evolution Chaotic Whale Optimization Algorithm (DECWOA) that employs sine chaos theory to create the algorithm's initial population, adaptive weights to update the position of the whales, and a differential variance algorithm for creating new populations. It is important to note that these modifications were made because Liu and Zhang [42] claim that WOA suffers from insufficient search capability and low convergence speed. On the other hand, Ma and Yue [43] proposed an improved whale optimization algorithm (RAV-WOA), which considers adaptive weights on the position of whales in addition to using reverse learning strategy and horizontal and vertical crossover to create the algorithm populations. Likewise, Ma and Yue [43] point out that WOA tends to fall into local optima and has convergence speed and convergence accuracy problems. Qiao et al. [44] introduced the worst individual disturbance (WD) and neighborhood mutation (NM) search strategies with the original WOA to create WDNMWOA. Similarly, Qiao et al [44]. indicate that the modified version aims to solve the problems of low exploratory capacity, a tendency to get trapped into local optima, and low optimization accuracy presented by the original algorithm. Seyyedabbasi [45] combined WOA with Sine Cosine Algorithm (SCA) and Lévy flight (LF) to create a hybrid algorithm called WOASCALF. The decision to combine WOA with SCA was made because the former has low performance in the exploitation phase, an aspect that can be corrected with the high performance shown by SCA in that same phase. Wang et al. [46] developed the Improved Surrogate-Assisted Whale Optimization Algorithm (ISAWOA), which adds three strategies to improve WOA's performance: a surrogate-assisted model, Lévy flight, and quadratic interpolation. Again, Wang et al. [46] indicate that WOA tends to be trapped in local optima. Finally, Wang et al. [47] proposed the Improved Whale Optimization Algorithm (IWOA), which considers logistic chaos to improve WOA's performance.

Slime Mould Algorithm (SMA) is a population-based algorithm proposed by Li et al. [31] that resembles the behavior of slime mould Physarum Polycephalum. Despite being a recent algorithm, just in 2022, it has been modified at least five times. As is typical for this kind of study, the authors of the original SMA reported that the algorithm "*benefits from competitive, often outstanding performance on different search landscapes*" [31]. The following are some of the modifications made to SMA. Ewees et al. [48] proposed Gradient-based Optimizer and Slime Mould Algorithm (GBOSMA), an algorithm obtained by combining Gradient-Based Optimize (GBO) and SMA. In the hybrid GBOSMA algorithm, the current population is operated either by SMA or GBO, with both algorithms having the same probability of being selected to modify the solutions. Kaveh et al. [49] developed an Improved Slime Mould Algorithm (ISMA), which includes an elitist population replacement strategy and a new rule for updating the solutions' position. In this case, Kaveh et al. [49] mention that the modifications were proposed to solve premature convergence problems to non-optimal solutions and the slow convergence rate presented by the original SMA. Örnek et al. [50] combined Sine Cosine Algorithm (SCA) and SMA to develop SCA-SMA. In SCA-SMA, sine and cosine equations are used to update the positions of the population solutions; also, a modified version of the Sigmoid function replaces the arctanh function used in SMA. Likewise, the authors justified the development of the new algorithm by mentioning that SMA tends to get trapped in local optima [50]. Qiu et al. [51] presented a modified version of SMA called Improved Slime Mould Algorithm (ISMA), which employed a Cauchy mutation mechanism with crossover and mutation operators from Differential Evolution (DE). Again, the tendency of the original SMA to get trapped in local optima is mentioned as one of the motivations for developing this new version [51]. Finally, Zhong et al. [52] proposed Teaching– Learning Slime Mould Algorithm (TLSMA), combining two population-based metaheuristics: SMA and Teaching–Learning Based Optimization (TLBO). In TLSMA, the population is divided into two subgroups: the first is operated by TLBO, while SMA operates the second. As the algorithm iterations progress, the SMA population is transferred to the TLBO-operated population to balance both algorithms' exploration and exploitation capabilities. Zhong et al. [52] mention that TLSMA was proposed to solve the problem of diversity loss suffered by SMA.

The multiple versions of WAO and SMA exemplify a long-known flaw in the metaheuristic design field: ignoring that these algorithms are component-based. This means it is possible to combine components from different frameworks to create algorithms that meet the needs of their designers. However, it is necessary to consider the following question: is adding components (operators or strategies) to a framework sufficient to publish the resulting method as an innovative algorithm or even as an improved version of the base algorithm? Answering this question is extremely difficult in the current condition of the field because similar frameworks are hidden under whimsical and unscientific names. To further contrast this idea, consider the following case. Kuyu and

Vatansever [53] and Jia et al. [54] each modified a PSO-like framework by adding a Cauchy-based mutation mechanism and opposition-based learning. Despite working on similar frameworks and adding the same components, the resulting algorithms are considered different since one "analogizes" to a forensic group [55] and the other to a flock of sparrows [56]. This subjective division leads many to consider that two algorithms represent different areas of study just because they have different names [57–60]. Keeping as valid this division between algorithms based more on their names than on their framework's structure would lead to an unnecessary number of algorithms. For example, all the PSO-like algorithms highlighted by Tzanetos and Dounias [61] could be modified by adding a Cauchy mutation mechanism and opposition-based learning, each being considered a different algorithm. Such a succession of studies would be considered frivolous by serious researchers, not only because they mix well-known components but also because they do not enrich our knowledge about optimization algorithms. Therefore, to identify the validity and contribution of these "improved" algorithms, one must first be sure that the underlying framework is genuinely unique and innovative.

Another interesting point observed in both WOA and SMA is that their respective authors reported outstanding performances in the original studies, which disagrees with the arguments outlined to justify the creation of their improved versions (premature convergence, lack of diversity, etc.). Both statements, those claiming superior performance and those alleging performance problems, can be seen as mutually exclusive. However, theory tells us that both viewpoints are likely to be true. A result of great relevance to the field of metaheuristics is the No Free Lunch (NFL) theorem [62], which shows that these metaheuristics (new and improved) are calibrated to solve the problems they were tested. For this reason, it is common to find outstanding performances in these studies. However, when these algorithms are used in different problems for which they were not calibrated, issues such as the lack of diversity or the tendency to be trapped in local optima appear. This point is addressed and explained in detail in Sect. 3.

3 Performance and No Free Lunch (NFL) Theorem

A common motivation employed by metaheuristic designers is to find an algorithm that shows superior performance to all other currently available options. This is known as the up-the-wall-game argument, where the only goal pursued by researchers is to obtain a better result than that achieved by the rest of their peers [19]. A priori, this argument would seem to be sufficient motivation to justify the overwhelming number of new metaheuristics recently proposed. However, the well-known No Free Lunch (NFL) theorem, proposed by Wolpert and Macready [62], directly opposes such an argument. Moreover, the same theorem raises doubts about the superior performances reported in virtually all studies proposing new metaheuristics. To point out these discrepancies between theoretical and reported results, a summary of the formulation by Wolpert and Macready [62] to derive NFL is given below.

3.1 No Free Lunch (NFL) Theorem

Consider a finite search space X associated to a finite space Y through an objective function $f : X \rightarrow Y$ where $Y \subset \mathbb{R}$. Let F be the space of all possible optimization problems. Optimization problems (also referred to as "objective functions") are represented using probability theory and a uniform distribution $P(f)$, defined over F, which gives the probability that each $f \in F$ is the optimization problem under consideration, namely:

$$P(f) = P(f(x_1), f(x_2), ..., f(x_{|X|})) \tag{5}$$

Wolpert and Macready [62] indicate that employing a probability distribution also allows for analyses that consider a single objective function whose uncertainties are encoded in $P(f)$. Likewise, the search algorithms are considered to check a total of m distinct solutions $x \in X$ with their respective objective function evaluations $y = f(x) \in Y$. This sample of solutions x and evaluations y can be defined as a set of different and chronologically ordered configurations $\overline{d}_m$, that is:

$$\overline{d}_m \equiv \left\{ \left(d_m^x(1), d_m^y(1)\right), ..., \left(d_m^x(m), d_m^y(m)\right) \right\} \tag{6}$$

The probability that any search algorithm α has of finding a particular sample $\overline{d}_m$ is dependent on the objective function f, the iteration m and the algorithm itself employed, this is denoted as $P(d_m^y | f, m, \alpha)$. Denoting any pair of algorithms as a and b, the first theorem of NFL states:

$$\sum_f P\left(d_m^y | f, m, a\right) = \sum_f P\left(d_m^y | f, m, b\right) \tag{7}$$

This result implies that, for any metric ϕ considered, the mean performance of any two search algorithms is identical when applied to all possible optimization problems. Theoretically, this result holds even when competing a simple algorithm such as random search with more rational ones such as Genetic Algorithms or Simulated Annealing. Additionally, for equality (7) to hold, high-performance values of an algorithm on a specific set of problems must be paid with low-performance values on the rest of the possible problems. To further clarify this idea, it is recommended to consider the geometric interpretation of NFL. Stavros et al.

[63] pointed out that the performance of a search algorithm α is defined by the inner product of two vectors acting on a particular objective function f.

$$\phi = \overline{v}_{\alpha} \cdot \overline{P} \tag{8}$$

where $\overline{v}_{\alpha} \equiv P(d_m^y | f, m, \alpha)$ contains only the information of the employed search algorithm, while $\overline{P} \equiv P(f)$ describes the uncertainties of the considered objective function f [64]. This interpretation indicates that the algorithm's performance is measured "*by how well it is 'aligned' with the distribution $P(f)$ that governs the problems on which that algorithm is run*" [64]. It is for this reason that high optimization capabilities are claimed for all algorithms, but also a tendency to get trapped in premature local optima is frequent. This is because the proposed algorithms are "aligned" (show high-performance measures) to some specific set of problems they are assessing on, which in turn causes a "misalignment" (show low-performance measures) with the rest of the possible optimization problems.

3.2 Criticism and Free Lunch Theorems

In contrast to NFL, some researchers have claimed that the theorem has limited relevance in practice because it considers in its formulation tight and unrealistic assumptions [65, 66]. The main assumptions on which NFL works are: 1) optimization problems follow a uniform probability distribution; 2) the search space is finite; and 3) the algorithm does not revisit solutions during the search process. On the first assumption, Wolpert [64] has shown that NFL holds even when a set of probability distributions $P(f)$'s are considered; this because the characteristics of the search algorithm employed are not dependent on the distribution $P(f)$ chosen. Although assumption (2) might make it seem that continuous optimization problems are not subject to NFL, the truth is that no computer can create a truly infinite search space up to now, no matter its precision. Assumption (3) is violated in most search algorithms (with some exceptions such as tabu search); however, it has yet to be formally demonstrated that this is a sufficient argument to dismiss the validity of NFL.

Despite researchers' general acceptance of the NFL theorem, there are cases where the theorem does not hold. Köppen et al. [67] indicated that NFL is not valid for cases where one of the search algorithms employs strategy stealing, i.e., scenarios in which an additional move can never be considered a disadvantage. Similarly, Droste et al. [65] have pointed out that NFL does not hold for some black-box optimization scenarios considering complexity theory aspects. Despite this result, the authors point out that the possible performance increments of one algorithm with respect to others are small. Another interesting result is provided by Corne and Knowles [68], who indicated that there are multi-objective optimization algorithms that can perform better than others, i.e., that NFL does not hold for this type of problems. This is due to the different ways in which single and multi-objective algorithms map the search space. While single-objective algorithms keep the best solution so far, multi-objective algorithms create a limited length record with different solutions. This feature allows the existence of a Free Lunch Theorem when considering algorithms with different memory capacities. Additionally, Wolpert and Macready [69] showed that NFL is no longer valid in coevolution and self-play problems, i.e., situations where two competitors collaborate to obtain a champion. Other studies pointing out particular scenarios for which NFL does not hold can be found in Kimbrough et al. [70] and Auger and Teytaud [71].

Interestingly, instead of supporting proposals for new metaheuristics with the above findings, most authors simply ignore this fact of optimization theory. From the 111 articles surveyed for the preparation of this study, 57% omitted any mention of the NFL theorem. On the other hand, in the remaining 43% that mention it, the authors provided an incomplete description or misinterpreted it by using it to support the development of new metaheuristics For example, Braik et al. [72], Gezici and Livatyali [73], Hassan et al. [74], and others mention that researchers are motivated to propose and develop better metaheuristics since NFL prohibits a single algorithm from solving all possible optimization problems. This approach of constantly searching for an algorithm that always performances better than others would be valid if authors focused on the exceptional cases where Free Lunch Theorems hold. However, the norm seems to propose metaheuristics with repeated concepts [15] to solve the same problems. This last point is associated with the next section, where it is reviewed the problems commonly used to demonstrate, in an informal way, the superior performance shown by all the newly proposed metaheuristics.

4 Types of Solved Problems

It is common for newly proposed algorithms to be put into competition with other metaheuristics to demonstrate their optimization capabilities. To learn more about this widespread practice, the problems used to compare the algorithm's performances are analyzed. For this analysis, the sampled articles were divided into four groups: (1) studies focused on solving specific problems, (2) studies that only solve benchmarking problems, (3) studies that solve benchmarking and engineering problems, and (4) studies that consider both benchmarking and specific problems. The first group considers studies where an algorithm is proposed for a particular application. These can be as diverse as object identification in images [75], power grid

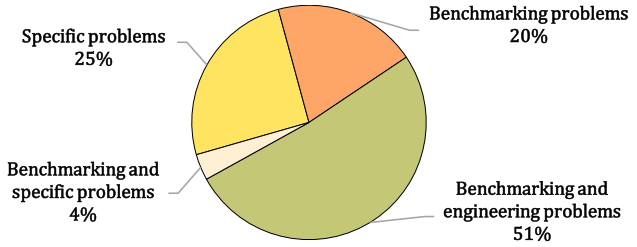

Fig. 6 Percentages of problem types solved in the sample

optimization [76], structural optimization [77, 78], chemical process optimization [79], or disease identification [80]. The second group contains those studies that only solve benchmarking problems proposed for the Congress on Evolutionary Computation (CEC). These problems are sets of continuous and scalable functions, which can be modal or multimodal, single or multi-objective, among other characteristics [81–84]. The third group considers studies that solve both benchmarking and engineering problems. The latter can be the Welded Beam Design Problem, Tension/Compression Spring Design Problem, Pressure Vessel Design Problem, Speed Reducer Design Problem, or others. The last group is cases where benchmarking problems are solved first, and then the proposed algorithm is used to solve a specific problem. Only a few articles treat the last group. The occurrence percentages per group are shown in Fig. 6.

According to Fig. 6, the most frequent studies are those that solve benchmarking and engineering problems, with 51% of the sample. The second largest group is the one where only a specific problem is considered, representing 25% of the sample. On the other hand, the most infrequent studies, with 4%, are those that consider benchmarking and specific problems. According to the data, 75% of the articles studied solve benchmarking problems, while specific problems are addressed in 29% of the sample. A fascinating discussion on the benchmarking problems used to study the performance of search algorithms is presented by Kudela [85]. According to his analysis, the benchmarking problems considered in many studies tend to present their global optimum in the center of the search space. This characteristic causes algorithms with a propensity to search in this area to perform better than those without such bias. For this reason, many of the algorithms that show high performance in benchmarking problems may give poor results in other more general search spaces.

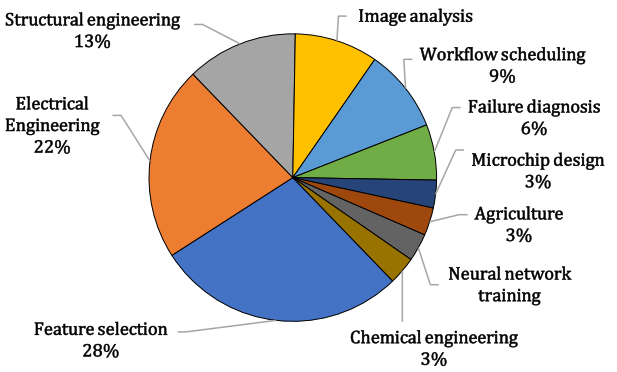

Fig. 7 Specific problem types included in the sample

Additionally, the different types of specific problems included in the sample are shown in Fig. 7 with their respective percentages of occurrence. As can be seen, the specific problems belong to diverse areas, such as chemical engineering, image analysis, and others. The most frequent fields of application are feature selection, electrical engineering, and structural engineering, with 28%, 22%, and 13%, respectively.

Since engineering problems are quite common (they are addressed by half of the sample) and are used to demonstrate that proposed algorithms can solve "real-world" optimization problems, they will be analyzed in detail in the following sections. Additionally, aspects such as the validity of these problems and some inconsistencies among the reported results will be discussed at the end of this section.

4.1 Engineering Problems

Engineering problems are used as a complement in many studies to demonstrate an algorithm optimization capability on real problems. Six engineering problems commonly found in the literature will be analyzed in detail. These are Cantilever Beam Design Problem, Welded Beam Design Problem, Pressure Vessel Design Problem, Tension/Compression Spring Design Problem, Speed Reducer Design Problem, and 3-Bar Truss Design Problem. For each problem, a brief description, its mathematical formulation, and a table comparing the results reported by the sample articles are given in the following sections. It is important to mention that the objective functions of the best solutions found in the following tables are presented respecting the numbers used in the original studies. Additionally, Sect. 4.2 mentions relevant features of the solutions found in the literature.

4.1.1 Cantilever Beam Design Problem

The Cantilever Beam Design Problem consists of defining the beam cross-sections so that its weight is minimized. Initially, this problem considered ten decision variables consisting of the depth and width of five cross-sections [86]. To date, the problem has been simplified to five decision variables representing the width (x_i) of five squared box cross-sections of constant thickness (t). A layout of the problem and a comparison of the best solutions found by state-of-the-art algorithms are presented in Fig. 8 and Table 1, respectively. The problem formulation is as follows:

Consider $\bar{x} = [x_1, x_2, x_3, x_4, x_5]$.
Minimize $f(\bar{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5)$.
Subject to:

$$g_1(\bar{x}) = \frac{60}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \tag{9}$$

where:

$$0.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100 \tag{10}$$

4.1.2 Welded Beam Design Problem

The Welded Beam Design Problem consists of defining the minimum cost of a welded rectangular beam. The problem considers different constraints associated with stresses and deflections [87]. The decision variables of the problem are the thickness (t) and length (l) of the weld, and the dimensions of the beam cross section (b, h). A layout of the problem and a comparison of best solutions found by state-of-the-art algorithms are presented in Fig. 9 and Table 2, respectively. The problem formulation is as follows:

Consider $\bar{x} = [h, l, t, b] = [x_1, x_2, x_3, x_4]$.
Minimize $f(\bar{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)$.
Subject to:

$$\begin{aligned} g_1(\bar{x}) &= \tau(\bar{x}) - 13,600 \leq 0 \\ g_2(\bar{x}) &= \sigma(\bar{x}) - 30,000 \leq 0 \\ g_3(\bar{x}) &= x_1 - x_4 \leq 0 \\ g_4(\bar{x}) &= 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \leq 0 \\ g_5(\bar{x}) &= \delta(\bar{x}) - 0.25 \leq 0 \\ g_6(\bar{x}) &= 6000 - p_c(\bar{x}) \leq 0 \end{aligned} \tag{11}$$

where:

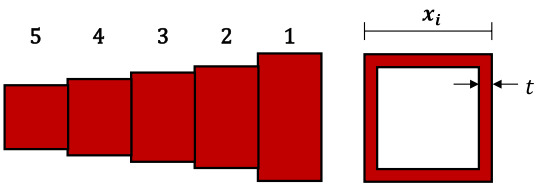

Fig. 8 Cantilever beam design problem layout

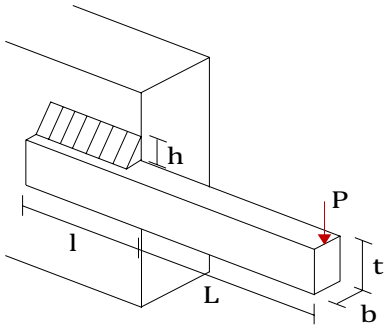

Fig. 9 Welded beam design problem layout

Table 1 Comparison of best solutions found for Cantilever beam design problem

Algorithm	×1	×2	×3	×4	×5	$f(x)$
Örnek et al. [50]	5.947314	4.862986	4.461483	4.472778	3.482863	1.301213
Liu et al. [121]	5.9271	5.3962	4.5081	3.476	2.1726	1.3404
El-Shorbagy and El-Refaey [122]	6.017563478	5.311392298	4.497733036	3.494192221	2.152821173	1.336523225
Zhong et al. [123]	6.0351	4.8313	4.469	3.4503	2.1587	13.0358
Kang et al. [124]	5.883	4.8123	4.6712	3.4479	2.1476	13.0468
Zhang et al. [125]	6.0202	6.0202	4.5124	3.5199	2.1219	1.3366
Wang et al. [126]	5.99349	5.33819	4.501471252	3.4892014	2.152033962	1.340002
Wan et al. [108]	6.044796	4.805171	4.431811	3.47176	2.196531	1.307284

Table 2 Comparison of best solutions found for Welded beam design problem

Algorithm	h	l	t	b	$f(x)$
Dehghani et al. [25]	NA	NA	NA	NA	1.723127
El-Kenawy et al. [41]	0.205604	3.479712	9.041001	0.205739	1.726738
Seyyedabbasi [45]	0.2057	3.4705	9.0366	0.2057	1.7249
Zhong et al. [52]	NA	NA	NA	NA	1.3717
Kuyu and Vatansever [53]	NA	NA	NA	NA	1.724
Braik et al. [72]	0.205729	3.470488	9.036623	0.205729	1.724852
El-Shorbagy and El-Refaey [122]	0.24436898	5.31042797	8.29147139	0.24436898	2.23269378
Zhong et al. [123]	0.2059	3.2665	9.0229	0.2064	1.6997
Zhang et al. [125]	0.20567	3.2542	9.0366	0.20573	1.6953
Wan et al. [108]	0.2043	3.273201	9.104938	0.205632	1.706809
Tang and Zhou [95]	0.205734	3.253036	9.036624	0.20573	1.695245
Qaraad et al. [96]	NA	NA	NA	NA	1.67043804
Lin et al. [127]	NA	NA	NA	NA	1.69998
Xu et al. [128]	0.181328	3.748321	9.036225	0.205748	1.7232
Daliri et al. [93]	NA	NA	NA	NA	1.1764
Wen et al. [129]	0.2062185	3.254893	9.020003	0.206489	1.699058
Zhao et al. [130]	0.1885	3.6205	9.02577	0.20623	1.72002
Hashim and Hussien [131]	0.2057	3.4705	9.0366	0.2057	1.72485193
Zhou et al. [97]	0.2046	3.3319	9.1025	0.2048	1.7245
Zitouni et al. [132]	NA	NA	NA	NA	1.67
Rao et al. [133]	0.205351	3.268419	9.069875	0.205621	1.70163394
Su et al. [94]	0.208	3.38	9.109139	0.23768	1.726769
Wu et al. [134]	0.2057	3.2531	9.0366	0.2057	1.6952
Dehghani and Trojovský [135]	0.20573	3.470489	9.036624	0.20573	1.724852
Trojovska and Dehghani [136]	0.20573	3.4705	9.0366	0.20573	1.7249
Yang et al. [137]	0.2405	5.3885	8.3354606	0.2443461	2.244137

$$
\begin{aligned}
&\tau(\bar{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}\\
&\tau' = \frac{6000}{\sqrt{2}x_1x_2}; \quad \tau'' = \frac{MR}{J}\\
&M = 6000\left(14 + \frac{x_2}{2}\right); \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}\\
&J = \frac{2x_1x_2}{\sqrt{2}}\left(\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right)\\
&\sigma(\bar{x}) = \frac{504,000}{x_4x_3^2}\\
&\delta(\bar{x}) = \frac{65,856,000}{x_4x_3^3(30\times10^6)}\\
&p_c(\bar{x}) = \frac{4.013(30\times10^6)\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{x_3\sqrt{\frac{30\times10^6}{4(12\times10^6)}}}{28}\right)\\
&0.1 \le x_1, x_4 \le 2.0; 0.1 \le x_2, x_3 \le 10.0
\end{aligned}
\tag{12}
$$

4.1.3 Pressure Vessel Design Problem

In the Pressure Vessel Design Problem, a cylindrical vessel capped at both ends by hemispherical heads is designed [87], looking for a geometry that will produce a minimum cost of production. The decision variables for this problem are the shell thickness (T_s), the head thickness (T_h), the inner radius (R), and the length of the vessel excluding the head (L). A layout of the problem and a comparison of best solutions found by state-of-the-art algorithms are presented in Fig. 10 and Table 3, respectively. The problem formulation is as follows:

Consider $\bar{x} = [T_s, T_h, R, L] = [x_1, x_2, x_3, x_4]$.

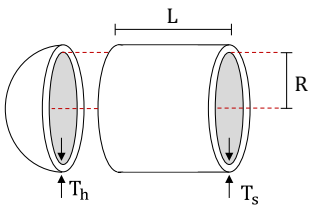

Fig. 10 Pressure vessel design problem layout

Table 3 Comparison of best solutions found for pressure vessel design problem

Algorithm	T_s	T_h	R	L	$f(x)$
Dehghani et al. [25]	NA	NA	NA	NA	5884.882
Qiao et al. [44]	0.8125	0.4375	41.99683	177.9013	6072.161
Seyyedabbasi [45]	0.7815	0.3878	40.4917	197.7095	5897.8
Örnek et al. [50]	0.7782	0.3847	40.3214	199.9756	5885.39056
Braik, Ryalat and Al-Zoubi [72]	12.450698	6.154386	40.319618	199.999999	5885.33277
Gezici and Livatyali [73]	0.8458	0.3094	43.4811	165.6206	5823.2523
Liu et al. [121]	0.8304795	0.3770664	44.00935	154.9557	5982.8355
El-Shorbagy and El-Refaey [122]	0.77816951	0.38464959	40.3196635	199.999377	5885.33425
Zhong, Li and Meng [123]	0.7796	0.3921	40.3598	199.4567	5912.114
Kang et al. [124]	0.8312	0.4084	44.6683	147.1969	5849.9629
Zhang et al. [125]	0.7745491	0.3832039	40.31962	200	5870.124
Tang and Zhou [95]	1.093571	0.001	65.225233	10	2310.11103
Qaraad et al. [96]	NA	NA	NA	NA	4543
Xu et al. [128]	0.8125	0.375	41.98927	182.4983	5982.1768
Wen et al. [129]	0.74257889	0.36838481	40.3338523	199.802664	5735.8501
Zhao et al. [130]	0.8125	0.4375	42.09845	176.6366	6059.714
Hashim and Hussien [131]	0.7819	0.3857	40.5752	196.5499	5887.52977
W. Zhou et al. [97]	0.875	0.04375	45.253	141.2125	6102.9011
Zitouni et al. [132]	NA	NA	NA	NA	6060
Rao et al. [133]	0.754364	0.366375	40.42809	198.5652	5752.40246
Su et al. [94]	1.5	0.6875	61.81515	47.9898	6060.217
Wu et al. [134]	0.7424	0.3702	40.3196	200	5734.9131
Dehghani and Trojovský [135]	0.778027	0.384579	40.31228	200	5882.901
Trojovska and Dehghani [136]	0.7782806	0.3847046	40.32536	199.819	5883.3629
Yang et al. [137]	0.8125	0.4375	42.09776	176.6454	6059.805
Yu et al. [138]	0.8125	0.4375	42.096383	176.662162	6059.9656

Minimize $f(\bar{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$.

Subject to:

$$\begin{aligned} g_1(\bar{x}) &= -x_1 + 0.0193x_3 \le 0 \\ g_2(\bar{x}) &= -x_2 + 0.00954x_3 \le 0 \\ g_3(\bar{x}) &= -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\ g_4(\bar{x}) &= x_4 - 240 \le 0 \end{aligned} \tag{13}$$

where:

$$0 \le x_1, x_2 \le 99; \quad 10 \le x_3, x_4 \le 200 \tag{14}$$

4.1.4 Tension/Compression Spring Design Problem

The Tension/Compression Spring Design Problem consists of minimize the weight of a spring. The constraints of the problem consider aspects like deflection, shear stress and surge frequency. The decision variables are the wire diameter (d), the mean coil diameter (D), and the number of coils (N) [87]. A layout of the problem and a comparison of best solutions found by state-of-the-art algorithms are presented in Fig. 11 and Table 4, respectively. The problem formulation is as follows:

Consider $\bar{x} = [d, D, N] = [x_1, x_2, x_3]$.
Minimize $f(\bar{x}) = x_1^2 x_2 (x_3 + 2)$.
Subject to:

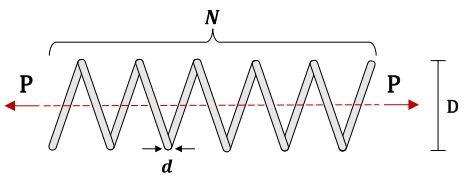

Fig. 11 Tension/compression spring design problem layout

Table 4 Comparison of best solutions found for tension/compression spring design problem

Algorithm	d	D	N	$f(x)$
Dehghani et al. [25]	NA	NA	NA	0.012652
El-Kenawy et al. [41]	0.051232	0.345805	11.95902	0.0126696
Qiao et al. [44]	0.051953	0.363103	10.92515	0.012668
Kuyu and Vatansever [53]	NA	NA	NA	0.0126
Braik et al. [72]	0.051691	0.356777	11.285441	0.012665
Gezici and Livatyali [73]	0.05	0.3189	13.8277	0.012621
Liu et al. [121]	0.050411	0.37384	9.7854	0.011196
El-Shorbagy and El-Refaey [122]	NA	NA	NA	0.012665
Zhong et al. [123]	0.0517	0.3568	11.3132	0.012703
Zhang et al. [125]	0.051868	0.36104	11.04	0.012666
Wan et al. [108]	0.051889	0.361544	11.011088	0.012666
Qaraad et al. [96]	NA	NA	NA	0.01266
Xu et al. [128]	0.051446	0.3508908	11.63895	0.0126663
Wen et al. [129]	0.05	0.37443	8.5497203	0.00987533
Zhao et al. [130]	0.051684	0.356589	11.29651	0.012665
Hashim and Hussien [131]	0.0511	0.3418	12.2222	0.01267254
Zitouni et al. [132]	NA	NA	NA	0.0127
Rao et al. [133]	0.05	0.374396	8.549078	0.009875
Wu et al. [134]	0.05	0.3744	8.5465	0.0099
Dehghani and Trojovský [135]	0.051689	0.356718	11.28897	0.012665
Trojovska and Dehghani [136]	0.0519443	0.362889	10.9361	0.012666
Yang et al. [137]	0.0516536	0.35586235	11.3400917	0.01266607
Yu et al. [138]	0.051762	0.358474	11.18673	0.012665
Zhou et al. [99]	0.05	0.31683	15	0.012764
Hu et al. [98]	0.475047	1.210989	4.938489	0.012754
Azizi et al. [139]	0.05168807	0.35669389	11.2903644	0.01266523

$$
\begin{aligned}
g_1(\bar{x}) &= 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \\
g_2(\bar{x}) &= \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \leq 0 \\
g_3(\bar{x}) &= 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \\
g_4(\bar{x}) &= \frac{x_1 + x_2}{1.5} \leq 0
\end{aligned}
\tag{15}
$$

where:

$$
0.05 \leq x_1 \leq 2; \quad 0.25 \leq x_2 \leq 1.3; \quad 2 \leq x_3 \leq 15 \tag{16}
$$

4.1.5 Speed Reducer Design Problem

The Speed Reducer Design Problem consists of designing a gear train with a minimum weight. Different conditions of stresses in the shafts are considered as constraints. The decision variables of the problem are the following: the face width (x_1), module teeth (x_2), number of teeth in the pinion (x_3), length of the first shaft between bearings (x_4), length of the second shaft between bearings (x_5) and the diameter of first (x_6) and second (x_7) shafts [88]. A layout of the problem and a comparison of best solutions found by state-of-the-art algorithms are presented in Fig. 12 and Table 5, respectively. The problem formulation is as follows:

$$
\begin{aligned}
&\text{Consider } \bar{x} = \left[x_1, x_2, x_3, x_4, x_5, x_6, x_7\right]. \\
&\text{Minimize } f(\bar{x}) = 0.7854 x_1 x_2^2 \left(3.3333 x_3^2 + 14.9334 x_3 - 43.0934\right) \\
&-1.508 x_1 \left(x_6^2 + x_7^2\right)
\end{aligned}
$$

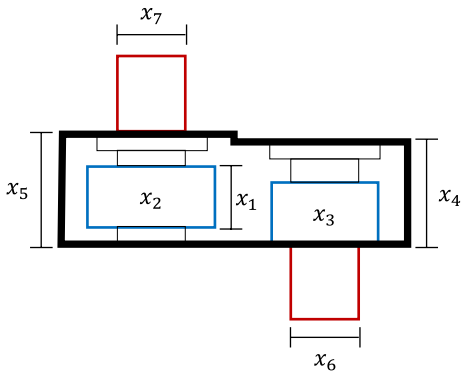

Fig. 12 Speed reducer design problem layout

Table 5 Comparison of best solutions found for speed reducer design problem

Algorithm	×1	×2	×3	×4	×5	×6	×7	$f(x)$
Dehghani et al. [25]	NA	NA	NA	NA	NA	NA	NA	2995.39
Qiao et al. [44]	3.5019	0.7	17	7.351	7.81312	3.350772	5.291852	3001.265
Örnek et al. [50]	3.5	0.7	17	7.3	7.8	3.3434	5.2854	2993.76177
Zhong et al. [52]	NA	NA	NA	NA	NA	NA	NA	3038.6128
Braik et al. [72]	3.5	0.69	17	7.3	7.715319	3.350214	5.286654	2994.47107
Liu et al. [121]	3.49767	0.7	17	7.3	7.8001	3.34982	5.28559	2995.4897
El-Shorbagy and El-Refaey [122]	3.5	0.7	17	7.3	7.715319	3.350214	5.286654	2994.47107
Zhang et al. [125]	3.4976	0.7	17	7.3	7.8	3.35006	5.2857	2995.5447
Qaraad et al. [96]	NA	NA	NA	NA	NA	NA	NA	2992.63
Daliri et al. [93]	NA	NA	NA	NA	NA	NA	NA	1400.2
Wen et al. [129]	3.497571	0.7	17	7.3	7.8	3.350057	5.28554	2995.43745
Hashim and Hussien [131]	3.4976	0.7	17	7.3	7.8	3.3501	5.2857	2995.54244
Zitouni et al. [132]	NA	NA	NA	NA	NA	NA	NA	2990
Su et al. [94]	3.17379	0.7	18	8	8.2	3.792757	5.425506	3018.1654
Wu et al. [134]	3.4975	0.7	17	7.3	7.8	3.35	5.2855	2995.4373
Dehghani and Trojovský [135]	3.5	0.7	17	7.3	7.8	3.350215	5.286683	2996.348
Trojovska and Dehghani [136]	3.5	0.7	17.0001	7.3	7.8	3.35022	5.28673	2996.4342
Yang et al. [137]	3.501	0.7	17	7.326	7.721	3.352	5.286868	2995.821
Yu et al. [138]	3.5	0.7	17	7.3	7.7	3.350215	5.286654	2994.4711
Zhou et al. [99]	3.6	0.7	17	8.3	8.05	3.4	5.375	2891.119
Azizi et al. [139]	3.50001119	0.70000014	17.0000009	7.30070577	7.71553368	3.35054862	5.28665896	2994.44537

$$+7.4777\left(x_6^3+x_7^3\right)+0.7854\left(x_4x_6^2+x_5x_7^2\right)$$

Subject to:

$$g_1(\bar{x})=\frac{27}{x_1x_2^2x_3}-1\le 0 \quad g_2(\bar{x})=\frac{397.5}{x_1x_2^2x_3^2}-1\le 0$$

$$g_4(\bar{x})=\frac{1.93x_5^3}{x_2x_3x_7^4}-1\le 0$$

$$g_5(\bar{x})=\frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2+16.9x10^6}}{110x_6^3}-1\le 0$$

$$g_6(\bar{x})=\frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2+157.5x10^6}}{85x_7^3}-1\le 0$$

$$g_7(\bar{x})=\frac{x_2x_3}{40}-1\le 0 \quad g_8(\bar{x})=\frac{5x_2}{x_1}-1\le 0$$

$$g_9(\bar{x})=\frac{x_1}{12x_2}-1\le 0 \quad g_{10}(\bar{x})=\frac{1.5x_6+1.9}{x_4}-1\le 0$$

$$g_{10}(\bar{x})=\frac{1.1x_7+1.9}{x_5}-1\le 0 \tag{17}$$

where:

$$\begin{aligned}&2.6\le x_1\le 3.6;\quad 0.7\le x_2\le 0.8;\quad 17\le x_3\le 28\\&7.3\le x_4\le 8.3;\quad 7.8\le x_5\le 8.3;\quad 2.9\le x_6\le 3.9\\&5.0\le x_7\le 5.5\end{aligned} \tag{18}$$

4.1.6 3-Bar Truss Design Problem

The 3-Bar Truss Design Problem seeks to minimize the weight of a metallic structure subject to stress constraints [88]. The decision variables are the cross-sectional areas (A_1, A_2) of the elements of the structure. These are grouped as shown in Fig. 13. A layout of the problem and a comparison of best solutions found by state-of-the-art algorithms are presented in the Fig. 13 and Table 6, respectively. The problem formulation is as follows:

Consider $\bar{x}=[A_1,A_2]=[x_1,x_2]$.

Minimize $f(\bar{x})=l\left(2\sqrt{2}x_1+x_2\right)$.

Subject to:

$$\begin{aligned}g_1(\bar{x})&=\sigma_1=\frac{\sqrt{2}x_1+x_2}{\sqrt{2}x_1^2+2x_1x_2}P\le 2\\g_2(\bar{x})&=\sigma_2=\frac{1}{x_1+\sqrt{2}x_2}P\le 2\\g_3(\bar{x})&=\sigma_3=\frac{x_2}{\sqrt{2}x_1^2+2x_1x_2}P\le 2\end{aligned} \tag{19}$$

where:

$$0\le x_1,x_2\le 1;\quad \mathrm{L}=100\,\mathrm{cm};\ \mathrm{P}=2\,kN/\mathrm{cm}^2 \tag{20}$$

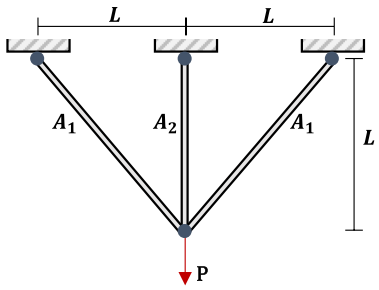

Fig. 13 3-Bar Truss design problem layout

Table 6 Comparison of best solutions found for 3-Bar Truss design problem

Algorithm	×1	×2	$f(x)$
Qiao et al. [44]	0.788987	0.407367	263.8961
Örnek et al. [50]	0.7312	0.3933	246.13978
Gezici and Livatyali [73]	0.7814	0.42822	263.84548
Zhang et al. [125]	0.78865	0.40823	263.8915
Wang et al. [126]	0.788674	0.408251	263.8958427
Wan et al. [108]	0.7860272	0.407114772	263.4634343
Tang and Zhou [95]	0.786848	0.288	186.3859
Qaraad et al. [96]	NA	NA	263.46343
Zitouni et al. [132]	NA	NA	264
Rao et al. [133]	0.788413	0.408121	263.8523
Wu et al. [134]	0.7884	0.4081	263.8523464
Yang et al. [137]	0.788733	0.408084	263.8958
Hu et al. [98]	0.76598	0.756812	264.0072

4.2 Observations on Engineering Problems

A noteworthy point about the engineering problems is presented in Sect. 4.1. is that they are not solved in a properly engineering way by any study. In these problems, the decision variables are composed of continuous values, which is inconsistent with the discrete and standardized values required by professional practice [89–92]. For this reason, the solutions provided to the problems are quite absurd. For example, has anyone seen a commercially available 0.408084 cm^2 bar like the one proposed for the 3-Bar Truss Design Problem? Or a 0.38464959 cm thick plate as the one recommended to be used in the Pressure Vessel Design Problem? Or a welder capable of welding with a thickness of 8.29147193 mm, as indicated in the Welded Beam Design Problem? Or can anyone explain what is meant by a spring with 11.285441 coils? It is evident that these kinds of results are explained by the fact that the proposed metaheuristics are born as algorithms focused on continuous problems. While it is true that modifying the formulation of an algorithm to solve discrete problems is a complicated task, this does not justify their authors claiming high capabilities to solve "real-world" optimization problems when this is not being authentically demonstrated. This characteristic of "solving" engineering problems that may not have a relation whatsoever with professional practice is just another example of the lack of rigor shown by this kind of study.

Another finding was that some studies present results that differ from those obtained when evaluating the objective function. The results reported for the Cantilever Beam Design Problem (see Table 1) are a clear example. As can be seen, two algorithms obtained structural weights of around 13 units, while the rest reported 1.3 units. When evaluating the objective function, it was found that the correct solutions presented an order of magnitude of 13 weight units. Other less obvious discrepancies in the reported results were found for the remaining problems. For the Welded Beam Design Problem, two solutions were found whose costs differed considerably from the other solutions. Zhong, Li and Meng [52] and Daliri et al. [93] reported solutions with a cost of 1.37 and 1.17, respectively, contrasting with the mean cost of 1.72 reported by other studies. Unfortunately, both studies omitted the values of their decision variables, so these results could not be verified. For the Pressure Vessel Design Problem, discrepancies were found as follows. Braik et al. [72] report a solution with a value of 5885 units; however, when evaluating the objective function, the actual result is 302 448 units. Su et al. [94] report a solution cost of 6060 units; nevertheless, when evaluating the objective function, the actual cost was 10 541 units. Tang and Zhou [95] and Qaraad et al. [96] presented a solution with a meager cost of 2310 and 4543 units, respectively. In the first case, it was found that such a solution does not meet the constraints $g_1(\overline{x})$ and $g_2(\overline{x})$, while the solution presented by Qaraad et al. [96] could not be verified since the values of the decision variables are omitted. Similarly, W. Zhou et al. [97] reported a solution whose decision variables do not satisfy the constraint $g_2(\overline{x})$ of the problem. For the Tension/Compression Spring Design Problem, Hu, Du, and Wang [98] reported a solution with a weight equal to 0.0127. When the objective function was evaluated, its actual weight was 1.89. For the Speed Reducer Design Problem, Daliri et al. [93] reported a solution with a weight of 1400 units, which is about half the weight of the other reported solutions. This result could not be verified because the study did not provide the values found for the decision variables. For the same problem, X. Zhou et al. [99] found a solution with a cost of 2891; when evaluating the objective function, it was found that the real cost was equal to 3120. Finally, for the 3-Bar Truss Design Problem, Tang and Zhou [95] reported a solution weighing 186, but the objective function evaluation gives a weight of 251. These discrepancies raise doubts about whether the

same problems are being resolved in the aforementioned studies and the adequacy of the review process carried out prior to publication.

A widely used argument to justify the creation of new metaheuristics is the need for more powerful algorithms to solve optimization problems. Since the considered engineering problems have been solved for at least two decades, a performance comparison between the results reported by older papers and the solutions reported in the sample is possible. To make this comparison, the mean and the standard deviation (SD) of the solutions reported by the so-called "state-of-the-art" algorithms are contrasted with those reported by papers published more than 20 years ago. The results are shown in Table 7. It is important to note that values that presented discrepancies with the objective function evaluation and those whose veracity could not be verified were not considered in the calculation. Also, the comparison of the Cantilever Beam Design Problem is omitted because its approach changed over the years. The following nomenclature is used: Welded Beam Design Problem (WB), Pressure Vessel Design Problem (PV), Tension/Compression Spring Design Problem (TCS), Speed Reducer Design Problem (SR), and 3-Bar Truss Design Problem (3B). As can be seen, the results obtained more than 20 years ago for the engineering problems considered are very similar to those reported by state-of-the-art algorithms. The Tension/Compression Spring Design Problem presented the highest performance increase, where 3.2% more economical solutions were found than the one reported by Coello and Montes [100] in 2002. At the other extreme is the Welded Beam Design Problem, where the solution found by Coello [87] in 2000 saves 0.5% more material than the average of the recently proposed algorithms. Regarding standard deviations, the most significant difference is observed in the Pressure Vessel Design Problem, where there is a difference of 1.4SD between the mean and the solution found by Coello and Montes [100]. For the rest of the problems, there is a marginal difference ($\leq$0.4SD) between the mean and the solutions reported by older articles.

These little performance differences reflect a stagnation of algorithm development and call into question the continuous improvement reported by metaheuristic designers today. As mentioned earlier, some recently proposed algorithms are noted for repeating concepts and strategies employed by previously presented frameworks. This recycling of concepts may be the cause of older algorithms presenting similar performances to those recently proposed. To exemplify how recent algorithms present already known components under different names, the following section reviews Black Window Optimization and Coral Reef Optimization algorithms.

5 Black Widow and Coral Reef Optimization

As mentioned in Sect. 2, many algorithms present identical formulations to others previously proposed. Unfortunately, these similarities can go unnoticed for years because they are hidden by a metaphorical language. In such cases, a solution may be named an empire, a remora, a whale, a black hole, or others. Something similar happens with the components used by the algorithms. To understand the true innovation of these algorithms, Sörensen [19] points out that it is necessary to go through a deconstruction process rather than relying solely on the algorithm's name or the analogy on which it is based. In the deconstruction process [101–104], the contributions (benefits and disadvantages) of each algorithm component are analyzed to distinguish between the truly innovative components and those shared with other algorithms.

In this section, the deconstruction of two algorithms that have recently presented modified versions is performed, namely: Black Widow Optimization [105] and Coral Reef Optimization [106]. Both algorithms were born from an analogy with the life cycle of the species they are named after. The analysis process will show that they not only have a solid resemblance to Genetic Algorithms [107] but also that some of their components make them more complex but not necessarily more efficientl.

Table 7 Comparison between older studies and the mean of the sample

	WB	PV	TCS	SR	3B
Sample mean	1.75637103	5912.4989	0.012293503	2993.2429	262.2491
Standard deviation (SD)	0.1492	107.0076	0.0009	26.4044	5.1081
Older study	1.74830941 [87]	6059.9464 [100]	0.012681000 [100]	2996.426 [140]	264.3763 [88]
(year)	(2000)	(2002)	(2002)	(1985)	(2001)
Difference	−0.5%	2.5%	3.2%	0.1%	0.8%
	(−0.1SD)	(1.4SD)	(0.4SD)	(0.1SD)	(0.4SD)

5.1 Black Widow Optimization

Black Widow Optimization is a population-based algorithm proposed by Hayyolalam and Pourhaji [105], inspired by the life cycle of black widow spiders. By December 15, 2022, Google Scholar indicated that the study where this algorithm was presented had obtained 329 citations. Additionally, in 2022, four improved versions of the algorithm were presented [98, 108–110]. The formulation of the algorithm, in the words of its creators, is as follows. The algorithm starts by creating an initial population of N_{pop} spiders. From this population of spiders, the next generation's parents are taken randomly. Because cannibalism exists among black widows, this algorithm considers three possibilities: (1) the female spider eats the male during mating, (2) the offspring eat each other, and (3) the offspring eat the mother. In all three cases, the value of the objective function is used to define which individuals will be cannibalized. Additionally, a mutation process after mating is used. At this point, anyone with knowledge of Genetic Algorithms can identify the similarities to Black Widow Optimization. However, to make it more straightforward, the components of the algorithm are cleaned of metaphorical vocabulary. The comparison between Genetic Algorithms and Black Widow Optimization is shown in Table 8.

As seen from Table 8, the original Black Widow Optimization formulation has the same components as Genetic Algorithms but adds a cannibalism operator, which is redundant. Arguably, Black Widow Optimization's contribution to the optimization field is that it introduces the destruction of low-quality solutions. However, this additional step is required for Black Widow Optimization because the algorithm's selection operator is random, i.e., there is no criterion based on the solution's quality to define which ones pass on their characteristics to the next generation. For this reason, many next-generation solutions are low-quality and must be destroyed somehow. If these low-quality solutions were not destroyed, their characteristics would negatively affect the quality of the following populations. This is an example of how blind recombination of components can give rise to more complex but not more efficient algorithms. It is needless to say, since the original version of Black Widow Optimization is a deficient version of Genetic Algorithms, then, by extension, its recently published improved versions are as well.

5.2 Coral Reef Optimization

Coral Reef Optimization is a population-based algorithm proposed by Salcedo et al. [106]. In the authors' words, the algorithm "*simulates a coral reef, where different corals (solutions to the optimization problem considered) grow and reproduce in coral colonies, fighting by choking out other corals for space in the reef*" [106]. By December 15, 2022, the original study where Coral Reef Optimization was presented had 195 citations registered in Google Scholar. Similarly to the Black Widow Optimization algorithm, two modified versions of the algorithm have been published just in 2022 [111, 112]. The formulation of the algorithm is as follows. The first step is to create a grid of size NM which serves to model an artificial reef. Each of the possible positions of the grid is a space where corals will be housed. The initial population of the algorithm is randomly distributed on the artificial reef leaving some empty spaces. The study points out that there are three ways in which corals reproduce: (1) Broadcast Spawning, (2) Brooding, and (3) Budding or Fragmentation. The first type occurs when corals release gametes into the water together; under this form, new corals are created once two reproductive cells meet. The second form of reproduction is like the first one, with the difference that the encounter between gametes occurs inside a coral. The new coral is released after having partially developed inside its parent. The last form of reproduction occurs when a new coral is born from the separation of a single coral. These three forms of reproduction are considered in the algorithm. Once a new coral has been created, it looks for a place to stay on the artificial reef. If it finds an empty space, it automatically occupies it. If another coral occupies

Table 8 Comparison between black widow optimization and genetic algorithms

Step	Black widow optimization (with metaphorical language)	Black widow optimization (without metaphorical language)	Genetic algorithms
1	Create a population of N_{pop} spiders	Create a set of N_{pop} initial solutions	Create a set of N_{pop} initial solutions
2	Select parents randomly	Select two solutions randomly	Select two solutions, either by roulette method, ranking, tournament, etc
3	Procreate offspring	Combine solutions to create new ones	Combine solutions to create two new ones
4	Apply one of three forms of cannibalism	Destroy solutions according to the value of their objective function	–
5	Mutation	Modify a percentage of the new solutions randomly	Modify a percentage of the new solutions randomly
6	Update population	Update the solution set	Update the solution set

the space, the values of the objective functions are compared to define whether the new coral is rejected or replaces the old one. Each new coral has k chances to find a position on the reef. At the end of the algorithm iteration, those lower-quality corals have a probability f_d to be eliminated; this step is justified as a natural predation process on the coral reef. Clearly, this algorithm presents a more complex formulation than Black Widow Optimization; however, eliminating the metaphorical language helps to study each of its components. Table 9 presents a comparison between Coral Reef Optimization and Genetic Algorithms.

As seen from Table 9, most of the components of Coral Reef Optimization have an equivalent in Genetic Algorithms. The core component of the algorithm, the artificial reef, is nothing more than an alternative way of storing the solutions in a two-dimensional space. In Genetic Algorithms, this is done by a one-dimensional vector. Additionally, because the solutions "*must fight for space in the reef*" [106] in step 6, it is necessary to introduce the predation operator in step 7. It is worth noting that steps 6 and 7 are more closely similar to Evolutionary Strategies (ES). In the ES-$(\mu + 1)$ version, the algorithm creates a population of size μ and a single solution per iteration. Once the solution is created, it is compared to the lowest-quality solution in the current population. If the new solution has a higher quality than the worst solution, the former replaces the latter. In the case of Coral Reef Optimization, this process is performed probabilistically in step 6, i.e., the new solution created is not directly compared with the worst of the current population. Clearly, the strategy employed by Coral Reef Optimization is not efficient. Since there is a low probability that the worst solution will be selected to be compared with the new one, it becomes necessary to create a complementary strategy that increases the chances of discarding the worst quality solutions and thus prevent their characteristics from being transferred to the new generation. This is done in step 7 by the predation operator. Like the case of Black Widow Optimization, Coral Reef Optimization introduces additional steps that aim to correct the shortcomings of a formulation where different evolutionary operators are simply mixed carelessly.

6 Discussion

Multiple flaws in studies where new metaheuristics are proposed were pointed out throughout the paper. These flaws include a lack of knowledge of the NFL theorem and its implications, the recycling of well-known ideas and strategies that are presented as new, the unrealistic "real-world" problems they solve, the low standards required during the reviewing process, or even the limited increase in performance that the algorithms have shown in the last 20 years. Based on the obtained results, the authors agree with the position of Aranha et al. [1], who considered that the dozens of new algorithms proposed each year are symptoms of a lack of scientific rigor rather than an authentic advance in the field. As demonstrated in the cases of Black Widow Optimization and Coral Reef Optimization, algorithms that do not present any innovation can be considered new frameworks for years. If we add to these studies their "improved" versions, what we have is a problem that multiplies itself

Table 9 Comparison between coral reef optimization and genetic algorithms

Step	Coral reef optimization (with metaphorical language)	Coral reef optimization (without metaphorical language)	Genetic algorithms
1	Create a grid with size NM	Define a matrix of size NM that will store the solutions to the problem	Define a vector of size N that will store the solutions to the problem
2	Create an initial population and randomly place them on the grid	Create an initial population and store it randomly in the matrix of step 1, leaving some empty spaces	Create an initial population of size N
3	Define the number of corals to be spawned by Broadcast Spawning (spawning between two corals)	Select two solutions and combine them to create a new one	Apply selection and crossover operators on two solutions to create a new one
4	The rest of the corals that are not reproduced by Broadcast Spawning will be reproduced by Brooding (asexual reproduction with mutation)	Selecting a solution and mutating it to create a new one	Applying only the mutation operator on a solution to create a new one
5	Higher quality corals have a f_a probability of reproducing by budding or fragmentation (asexual reproduction without mutation)	Applying an elitist strategy with probability f_a	Applying an elitist strategy
6	Each new coral has k chances of finding a place in the reef	Each solution created has k chances to find a space in the matrix of step 1 to be stored	–
7	Poorer quality corals have a f_b probability of being eliminated by predation	Destroy low-quality solutions with a probability f_b	–

yearly. Of course, the best option to solve this problem is not to analyze study by study once published but to identify the lack of scientific rigor during the review process or before sending them to reviewers. It is therefore evident that to solve this problem, it is necessary to raise awareness of the subject and increase the standards required for this type of study.

It is important to remember that research is a human activity and, therefore, is not indifferent to the motivations and interests of those who carry it out. A well-known example that demonstrates the link between scientific endeavor and the researcher's interests is the "publish or perish" philosophy. In this working way, researchers are under constant pressure to demonstrate high-performance metrics, either by number of publications or by number of citations received. As several studies point out [113–115], achieving high-performance metrics brings rewards such as greater prestige among peers but also economic benefits in the form of salary improvements or research funds. Because of this relationship between performance metrics and economic stability, researchers are willing to publish many scientifically weak papers despite multiple calls to stop that practice. In the case of new metaheuristic development, algorithm designers engage in questionable practices that help to provide an image of innovation to the work submitted but damage the field of research. Undoubtedly, the most harmful practice is the one where new algorithms are reinvented under new names, and their components are hidden under a metaphorical language. This practice not only generates fictitious areas of study [57–60] but also opens the door to "propose" improvements that have already been made.

The results obtained in this study reveal that this field of research must establish an evaluation framework to discern between speculative algorithms and those that genuinely provide new concepts and optimization strategies. Multiple authors have approached this topic from different perspectives to develop such an evaluation framework. Hooker [116] provides an interesting discussion revealing the drawbacks of comparing algorithms based on their execution time or the performance achieved in benchmark problems. Corstjens et al. [117] propose a methodology to evaluate the influence of the different algorithm components on its performance. Likewise, Campelo and Wanner [118] presented a statistical method to determine the experimental sample size when comparing different algorithms. Tzanetos and Dounias [15] proposed creating databases containing real optimization problems with which to compare the proposed algorithms. Finally, Franzin et al. [119] proposed a causal framework to explain the performance and results obtained by a search algorithm. Clearly, it is possible to develop a suitable evaluation framework based in whole or in part on the above works. However, the authors believe that an adequate evaluation framework should be based more on identifying the contributions of the algorithm's components than on the performances they achieve on benchmark problems. This is because there are several cases where a metaphorical language is used to hide an already-known formulation and present it as a totally new algorithm. In such cases, the "new" algorithm would perform identically to the algorithm it emulates, making a performance-based evaluation framework unenlightening. Additionally, it is well known that different configurations of an algorithm's hyperparameters can generate different performances [120]. This, in turn, can be exploited to create samples where the compared algorithms show poor performances, thus creating a favorable picture for the proposed new algorithm. All these problems can be avoided if the discussion focuses more on the theoretical contributions provided by the new algorithms rather than a competition between their performances.

7 Conclusions

In this paper, an analysis of the characteristics presented by a sample of 111 recent papers where metaheuristics described as "new", "novel", "advanced", "improved", or similar was conducted. Different aspects were studied, such as the number of citations received, the origin of the proposed optimization algorithms, as well as the problems they solve. Valuable conclusions emerged from the analysis. These are listed below.

- There is currently a trend to develop improved versions of established algorithms. Unfortunately, these studies present a deficient level of innovation because they only recombine well-known optimization components.
- The names used by algorithm designers to label their creations do not serve to catalog their features clearly and concisely. Moreover, they only generate a fictitious research field where the proposed algorithm is compared with others bearing the same name, avoiding contrasting frameworks with scientific rigor and objectivity.
- It is common for metaheuristic designers to have limited knowledge of the NFL theorem, which in turn causes them to erroneously use it as an argument in favor of proposing ever more powerful algorithms.
- Although some cases allow the existence of Free Lunch Theorems, they are simply ignored by metaheuristic designers, representing an unexploited field of research.
- The constant claims of algorithms with outstanding performance seem to be more a literary resource to get a new algorithm published than a fact. This is especially evident in the high number of "improved" versions that

seek to fix the flaws exhibited by algorithms that were once presented as superior.
- Despite the claim that the new proposed metaheuristics can solve "real world" problems, the truth is that the engineering problems they solve are far from being applicable in real practical situations.
- Studies that propose metaheuristics, whether new or improved, may have been subjected to poor review processes.
- From the review of their components, it was shown that some algorithms, like Black Widow Optimization and Coral Reef Optimization, are not really innovative but rather inefficient mixtures of evolutionary operators. By extension, this may also apply to their recently released improved versions.
- It is necessary that both editors and reviewers raise the standards required for this kind of study to only allow the publication of those that effectively present innovative ideas that enrich the research field.

Acknowledgements The first and second authors acknowledge the facilities provided by the Institute of Engineering, UNAM, Mexico, to develop this study.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Aranha C et al (2021) Metaphor-based metaheuristics, a call for action: the elephant in the room. Swarm Intell. https://doi.org/10.1007/s11721-021-00202-9
2. Sörensen K, Sevaux M, Glover F (2018) A history of metaheuristics. In: Martí R, Pardalos P, Resende M (eds) Handbook of heuristics. Springer
3. Weyland D (2010) A rigorous analysis of the harmony search algorithm: how the research community can be misled by a "novel" methodology. Int J Appl Metaheuristic Comput 1:50–60
4. Weyland D (2015) A critical analysis of the harmony search algorithm-How not to solve sudoku. Oper Res Perspect 2:97–105
5. Geem Z, Hoon J, Loganathan G (2001) A new heuristic optimization algorithm: harmony search. SIMULATION 76:60–68
6. Rechenberg I (1973) Evolutionsstrategie: optimierung techischer systeme nach prinzipien der biologischen evolution. Frommann Holzboog Verlag, Stuttgart
7. Camacho-Villalón CL, Dorigo M, Stützle T (2019) The intelligent water drops algorithm: why it cannot be considered a novel algorithm: a brief discussion on the use of metaphors in optimization. Swarm Intell 13:173–192
8. Shah H (2008) Intelligent water drops algorithm: a new optimization method for solving the multiple knapsack problem. Int J Intell Comput Cybern 1:193–212
9. Colorni A, Dorigo M, Maniezzo V (1991) Distributed optimization by ant colonies. In Proceedings of the first european conference on artificial life.
10. Camacho C, Stützle T, Dorigo M (2020) (2020) Grey wolf, firefly and bat algorithms: Three widespread algorithms that do not contain any novelty. In: Dorigo M, Stützle T, Blesa MJ, Blum C, Hamann H, Heinrich MK, Strobel V (eds) International conference on swarm intelligence. Springer, Cham, pp 121–133
11. Mirjalili S, Mirjalili S, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
12. Yang X (2009) Firefly algorithms for multimodal optimization. In: Watanabe O, Zeugmann T (eds) SAGA 2009, vol 5792. Springer, Berlin, pp 169–178
13. Yang X (2010) A new metaheuristic bat-inspired algorithm. In: González J, Pelta D, Cruz C, Terrazas G, Krasnogor N (eds) Nature inspired cooperative strategies for optimization (NICSO 2010): studies in computational intelligence, vol 284. Springer
14. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proc. ICNN 1995 Int Conf Neural Netw 4:1942–1948
15. Tzanetos A, Dounias G (2021) Nature inspired optimization algorithms or simply variations of metaheuristics? Artif Intell Rev 54:1841–1862
16. Journal of Heuristics (2015) Policies on heuristic search. https://www.springer.com/journal/10732/updates/17199246
17. ACM Transactions on Evolutionary Learning and Optimization (2022) Author Guidelines.
18. Dorigo M (2016) Swarm intelligence: a few things you need to know if you want to publish in this journal. Swarm Intell
19. Sörensen K (2013) Metaheuristics-the metaphor exposed. Int Trans Oper Res 22:3–18
20. Oyelade ON, Ezugwu AES, Mohamed TIA, Abualigah L (2022) Ebola optimization search algorithm: a new nature-inspired metaheuristic optimization algorithm. IEEE Access 10:16150–16177
21. Braik M, Hammouri A, Atwan J, Al-Betar MA, Awadallah MA (2022) White shark optimizer: a novel bio-inspired metaheuristic algorithm for global optimization problems. Knowl-Based Syst 243:108457
22. Rahmani AM, AliAbdi I (2022) Plant competition optimization: a novel metaheuristic algorithm. Expert Syst. https://doi.org/10.1111/exsy.12956
23. Khalid AM, Hamza HM, Mirjalili S, Hosny KM (2022) BCOVIDOA: a novel binary coronavirus disease optimization algorithm for feature selection. Knowl-Based Syst 248:108789
24. Lin N et al (2022) A novel nomad migration-inspired algorithm for global optimization. Comput Electr Eng 100:107862
25. Dehghani M, Trojovská E, Zuščák T (2022) A new human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training. Sci Rep 12:1–25
26. Trojovská E, Dehghani M (2022) A new human-based metahurestic optimization method based on mimicking cooking training. Sci Rep 12:1–25
27. Goodarzimehr V, Shojaee S, Hamzehei-Javaran S, Talatahari S (2022) Special relativity search: a novel metaheuristic method

based on special relativity physics. Knowledge-Based Syst 257:109484
28. Nouhi B et al (2022) The fusion–fission optimization (FuFiO) algorithm. Sci Rep 12:1–44
29. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
30. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
31. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Futur Gener Comput Syst 111:300–323
32. Xue Y, Zhang Q, Zhao Y (2022) An improved brain storm optimization algorithm with new solution generation strategies for classification. Eng Appl Artif Intell 110:104677
33. Shi Y (2011) Brain storm optimization algorithm. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **6728 LNCS:**303–309.
34. Li Y, Xu F (2022) Acoustic emission sources localization of laser cladding metallic panels using improved fruit fly optimization algorithm-based independent variational mode decomposition. Mech Syst Signal Process 166:108514
35. Chakraborty S, Nama S, Saha AK (2022) An improved symbiotic organisms search algorithm for higher dimensional optimization problems. Knowl-Based Syst 236:107779
36. Tizhoosh H (2005) Opposition-based learning: a new scheme for machine intelligence. In International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06), pp. 695–701. https://doi.org/10.1109/CIMCA.2005.1631345.
37. Mahdavi S, Rahnamayan S, Deb K (2018) Opposition based learning: a literature review. Swarm Evol Comput 39:1–23
38. Aydemir SB (2022) A novel arithmetic optimization algorithm based on chaotic maps for global optimization. Evol Intell. https://doi.org/10.1007/s12065-022-00711-4
39. Viswanathan GM et al (2000) Levy flights in random searches. Phys A Stat Mech its Appl 282:1–12
40. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
41. El-Kenawy ESM et al (2022) Novel meta-heuristic algorithm for feature selection, unconstrained functions and engineering problems. IEEE Access 10:40536–40555
42. Liu L, Zhang R (2022) Multistrategy improved whale optimization algorithm and its application. Comput Intell Neurosci 2022:1–16
43. Ma G, Yue X (2022) An improved whale optimization algorithm based on multilevel threshold image segmentation using the Otsu method. Eng Appl Artif Intell 113:104960
44. Qiao S et al (2022) Individual disturbance and neighborhood mutation search enhanced whale optimization: performance design for engineering problems. J Comput Des Eng 9:1817–1851
45. Seyyedabbasi A (2022) WOASCALF: a new hybrid whale optimization algorithm based on sine cosine algorithm and levy flight to solve global optimization problems. Adv Eng Softw 173:103272
46. Wang S, Hu W, Riego I, Yu Y (2022) Improved surrogate-assisted whale optimization algorithm for fractional chaotic systems ' parameters identification. Eng Appl Artif Intell 110:104685
47. Wang Y, Zhang Y, Xu D, Miao W (2022) Improved whale optimization-based parameter identification algorithm for dynamic deformation of large ships. Ocean Eng 245:110392
48. Ewees AA, Ismail FH, Sahlol AT (2023) Gradient-based optimizer improved by Slime Mould algorithm for global optimization and feature selection for diverse computation problems. Expert Syst Appl 213:118872
49. Kaveh A, Biabani K, Kamalinejad M (2022) Improved slime mould algorithm with elitist strategy and its application to structural optimization with natural frequency constraints. Comput Struct 264:106760
50. Örnek BN, Aydemir SB, Düzenli T, Özak B (2022) A novel version of slime mould algorithm for global optimization and real world engineering problems: enhanced slime mould algorithm. Math Comput Simul 198:253–288
51. Qiu F et al (2022) Mutational Slime Mould algorithm for gene selection. Biomedicines 10:1–37
52. Zhong C, Li G, Meng Z (2022) A hybrid teaching–learning slime mould algorithm for global optimization and reliability-based design optimization problems. Neural Comput Appl 34:16617–16642
53. Kuyu YÇ, Vatansever F (2022) Modified forensic-based investigation algorithm for global optimization. Eng Comput 38:3197–3218
54. Jia J et al (2022) Improved sparrow search algorithm optimization deep extreme learning machine for lithium-ion battery state-of-health prediction. iScience 25:103988
55. Chou JS, Nguyen NM (2020) FBI inspired meta-optimization. Appl. Soft Comput. J. 93:106339
56. Xue J, Shen B (2020) A novel swarm intelligence optimization approach: sparrow search algorithm. Syst Sci Control Eng 8:22–34
57. Salcedo-Sanz S (2017) A review on the coral reefs optimization algorithm: new development lines and current applications. Prog Artif Intell 6:1–15
58. Makhadmeh SN et al (2022) Recent advances in butterfly optimization algorithm, its versions and applications. Arch Comput Methods Eng. https://doi.org/10.1007/s00521-022-07704-5
59. Shehab M et al (2022) Harris hawks optimization algorithm: variants and applications. Arch Comput Methods Eng 29:5579–5603
60. Negi G, Kumar A, Pant S, Ram M (2021) GWO: a review and applications. Int J Syst Assur Eng Manag 12:1–8
61. Tzanetos A, Dounias G (2020) A comprehensive survey on the applications of swarm intelligence and bio-inspired evolutionary strategies. In: Tsihrintzis GA, Jain LC (eds) Machine learning paradigms: advances in deep learning-based technological applications. Springer, Cham
62. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1:67–82
63. Stavros P, Alexandropoulos S, Pardalos P, Michael N (2019) No free lunch theorem: a review. In: Demetriou I, Pardalos P (eds) Approximation and optimization. Springer, Cham, pp 57–82
64. Wolpert DH (2012) What the no free lunch theorems really mean; how to improve search algorithms. Sfi Work. Pap. 2012–10–17, pp. 1–13
65. Droste S, Jansen T, Wegener I (2002) Optimization with randomized search heuristics—the (A)NFL theorem, realistic scenarios, and difficult functions. Theor Comput Sci 287:131–144
66. Yang X-S (2012) Swarm-based metaheuristic algorithms and no-free-lunch theorems. Theory New Appl Swarm Intell. https://doi.org/10.5772/30852
67. Köppen M, Wolpert D, Macready W (2001) Remarks on a recent paper on the "no free lunch" theorems. IEEE Trans Evol Comput 5:295–296
68. Corne D, Knowles J (2003) Some multiobjective optimizers are better than others. 2003 Congr Evol Comput—Proc 4:2506–2512
69. Wolpert DH, Macready WG (2005) Coevolutionary free lunches. IEEE Trans Evol Comput 9:721–735
70. Kimbrough SO, Koehler GJ, Lu M, Wood DH (2008) On a feasible-infeasible two-population (FI-2Pop) genetic algorithm for

constrained optimization: distance tracing and no free lunch. Eur J Oper Res 190:310–327
71. Auger A, Teytaud O (2010) Continuous lunches are free plus the design of optimal optimization algorithms. Algorithmica (New York) 57:121–146
72. Braik M, Ryalat MH, Al-Zoubi H (2022) A novel meta-heuristic algorithm for solving numerical optimization problems: Ali Baba and the forty thieves. Neural Comput Appl 34:409–455
73. Gezici H, Livatyali H (2022) Chaotic Harris hawks optimization algorithm. J Comput Des Eng 9:216–245
74. Hassan IH, Abdullahi M, Aliyu MM, Yusuf SA, Abdulrahim A (2022) An improved binary manta ray foraging optimization algorithm based feature selection and random forest classifier for network intrusion detection. Intell Syst with Appl 16:200114
75. Alam S, Zafar A (2022) A fusion of dolphin swarm optimization and improved sine cosine algorithm for automatic detection and classification of objects from surveillance videos. Meas J Int Meas Confed 192:110921
76. Akdag O (2022) A improved archimedes optimization algorithm for multi/single-objective optimal power flow. Electr Power Syst Res 206:107796
77. Kaveh A, Biabani K (2022) Improved arithmetic optimization algorithm and its application to discrete structural optimization. Structures 35:748–764
78. Nouhi B, Jahani Y, Talatahari S, Gandomi AH (2022) A swarm optimizer with modified feasible-based mechanism for optimum structure in steel industry. Decis Anal J 5:100129
79. Wang T, Ye Z, Wang X, Li Z, Du W (2022) Improved distributed optimization algorithm and its application in energy saving of ethylene plant. Chem Eng Sci 251:117449
80. Xie W, Wang L, Yu K, Shi T, Li W (2023) Improved multi-layer binary firefly algorithm for optimizing feature selection and classification of microarray data. Biomed Signal Process Control 79:104080
81. Liang J, Qu B, Suganthan P (2014) Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. In 2014 IEEE congress on evolutionary computation
82. Helbig M, Engelbrecht A (2015) Benchmark functions for CEC 2015 special session and competition on dynamic multi-objective optimization. In 2015 IEEE congress on evolutionary computation
83. Cheng R. et al. (2017) Benchmark functions for CEC 2017 competition on evolutionary many-objective optimization. In 2017 IEEE congress on evolutionary computation
84. Luo W, Lin X, Li C, Yang S, Shi Y (2022) Benchmark functions for CEC 2022 competition on seeking multiple optima in dynamic environments, pp. 1–17.
85. Kudela J (2022) A critical problem in benchmarking and analysis of evolutionary computation methods. Nat Mach Intell 4:1238–1245
86. Thanedar P, Vanderplaats G (1995) Survey of discrete variable optimization for structural design. J Struct Eng 121:301–306
87. Coello Coello CA (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41:113–127
88. Ray T, Saini P (2001) Engineering design optimization using a swarm with an intelligent information sharing among individuals. Eng Optim 33:735–748
89. Atabay Ş (2009) Cost optimization of three-dimensional beamless reinforced concrete shear-wall systems via genetic algorithm. Expert Syst Appl 36:3555–3561
90. Velasco L, Hospitaler A, Guerrero H (2022) Optimal design of the seismic retrofitting of reinforced concrete framed structures using BRBs. Bull Earthq Eng. https://doi.org/10.1007/s10518-022-01394-z
91. Mathirajan M, Chandru V, Sivakumar AI (2007) Heuristic algorithms for scheduling heat-treatment furnaces of steel casting industries. Sadhana—Acad Proc Eng Sci 32:479–500
92. O'Hara A, Faaland B, Bare B (1988) Spatially constrained timber harvest scheduling. Can J For Res 19:715–724
93. Daliri A, Asghari A, Azgomi H, Alimoradi M (2022) The water optimization algorithm: a novel metaheuristic for solving optimization problems. Appl Intell. https://doi.org/10.1007/s10489-022-03397-4
94. Su H et al (2022) A horizontal and vertical crossover cuckoo search: optimizing performance for the engineering problems. J Comput Des Eng. https://doi.org/10.1093/jcde/qwac112
95. Tang Y, Zhou F (2023) An improved imperialist competition algorithm with adaptive differential mutation assimilation strategy for function optimization. Expert Syst Appl 211:118686
96. Qaraad M et al (2022) Comparing SSALEO as a scalable large scale global optimization algorithm to high-performance algorithms for real-world constrained optimization benchmark. IEEE Access 10:1
97. Zhou W, Wang P, Heidari AA, Zhao X, Chen H (2022) Spiral Gaussian mutation sine cosine algorithm: framework and comprehensive performance optimization. Expert Syst Appl 209:118372
98. Hu G, Du B, Wang X (2022) An improved black widow optimization algorithm for surfaces conversion. Appl Intell. https://doi.org/10.1007/s10489-022-03715-w
99. Zhou X et al (2022) Advanced orthogonal learning and Gaussian barebone hunger games for engineering design. J Comput Des Eng 9:1699–1736
100. Coello CAC, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inf 16:193–203
101. Watson JP, Howe AE, Darrell Whitley L (2006) Deconstructing Nowicki and Smutnicki's i-TSAB tabu search algorithm for the job-shop scheduling problem. Comput Oper Res 33:2623–2644
102. Franzin A, Stützle T (2019) Revisiting simulated annealing: a component-based analysis. Comput Oper Res 104:191–206
103. Johnson DS, Aragon CR, McGeoch LA, Schevon C (1989) Optimization by simulated annealing: an experimental evaluation—part I: graph partitioning. Oper Res 37:865–892
104. Johnson DS, Aragon CR, McGeoch LA, Schevon C (1991) Optimization by simulated annealing: an experimental evaluation—part II: graph coloring and number partitioning. Oper Res 39:378–406
105. Hayyolalam V, Pourhaji Kazem AA (2020) Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Eng Appl Artif Intell 87:103249
106. Salcedo S, Del Ser J, Landa-Torres I, Gil-López S, Portilla-Figueras JA (2014) The coral reefs optimization algorithm: a novel metaheuristic for efficiently solving optimization problems. Sci World J 2014:1–15
107. Holland J (1975) Adaptation in natural and artificial systems. University of Michigan Press
108. Wan C et al (2022) Improved black widow spider optimization algorithm integrating multiple strategies. Entropy 24:1640
109. Hu G, Du B, Wang X, Wei G (2022) An enhanced black widow optimization algorithm for feature selection. Knowl-Based Syst 235:107638
110. Semchedine M, Bensoula N (2022) Enhanced black widow algorithm for numerical functions optimization. Rev Intell Artif 36:1–11
111. Yadav LK, Verma MK, Joshi P (2022) Novel real valued improved coral-reef optimization algorithm for optimal integration of classified distributed generators. IEEE Access 10:80623–80638

112. Shieh CS, Nguyen TT, Lin WW, Nguyen DC, Horng MF (2022) Modified coral reef optimization methods for job shop scheduling problems. Appl Sci 12:9867
113. Swidler S, Goldreyer E (1998) The value of a finance journal publication. J Finance 53:351–363
114. Baser O, Pema E (2003) The return of publications for economics faculty. Econ Bull 1.
115. van Dalen HP (2021) How the publish-or-perish principle divides a science: the case of economists. Scientometrics 126:1675–1694
116. Hooker JN (1995) Testing heuristics: we have it all wrong. J Heuristics 1:33–42
117. Corstjens J, Depaire B, Caris A, Sörensen K (2020) A multi-level evaluation method for heuristics with an application to the VRPTW. Int Trans Oper Res 27:168–196
118. Campelo F, Wanner EF (2020) Sample size calculations for the experimental comparison of multiple algorithms on multiple problem instances. J Heuristics 26:851–883
119. Franzin A, Stützle T (2022) Technical Report No. TR/IRIDIA/2022-007. A causal framework for optimization algorithms. IRIDIA, Institut de Recherches Interdisciplinaires et de D´eveloppements en Intelligence Artificiell
120. Velasco L, Guerrero H, Hospitaler A (2022) Can the global optimum of a combinatorial optimization problem be reliably estimated through extreme value theory? Swarm Evol Comput 75:101172
121. Liu Q et al (2022) A hybrid arithmetic optimization and golden sine algorithm for solving industrial engineering design problems. Mathematics 10:1567
122. El-Shorbagy MA, El-Refaey AM (2022) A hybrid genetic-firefly algorithm for engineering design problems. J Comput Des Eng 9:706–730
123. Zhong C, Li G, Meng Z (2022) Beluga whale optimization: a novel nature-inspired metaheuristic algorithm. Knowl-Based Syst 251:109215
124. Kang H, Liu R, Yao Y, Yu F (2023) Improved Harris hawks optimization for non-convex function optimization and design optimization problems. Math Comput Simul 204:619–639
125. Zhang YJ, Wang YF, Yan YX, Zhao J, Gao ZM (2022) LMRAOA: An improved arithmetic optimization algorithm with multi-leader and high-speed jumping based on opposition-based learning solving engineering and numerical problems. Alexandria Eng J 61:12367–12403
126. Wang Z, Huang X, Zhu D (2022) A multistrategy-integrated learning sparrow search algorithm and optimization of engineering problems. Comput Intell Neurosci 2022:1–21
127. Lin C, Wang P, Zhao X, Chen H (2022) Double mutational salp swarm algorithm: from optimal performance design to analysis. J Bionic Eng. https://doi.org/10.1007/s42235-022-00262-5
128. Xu Z et al (2023) Enhanced Gaussian bare-bones grasshopper optimization: mitigating the performance concerns for feature selection. Expert Syst Appl 212:118642
129. Wen C et al (2022) Modified remora optimization algorithm with multistrategies for global optimization problem. Mathematics 10:3604
130. Zhao D et al (2022) Opposition-based ant colony optimization with all-dimension neighborhood search for engineering design. J Comput Des Eng 9:1007–1044
131. Hashim FA, Hussien AG (2022) Snake optimizer: a novel meta-heuristic optimization algorithm. Knowl-Based Syst 242:108320
132. Zitouni F, Harous S, Belkeram A, Hammou LEB (2022) The archerfish hunting optimizer: a novel metaheuristic algorithm for global optimization. Arab J Sci Eng 47:2513–2553
133. Rao H et al (2022) A modified group teaching optimization algorithm for solving constrained engineering optimization problems. Mathematics. https://doi.org/10.3390/math10203765
134. Wu T et al (2022) A modified gorilla troops optimizer for global optimization problem. Appl Sci 12:10144
135. Dehghani M, Trojovský P (2022) Serval optimization algorithm: a new bio-inspired approach for solving optimization problems. Biometrics. https://doi.org/10.3390/biomimetics7040204
136. Trojovska E, Dehghani M (2022) Clouded leopard optimization: a new nature-inspired optimization algorithm. IEEE Access 10:102876–102906
137. Yang X et al (2023) An adaptive quadratic interpolation and rounding mechanism sine cosine algorithm with application to constrained engineering optimization problems. Expert Syst Appl 213:119041
138. Yu H et al (2022) Laplace crossover and random replacement strategy boosted Harris hawks optimization: performance optimization and analysis. J Comput Des Eng 9:1879–1916
139. Azizi M, Talatahari S, Gandomi AH (2022) Fire hawk optimizer: a novel metaheuristic algorithm. Artif Intell Rev. https://doi.org/10.1007/s10462-022-10173-w
140. Li HL, Papalambros W (1985) A production system for use of global optimization knowledge. J Mech Des Trans ASME 107:277–284