Review

Review of Metaheuristics Inspired from the Animal Kingdom

Elena Niculina Dragoi [1,2,]* and **Vlad Dafinescu** [2,3]

1 Faculty of Automatic Control and Computer Engineering, "Gheorghe Asachi" Technical University, Bld. Dimitrie Mangeron, No. 27, 700050 Iaşi, Romania

2 Faculty of Chemical Engineering and Environmental Protection "Cristofor Simionescu", "Gheorghe Asachi" Technical University, Bld. Dimitrie Mangeron, No. 73, 700050 Iaşi, Romania; vdafinescu@gmail.com

3 Emergency Hospital "Prof. Dr. N. Oblu", Str. Ateneului No. 2, 700309 Iaşi, Romania

* Correspondence: elena-niculina.dragoi@academic.tuiasi.ro; Tel.: +40-232-278683

Abstract: The search for powerful optimizers has led to the development of a multitude of metaheuristic algorithms inspired from all areas. This work focuses on the animal kingdom as a source of inspiration and performs an extensive, yet not exhaustive, review of the animal inspired metaheuristics proposed in the 2006–2021 period. The review is organized considering the biological classification of living things, with a breakdown of the simulated behavior mechanisms. The centralized data indicated that 61.6% of the animal-based algorithms are inspired from vertebrates and 38.4% from invertebrates. In addition, an analysis of the mechanisms used to ensure diversity was performed. The results obtained showed that the most frequently used mechanisms belong to the niching category.

Keywords: metaheuristics; optimization; animal-inspired; exploration; exploitation

1. Introduction

A metaheuristic is a high level, problem-independent framework that provides a series of steps and guidelines used to develop heuristic optimizers [1]. Nowadays, the tendency is to use the term for both the general framework and for the algorithms built based on its rules [1]. In the latest years, the literature has shown an increase in the number of proposals of new optimization metaheuristics and their improvements through step alterations, local search procedures or hybridizations [2]. For a few well-known metaheuristics, the numerical evolution of the number of papers (journal and conferences) from the IEEE library is provided in [3]. The work of Hussain et al. in [2] presents a detailed distribution of types of research (basic, improvement, applications) focusing on all metaheuristics and, in [4], a timeline of the history of a set of representative techniques is provided.

This increase has been fueled by the need to efficiently find good solutions for difficult problems, especially for those where classical techniques fail to provide acceptable results within a reasonable amount of time and resources consumed.

All of these new optimizers (as well as the existing ones) follow the principles of the No Free Lunch Theorem (NFL), which states that if "an algorithm gains in performance on one class of problems it necessarily pays for on the remaining problems" [5]. In a simplistic view, this can be interpreted as meaning that one specific algorithm cannot outperform its counterparts on all problems but only on specific types or classes of problems, and it was shown that it is theoretically impossible to have a best general-purpose optimization strategy [6] (more details about NFL and its detailed analysis can be found in [7]). Consequently, researchers will probably never be satisfied with the existing metaheuristics [6], and this gives room for the development of new algorithms, improvements and strategies.

The oldest type of metaheuristic optimizers (from the 60s and 70s) is represented by genetic algorithms, based on the evolutionary processes; however, in their quest for better optimization of metaheuristics, researchers have turned to new sources of inspiration. Nowadays, the world of metaheuristics is large and covers ideas varying from the behavior of the very small, e.g., viruses and bacteria, to the mechanisms of galaxies. This multitude of algorithms can be somewhat overwhelming and, therefore, the objective of this paper is to identify the main directions of research, in terms of sources of inspiration, and to shed some light on the mechanisms used to generate powerful optimizers.

2. Classification and Categorization

When trying to identify the main classes of metaheuristics, various criteria can be applied. Examples include: search path, memory use, neighborhood exploration, number of solutions transferred from one iteration to the next and parallelization ability [8,9]. An extensive discussion related to the issue of classification and categorization for metaheuristics and the different schemes used can be found in [10].

In terms of categorizations, different aspects can be considered. For example, the type of candidate solutions is one of the most used criteria, and it splits the metaheuristic into: (i) individual-based, also known as single solutions, trajectory methods [1] or individualist algorithm [11] and (ii) population-based or collective algorithms [11]. In the individual-based group, a single solution is evolved. The main advantages of these methods consist in simplicity, lower computational costs and a lower number of function evaluation [11]. However, in their basic form, they can become trapped in the local optima and, since there is no information sharing, as there is just one solution, issues such as isolation of optima, deceptiveness and bias of the search space need to be dealt with [12]. Examples of algorithms that belong to this class are: Simulated Annealing (SA) [13]; Tabu Search (TS) [14]; Variable Neighborhood Search (VNH) [15]; Iterated Local Search (ILS) [16], proposed before 2006; Vortex Search (VS) [9], proposed after 2006. In the case of the population-based algorithms, multiple solutions are generated and improved. Distinctive from the individual-based algorithms, the population-based approaches allow some information exchange between the candidate solutions and thus can handle aspects that the individual-based approaches struggle with [12]. However, the cost of the improved performance is higher complexity and a larger number of function evaluations. The majority of metaheuristics are population-based and can themselves be classified into approaches that [17]: (i) increase the population diversity through the variation of control parameters; (ii) maintain population diversity through the replacement of individuals in the current population; (iii) include memory to store promising solutions; (iv) divide the population into subpopulations; (v) combine multiple methods, i.e., hybrid approaches.

When type of search is considered, metaheuristics can be local or global. The local search approaches tend to be more exploitative while the global algorithms are more explorative in nature [2]. On the other hand, in the latest years, the trend is to create hybrids that combine the two types of searches. The best-known examples of local search algorithms are TS, ILS and Greedy Randomized Adaptive Search Procedure (GRASP). Differential Evolution (DE), Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) are examples of global search algorithms. Although the global search approaches can be hybridized to also include local procedures as a means to improve a previously proposed version, the literature presents algorithms that include this global–local search combination from the first version, e.g., the Bat Algorithm (BA) [18], the Shuffle Frog Leap Algorithm (SFL) [19] or Water Wave Optimization (WWO) [20].

When considering the source of inspiration, the majority of authors identify the metaheuristics as evolutionary and swarm intelligence techniques [11]. An extended categorization can be considered, such as the one in [21], where two more groups are included: stochastic and physical. In the latest years, various sources of inspiration have been used for metaheuristics. Therefore, this categorization must be extended to include all of the new methods. As a result, this work performs an extensive literature review of the proposed approaches covering the years 2006–2021. The review is organized around the biological classification of living things (kingdom-phylum-class), and its aim is to determine the main directions of research followed in the last 15 years and to identify new potential directions. The work [22] has a similar aim but focuses on all types of sources of inspirations for metaheuristics. Taking into account the variety of aspects that can be analyzed and the number of algorithms, this work only considers the metaheuristics with a biological base.

Regarding classification of metaheuristics, the work of Stegherr et al. [10] presents a seven-layer classification system. It focuses on structure (with criteria that include discontinuances, population, local search and memory), behavior (with criteria that include the strategy to create new solutions, groups and sub-populations), search (with criteria dealing with the intensification and diversification capabilities), algorithm (with criteria including the basic components incorporated), specific features (dealing with capabilities, i.e., use of adaptive parameters), evaluation (concerning the efficiency on various types of problems) and metaheuristics (which contains the specific algorithm that corresponds to the characteristics form the previous levels). If the first six levels are viewed from a framework perspective, the metaheuristic level deals with algorithms.

3. Source of Inspiration

In order to perform the current review, the main databases searched were: ScienceDirect (https://www.sciencedirect.com/, accessed on 6 August 2021), Web of Science (https://apps.webofknowledge.com/, accessed on 6 August 2021), Google Scholar (https://scholar.google.ro/, accessed on 6 August 2021), Springer Link (https://link.springer.com/, accessed on 6 August 2021) and IEEE Xplore Digital Library (https://ieeexplore.ieee.org/Xplore/home.jsp, accessed on 6 August 2021). The terms used in the search process were "metaheuristics", "nature-inspired optimizers" and "bio-inspired algorithms". The strategy to use both nature-inspired and bio-inspired terms is related to the fact that, in many works, there is not a clear distinction between the two and they are used to describe a variety of metaheuristics. Based on the identified sources, a drill down (study of the references used) and drill up approach (study of the papers citing a specific work) were applied in order to determine additional appropriate manuscripts. For the covered period, 283 algorithms were identified. Their distribution, based on the inspiration source, is presented in Figure 1.

By analyzing the identified categories, two main groups can be distinguished: biological and non-biological sources. The biological sources include animals, plants and humans, while the non-biological sources are represented by the chemical and physical laws of nature. Therefore, broadly speaking, the optimization metaheuristics can be grouped into: (i) biologically-inspired and (ii) nature-inspired.
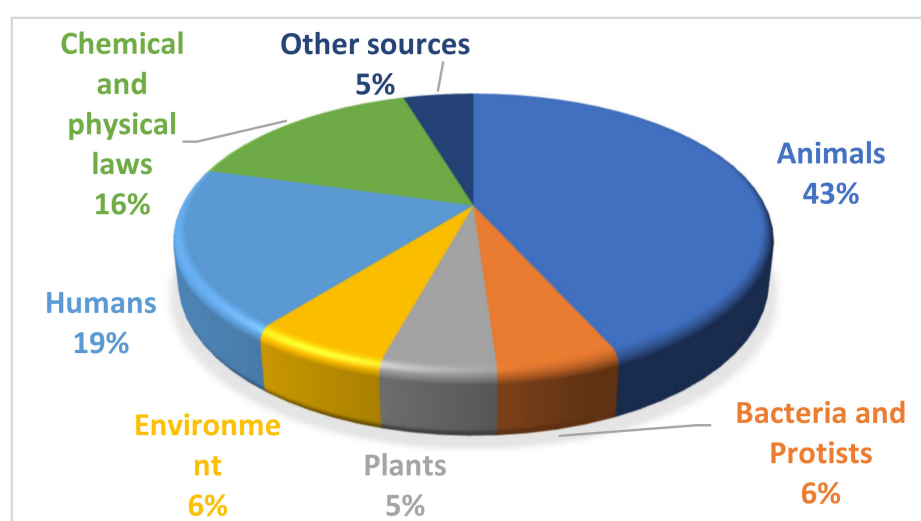
Figure 1. Distribution of newly proposed algorithms in the period 2006–2021, based on their inspiration source.

As it can be observed from Figure 1, the largest group of newly proposed metaheuristics in the considered period have animals as a source of inspiration, and this group is the main focus of the current work. Although humans, from a biological point of view, belong to the animal group, Chordata (vertebrates) phylum, in this work they were not included because they deserve a separate discussion, considering the unique ways of thinking, behaving and interacting with the environment.

In the latest years, various reviews have tried to shed light on the novel approaches that are constantly developed. Examples include: (i) a comprehensive list of algorithms and the steps of a few selected approaches [23]; (ii) a detailed discussion about the main research aspects specific to the field of nature-inspired metaheuristic optimizers [2]. In most works, researchers present the names of the best-known metaheuristics and a few details about the general ideas. This work aims to provide a series of details (such as: source code availability, improvement, applications, mechanisms for controlling the exploration-exploitation balance) in a systematic manner, for each algorithm considered.

3.1. Vertebrates

Most algorithms inspired by animals simulate two main general behaviors: (i) food search (foraging) and (ii) mating. For foraging, there are a number of theoretical models developed to predict the behavior of living things: the optimal foraging theory, the ideal free distribution, game theory and predator-prey models [24]. The theory of optimal foraging was developed to explain the dietary patterns and the resource use, and it states that the individuals using their energy more efficiently for finding food are favored by natural selection [25]. Foraging for food can be an individual activity (solitary foraging—where each individual searches for its food) or can be a social activity (social foraging—where foraging is a group behavior) [26]. The topics of social foraging include: (i) the mechanisms used by the members to find food; (ii) the manner in which the food locations are communicated to other members; (iii) the division of food between group members. The majority of foraging inspired optimization algorithms focus on the first two topics [26]. A taxonomy of foraging inspired algorithms is proposed in [27], where three main categories are identified: vertebrates (with backbone), invertebrates (without backbone) and non-neuronal (organisms that do not possess a central nervous system or brain).

Concerning the mating behavior, different theoretical models that simulate the mating mechanisms of specific species exist. For example, in birds, different strategies are used to display the quality of genes to the potential mates by showing the main physical features: color, shape of specific body parts, etc. In terms of partner combinations, five strategies are encountered: monogamy, polygyny, polyandry, parthenogenesis and promiscuity [28]. These mechanisms are included, in different forms, in the metaheuristic optimizers with the objective of improving diversity and thus increasing performance.

From the total of algorithms inspired from animals, the ones based on vertebrates represent 61.6%, with the most represented sub-groups being birds (Section 3.1.1) and mammals (Section 3.1.2).

3.1.1. Birds
Mating Behavior

One of the best-known algorithms inspired from bird behavior is Cuckoo Search (CS) [29]. It simulates the brood parasitic behavior of some species of cuckoo and, in order to search for new solutions, it uses Levy flight random walk (mutation based on the best solution found so far) and biases/selective random walk (crossover between a current solution and its mutation) [30]. The Levy flight is a random process from the non-Gaussian class, a step which is based on the Levy distribution [31]. The steps of the CS algorithm express three idealized rules: (i) each cuckoo lays an egg and places it into a randomly chosen nest; (ii) the nests with high-quality eggs will be further used in the next generations; (iii) there is a fixed number of nests and there is a probability that the host will discover the foreign egg [32]. The main disadvantage of this algorithm is the fixed value of the scaling factor (that controls the step size) [30] and, in order to improve its performance, various strategies have been applied. A list of different modifications of CS and its applications can be found in [32,33]. Inspired by the same cuckoo breeding behavior, the Cuckoo Optimization Algorithm (COA) was proposed in [34]. When comparing COA and CS, it can be observed that COA is more complex, in the sense that it contains an additional behavioral aspect represented by the immigration process. Also, it uses the k-means clustering algorithm to identify the group that a cuckoo belongs to.

Bird Mating Optimizer (BMO) [28] uses the birds mating process as a framework. Throughout generations, the birds (the population of solutions) apply a probabilistic method to improve the quality of their offspring. The population is divided into males and females. The males can be monogamous, polygamous or promiscuous. On the other hand, the females can be parthenogenetic and polyandrous. In BMO, five species are simulated, and each one has its specific updating pattern.

Developed to adjust the parameters of adaptive neuro-fuzzy inference systems, the Satin Bowerbird Optimizer (SBO) [35] simulates the mating behavior of bowerbirds (a close relative species of the birds-of-paradise). In each iteration, a target individual is determined through roulette wheel selection. The other individuals try to follow it, i.e., they change their position accordingly, and try to improve their strategies by mutation.

Food Search

The flight of eagles (as does, in fact, the flight behavior of many animals and insects) has the typical characteristics of the Levy flight [36]. Based on this idea, the Eagle Strategy (ES) was proposed in [36]. It simulates an idealized two stage strategy (find and chase) [37]. The find step represents the exploration phase realized by the Levy walk and the chase step is an exploitation phase where an intensive local search is performed by the Firefly Algorithm (FA). ES represents a strategy and not an algorithm, and its authors indicate that different algorithms can be used at different stages of the iterations [38]. For example, in [38–40], Differential Evolution (DE) [41] performs the local search procedure that was initially solved by FA.

Chicken Swarm Optimization (CSO) is inspired by swarm behavior [42]. It mimics the flock and foraging behavior of chickens and is based on four simplified rules: (i) the

swarm is comprised of several groups and each group has a dominant rooster, some hens and chicks; (ii) the selection of individuals representing each type of bird is based on the fitness value; (iii) the dominance and hen–chick relationship remains unchanged for several iterations; (iv) the search for food is performed around the dominant entity. CSO is a multi-swarm algorithm and its performance analysis showed that it can be efficiently applied to solve benchmarks and real-world problems.

The Crow Search Algorithm (CSA) [43] simulates the behavior of crows when it comes to food, i.e., storing excess food and thievery. Its main principles are: (i) crows live in flocks; (ii) each crow memorizes their hiding places; (iii) crows follow each other to steal food; (iv) crows try to protect their stashes. The algorithm includes a memory of good solutions and, since the movement of crows is performed regardless of whether a newly generated position is worse than the current one, distinctively from many metaheuristics, CSA is not a greedy algorithm.

Simulating the auditory-based hunting mechanism employed by owls, the Owl Search Algorithm (OSA) [44] assumes that the fitness value of each individual is correlated to the intensity of the received sound and that the search space has one global optimum.

The hummingbird's optimization algorithm (HOA) [45] focuses on the foraging processes of hummingbirds. It includes a self-searching phase, based on the individual accumulated experience (using a Levy flight mechanism), and a guide-searching phase that includes information from dominant individuals.

Simulating the social roosting and foraging behavior of ravens, in [26], the Raven Roosting Optimization (RRO) is proposed. The algorithm includes four main components: (i) the perception capability of each individual to find food; (ii) a memory related to the position of previous foraging locations; (iii) transmitting and receiving information about food locations; (iv) probabilistic movement when searching for new resources.

Based on the cooperative hunting behavior of Harris hawks, the Harris Hawks Optimization (HHO) [46] algorithm simulates a series of aspects such as prey exploration, surprise pounce, and attack strategies. The algorithm complexity is O(population_size $\times$ (iterations + iterations $\times$ dimensionality + 1)) and, in exploring or exploiting the search space, a series of strategies such as diversification mechanism, progressive selection scheme and adaptive and time-varying parameters were used.

Aquila is a very successful bird of prey located in the Northern hemisphere that represents the source of inspiration for the Aquila Optimizer (AO) [47]. The model that the AO is based on simulates four hunting methods: (i) high soar with a vertical stoop (corresponding to an expanded exploration step); (ii) contour flight with glide attack (narrowed exploration step); (iii) low flight and slow descent (expanded exploitation step); (iv) walking and prey grabbing (narrowed exploitation). The AO computational complexity is O(solution_number $\times$ (iterations x dimensionality + 1)).

The hunting mechanisms of golden eagles (spiral trajectory for searching food and straight path when attacking, a tendency of cruising at the beginning of the search and attacking at the end, capability to easily change between cruising and attaching) is simulated in the Golden Eagle Optimizer (GEO) [48]. The attack phase corresponds to exploitation and cruising to exploration. To extend applicability, two variants were proposed: the GEO version (single objective) and the MOGEO (multi-objective) version. The GEO computational complexity is O(population_size $\times$ dimensionality $\times$ iterations) and that for MOGEO is O(population_size $\times$ dimensionality $\times$ iterations $\times$ objectives $\times$ archive).

Movement

Based on the characteristics of geese's flight and the general PSO model, a geese-inspired hybrid PSO (Geese-PSO) was proposed in [49]. Although it does not use a completely novel metaphor and the algorithm is a hybridization, it is considered in this work because, to the authors' knowledge, the principle of following the particle ahead and the application of a unidirectional flow of information was not used prior to the proposal of the Geese-PSO approach. In the same direction of research, Migrating Bird

Optimization (MBO) [50] is inspired by the "V" flight formation of migrating birds. MBO is a neighborhood search approach where each solution is improved based on its neighbors.

The swarming behavior of passenger pigeons (the common name given to Blue Pigeons, Merne Rouck Pigeons, wandering long tailed Doves and Wood pigeons) represents the inspiration of the Pigeon Optimization Algorithm (POA) [51]. Pigeons have a specific behavior that can be simplified into a series of rules: (i) in order to enhance the search and reduce the probability of being prey for other animals, flight is performed in flocks; (ii) different flocks have their own solution for movement, which influences the shape of the group; (iii) there is communication between birds through "keck" and "tweet" calls; (iv) the behavior of other pigeons can be imitated; (v) the pigeons responding to the calls are the closest to the source of the call; (vi) in order to have a better probability of survival, each pigeon must lead. Another algorithm inspired by pigeons is Pigeon Inspired Optimization (PIO) [52]. However, in PIO the main idea is to simulate the homing behavior and the mechanisms used by a pigeon to move from a point A to a point B, i.e., orientation through magnetoreception and the sun, and recall of known landmarks close to the destination.

The migratory and attack behavior of seagulls is imitated in the Seagull Optimization Algorithm (SeOA) [53]. Several simplified rules are considered: (i) the migration is performed in groups; (ii) all the individuals travel towards the one with the best fitness; (iii) the attack follows a spiral-like movement. In addition, during the migration process, a mechanism for collision avoidance is included. The Sooty Tern Optimization Algorithm (STOA) [54] has a similar source of inspiration as the SeOA, but based on Sooty Tern seabirds. In this case, the migration behavior represents the exploration phase and the attacking behavior the exploitation one. The complexity of the STOA is O(problem_dimension × iteration × objective_number × population_size × objective_function) and the space complexity is O(objective_number × population_size).

In their fight to survive the harsh conditions of the polar regions, the emperor penguins use a specific strategy of huddling. This represents the main source of inspiration for the Emperor Penguin Optimizer (EPO) [55], where operations such as huddle boundary determination, temperature computation, distance determination and identification of the effective move are mathematically modeled and simulated in order to perform the optimization. The time complexity of EPO is O(k × individual_length × iterations × dimensionality × population_size), where the algorithm termination criteria requires O(k) time. The same mechanisms are also simulated in Emperor Penguins Colony (EPC) [56]. The main difference between the EPO and EPC consists in the manner in which the movement is realized, i.e., in EPC the individuals perform a spiral-like movement.

The foraging and navigation behaviors of African vultures is modeled in the African Vultures Optimization Algorithm (AVOA) [57], where multiple mechanisms to improve the exploration–exploitation balance were proposed: the use of a coefficient vector to change between these phases, use of phase-shift to precent premature convergence and local optimum escape and inclusion of Levy Flight. The AVOA computational complexity is based on initialization, fitness evaluation and vulture update and is O(iteration × population_size) + O(iteration × population_size × dimensionality).

Table 1 summarizes the algorithms briefly presented in this section and shows a series of examples for improvements and applications. In Table 1, where a link to the source code exists, if not specifically indicated, the implementation is provided by a third party. In the application column, due to the fact that it is common to test the performance of a newly proposed algorithm on a set of problems with known characteristics, the standard benchmarks were not specified.

Table 1. Improvements and applications for bird-inspired metaheuristics (alphabetically sorted).

Algorithm	Source Code	Modifications and Improvements	Applications
Bird Mating Optimizer (BMO) [28]		− adaptive population size [58] − hybridization with Differential Evolution [59,60], Teaching Learning-based Optimization [61]	− image segmentation [59] − optimal expansion planning [58] − structural damage [62] − structural design [62] − engineering design [60] − electrochemical discharging machining [63] − photovoltaic modules [64]
Chicken Swarm Optimization (CSO) [42]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/48204-cso, accessed on 7 January 2020	− multi-objective mechanism [65] − inclusion of penalty [66] − learning mechanism [67]	− crude oil price prediction (in combination with ANNs) [68] − projection pursuit evaluation [69] − error control [70]
Crow Search Algorithm (CSA) [43]	(MATLAB–author source) https://www.mathworks.com/matlabcentral/fileexchange/57867-crow-search-algorithm-for-constrained-optimization, accessed on 19 June 2021	− constraint handling [71] − inclusion of Levy flight [72] − multi-objective adaptation [73] − chaotic systems [73,74] − parameter control through diversity population information [75]	− structural design [71] − Parkinson diagnosis [76] − energy problems [72] − image processing [77]
Cuckoo Search (CS) [29]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/29809-cuckoo-search-cs-algorithm, accessed on 4 February 2019	− hybridization with PSO [78] − Shuffle Frog Optimization Algorithm [79] − improvement of specific steps [32] − varying the control parameters [30,80]	− linear antenna array optimization [81] − operating schedule of battery, thermal energy storage, and heat source in a building energy system [82] − power load dispatch [83] − synchronization of bilateral teleoperation systems [84] − 0–1 knapsack problem [79]
Cuckoo Optimization Algorithm (COA) [34]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/35635-cuckoo-optimization-algorithm, accessed on 4 February 2019	− hybridization with HS [85] − adaptation to discrete spaces [86]	− water allocation and crop planning [87] − load frequency control [85] − bilateral teleoperation system [84] − inverse kinematic problem [88] − PID design [34]
Emperor Penguin Optimizer (EPO) [55]		− binary version [89] − multi-objective variant [90,91] − hybridization with Salp Swarm algorithm [92], Social Engineering Optimization [93]	− ranking of cloud service providers [90] − color image segmentation [94] − medical data classification (in combination with Support Vector Machines) [93]
Emperor Penguins Colony (EPC) [56]		− introduction of mutation and crossover operators [95]	− inventory control problem [96] − neuro-fuzzy system [97]

Table 1. *Cont.*

Algorithm	Source Code	Modifications and Improvements	Applications
Harris Hawks Optimization (HHO) [46]	(MATLAB—author source) https://github.com/aliasgharheidaricom/Harris-Hawks-Optimization-Algorithm-and-Applications, accessed on 10 December 2020	— use of chaos [98] — binary version [99] — hybridization with Differential Evolution [100], Salp Swarm Algorithm [99]	— parameter identification photovoltaic cells [98] — productivity prediction of solar still (in combination with Artificial Neural Networks) [101]
Migrating Bird Optimization (MBO)[50]	(Java) http://mbo.dogus.edu.tr, accessed on 15 November 2020	— hybridization with Harmony Search [102], Differential Evolution [103] — new mechanism for leader selection [104], neighborhood search strategy [105], age mechanism [106], crossover mechanism [107], Glover generator in the initialization phase [108] — use of parallel micro-swarms [106]	— scheduling [102,104–106,108] — manufacturing [107]
Owl Search Algorithm (OSA) [109]		— inclusion of opposition-based learning [110], chaos [111] — binary version [112]	— image segmentation [110] — bilateral negotiations [111] — feature selection [112]
Pigeon Inspired Optimization (PIO) [52]	(MATLAB) http://read.pudn.com/downloads713/sourcecode/math/2859919/Code%20of%20Basic%20PIO/Code%20of%20Basic%20PIO.txt__htm, accessed on 15 December 2020	— discretization [113] — inclusion of the heterogeneity principle [114] — use of Cauchy distribution [115], probability factors to adapt the velocity [116] — multi-objective [117] — predator-prey concept [118]	— travelling salesman problem [113] — prediction of bulk commodity futures prices (in combination with extreme learning machine) [119] — automatic carrier landing [115,116] — current motor parameter design [117]
Raven Roosting Optimization (RRO) [26]		— subpopulations with different behavior [120] — hybridization with CSO [121]	— task scheduling [121]
Satin Bowerbird Optimizer (SBO) [35]		— encoding based on complex values [122]	— solid oxide fuel cells [123] — neuro-fuzzy inference systems [35]
Seagull Optimization Algorithm (SeOA) [53]	(Matlab—author code) https://www.mathworks.com/matlabcentral/fileexchange/75180-seagull-optimization-algorithm-soa, accessed on 12 February 2021	— multi-objective [124] — hybridization with Whale Optimization [125], Cuckoo Search [126], Thermal Exchange Optimization [127]	— feature selection [127]
Sooty Tern Optimization Algorithm (STOA) [54]	(Matlab—author code) https://jp.mathworks.com/matlabcentral/fileexchange/76667-sooty-tern-optimization-algorithm-stoa, accessed on 20 June 2021		— model predictive control [128] — industrial engineering problems [54]

3.1.2. Mammals

Food search

Based on the principles of echolocation used by bats to find food, the Bat Algorithm (BA) [18] employs several idealized rules: (i) echolocation is used for distance sensing and prey identification; (ii) the flying pattern is random, with characteristics such as velocity, pulse rate and loudness; (iii) the variation of loudness is assumed to move from a large value to a minimum (constant). The role of the pulse rate and loudness is to balance exploration and exploitation [129]. Another algorithm simulating bats is the Directed Artificial Bat Algorithm (DABA) [130]. DABA considers the individual flight of bats with no interaction between individuals, while in BA, the bat behavior is similar to the PSO particles. Although, compared to BA, the DABA model is closer to the natural behavior, in terms of optimization performance, BA is better [131]. On the other hand, the Dynamic Virtual Bats Algorithm (DVBA) [131] has a population comprised of only two individuals, i.e., explorer and exploiter bats, that dynamically exchange their roles based on their locations.

The echolocation mechanism is not specific to bats; other animals also using it to navigate and to find food, e.g., Dolphin Echolocation [132]. The abbreviation given by its authors is DE; however, so as not to confuse it with Differential Evolution, Dolphin Echolocation will be denoted by DEO in this work. It mimics the manner in which sound, in the form of clicks, is used to track and aim objects. Distinctively from the bat sonar system, which has a short range of 3–4 m, the range of the dolphin sonar varies from a few tens of meters to over a hundred meters. This aspect and the differences in the environmental characteristics lead to the development of totally different sonar systems, and a direct comparison between the two may be difficult.

In their search for food, sperm whales go as deep as 2000–3000 m and can stay underwater without breathing for about 90 min [133]. They are social animals, travel in groups and only the weaker specimens are attacked by predators such as orcas. This behavior was modeled in the Sperm Whale Algorithm (SWA) [133], where the population is divided into subgroups. In each cycle of breathing and feeding, the individual experiences two opposite poles (surface and bottom of the sea); however, because computing the mirror place is expensive and its influence on the search process is limited, it is applied only to the worst solutions. In order to simulate the hunting behavior of humpback whales, i.e., the bubble net feeding method, the Whale Optimization Algorithm (WOA) [134] searches for prey (the exploration phase) and then uses the shrinking encircling mechanism and the spiral updating position (the exploitation phase). A detailed review covering the multiple aspects of WOA is presented in [135].

The Grey Wolf Optimizer (GWO) [136] models a strict social dominant hierarchy and the group hunting mechanisms—tracking, chasing, approaching and attacking the prey–of grey wolfs (Canis lupus). The complexity of GWO is O(problem_dimension × iteration × objective_number × population_size × objective_function) [54]. Similar to other bio-inspired approaches, the GWO suffers from premature convergence. The prey weight and astrophysics concepts were applied in the Astrophysics Inspired Grey Wolf Optimizer (AGWO) [137] to simultaneously improve exploration and exploitation. Although wolves live in packs and communicate over long distances by howling, they have developed unique semi-cooperative characteristics [138]. By focusing on the independent hunting ability, as opposed to the GWO, which uses a single leader to direct the search in a cooperative manner, the Wolf Search Algorithm (WSA) [138] functions with multiple leaders swarming from multiple directions towards the optimal solution.

Spider monkeys are specific to South America and their behavior falls in the category of fission–fusion social structure [139], i.e., based on the scarcity or availability of food, they split from large to smaller groups and vice versa. The algorithm that simulates this structure is called Spider Monkey Optimization (SMO) [139]. It consists of six phases and, unlike the natural system, the position of leader (local or global) is not fixed, depending instead on its ability to search for food. In addition, the optimization procedure does not

include the communication tactics specific to spider monkeys. Distinctively, the individual intelligence of chimps used for group hunting is modelled into the Chimp Optimization Algorithm (ChOA), where four types of hunting are included: driving, chasing, blocking and attacking [140]. Another type of ape is represented by gorillas and, in the Artificial Gorilla Troops Optimizer (GTO), their collective life is mathematically modeled to include exploration–exploitation mechanisms [141].

Spotted hyenas have a behavior similar to that of wolves and whales, which uses collective behavior to encircle the prey and attack. Their model is used in the Spotted Hyena Optimizer (SHO) [142], which saves the best-so-far solution, simulates the encircling prey through a circle-shaped neighborhood that can be extended to higher dimensions and controls the exploration–exploitation balance through control parameters. The time complexiy of SHO is O(problem_dimension $\times$ G $\times$ iteration $\times$ objective_number $\times$ population_size $\times$ objective_function), where the time to define the groups of individuals is O(G).

In the Squirrel Search Algorithm (SSA) [109], the gliding behavior of flying squirrels when exploring different areas of a forest in search for food is simulated by considering some simplifications of the natural mechanisms: (i) a squirrel is assumed to be on one tree; (ii) in the forest there are only three types of trees: normal, oak and hickory; (iii) the region under consideration contains three oaks and one hickory tree. It is considered that the squirrel with the best fitness is positioned on a hickory tree and the next three individuals with the best fitness are on oak trees. The other individuals in the population move towards the oak or the hickory, depending on their daily energy requirements. In SSA, the seasonal changes are modeled through control parameters and influence the behavior of the individuals in the population.

Social Behavior

The Lion's Algorithm (LA) [143] is based on the social behavior of lions. It simulates the process of pride forming through mating, removing weak cubs, territorial defense and takeover. The population is formed of males and females and the cub population is subjected to gender grouping (through the application of k-means clustering). LA is not the only approach inspired by lions; the Lion Optimization Algorithm (LOA) [144] is also an example. Distinctively from LA, LOA includes the hunting and migration mechanisms and the mating process is based on differentiation rather than on crossover and mutation. Another lion-inspired approach is the Lion Pride Optimization (LPOA) [145].

Similar to honey bees or ant colonies, blind naked mole rats (a species specific to Africa) have a complex social behavior: (i) they live in large colonies; (ii) a queen and a reduced number of males are responsible for offspring generation; (iii) there are individuals specialized in food search and domestic activities, i.e., taking care of the nest and of the young and in protection against invaders [146]. These mechanisms, in a simplified form, are simulated in the Blind Naked Mole Rats (BNMR) algorithm [146].

Elephants are the largest walking mammals and their successful survival is influenced, among other things, by their social and behavioral structures. The adult males solitarily roam into the wild, they do not commit to any family and can potentially mate over thirty times a year, while the female elephants form matriarchal societies that allow better protection and safe rearing of young calves. The Elephant Search Algorithm (ESA) [147] and Elephant Herding Optimization [148,149] simulate these mechanisms and perform the search.

Table 2 summarizes the algorithms briefly presented in this section and shows a series of examples for improvements and applications. The same structure and idea as in Table 1 are applied.

Table 2. Improvements and applications for some mammal-inspired metaheuristics (alphabetically sorted).

Algorithm	Source Code	Modifications and Improvements	Applications
Bat Algorithm (BA) [18]	(Python) https://github.com/buma/BatAlgorithm, accessed on 20 December 2019	− discrete version [150] − introducing directional echolocation [151] − multi-population, chaotic sequences [129] − inclusion of Doppler effect [152] − hybridization with Invasive Weed Optimization [153], Differential Evolution [154], − binary version [155]	− drugs distribution problem [150] − forecasting motion of floating platforms (in combination with Support Vector Machines -SVM-) [156] − flood susceptibility assessment (in combination with adaptive network-based fuzzy inference system) [157] − job shop scheduling [158] − travelling salesman problem [159] − battery energy storage [160] − constraint [161] and structural optimization [162]
Blind Naked Mole Rats (BNMR) [146]			− data clustering [163]
Chimp optimization algorithm (ChOA), [140]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/76763, accessed on 20 August 2021	− use of sine-cosine functions to update the search process of ChOA [164]	− combination with ANNs for underwater acoustical classification [165] − high level synthesis of data paths in digital filters [164]
Directed Artificial Bat Algorithm (DABA) [130]			− travelling salesman problem [130]
Dolphin Echolocation (DEO) [132]		− exploration improvement [166]	− plastic analysis of moment frames [167] − design of steel frame structure [168] − reactive power dispatch [169]
Dynamic Virtual Bats Algorithm (DVBA) [131]		− parameter setting [170]	
Elephant Herding Optimization (EHO) [148,149]	(MATLAB—author source) http://www.mathworks.com/matlabcentral/fileexchange/53486, accessed on 17 January 2020	− alpha tuning, cultural-based algorithm, biased initialization [171] − hybridization with Cultural Algorithm [172] − multi-objective and discrete [173] − introduction of chaotic maps [174]	− structural design [171] − batch fermented for penicillin production [171] − network detection intrusion (in combination with SVM) [175] − image processing [176] − SVM parameter tuning [177]
Elephant Search Algorithm (ESA) [147]		− chromosome representation, elephant deep search, and baby elephant birth [178]	− data clustering [179,180] − snack food distribution [178] − travelling salesman problem [181]
Grey Wolf Optimizer (GWO) [136]	(MATLAB—author source) http://www.alimirjalili.com/Projects.html, accessed on 6 June 2021	− two phase mutation [182] − introduction of random walk [183] − introduction of cellular topological structure [184] − modification of parameter behavior [185] − binary [186] − multi-objective [187]	− fluid dynamic problems [188] − power systems [185,189] − combination with ANNs [190] − structural engineering [137] − maximum power tracking [191]

Table 2. *Cont.*

Algorithm	Source Code	Modifications and Improvements	Applications
Lion's Algorithm (LA) [143]		− fertility evaluation, a modified crossover operator and gender clustering [192] − hybridization with a heuristic specific to job shop scheduling [193]	− system identification [192] − rescheduling based congestion management [194] − job shop scheduling [193]
Lion Optimization Algorithm (LOA) [144]			− clustering mixed data [195]
Lion Pride Optimization Algorithm (LPOA) [145]			− double layer barrel vault structures [196] − structural design [145]
Sperm Whale Algorithm (SWA) [133]			− natural gas production optimization [133] − ANN parameter identification [197]
Spider Monkey Optimization (SMO) [139]	(MATLAB, C++, Python–author sources) http://smo.scrs.in, accessed on 10 January 2021	− inclusion of chaos [198], levy flight [199], quadratic approximation [200], age principle for population [201] − hybridization with Limacon curve [202], Nelder-Mead [203] − binary [204]	− load frequency control [205] − irrigation [206] − diabetes classification [207] − capacitor optimal placement [202] − antenna array [204]
Spotted Hyena Optimizer (SHO) [142]		− multi-objective [208]	− structural design [209] − neural network training [210] − airfoil design [211]
Squirrel Search Algorithm (SSA) [109]			− heat flow [109]
Whale Optimization Algorithm (WOA) [134]	(MATLAB—author source) http://www.alimirjalili.com/Projects.html, accessed on 6 June 2021	− hybridization with Nawaz–Enscore–Ham [212], Simulated Annealing [213], Differential Evolution [214] − mechanism of exploration phase [215] − introduction of chaotic maps [216]	− power system [217] − optimal control [218] − structural engineering [215] − drug toxicity [219] − parameter optimization for Elman Networks applied to polymerization process [216] − robot path planning [220] − handwritten binarization [221]

3.1.3. Other Vertebrates

This category includes other sources of inspiration from the vertebrate group that do not belong to the bird and mammal classes.

The SailFish Optimizer (SFO) [222] is inspired by the group hunting of sailfish (*Istiophorus platypterus*), one of the fastest fish in the ocean. This mechanism of alternating attacks on schools of sardines is modeled through the use of energy-based approaches, where, at the beginning of the hunt, both the predator and the prey are energetic and not injured; however, as the hunt continues, the power of the sailfish will decrease and the sardines will become tired and have reduced awareness.

The food catching behavior of Agama lizards is modelled in the Artificial Lizard Search Optimization (ALSO) [223]. The algorithm focuses on new discoveries regarding the mechanisms for movement control through the tail during prey hunting.

The manner in which chameleons catch prey using their long and sticky tongue represents the basis for the Chameleon Swarm Algorithm [224]. The notation given by the authors of this algorithm is CSA, however, since the same notation is used to represent the Crow Search Algorithm, in this work, the Chameleon Swarm Algorithm will be indicated by the ChSA notation. The ChSA follows three main strategies for catching prey: tracking (modelled as a position update step), eye pursuing (modeled as position update in accordance with the position of the prey) and attacking (based on tongue velocity). Distinctively from the majority of metaheuristics, which tend to have less than three parameters, ChSA has five parameters that help in controlling the exploration–exploitation balance.

3.1.4. General

Unlike the other algorithms mentioned in this work that have a source of inspiration represented by a single animal, in the case of the general class, the metaheuristics are based on a general aspect that can be specific to multiple animals or types of animals. Examples proposed prior to 2008 include algorithms such as Genetic Algorithms (where the genetic principles of mutation and crossover are applicable to all species) and Extremal Optimization [225], based on the Bak–Sneppen mechanism, a model of co-evolution between interacting species which reproduces nontrivial features of paleontological data.

Inspired from the encircling mechanisms used by group hunters such as lions, wolves and dolphins, the Hunting Search (HuS) [226] simulates the cooperation of members to catch food. As a perfect correlation between nature and an optimization process cannot be achieved, a set of differences from the real world are taken into account: (i) in the majority of cases, the location of the optimum of a problem is not known, while, in the real world, the hunters can see the prey or sense its presence; (ii) the optimum is set, however, in the real world, the prey dynamically changes its position. Unlike the DEO and GWO, which emulate the specific hunting approaches used by dolphins and wolves, HuS is focused on the cooperation aspect and the repositioning during the hunt. Other approaches which simulate the food searching mechanisms include: the Backtracking Search Algorithm Optimization (BSA) [227], Optimal Foraging Algorithm (OFA) [25], Fish Electrolocation Optimization (FEO) [228] and Marine Predators Algorithm (MPA) [229]. BSA is based on the return of a living creature to previously found fruitful areas. At its core, it is an evolutionary approach that, although it has a very similar structure to the other EAs, differs as follows: (i) mutation is applied to a single individual; (ii) there is a more complex crossover strategy compared with DE; (iii) it is a dual population algorithm; (iv) it has boundary control mechanisms. Distinctively from the BSA, the OFA algorithm is based on the Optimal Foraging Theory developed to explain the dietary patterns of animals. In the OFA, the animal foraging is an individual and its position represents a solution. Its time complexity is O(group_size × dimensionality × iterations) and its space complexity is O(group_size × dimensionality × (iterations + 1)). The FEO simulates the active and passive electrolocation mechanisms used by sharks and "elephant nose fishes" to find prey. A series of electric waves are generated and reflected back to the fish after hitting the surrounding objects, which creates an electric image that is then

analyzed. In the case of the MPA, the different strategies used for finding food and the interaction between predator and prey are modeled in different scenarios through Brownian and Levy strategies. The MPA algorithm complexity is O(iterations × (agent_number × dimensionality + Cost_function_evaluation × agent_number)).

Another aspect specific to all species in their quest to survive is represented by the competition for food, resources or mates. Two metaheuristic optimizers based on competition were identified: Competition over Resources (COR) [230] and the Competitive Optimization Algorithm (COOA) [231]. The COR algorithm mimics the competition for food of wild animals. The groups with the best approach to storing food have improved scores while the worst performance groups are starving and, after a few generations, die and are removed from the population. In the COOA approach, the competition is simulated by the Imperialist Competitive Algorithm [232] and the groups are represented by the populations of various metaheuristics.

Migration behavior is encountered in all major animal groups. Among the first bio-inspired metaheuristics that contain elements specific to migration is the Biogeography-based Optimization (BBO) [233]. However, the BBO imitates a much larger phenomenon—island biogeography—that includes both migration and mutation [234]. Another algorithm that has the migration principle at its core is the Migrating Birds Optimization [50]. As it simulates the features of the "V" flight of birds, the MBO was included in the bird inspired metaheuristic section. The Animals Migration Optimization (AMO) [235] simulates the animal migration model proposed by ecologists and uses two idealized assumptions: (i) the leader animal will survive to the next generation; (ii) the number of animals in the population is fixed. The algorithm has two phases: the migration process (where the individuals respect three rules: move in the same direction as the neighbors, remain close to the neighbors and avoid collision with neighbors) and population update (where some individuals leave the group and others join it).

Table 3 summarizes the algorithms briefly presented in this section and shows a series of examples for improvements and applications. The same structure and idea as in Tables 1 and 2 are applied.

Table 3. Improvements and applications for metaheuristics inspired from general behavior (alphabetically sorted).

Algorithm	Source Code	Modifications and Improvements	Applications
Animals Migration Optimization (AMO) [235]		− inclusion of an interactive learning behavior [236]; − use of Opposition Based Learning [237] − hybridization with Association Rule Mining [238] − population updating step [239]	− bridge reinforcement [240] − multilevel image thresholding [241] − data mining [238] − data clustering analysis [239]
Backtracking Search Algorithm Optimization (BSA) [227]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/44842, accessed on 10 December 2019	− multiple mutation strategies [242] − discrete variant [243] − use of Opposition Based Learning [244] − hybrid mutation and crossover strategy [243] − hybridization with TLBO [245] − constraint handling mechanisms [246]	− casting heat treatment charge plan problem [243] − electricity price forecasting (in combination with adaptive network-based fuzzy inference system) [247] − parameter estimation for frequency-modulated sound waves [227] − engineering design problems [227,246]
Biogeography based Optimization (BBO) [233]		− inclusion of re-sampling [248] − inclusion of mutation strategy [234] − inclusion of chaos maps [249]	− flood susceptibility assessment (in combination with adaptive network-based fuzzy inference system) [157] − soil consolidation (in combination with artificial neural networks) [250] − power fuel cells [234]
Competition over Resources (COR) [230]	(MATLAB—author source) http://freesourcecode.net/matlabprojects/71991/competition-over-resources--a-new-optimization-algorithm-based-on-animals-behavioral-ecology-in-matlab, accessed on 25 June 2020		− building lighting system [251] − magnetic actuators [252]
Hunting Search (HuS) [226]		− hybridization with Harmony Search [253]	− artificial neural network training [253] − steel cellular beams [254]
Marine Predators Algorithm [229]	(MATLAB–author source) au.mathworks.com/matlabcentral/fileexchange/74578, accessed on 08 August 2021	− hybridization with Moth Flame Optimization [255], Teaching-learning based optimization [256] − binary version with V-shaped and S-shaped transfer functions [257]	− parameter extraction of photovoltaic models [258] − multi-level thresholding for image segmentation [255]
Optimal Foraging Algorithm (OFA) [25]	(MATLAB-author source) https://www.mathworks.com/matlabcentral/fileexchange/62593, accessed on 24 April 2020	− chaos [259] − constraint handling mechanisms [259]	− drilling path optimization [260] − SVM Parameter optimization [261] − white blood cell segmentation [259]

3.2. Invertebrates

From the total of algorithms inspired from animals, the ones based on invertebrates represent 38.4%, with the main sub-group indicated by insects (Section 3.2.1). As the number of algorithms inspired from other invertebrate sub-groups were small, the ones not belonging to insects were included in a separate section (Section 3.2.2).

3.2.1. Insects

Although the majority of insects are solitary, several types of insects are organized in colonies or swarms [262]. As insect swarms have several desirable attributes, a high percentage of insect-inspired metaheuristic optimizers belong to the swarm intelligence class.

Swarm intelligence has two main key components: self-organization (global response through interactions among low level components that do not have a central authority) and division of labor (the tasks are performed by specialized individuals) [139,263]. It follows three basic principles: (i) separation (static collision avoidance); (ii) alignment (velocity matching); (iii) cohesion (the tendency of individuals to go towards the center of the mass of the swarm) [264].

While, in the classic swarm approaches, the individuals considered are unisex and perform virtually the same behavior, thus wasting the possibility of adding new operators [265], in the newer bio-inspired metaheuristics researchers began to incorporate different types of individuals in the population(s), and the results obtained show an improvement of several characteristics, such as search ability and population diversity. However, the use of different operators leads to an increase in complexity and, until now, theoretical studies that can explain the influence of these operators and the context in which they are recommended have been very scarce.

Hymenoptera

This order includes some of the best-known social insects: wasps, bees and ants. The main characteristics of these insects are: (i) the presence of a pair of membranous wings; (ii) antennae longer than the head; (iii) complete metamorphosis.

- *Bees*

The social bees show all the characteristics of eusociality: generation overlapping, separation into fertile and infertile groups, labor division and brood care. In addition, the beehive can be considered as a self-organizing system with multiple agents [266].

In a comprehensive review regarding the algorithms inspired by honey bees, the authors identified five main characteristics that were modeled: (i) mating; (ii) foraging and communication; (iii) swarming; (iv) spatial memory and navigation; (v) division of labor [267]. However, [268] considers that, alongside mating and foraging, the third class is represented by nest-site selection process, and thus proposed the Bee Nest-Site Selection Scheme (BNSS)—a framework for designing optimization algorithms.

In addition to the algorithms presented in [267], other approaches that simulate bee behavior are: Bumblebees (B) [269], Bee Colony Inspired Algorithm (BCiA) [266] and Bumble Bee Mating Optimization (BBMO) [270]. The B algorithm is based on a simplified model of the evolution of bumblebees and can be regarded as a loose implementation of the concepts of the evolutionary model proposed by [271]. On the other hand, the BCiA focuses on the foraging behavior and the BBMO simulates the mating process.

- *Ants*

Among the first algorithms that simulate ant behavior is the Ant System [272]. However, the best-known approach is the Ant Colony Optimization (ACO), used to find the path of minimum length in a graph. To the authors' knowledge, the only other approach simulating ant behavior, which is not based on the ACO, is Termite-hill [273]. It is a swarm-based algorithm designed for wireless sensor networks, i.e., an on-demand and multipath routing algorithm.

Diptera

- *Flies*

Due to their short life-span and easiness of breeding and of providing an adequate living environment, the fruit fly is widely studied in laboratory conditions. Consequently, their behavior is known in detail and specific mechanisms for finding food are sources of inspiration for new algorithms. To the authors' knowledge, there are two metaheuristics that simulate the fruit fly: the Fruit Fly Optimization (FOA) and the Drosophila Food Search Optimization (DFO) [274]. Similar to the DFO, the FOA is also based on the Drosophila fly and the literature shows that there are at least two different implementations, proposed by [275] and [276].

- *Mosquitoes*

The host seeking behavior of female mosquitos is mimicked by the Mosquito host seeking algorithm (MHSA) [277]. The general idea is simple and it is based on the following idealized rules: (i) the mosquito looks for carbon dioxide or some other scent; (ii) if found, the mosquito moves toward the location with the highest concentration; (iii) it descends when the heat radiating from the host is felt. The algorithm was developed specifically to solve the travelling salesman problem and it has several advantages that include: (i) the ability to perform large-scale distributed parallel optimization; (ii) it can describe complex dynamics; (iii) it can perform multi-objective optimization; (iv) it is claimed to be independent of the initial conditions and problem size [278]

Lepidoptera

The insects that belong to this class have wings covered with overlapping small scales. The best-known examples include butterflies and moths.

- *Butterflies*

The Monarch Butterfly Optimization (MBO) [279] simulates the migration behavior of monarch butterflies through the use of a set of idealized rules: (i) the entire population of butterflies is located in two areas, i.e., Land1 and Land2; (ii) each offspring is generated by the migration operator applied to individuals from Land1 or Land2; (iii) once an offspring is generated, the parent butterfly dies if its fitness is worse than that of the offspring; (iv) the butterflies with the best fitness survive to the next generation. Similar to the MBO, the Monarch Migration Algorithm (MMA) [280] models the migration behavior of monarch butterflies. The main differences between the MBO and the MMA consist in the mechanisms used for movement, for new individual creation and for population size control.

If the MBO and the MMA focus on migration aspects, the Butterfly Optimizer (BO) simulates the mate-location, behavior–perching and patrolling of male butterflies [281]. The initial BO version is developed for unconstrained optimization and is a dual population algorithm that includes male butterflies and auxiliary butterflies. The Artificial Butterfly Optimization (ABO) [282] is inspired from the same mating strategy as BO. However, the ABO is a single-population optimizer that contains two types of butterflies: sunspot and canopy, and the rules that it follows are different. In the BO, the following rules are considered: (i) the male butterflies are attracted to the highest UV/radiation object; (ii) the best perching position and the flying direction is memorized; (iii) the flying velocity

is constant; (iii) the flying direction is changed if necessary [281]. On the other hand, the ABO considers the following generalized rules: (i) all male butterflies attempt to fly towards a better location (sunspot); (ii) in order to occupy a better position, the sunspot butterflies try to fly to the neighbor's sunspot; (iii) the canopy butterflies continually fly towards the sunspot butterflies to contend for the sunspot [282]. In ABO, three flight strategies are considered and their combination leads to two other variants of the algorithm.

The Butterfly Optimization Algorithm (BOA) [283] considers the foraging behavior and focuses on the smell of butterflies as the strategy used for determining the location of food or of a mating partner. In order to model this behavior, a set of idealized rules are used: (i) all butterflies emit fragrances that attract each other; (ii) the movement of the butterfly is random or towards the most fragrant butterfly; (iii) the stimulus intensity is influenced by the landscape of the objective function.

- *Moths*

The transverse orientation navigation mechanism of moths represents the source of inspiration for the Moth Flame Optimization (MFO) [284]. The population of moths updates their position in accordance with a flame. The group of flames represents the best solutions and serves as guidance for the moths [285]. The complexity of the MFO is O(problem_dimension × iteration × objective_number × population_size × objective_function) [54]. While the MFO contains a population of moths and flames, in the case of the Moth Swarm Algorithm (MSA) [286], which is also inspired by the navigation behavior of moths, the population is formed of three groups of moths: pathfinders (with the ability to discover new areas of the search space), prospectors (that tend to wander in spiral) and onlookers (that drift directly to the solutions obtained by the prospectors). Distinctively from MFO and MSA, the Moth Search (MS) algorithm [287] considers the phototaxis and the Levy flight of the moths as a source of inspiration. In this case, the population is formed of two subpopulations. One follows the Levy movement and the other simulates the straight flight.

Ortoptera

This order includes, among others, insects such as grasshoppers, crickets and locusts. They are insects that move with great agility and have many shapes and characteristics.

The first metaheuristic optimizer that used the locust swarm metaphor is the Locust Swarm [288], a multi-optima search technique developed for non-globally convex search spaces. However, in order to identify the starting points for the search, it uses the PSO as part of its search [288], and one may argue that it is not a new metaheuristic, but a hybridization of the PSO. On the other hand, the Locust Swarm proposed in [289] emulates the interaction of a locust cooperative swarm. Since the two algorithms have the same name, in this work they are referred to as LS1 for the version proposed in [288] and LS2 for the [289] version. LS2 considers both solitary and social behavior, and consists of three parts: initialization, solitary operation and social processes [289]. The solitary phase performs the exploration of the search space, while the social phase is dedicated to exploitation.

The grasshoppers' social interactions represent the basis for the Grasshopper Optimization Algorithm (GOA) [290]. Both larvae, which corresponds to the feeding stage, and the adult form, which corresponds to the exploration stage, are considered. While in nature, the individual evolves from larvae to adult (local, then global), and due to the nature of the search space, the optimization algorithm first needs to find a promising region and, after that, exploit it (global, then local).

Other Insects

- *Hunting*

The mechanisms used by antlions to hunt ants is simulated in the Ant Lion Optimizer (ALO) [291]. Antlions have two phases: larvae, focused on feeding, and adults, focused on mating. The ALO is based on the larvae form and, in order to perform the optimization, a series of conditions are considered: (i) a random walk imitates the movement of an ant; (ii) the random walks are affected by the traps of the antlions; (iii) the pits built by antlions (and the probability of catching ants) are proportional with the antlion fitness; (iv) the random walk range decreases adaptively; (v) the ant is caught when its fitness is worse than that of an antlion; (vi) after catching a prey, the antlion repositions and builds another pit.

- *Mixed behavior*

Inspiration for metaheuristics comes not only from insects that are generally considered useful, e.g., bees, but also from insects that are considered pests, such as cockroaches. Among the first approaches that included the social behavior of cockroaches is the Roach Infestation Optimization (RIO) [292]. It is an adaptation of the PSO that implements three elements: finding the darkness, finding friends and finding food. Other algorithms simulating cockroach behavior are: the Cockroach Swarm Optimization (CSO) [293] and the Cockroach-inspired Swarm Evolution (CSE) [294]. The paper containing the initial CSO version was retracted from the IEEE database due to violations of publication principles, however, this did not stop other researchers from using and improving CSO; Google Scholar indicates that, as of the end of February 2019, there are 32 articles citing the retracted paper. Unlike the RIO, the CSE considers competition, space endurance and migration of cockroaches, beside cooperative behavior [294].

The Dragonfly Algorithm (DA) [264] is inspired from the static (feeding) and dynamic (migratory) swarming behavior of dragonflies. In the feeding stage, the dragonflies are organized into small groups that cover a small area to hunt, through back-and-forth movement with abrupt changes in the flying path. In the dynamic stage, a large group of individuals form a swarm and migrate in one direction over long distances.

The Pity Beetle Algorithm (PBA) [295] is based on the aggregation behavior and search mechanisms used for nesting and food finding of Pityogenesis chalcographus, a beetle also known under the name of "six toothed spruce bark beetle". The PBA follows three stages: searching (where the chemicals emitted by the weakened trees are used to identify a suitable host), aggregation (were multiple individuals feed on the host and attract more individuals—both male and female) and anti-aggregation (that is specific to the situation when the population size surpasses a specific threshold).

Table 4 summarizes the algorithms inspired from inspects and presents a series of examples for improvements and applications. The same structure and idea as in the previous tables are applied.

Table 4. Improvements and applications for insect-inspired metaheuristics (alphabetically sorted).

Algorithm	Source Code	Modifications and Improvements	Applications
Ant Lion Optimizer (ALO) [291]	(MATLAB-author source) http://www.alimirjalili.com/ALO.html, accessed on 6 June 2021	− multi-objective optimization [296] − binary [297] − inclusion of chaos [298]	− Artificial Neural Network training [299] − multi-objective engineering design problems [296] − automatic generation control [300] − feature selection [297]
Bee Colony Inspired Algorithm (BCiA) [266]			− vehicle routing problem with time windows [266]
Bumble Bee Mating Optimization (BBMO) [270]		− inclusion of combinatorial neighborhood topology [301] − parameter adaptation [302]	− multicast routing, traveling salesman problem [302] − feature selection [303] − vehicle routing problem with stochastic demands [301]
Butterfly Optimizer (BO) [281]		− inclusion of constraints [304] − inclusion of covariance matrix [305]	
Butterfly Optimization Algorithm (BOA) [283]	(MATLAB-author source) https://www.mathworks.com/matlabcentral/fileexchange/68209-butterfly-optimization-algorithm-boa/, accessed on 12 December 2019	− inclusion of mutualism principle [306], cross-entropy [307], learning automata [308] − binary approach [309] − the search is modified to use a normal distribution	− maximum power point tracking in photovoltaic systems [310] − feature selection [309]
Pity Beetle Algorithm (PBA) [295]		− new search and population reproduction mechanism, parameter adaptation [311] − inclusion of the opposition-based principle [312]	− wireless multimedia sensors [311] − lung cancer classification [312]
Dragonfly Algorithm (DA) [264]	(MATLAB-author source) http://www.alimirjalili.com/DA.html, accessed on 6 June 2021	− binary [264] − multi-objective [264] − inclusion of memory mechanisms specific to PSO [313], chaos theory [314]	− feature selection [314–316] − proton exchange fuel cells [317] − engineering design [313] − submarine propeller optimization [264]
Drosophila Food Search Optimization (DFO) [274]			− winner takes all circuit [318]
Firefly algorithm (FF) [37]	(MATLAB) http://yarpiz.com/259/ypea112-firefly-algorithm, accessed on 12 December 2019	− chaotic maps [319] − hybridization with Patter Search [320], Harmony Search [321], Group Search Optimizer [322]	− load frequency controller design [323] − ophthalmology [324] − discrete optimization [325]

Table 4. *Cont.*

Algorithm	Source Code	Modifications and Improvements	Applications
Fruit Fly Optimization (FOA) [276]	(MATLAB—author source) http://www.oitecshop.byethost16.com/FOA.html?i=1, accessed on 15 June 2020	— multi-swarm [326,327] — adaptive cooperative learning [328] — introduction of random perturbation [329] — use of a cloud-based model [330]	— shortest path in mobile ad-hoc networks [331] — image processing [328] — joint replenishment problems [329] — parameter identification of synchronous generator [327]
Grasshopper Optimization Algorithm (GOA) [290]	(MATLAB-author source) http://www.alimirjalili.com/GOA.html, accessed on 6 June 2021	— binary [332] — multi-objective [333] — inclusion of chaos [334] — inclusion of levy flight mechanism [335]	— feature selection [336] — Support Vector Machine optimization [336] — financial stress prediction (in combination with extreme learning machine) [335] — Artificial Neural Network training [332] — decision making for self-driving vehicles [337]
Locust Swarm (LS1) [288]			— joint replenishment problems [338]
Locust Swarm (LS2) [289]	(MATLAB-author source) https://www.mathworks.com/matlabcentral/fileexchange/53271-locust-search-ls-algorithm, accesed on: 20 December 2019		— image segmentation [339,340]
Mayfly optimization algorithm (MA) [341]	(MATLAB-author source) https://in.mathworks.com/matlabcentral/fileexchange/76902-a-mayfly-optimization-algorithm, accessed on 15 August 2021	— hybridization with Harmony Search [342]	— feature selection [342] — optimal design of energy renewable sources (in combination with radial basis neural networks) [343]
Monarch Butterfly Optimization (MBO) [279]	(C++, MATLAB) https://github.com/ggw0122/Monarch-Butterfly-Optimization, accessed on 12 December 2019	— binary adaptation [309] — hybridization with Differential Evolution [344] — inclusion of crossover operator [345] — self-adaptive strategies [346]	— 0–1 knapsack problem [347] — osteoporosis classification (in combination with Artificial Neural Networks) [348] — vehicle routing problem [349]
Mosquito host-seeking algorithm (MHSA) [277]		— inclusion of random walk and game of life [350]	— travelling salesman problem [278]
Moth Flame Optimization (MFO) [284]	(MATLAB-author source) http://www.alimirjalili.com/MFO.html, accessed on 6 June 2021	— inclusion of Gaussian mutation [351] — multi-objective optimization [352] — inclusion of chaos [298,353] — inclusion of levy flight mechanism [354,355]	— non-linear feedback control design [356] — medical diagnosis (in combination with extreme learning machine) [353] — reactor power dispatch [285] — engineering design problems [355]

Table 4. *Cont.*

Algorithm	Source Code	Modifications and Improvements	Applications
Moth Swarm Algorithm (MSA) [286]	(MATLAB-author source) https://www.mathworks.com/matlabcentral/fileexchange/57822-moth-swarm-algorithm-msa, accessed on 8 February 2020	— inclusion of Opposition Based Learning [357] — hybridization with Gravitational Search Algorithm [358] — inclusion of arithmetic crossover [359] — inclusion of chaos [360]	— power flow [359] — threshold image segmentation [361]
Moth Search (MS) algorithm [287]	(MATLAB-author source) https://in.mathworks.com/matlabcentral/fileexchange/59010-moth-search-ms-algorithm, accessed on 8 February 2020	— inclusion of disruptor operator [362] — binary optimization [363] — alteration at step level [364] — hybridization with Ant Colony Optimization [365]	— photovoltaic parameter identification [362] — knapsack problem [363] — drone placement [366]
Roach Infestation Optimization (RIO) [292]	(C#, VB) https://msdn.microsoft.com/en-us/magazine/mt632275.aspx, accessed on 6 February 2020	— introduction of the center agent concept [367] — addition of cannibalism components [368] — dynamic step size adaptation [369]	— Artificial Neural Networks training [367] — engineering design [367]
Water strider algorithm (WfSA) [370]		— inclusion of chaos [371], of quasi-opposition and elite-guide evolution mechanism [372], adaptable parameters [373]	— optimal design of renewable energy systems [372]

3.2.2. Other Invertebrates

This group includes algorithms inspired from different invertebrates that do not belong to the insect class.

- *Arachnids*

The social behavior of spiders, i.e., communication using vibrations throughout the web, represents the source of inspiration for the Social Spider Optimization (SSO) [265]. It emulates a group of spiders that contains both males and females and applies different evolutionary operators to mimic the distinct behaviors typically found in the colony. In addition to the cooperation behavior, a mating operator, applicable only to the strongest individuals, is introduced to increase diversity. A comprehensive review of the SSO, which covers its main variants and applications, is the work of Luque–Chang et al. [374].

Distinctively from the SSO, which models the cooperative behavior and exchange of information through the web, the Social Spider Algorithm (SSA) [375] simulates the foraging behavior of social spiders. The SSA does not distinguish the individuals by sex; all the spiders share the same search operations. Compared to the SSO, the SSA is simpler, it uses a single random move operator and depends on the parameter settings to control the search [375]. A parameter sensitivity analysis (through advanced non-parametric statistical tests) indicated that medium population, small to medium attenuation rate, medium crossover probability and small mutation probability lead to good results for the majority of the problem being tested [376].

- *Crustacea*

The Krill Herd Algorithm (KHA) [377] is inspired from the herding behavior of krill individuals and was developed to solve non-complex optimization problems. It is a population-based approach that uses three main ways to explore the search space: (i) movement, induced by other individuals; (ii) foraging; (iii) random diffusion. A review that covers the main improvements and applications of the KHA is [378]. Newer studies (after 2017) that use the KHA to solve specific problems are presented in Table 5.

- *Annelid worms*

The reproduction mechanisms used by earthworms are the source of inspiration for the Earthwork Optimization Algorithm (EWA) [379]. The idealized rules that the EWA follows are: (i) all the earthworms can produce offspring using only two types of reproduction; (ii) the number of genes of the offspring is the same as the parent's; (iii) the best individuals go directly, without change, to the next generation so as to ensure that the population cannot deteriorate throughout generations.

- *Tunicata*

Tunicates are marine, small bio-luminescent invertebrates with a unique mode of jet propulsion. The movement strategy and the swarming behavior of tunicates was modelled in the Tunicate Swarm Algorithm (TSA) [380], its performance for a set of benchmarking problems being similar with state-of-the-art approaches. The time complexity of the TSA is O(iterations $\times$ population_size $\times$ dimensionality $\times$ N) where N indicates the jet propulsion and swarm behaviors.

Table 5. Improvements and applications for other invertebrates-based metaheuristics (alphabetically sorted).

Algorithm	Source Code	Modifications and Improvements	Applications
Earthwork Optimization Algorithm (EWA) [379]	(MATLAB—author source) https://in.mathworks.com/matlabcentral/fileexchange/53479-earthworm-optimization-algorithm-ewa?s_tid=FX_rc3_behav, accessed on 15 March 2021	− hybridization with Differential Evolution [381]	− home energy management system [381–383]
Krill Herd Algorithm (KHA) [377]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/55486-krill-herd-algorithm, accessed on 9 February 2020	− hybridization with Ant Colony Optimization [384], Bat Algorithm [385], Clonal Selection [386] − modification at inner lever [387]	− Artificial Neural Network training [388] − planning and scheduling [385] − feature reduction [389] − text clustering [387]
Social Spider Optimization (SSO) [265]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/46942-a-swarm-optimization-algorithm-inspired-in-the-behavior-of-the-social-spider, accessed on 26 April 2020	− modification of the solution generation mechanism [390] − inclusion of rough sets [391] − constraint handling [392]	− reactive power dispatch [390] − minimum number attributes reduction problem [391] − Artificial Neural Network training [393]
Social Spider Algorithm (SSA) [375]	(MATLAB, C++, Python—author source) https://github.com/James-Yu/SocialSpiderAlgorithm, accessed on 18 January 2020	− inclusion of differential mutation [394], chaos [395] − new mutation strategy [396]	− train energy optimization [397] − scheduling [394,395] − load dispatch problem [398]
Tunicate Swarm Algorithm (TSA) [380]	(MATLAB—author source) https://www.mathworks.com/matlabcentral/fileexchange/75182-tunicate-swarm-algorithm-tsa, accessed on 17 July 2021	− inclusion of local escaping operator [399], Levi Flight distribution [400] − hybridization with Salp Swarm Optimizer [401]	− control and operation of automated distribution networks [400] − connectivity and coverage optimization in wireless sensor networks [401]

4. The Exploration–Exploitation Balance

Although based on different ideas, for all metaheuristic optimizers, the mechanisms used to simulate the optimization behaviors are similar. Generally speaking, all the algorithms in this class start with an initial population of potential solutions, usually randomly generated, which is evolved, i.e., modified by a series of mechanisms that can include—among others—selection, crossover and mutation, until a stopping criterion is satisfied. The actual strategies used to perform these steps and the mechanisms used to control the exploration–exploitation balance (EEB) influence the performance behavior and represent the main elements that make the distinction between algorithms.

The research in the area of metaheuristics often mentions the exploration and exploitation aspects of the algorithms; however, these terms have never been formally defined [402]. Informally, exploration is defined as the process of visiting entirely new regions of the search space, also known as the global search ability of the algorithm, while exploration is the process of visiting those regions of the search space within the neighborhood of previously visited points, which represents the local search ability [402]. Pure exploration leads to a decrease in precision but increases the ability of finding new, good solutions, while pure exploitation refines the existing solution and drives the search to a local optimum [403]. Because it indicates how the resources are allocated, knowing the EEB information can be useful to determine the impact of specific aspects of the algorithm [404]. The EEB can be seen from two points of view: (i) exploration and exploitation as opposing forces; (ii) exploration and exploitation as orthogonal forces [404]. However, it was shown that the opposing forces view is a special case of the orthogonal view and, thus, EEB monitoring must involve a metric for the exploration axis and one for the exploitation axis [404].

For evolutionary algorithms, it was shown that different operators, depending on the algorithm, are acting as exploitation or exploration procedures [1]. In population-based algorithms, the EEB is connected to the population diversity: when this is high, the algorithm is explorative and when it is low, the behavior is exploitative [1]. Although a diverse population is a prerequisite rather than a guarantee for the EEB and a good EEB can be reached through other means, e.g., fitness, using diversity is one of the simplest methods for achieving it [402]. Due to the fact that the problems to be solved must be encoded into a binary or real-valued vector, a clear distinction between the genotype (the encoded structure) and the phenotype (the actual problem) must be done. In this context, the diversity can be measured at the genotype level, at the phenotype level or using complex or composite measures that combine the genotype and the phenotype.

In [402], the diversity-based approaches applied to the EEB are classified as: (i) diversity maintenance—in this case it is assumed that the techniques will maintain diversity per se; (ii) diversity control, where feedback from measured individual fitness and or/fitness improvement is used to direct the evolution towards exploration or exploitation; (iii) diversity learning—a long-term history in combination with machine learning approaches is used to learn unexplored search areas; iv) other direct approaches (Figure 2). In the case of diversity maintenance, two categories can be indicated: niching and non-niching. The niching techniques represent extensions of the algorithms to multi-modal domains [405]. One of the most comprehensive definitions for niching is given in [406]: "Niching is a two-step procedure that (a) concurrently or subsequently distributes individuals into distinct basins of attraction and (b) facilitates approximation of the corresponding (local) optimizers".
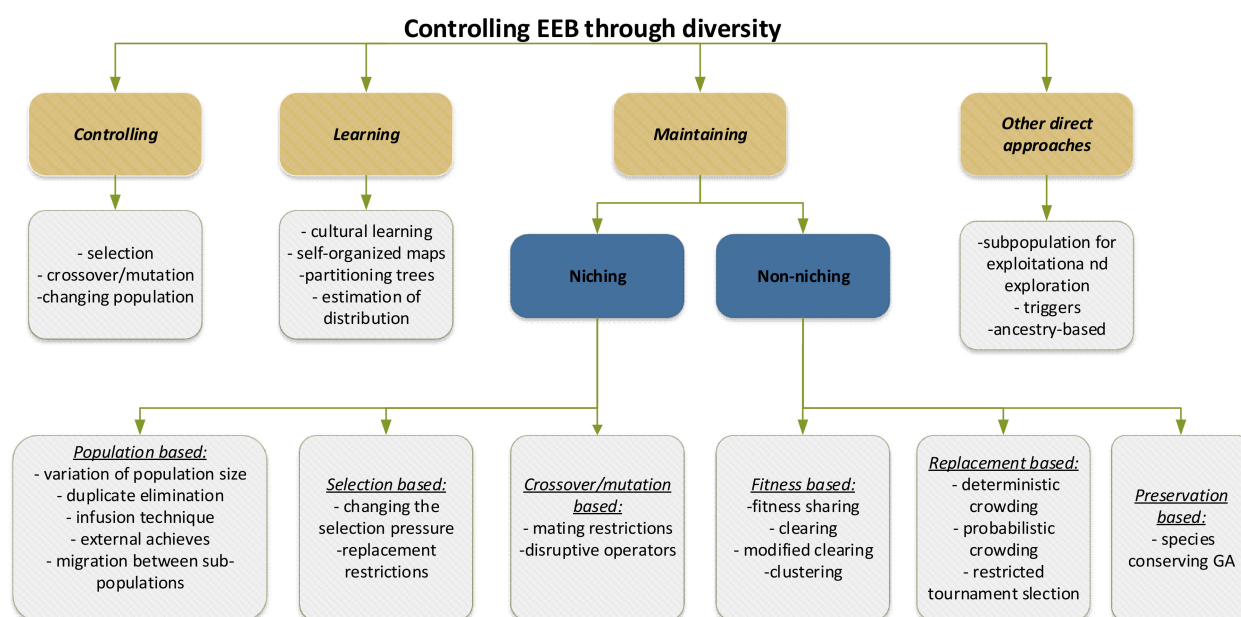
Figure 2. Mechanisms used to control the exploration–exploitation balance through diversity.

Table 6 shows the different mechanisms for the EEB used by the initial versions of the algorithms presented in Section 3. The subsequent modifications performed to the base versions are not considered in this table. The following notations are used: *C* indicates the controlling mechanisms, *L* the learning approaches, *OD* are the other direct approaches and *H.* represents the hybrid techniques. *Pop.* represents the population-based techniques, *Sel.* the selection based, *Crs.* the crossover/mutation based, *Fit.* the fitness based, *Rep.* the replacement based and *Pres.* the preservation-based approaches. In Table 6, in each case a specific approach is encountered in the algorithm, an x is set in the corresponding column.

Table 6. The diversity-based approaches used for EEB by the bio-inspired metaheuristic optimizers (alphabetically sorted).

Algorithm	C	L	Maintaining								OD	Description of the Mechanisms
			Niching				Non-Niching					
			Pop.	*Sel.*	*Crs.*	*H.*	*Fit.*	*Rep.*	*Pres.*	*H.*		
Backtracking Search Algorithm Optimization (BSA)			x		x						x	− dual-population with a complex crossover approach
Bat Algorithm (BA)			x								x	− a local search is performed to randomly selected best solutions; -new solutions are generated by flying randomly and added to the population if their fitness is good
Bee Colony Inspired Algorithm (BCiA)			x								x	− 2 populations with individuals that migrate between them
Bird Mating Optimizer (BMO)			x		x							− new individuals are randomly inserted after a certain number of generations − the mating behavior of male and female is different
Blind Naked Mole Rats (BNMR)			x									− a new solution is generated and added to the population its fitness is good
Bumblebees (B)			x									− infusion of new individuals in the population and elimination of the worst
Bumble Bee Mating Optimization (BBMO)					x							− there are mating restrictions with division of roles within individuals in the population
Chicken Swarm Optimization (CSO)			x									− different subpopulations are considered
Competition over Resources (COR)			x								x	− 2 subgroups performing a separate search
Cuckoo Search (CS)			x									− worst individuals are removed and new ones are added
Cuckoo Optimization Algorithm (COA)								x				− k-means is used to cluster the cuckoos into groups
Drosophila Food Search Optimization (DFO)											x	− Redundant Search algorithm is used to perform a neighborhood search
Elephant Herding Optimization (EHO)			x									− each iteration, the males- i.e., the worst individuals from each clan are replaced by new individuals (separating operator)
Elephant Search Algorithm (ESA)											x	− the population is formed by two groups: male (that performs exploration) and females (that performs exploitation)
Grey Wolf Optimizer (GWO)												− the balance between exploration and exploitation is performed through control parameters
Hunting Search (HuS)	x											− when the individuals in the population are close together, a reset procedure is applied
Krill Herd Algorithm (KHA)							x					− the fitness function is used to simulate the motion induced by other krill individuals

Table 6. *Cont.*

Algorithm	C	L	Maintaining								OD	Description of the Mechanisms
			Niching					Non-Niching				
			Pop.	*Sel.*	*Crs.*	*H.*	*Fit.*	*Rep.*	*Pres.*	*H.*		
Locust Swarm (LS2)											x	− the exploration and exploitation steps are one after the other
Lion's Algorithm (LA)			x					x				− k-means is used to perform gender grouping − at pride update, sick/weak cubs are killed/eliminated
Lion Optimization Algorithm (LOA)			x		x							− the behavior of males and females in the pride is different; -there are different populations (nomad and pride); -restricted mating between males and females
Lion Pride Optimization Algorithm (LPOA)			x									− multiple subpopulations (prides) are considered
Monarch Butterfly Optimization (MBO)			x								x	− 2 subpopulations with individuals that migrate between them
Moth Swarm Algorithm (MSA)	x											− uses an adaptive crossover based on diversity
Moth Search (MS) algorithm			x								x	− at each generation the population recombines all the individuals and splits them into 2 groups based on fitness
Pigeon Inspired Optimization (PIO)	x											− the landmark operation implies reducing the number of individuals to half
Satin Bowerbird Optimizer (SBO)									x			− the probability of finding a mate is based on the fitness function
Sperm Whale Algorithm (SWA)					x							− restricted mating (strongest male mates with several females)
Social Spider Optimization (SSO)					x						x	− the individuals are gender oriented, with different behavior; the offspring generation is restricted to the classical male-female mating
Spider Monkey Optimization (SMO)											x	− multiple sub-populations
Spotted Hyena Optimizer (SHO)	x											− the parameters controlling the prey encircling are changed as the search progresses
Squirrel Search Algorithm (SSA)			x									− when seasonal conditions are satisfied, the squirrels are randomly relocated

As it can be observed from Table 6, some strategies are more popular than others; the non-niching techniques based on population are the most used approaches for diversity maintenance. This can be explained by the fact that the bio-inspired metaheuristics are population-based and, therefore, the most intuitive methods consist in modifying the characteristics of the population as a means of improving performance.

5. Algorithm Selection

As it can be observed, the list of algorithms, even when the source of inspiration is restrung to a single category, is extensive. In practice, the most common question is what is the best suited algorithm and how can it can be successfully applied for a specific problem? Unfortunately, answering this question is not an easy task, as by their definition, heuristics can provide sufficiently good solutions to an optimization problem. Thus, depending on the accepted level of precision, there can be more than one heuristic that can generate acceptable solutions in terms of quality. However, performance is not the only aspect that can be taken into account [407]. The issue of algorithm selection was formalized by Rice in [408], and involves: (i) a problem space P; (ii) an algorithm space A; (iii) a mapping PxA onto R (also known as performance model). This implies that there must exist an extensive set of problem instances and features that describe them and the algorithm state at any time [409]. Thus, although advances regarding algorithm selection were made [409,410], the most used strategy in the metaheuristic filed is based on comparison. The work of LaTorre et al. [411] presents a series of methodological guidelines for comparing metaheuristics that involves: (i) selection of benchmarks and refence algorithms; (ii) validation of results (with statistical analysis and visualization); (iii) parameter tunning; (iv) identification of usefulness.

As a demonstration, in this work, the three most used real-world benchmark problems were selected (Table 7) and used to determine (based on already reported results) what the best performing animal-inspired metaheuristics are. The centralized results, organized from the best to the worst solution, are presented in Table 8 for the pressure vessel design, Table 9 for the welded beam design and Table 10 for the tension/compression spring design. All the considered problems are constrained minimization problems and their description can be found in the references from the column "Reported work" of Tables 8–10.

Table 7. Real-world problems characteristics.

Problem	Decision Variables	Inequality Constraints
Tension/Compression spring	3 (diameter, mean coil diameter, number of active coils)	4
Pressure vessel design	4 (thinckness of the shell, thinkness of th head, inner radius, length of the cylindrical section)	4
Welded beam design	4 (thikness of the weld, length of the attached part of the bar, height of the bar and thickess of the bar)	7

Table 8. Solutions for the pressure vessel design problem.

Algorithms	Reported Work	Modified Version	Ts	Th	R	L	Optimal Cost
Sooty Tern Optimization Algorithm (STOA) [54]	[54]	No	0.778095	0.38324	40.31511	200	5879.1253
Emperor Pinguin Optimization (EPO) [55]	[55]	No	0.778099	0.383241	40.31512	200	5880.07
Chameleon Swarm Algorithm (ChSA) [224]	[224]	No	12.450698	6.154387	40.31961	200	5885.3327
Memory based Dragonfly algorithm (MHDA) [313]	[313]	Yes	0.778169	0.384649	40.3196	200	5885.3353
COOT [412]	[412]	No	0.77817	0.384651	40.31961	200	5885.3487
Marine Predator Algorithm (MPA) -continuous variant [229]	[229]	No	0.77816876	0.3846497	40.31962	199.99999	5885.3353
Spotted Hyena Optimizer (SHO) [142]	[55]	No	0.77821	0.384889	40.31504	200	5885.5773
Modified Spider Monkey Optimization (SMONM) [203]	[412]	Yes	0.778322	0.384725	40.32759	199.8889	5885.595
African Vulture Optimization Algorithm (AVOA) [57]	[57]	No	0.778954	0.3850374	40.36031	199.43429	5886.67659
Grey Wolf Optimizer (GWO) [136]	[55]	No	0.779035	0.38466	40.32779	199.65029	5889.3689
Dragonfly Algorithm (DA) [264]	[313]	No	0.782825	0.384649	40.3196	200	5923.11
Aquila Optimization (AO) [47]	[47]	No	1.0540	0.182806	59.6219	38.8050	5949.2258
Improved Grasshoper Oprimization (OBLGOA) [413]	[412]	Yes	0.81622	0.4035	42.29113	174.81119	5966.6716
Slime Mould Algorithm (SMA) [414]	[414]	No	0.7931	0.3932	40.6711	196.2178	5994.1857
Harris Hawk Optimization (HHO) [46]	[412]	No	0.81758383	0.4072927	42.09174	176.71963	6000.46259
Improved Artificial bee Colony (I-ABC greedy) [415]	[412]	Yes	0.8125	0.4375	42.0984	176.6369	6059.7124
Firefly Algorithm (FA) [416]	[412]	No	0.8125	0.4375	42.09844	176.63659	6059.7143
Moth-flame Optimization (MFO) [284]	[412]	No	0.8125	0.4375	42.09844	176.63659	6059.7143
Marine Predator Algorithm (MPA) -mixed integer variant [229]	[229]	No	0.8125	0.4375	42.09844	176.63660	6059.7144
Sine-Cosine Grey Wolf Optimizer (SC-GWO) [417]	[412]	Yes	0.8125	0.4375	42.0984	176.6370	6059.7179
Co-evolutionary Differential Evolution (CDE) [418]	[229]	Yes	0.8125	0.4375	42.09841	176.6376	6059.734
Whale Optimization Algorithm (WOA) [134]	[412]	No	0.8125	0.4375	42.09826	176.63899	6059.741
Bacterial foraging Optimization (BFOA) [419]	[229]	No	0.8125	0.4375	42.09639	176.68323	6060.46
Co-evolutionary Particle Swarm Optimization (CPSO) [420]	[313]	Yes	0.8125	0.4375	42.09126	176.7465	6061.077
Artificial Immune System-Genetic Algorithm (HGA-1) [421]	[229]	Yes	0.8125	0.4375	42.0492	177.2522	6065.821
Artificial Immune System-Genetic Algorithm (HGA-2) [421]	[229]	Yes	1.125	0.5625	58.1267	44.5941	6832.583
Harmony Search (HS) [422]	[229]	No	1.125	0.625	58.2789	43.7549	7198.433

Table 9. Solutions for the welded beam design.

Algorithms	Reported Work	Modified Version	τ	σ	Pc	δ	Optimal Cost
Aquila Optimization (AO) [47]	[47]	No	0.1631	3.3652	9.0202	0.2067	1.6566
Butterfly Optimization Algorithm (BOA) [283]	[283]	No	0.1736	2.969	8.7637	0.2188	1.6644
COOT [412]	[412]	No	0.19883	3.33797	9.19199	0.19883	1.6703
Memory based Dragon Fly algorithm(MHDA)) [313]	[313]	Yes	0.20573	3.25312	9.03662	0.20573	1.69525
Slime Mould algorithm (SMA) [414]	[414]	No	0.2054	3.2589	9.0384	0.2058	1.696
Dragonfly Algorithm (DA) [264]	[313]	No	0.19429	3.46681	9.04543	0.2057	1.70808
Tunicate Swarm Algorithm (TSA) [380]	[412]	No	0.20329	3.47114	9.0351	0.20115	1.72102
Seagull optimization algorithm (SOA) [53]	[53]	No	0.205408	3.472316	9.035208	0.20114	1.723485
Emperor Pinguin Optimization (EPO) [55]	[55]	No	0.205411	3.472341	9.035215	0.20115	1.723589
Sooty Tern Optimization Algorithm (STOA) [54]	[54]	No	0.205415	3.472346	9.03522	0.20116	1.72359
Improved Artificial bee Colony (I-ABC greedy) [415]	[412]	Yes	0.20573	3.47049	9.03662	0.20573	1.72482
Co-evolutionary Particle Swarm Optimization (CPSO) [420]	[313]	Yes	0.20573	3.47049	9.03662	0.20573	1.72485
Modified Artificial Bee Colony (ABC) [263]	[313]	Yes	0.20573	3.47049	9.03662	0.20573	1.72485
Modified Spider Monkey Optimization (SMONM) [203]	[412]	Yes	0.20573	3.47049	9.03662	0.20573	1.72485
Chameleon Swarm Algorithm (ChSA) [224]	[224]	No	0.205730	3.470489	9.036624	0.20573	1.724852
African Vulture Optimization Algorithm (AVOA) [57]	[57]	No	0.20573	3.470474	9.03662	0.20573	1.724852
Moth-flame Optimization (MFO) [284]	[370]	No	0.20573	3.47049	9.03662	0.20573	1.7249
Water Strider Algorithm (WSA) [370]	[370]	No	0.20573	3.47049	9.03662	0.20573	1.7249
Marine Predator Algorithm (MPA) [229]	[229]	No	0.20573	3.47051	9.03662	0.20573	1.72485
Salp Swarm Algorithm (SSA) [11]	[414]	No	0.2057	3.4714	9.0366	0.2057	1.7249
Derivative free Simulated Annealing (SA) [423]	[313]	Yes	0.20564	3.47258	9.03662	0.20573	1.725
Spotted Hyena Optimizer(SHO) [142]	[412]	No	0.20556	3.47485	9.0358	0.20581	1.72566
Improved Grasshopper Optimization Algorithm (OBLGOA) [413]	[412]	Yes	0.20577	3.47114	9.03273	0.20591	1.7257
Grey Wolf Optimizer (GWO) [136]	[414]	No	0.2057	3.4784	9.0368	0.2058	1.7262
Whale Optimization Algorithm (WOA) [134]	[134]	No	0.205396	3.484293	9.037426	0.20627	1.730499
Harris Hawk Optimization (HHO) [46]	[412]	No	0.20404	3.53106	9.02746	0.20615	1.73199
Sailfish Optimizer (SFO) [222]	[222]	No	0.2038	3.6630	9.0506	0.2064	1.73231
Co-evolutionary Differential Evolution (CDE) [418]	[412]	Yes	0.20314	3.543	9.0335	0.20618	1.73346
Levy Flight Distribution (LFD) [424]	[412]	No	0.1857	3.907	9.1552	0.2051	1.77
Harmony Search and Genetic Algorithm (HSA-GA) [425]	[229]	Yes	0.2231	1.5815	12.8468	0.2245	2.25
Improved harmony Search (HS) [426]	[283]	Yes	0.2442	6.2231	8.2915	0.2443	2.3807
Differential Evolution with stochastic selection (DSS-DE) [427]	[229]	Yes	0.2444	6.1275	8.2915	0.2444	2.381
APPROX [428]	[134]	No	0.2444	6.2189	8.2915	0.2444	2.3815
Ragsdell [428]	[370]	No	0.2455	6.196	8.2915	0.2444	2.38154
David [428]	[134]	No	0.2434	6.2552	8.2915	0.2444	2.3841
Bacterial Foraging Optimization (BFOA) [419]	[229]	No	0.2057	3.4711	9.0367	0.2057	2.3868
Simplex [428]	[370]	No	0.2792	5.6256	7.7512	0.2796	2.5307
Random [428]	[134]	No	0.4575	4.7313	5.0853	0.66	4.1185

Table 10. Solutions for the tension/compression spring.

Algorithms	Reported Work	Modified Version	d	D	N	Optimal Cost
Aquila Optimization (AO) [47]	[47]	No	0.050243	0.35262	10.5425	0.011165
Butterfly Optimization Algorithm (BOA) [283]	[283]	No	0.051343	0.334871	12.9227	0.011965
Emperor Pinguin Optimization (EPO) [55]	[55]	No	0.051087	0.342908	12.0898	0.012656
Sooty Tern Optimization Algorithm (STOA) [54]	[54]	No	0.05109	0.34291	12.09	0.012656
FireFly algorithm (BA) [416]	[412]	No	0.05169	0.35673	11.2885	0.012665
Pathfinder algorithm (PFA) [429]	[229]	No	0.051726	0.357629	11.235724	0.012665
Marine Predator Algorithm (MPA) [229]	[229]	No	0.0517244	0.35757003	11.2391955	0.012665
Improved Artificial bee Colony (I-ABC greedy) [415]	[412]	Yes	0.051686	0.356014	11.202765	0.012665
COOT [412]	[412]	No	0.0516527	0.3558442	11.340383	0.012665
African Vulture Optimization Algorithm (AVOA) [57]	[57]	No	0.051669	0.3562553	11.316126	0.0126652
Chameleon Swarm Algorithm (ChSA) [224]	[224]	No	0.051778	0.358851	11.164981	0.0126653
Harris Hawk Optimization (HHO) [46]	[412]	No	0.0517963	0.3593053	11.138859	0.01266
Grey Wolf Optimizer (GWO) [136]	[412]	No	0.05169	0.356737	11.28885	0.012666
Modified Spider Monkey Optimization (SMONM) [203]	[412]	Yes	0.051918	0.362248	10.97194	0.012666
Moth-flame Optimization (MFO) [284]	[412]	No	0.0519944	0.36410932	10.868422	0.012666
Artificial Immune System-Genetic Algorithm (HGA-1) [421]	[229]	Yes	0.051302	0.347475	11.852177	0.012668
Co-evolutionary Differential Evolution (CDE) [418]	[229]	Yes	0.051609	0.354714	11.410831	0.01267
Improved harmony Search (HS) [426]	[283]	Yes	0.051154	0.349871	12.076432	0.012670
Bacterial Foraging Optimization (BFOA) [420]	[229]	No	0.051825	0.359935	11.107103	0.012671
Sine-Cosine Grey Wolf Optimizer (SC-GWO) [417]	[412]	Yes	0.051511	0.352376	11.5526	0.012672
Spotted Hyena Optimizer (SHO) [142]	[412]	No	0.051144	0.343751	12.0955	0.012674
Co-evolutionary Particle Swarm Optimization (CPSO) [420]	[229]	Yes	0.051728	0.357644	11.244543	0.012674
Whale Optimization Algorithm (WOA) [134]	[412]	No	0.051207	0.345215	0.004032	0.012676
Salp Swarm Algorithm (SSA) [11]	[412]	No	0.051207	0.345215	12.004032	0.012676
Improved Grasshopper Optimization Algorithm (OBLGOA) [413]	[412]	Yes	0.0530178	0.38953229	9.6001616	0.012701
Mathematical_optimization [430]	[283]	-	0.053396	0.39918	9.1854	0.012730
Constraint_correction [431]	[283]	-	0.05	0.3159	14.25	0.012833

In Tables 8–10, the column "Reported work" indicates the paper where the specific results were reported and where the control parameters used to obtain those results where indicated. The column "Modified version" indicates if the specific algorithm is a modified version of the base variant. As it can be observed, the majority of the algorithms in the top of the list represent base variants of metaheuristics proposed in the last five years. This indicates that, for this type of constrained problem with a reduced number of parameters, the newer metaheuristics tend to perform better than the older, more known metaheuristics, as well as the classical mathematical approaches.

To determine which algorithm is best suited to all the considered engineering problems, a Condorcet-based approach was applied [432]. It is based on the idea of the voting system, the problem of determining a rank of algorithms becoming an electoral problem. In this context, the considered algorithms represent the candidates and their solutions for each problem indicate the voters. Thus, as a majority-based method, the Condorcet algorithm determines the winner of an election as the candidate who outperforms or is equal to each candidate in a pair-wise comparison. As not all the algorithms considered were tested on

all of the problems, the Condorcet algorithm was applied for the metaheuristics tested on all three problems. The obtained results identified the top four metaheuristics as: EPO (45 votes), AO (42 votes), COOT (40 votes) and ChSA (34 votes). In the fifth and sixth pace, at equality with 32 votes, are I-ABC greedy and AVOA. As it can be observed, the difference between the algorithms placed in the first three positions is relatively small (2 votes). Similarly, there is a small difference between the algorithm placed in positions four, five and six. On the other hand, the difference between place three and four is larger (6 votes), indicating that the first three algorithms, when applied for the three engineering problems considered, performed substantially better than the next three. To test if there is a significant difference between the two groups, a *t*-Test Paired Sample was performed. The results obtained indicate a Pearson correlation of 0.999 and a P(T <= t) two-tail of 0.3168. As it is higher than 0.05, the null hypothesis is accepted, resulting in that there are no statistically significant differences between the results provided by the best three algorithms versus the results provided by the next three best algorithms. Thus, it can be concluded that, although from the 17 algorithms considered EPO is the winner, all of the first six algorithms can provide similar results and can be used successfully for solving the three engineering problems considered.

6. General Issues

Eighty five percent of articles that propose new bio-inspired metaheuristics have a high number of citations, i.e., more than 20/year, in a relatively reduced period in comparison with the norm in the area of artificial intelligence, where, during a year, the average number of citations is around 5. This indicates that the issues of finding good optimizers that can be easily applied to solve different problems is of high interest. However, a high percentage of the research performed and the subsequently published articles is focused on applications. An in-depth analysis of the theoretical aspects that influence the performance of the different operators used and their combination is relatively scarce. However, researchers are trying to correct this aspect and, in the latest years, a series of studies focusing on the analysis of theoretical and practical aspects were published [10,409,411,433–436].

The high number of citations was observed mostly for the algorithms for which the source-code is provided or easy to find. For the majority of these highly cited metaheuristics, the research focused on two main directions: (i) improvement or hybridization and (ii) applications—usually without any analysis or motivation for the selection of a particular algorithm. However, although the rate of publishing new algorithms (and the variants proposed) is high, studies focusing on the aspects that make an algorithm successful or on the mechanisms that lead to improvements in performance are quite rare. Therefore, in order to further advance the knowledge in the area and to establish some comprehensive basis on which newer, faster and efficient approaches are developed and successfully applied to problems from various domains, the mechanisms and the influence of different aspects of the problem/optimizer domain must be analyzed in depth. In the last years, it was observed that the manuscripts publishing new metaheuristics contain a more detailed analysis and comparison with other algorithms. However, these studies are predominantly based on empirical observations gathered from simulations performed on a handful of benchmarks (mathematical functions such as those proposed in the CEC competitions and engineering problems such as welded beam, pressure vessel or tension/compression spring design). The fact that the CEC test problems are considered, until recently, only in C++ and Matlab can be one explanation for the fact that the majority of these metaheuristics are implemented in Matlab.

This paper presented a comprehensive list of metaheuristics, with a focus on animals as a source of inspiration. All the studied works have a similar organization. First, a general description of the domain is presented, followed by the natural mechanisms used as sources of inspiration and a section with the implementation strategies used to simulate the natural mechanisms. In the results section, a set of problems are selected to demonstrate the strengths and weaknesses of the proposed approach. Although this seems straightforward

and easy to understand, in the metaheuristics area, the main issues are related to the fact that:

- The biological terminology used is complex and, in many cases, difficult to understand, which conceals that, in the implementation phase, the mechanisms used are simple and well-known and are, in fact, variations on the same theme; the work of [437] tries to shed some light onto the computational mechanisms used by the best-known metaheuristics. Also, in terms of the real-world mechanisms modeled, some of the algorithms are 'weak inspired', in the sense that the so-called modeled behavior is not met in the species that give the name of the algorithm [438];

- After overcoming the terminology barrier, upon a closer analysis, some of the so-called new algorithms not only do not have any novel aspect, but the papers describing them are incomplete or an implementation following the pseudo-code identifies other problems. In this regard, the work of Nguyen Van Thieu is worth mentioning, wherein he strides to implement these in a comprehensive python module with metaheuristics [438], and identified some of these dummy metaheuristics;

- Although some algorithms are inspired from the same source, the mechanisms modeled are different. For example, for Pidgeon inspired approaches, two algorithms were identified: PIO, which focuses on the movement of an individual from point A to point B and POA, which focuses on the movement of pigeons, taking into account the social interactions;

- There are multiple benchmark libraries that can be used and, in the majority of cases, the problems chosen by the authors to test the performance are very varied; thus, a comparison of performance between multiple algorithms based on the published literature is not always possible. The work of [439] presents the winners of some well-known competitions where standard benchmarks are used. In addition, as publishing bad results is sometimes discouraged, only the problems with the best results are chosen. In [2], it was shown that, in the comparison phase, the number of algorithms, the number of problems tested and the statistics used can lead to wrong conclusions if not properly selected;

- In an attempt to create high performance algorithms, the tendency is to include multiple strategies that have proven efficient over the years, e.g., self-adaptation, chaos, local search, etc. However, this has led to over-complicated methods that do not always show a direct correlation between complexity and performance. For example, for two winners of the CEC2016 competition, simpler versions (without operators biased towards 0) proved competitive against a large number of metaheuristics and even performed better for problems with solutions not close to 0 [435].

It can be observed that the source of inspiration follows the main classes identified in the biological taxonomy. Although the inspiration sources are varied and range from the behavior of simple organisms to the mechanisms used for survival of the species by large animals, the simulation of these sources is focused on exploration and exploitation, which translates into mathematical relations that make changes on the individuals. As it can be observed from Table 6, the majority of mechanisms used to control the exploration–exploitation balance in the standard versions belong to the niching class and are population-based. Overall, the manner in which the mechanisms that simulate the real-life behavior of animals are implemented and the combinations used represent the main aspects that differentiate the algorithms and that make them more sensitive or insensitive to the characteristics of the problem being solved, e.g., multi-modality, separability, etc.

Based on the aspects described above, the following potential directions of research can be identified:

- Performance measurement: the issue of performance is a complex aspect, especially taking into account that different metrics can be used. Although the tendency is to see performance as the capability to provide the best solution, other aspects such as complexity, computational resources consumed and stability can be employed. Moreover, how the best solution is identified is usually based on experiments with

mean and standard deviation as validation criteria [440], and a standardization of all these metrics and criteria of evolution can be a further step in the development of a general framework for metaheuristics.

- Performance analysis and improvement: identifying the main mechanisms that make a particular algorithm efficient for a particular class or group of problems. In this context, a better understanding of the exploration–exploitation balance, convergence analysis, diversity and the strategies that focus these aspects to a direction or another would help in providing a better foundation for the improvement of existing variants and creating new ones. In this regard, some studies focusing on these aspects were published (examples include: convergence analysis [441–444], fitness landscape analysis [445–448], exploration–exploitation [449–451]). However, additional research is required to reach field maturity.

- Algorithm selection: procedures and algorithms that can automatically select the best metaheuristic for a specific problem or group of problems. A wide level of applicability is one of the reasons for the popularity of metaheuristics. Thus, better strategies that can allow an easy identification of suitable algorithms are necessary. In this context, in the last few years, various methodologies and strategies to compare and select algorithms were proposed [2,411,433] and recommender systems were developed [409]. However, they are not widely accepted and applied and additional research in simplifying and generalizing these aspects is required.

7. Conclusions

This work is a review that focus on the animal-inspired metaheuristics proposed between 2006–2021. It was observed that, despite the rising number of critiques addressed to the entire metaheuristic community, the trend of proposing algorithms based on novel ideas and sources of inspiration does not seem to slow down considerably. In fact, it maintains the growth rate already observed a few years ago, mainly due to the large area of applicability and popularity of both the older, more established algorithms such as the GA, and newer approaches, for which the tendency is to provide the source code and thus increase the ease of use.

Regarding the animal-based metaheuristics, the most used source of inspiration is represented by the vertebrates, where easily observable behaviors such as food finding and mating are mathematically modeled using various approaches. However, a closer analysis of the inspiration sources indicated that all the main branches of the biological classification are represented in the metaheuristic world. This shows that researchers are actively searching for new ideas in unusual places and are not hindered by the difficulties associated with identifying the mechanisms of the behaviors of hard to analyze sources, such as animals living in remote and difficult to reach areas. In fact, the more exotic the inspiration source and the more uncommon the behavior, the higher the probability of finding new mechanisms that can be translated into truly novel approaches.

The main directions of research that were identified focus on the proposal of new metaheuristics and their application for various types of problems and only a few studies tackle the influence of specific operators or mechanisms on performance. Better performing algorithms are always desired and using nature as a source of inspiration can lead to new advances in this field of metaheuristics. However, attention must be paid not only to the source of inspiration but also to how this inspiration is modelled and put into practice. Similarity with existing variants, performance, complexity, exploration–exploitation balance, proper comparison and use of benchmarks must also be taken into account. An in-depth analysis of all the aspects that influence the performance behavior and the relations with the characteristics of the problems being solved can benefit both the metaheuristic community and the areas where these algorithms are applied.

Author Contributions: E.N.D. design the study and drafted the work. V.D. performed the literature search, revised and completed the manuscript. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Salcedo-Sanz, S. Modern meta-heuristics based on nonlinear physics processes: A review of models and design procedures. *Phys. Rep.* **2016**, *655*, 1–70. [CrossRef]
2. Hussain, K.; Salleh, M.N.M.; Cheng, S.; Shi, Y. Metaheuristic research: A comprehensive survey. *Artif. Intell. Rev.* **2018**, *52*, 2191–2233. [CrossRef]
3. Nabaei, A.; Hamian, M.; Parsaei, M.R.; Safdari, R.; Samad-Soltani, T.; Zarrabi, H.; Ghassemi, A. Topologies and performance of intelligent algorithms: A comprehensive review. *Artif. Intell. Rev.* **2016**, *49*, 79–103. [CrossRef]
4. Del Ser, J.; Osaba, E.; Molina, D.; Yang, X.S.; Salcedo-Sanz, S.; Camacho, D.; Das, S.; Suganthan, P.N.; Coello, C.A.C.; Herrera, F. Bio-inspired computation: Where we stand and what's next. *Swarm Evol. Comput.* **2019**, *48*, 220–250. [CrossRef]
5. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [CrossRef]
6. Lam, A.; Li, V.O.K. Chemical-Reaction-Inspired Metaheuristic for Optimization. *IEEE Trans. Evol. Comput.* **2009**, *14*, 381–399. [CrossRef]
7. Adam, S.P.; Alexandropoulos, S.A.N.; Pardalos, P.M.; Vrahatis, M.N. No Free Lunch Theorem: A Review. In *Approximation and Optimization: Algorithms, Complexity and Applications*; Demetriou, I.C., Pardalos, P.M., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 57–82.
8. Hosseini, S.; Al Khaled, A. A survey on the Imperialist Competitive Algorithm metaheuristic: Implementation in engineering domain and directions for future research. *Appl. Soft Comput.* **2014**, *24*, 1078–1094. [CrossRef]
9. Doğan, B.; Ölmez, T. A new metaheuristic for numerical function optimization: Vortex Search algorithm. *Inf. Sci.* **2015**, *293*, 125–145. [CrossRef]
10. Stegherr, H.; Heider, M.; Hähner, J. Classifying Metaheuristics: Towards a unified multi-level classification system. *Nat. Comput.* **2020**, 1–17. [CrossRef]
11. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
12. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [CrossRef]
13. Kirkpatrick, S.; Gelatt, C.D., Jr.; Vecchi, M.P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680. [CrossRef]
14. Glover, F. Tabu search—Part I. *ORSA J. Comput.* **1989**, *1*, 190–206. [CrossRef]
15. Mladenović, N.; Hansen, P. Variable neighborhood search. *Comput. Oper. Res.* **1997**, *24*, 1097–1100. [CrossRef]
16. Lourenço, H.R.; Martin, O.C.; Stützle, T. Iterated Local Search. In *Handbook of Metaheuristics*; Springer: Boston, MA, USA, 2003; pp. 320–353.
17. Turky, A.M.; Abdullah, S. A multi-population electromagnetic algorithm for dynamic optimisation problems. *Appl. Soft Comput.* **2014**, *22*, 474–482. [CrossRef]
18. Yang, X.S. A New Metaheuristic Bat-Inspired Algorithm. *Nicso 2010 Nat. Inspired Coop. Strateg. Optim.* **2010**, *284*, 65–74.
19. Ahandani, M.A. A diversified shuffled frog leaping: An application for parameter identification. *Appl. Math. Comput.* **2014**, *239*, 1–16. [CrossRef]
20. Zheng, Y.-J. Water wave optimization: A new nature-inspired metaheuristic. *Comput. Oper. Res.* **2015**, *55*, 1–11. [CrossRef]
21. Moghdani, R.; Salimifard, K. Volleyball Premier League Algorithm. *Appl. Soft Comput.* **2018**, *64*, 161–185. [CrossRef]
22. Molina, D.; Poyatos, J.; Del Ser, J.; García, S.; Hussain, A.; Herrera, F. Comprehensive taxonomies of nature-and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis recommendations. *Cogn. Comput.* **2020**, *12*, 897–939. [CrossRef]
23. Fausto, F.; Reyna-Orta, A.; Cuevas, E.; Andrade, G.; Perez-Cisneros, M. From ants to whales: Metaheuristics for all tastes. *Artif. Intell. Rev.* **2019**, *53*, 753–810. [CrossRef]
24. Brabazon, A.; McGarraghy, S. Formal Models of Foraging. In *Foraging-Inspired Optimisation Algorithms*; Brabazon, A., McGarraghy, S., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 23–44.
25. Zhu, G.-Y.; Zhang, W.-B. Optimal foraging algorithm for global optimization. *Appl. Soft Comput.* **2017**, *51*, 294–313. [CrossRef]
26. Brabazon, A.; Cui, W.; O'Neill, M. The raven roosting optimisation algorithm. *Soft Comput.* **2015**, *20*, 525–545. [CrossRef]
27. Brabazon, A.; McGarraghy, S. Introduction to Foraging-Inspired Algorithms. In *Foraging-Inspired Optimisation Algorithms*; Brabazon, A., McGarraghy, S., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 87–101.
28. Askarzadeh, A. Bird mating optimizer: An optimization algorithm inspired by bird mating strategies. *Commun. Nonlinear Sci. Numer. Simul.* **2014**, *19*, 1213–1228. [CrossRef]
29. Yang, X.-S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), IEEE, Coimbatore, India, 9–11 December 2009.

30. Wang, L.; Yin, Y.; Zhong, Y. Cuckoo search with varied scaling factor. *Front. Comput. Sci.* **2015**, *9*, 623–635. [CrossRef]
31. Chawla, M.; Duhan, M. Levy Flights in Metaheuristics Optimization Algorithms—A Review. *Appl. Artif. Intell.* **2018**, *32*, 802–821. [CrossRef]
32. Rakhshani, H.; Rahati, A. Snap-drift cuckoo search: A novel cuckoo search optimization algorithm. *Appl. Soft Comput.* **2017**, *52*, 771–794. [CrossRef]
33. Joshi, A.; Kulkarni, O.; Kakandikar, G.; Nandedkar, V. Cuckoo Search Optimization—A Review. *Mater. Today Proc.* **2017**, *4*, 7262–7269. [CrossRef]
34. Rajabioun, R. Cuckoo Optimization Algorithm. *Appl. Soft Comput.* **2011**, *11*, 5508–5518. [CrossRef]
35. Moosavi, S.H.S.; Bardsiri, V.K. Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation. *Eng. Appl. Artif. Intell.* **2017**, *60*, 1–15. [CrossRef]
36. Yang, X.S.; Deb, S. Eagle strategy using Levy walk and firefly algorithms for stochastic optimization. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 101–111.
37. Yang, X.-S. *Nature-inspired metaheuristic algorithms*; Luniver press: Bristol, UK, 2010.
38. Yang, X.-S.; Deb, S. Two-stage eagle strategy with differential evolution. *Int. J. Bio-Inspired Comput.* **2012**, *4*, 1–5. [CrossRef]
39. Gandomi, A.H.; Yang, X.-S.; Talatahari, S.; Deb, S. Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization. *Comput. Math. Appl.* **2012**, *63*, 191–200. [CrossRef]
40. Talatahari, S.; Gandomi, A.H.; Yang, X.-S.; Deb, S. Optimum design of frame structures using the Eagle Strategy with Differential Evolution. *Eng. Struct.* **2015**, *91*, 16–25. [CrossRef]
41. Storn, R.; Price, K. Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces. *J. Glob. Optim.* **1995**, *11*, 341–359. [CrossRef]
42. Meng, X.; Liu, Y.; Gao, X.; Zhang, H. A New Bio-inspired Algorithm: Chicken Swarm Optimization. In *Advances in Swarm Intelligence, Pt1*; Tan, Y., Shi, Y., Coello, C.A.C., Eds.; Springer: Cham, Switzerland, 2014; pp. 86–94.
43. Askarzadeh, A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Comput. Struct.* **2016**, *169*, 1–12. [CrossRef]
44. Jain, M.; Maurya, S.; Rani, A.; Singh, V. Owl search algorithm: A novel nature-inspired heuristic paradigm for global optimization. *J. Intell. Fuzzy Syst.* **2018**, *34*, 1573–1582. [CrossRef]
45. Zhuoran, Z.; Changqiang, H.; Hanqiao, H.; Shangqin, T.; Kangsheng, D. An optimization method: Hummingbirds optimization algorithm. *J. Syst. Eng. Electron.* **2018**, *29*, 386–404.
46. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Futur. Gener. Comput. Syst.* **2019**, *97*, 849–887. [CrossRef]
47. Abualigah, L.; Yousri, D.; Elaziz, M.A.; Ewees, A.A.; Al-Qaness, M.A.; Gandomi, A.H. Aquila Optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* **2021**, *157*, 10725. [CrossRef]
48. Mohammadi-Balani, A.; Nayeri, M.D.; Azar, A.; Taghizadeh-Yazdi, M. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. *Comput. Ind. Eng.* **2020**, *152*, 107050. [CrossRef]
49. Sun, J.; Lei, X. Geese-inspired hybrid particle swarm optimization algorithm for traveling salesman problem. In Proceedings of the 2009 International Conference on Artificial Intelligence and Computational Intelligence, IEEE, Shanghai, China, 7–8 November 2009.
50. Duman, E.; Uysal, M.; Alkaya, A.F. Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem. *Inf. Sci.* **2012**, *217*, 65–77. [CrossRef]
51. Goel, S. Pigeon Optimization Algorithm: A Novel Approach for Solving Optimization Problems. In Proceedings of the 2014 International Conference on Data Mining and Intelligent Computing (Icdmic), IEEE, Delhi, India, 5–6 September 2014.
52. Duan, H.; Qiao, P. Pigeon-inspired optimization: A new swarm intelligence optimizer for air robot path planning. *Int. J. Intell. Comput. Cybern.* **2014**, *7*, 24–37. [CrossRef]
53. Dhiman, G.; Kumar, V. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowl. Based Syst.* **2018**, *165*, 169–196. [CrossRef]
54. Dhiman, G.; Kaur, A. STOA: A bio-inspired based optimization algorithm for industrial engineering problems. *Eng. Appl. Artif. Intell.* **2019**, *82*, 148–174. [CrossRef]
55. Dhiman, G.; Kumar, V. Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowl. Based Syst.* **2018**, *159*, 20–50. [CrossRef]
56. Harifi, S.; Khalilian, M.; Mohammadzadeh, J.; Ebrahimnejad, S. Emperor Penguins Colony: A new metaheuristic algorithm for optimization. *Evol. Intell.* **2019**, *12*, 1–16. [CrossRef]
57. Abdollahzadeh, B.; Gharehchopogh, F.S.; Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Comput. Ind. Eng.* **2021**, *158*, 107408. [CrossRef]
58. Amiri, K.; Niknam, T. Optimal Planning of a Multi-carrier Energy Hub Using the Modified Bird Mating Optimizer. *Iran. J. Sci. Technol. Trans. Electr. Eng.* **2018**, *43*, 517–526. [CrossRef]
59. Ahmadi, M.; Kazemi, K.; Aarabi, A.; Niknam, T.; Helfroush, M.S. Image segmentation using multilevel thresholding based on modified bird mating optimization. *Multimed. Tools Appl.* **2019**, *78*, 23003–23027. [CrossRef]
60. Sadeeq, H.; Abdulazeez, A.; Kako, N.; Abrahim, A. A Novel Hybrid Bird Mating Optimizer with Differential Evolution for Engineering Design Optimization Problems. In Proceedings of the International Conference of Reliable Information and Communication Technology, Johor Bahru, Malaysia, 23–24 April 2017.

61. Zhang, Q.; Yu, G.; Song, H. A hybrid bird mating optimizer algorithm with teaching-learning-based optimization for global numerical optimization. *Stat. Optim. Inf. Comput.* **2015**, *3*, 54–65. [CrossRef]
62. Zhu, J.; Huang, M.; Lu, Z. Bird mating optimizer for structural damage detection using a hybrid objective function. *Swarm Evol. Comput.* **2017**, *35*, 41–52. [CrossRef]
63. Goswami, D.; Chakraborty, S. Multi-objective optimization of electrochemical discharge machining processes: A posteriori approach based on bird mating optimizer. *Opsearch* **2016**, *54*, 306–335. [CrossRef]
64. Skarzadeh, A.; Coelho, L.D.S. Determination of photovoltaic modules parameters at different operating conditions using a novel bird mating optimizer approach. *Energy Convers. Manag.* **2015**, *89*, 608–614. [CrossRef]
65. Zouache, D.; Arby, Y.O.; Nouioua, F.; Ben Abdelaziz, F. Multi-objective chicken swarm optimization: A novel algorithm for solving multi-objective optimization problems. *Comput. Ind. Eng.* **2019**, *129*, 377–391. [CrossRef]
66. Chen, Y.L.; He, P.L.; Zhang, Y.H. Combining Penalty Function with Modified Chicken Swarm Optimization for Constrained Optimization. In Proceedings of the First International Conference on Information Sciences, Machinery, Materials and Energy, Congqing, China, 11–13 April 2015.
67. Wu, D.; Kong, F.; Gao, W.; Shen, Y.; Ji, Z. Improved chicken swarm optimization. In Proceedings of the 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER); IEEE, Shenyang, China, 8–12 June 2015.
68. Khan, A.; Shah, R.; Bukhari, J.; Akhter, N.; Attaullah; Idrees, M.; Ahmad, H. A Novel Chicken Swarm Neural Network Model for Crude Oil Price Prediction. In *Advances on Computational Intelligence in Energy*; Springer: Cham, Switzerland, 2019; pp. 39–58.
69. Liu, D.; Liu, C.; Fu, Q.; Li, T.; Khan, M.I.; Cui, S.; Faiz, M.A. Projection pursuit evaluation model of regional surface water environment based on improved chicken swarm optimization algorithm. *Water Resour. Manag.* **2018**, *32*, 1325–1342. [CrossRef]
70. Banerjee, S.; Chattopadhyay, S. Improved serially concatenated convolution turbo code (SCCTC) using chicken swarm optimization. In Proceedings of the 2015 IEEE Power, Communication and Information Technology Conference (PCITC), IEEE, Bhubaneswar, India, 15–17 October 2015.
71. Javidi, A.; Salajegheh, E.; Salajegheh, J. Enhanced crow search algorithm for optimum design of structures. *Appl. Soft Comput.* **2019**, *77*, 274–289. [CrossRef]
72. Díaz, P.; Pérez-Cisneros, M.; Cuevas, E.; Avalos, O.; Gálvez, J.; Hinojosa, S.; Zaldivar, D. An Improved Crow Search Algorithm Applied to Energy Problems. *Energies* **2018**, *11*, 571. [CrossRef]
73. Hinojosa, S.; Oliva, D.; Cuevas, E.; Pajares, G.; Avalos, O.; Gálvez, J. Improving multi-criterion optimization with chaos: A novel Multi-Objective Chaotic Crow Search Algorithm. *Neural Comput. Appl.* **2018**, *29*, 319–335. [CrossRef]
74. Sayed, G.I.; Hassanien, A.E.; Azar, A.T. Feature selection via a novel chaotic crow search algorithm. *Neural Comput. Appl.* **2017**, *31*, 171–188. [CrossRef]
75. Dos Santos Coelho, L.; Richter, C.; Mariani, V.C.; Askarzadeh, A. Modified crow search approach applied to electromagnetic optimization. In Proceedings of the 2016 IEEE Conference on Electromagnetic Field Computation (CEFC), IEEE, Miami, FL, USA, 11–13 November 2016.
76. Gupta, D.; Sundaram, S.; Khanna, A.; Hassanien, A.E.; De Albuquerque, V.H.C. Improved diagnosis of Parkinson's disease using optimized crow search algorithm. *Comput. Electr. Eng.* **2018**, *68*, 412–424. [CrossRef]
77. Oliva, D.; Hinojosa, S.; Cuevas, E.; Pajares, G.; Avalos, O.; Galvez, J. Cross entropy based thresholding for magnetic resonance brain images using Crow Search Algorithm. *Expert Syst. Appl.* **2017**, *79*, 164–180. [CrossRef]
78. Chi, R.; Su, Y.-X.; Zhang, D.-H.; Chi, X.-X.; Zhang, H.-J. A hybridization of cuckoo search and particle swarm optimization for solving optimization problems. *Neural Comput. Appl.* **2017**, *31*, 653–670. [CrossRef]
79. Feng, Y.; Wang, G.-G.; Feng, Q.; Zhao, X.-J. An Effective Hybrid Cuckoo Search Algorithm with Improved Shuffled Frog Leaping Algorithm for 0-1 Knapsack Problems. *Comput. Intell. Neurosci.* **2014**, *2014*, 857254. [CrossRef]
80. Wang, L.; Zhong, Y. Cuckoo Search Algorithm with Chaotic Maps. *Math. Probl. Eng.* **2015**, *2015*, 1–14. [CrossRef]
81. Khodier, M. Comprehensive study of linear antenna array optimisation using the cuckoo search algorithm. *IET Microw. Antennas Propag.* **2019**, *13*, 1325–1333. [CrossRef]
82. Ikeda, S.; Ooka, R. Metaheuristic optimization methods for a comprehensive operating schedule of battery, thermal energy storage, and heat source in a building energy system. *Appl. Energy* **2015**, *151*, 192–205. [CrossRef]
83. Afzalan, E.; Joorabian, M. An improved cuckoo search algorithm for power economic load dispatch. *Int. Trans. Electr. Energy Syst.* **2014**, *25*, 958–975. [CrossRef]
84. Shokri-Ghaleh, H.; Alfi, A. A comparison between optimization algorithms applied to synchronization of bilateral teleoperation systems against time delay and modeling uncertainties. *Appl. Soft Comput.* **2014**, *24*, 447–456. [CrossRef]
85. Gheisarnejad, M. An effective hybrid harmony search and cuckoo optimization algorithm based fuzzy PID controller for load frequency control. *Appl. Soft Comput.* **2018**, *65*, 121–138. [CrossRef]
86. Mahmoudi, S.; Lotfi, S. Modified cuckoo optimization algorithm (MCOA) to solve graph coloring problem. *Appl. Soft Comput.* **2015**, *33*, 48–64. [CrossRef]
87. Mohammadrezapour, O.; YoosefDoost, I.; Ebrahimi, M. Cuckoo optimization algorithm in optimal water allocation and crop planning under various weather conditions (case study: Qazvin plain, Iran). *Neural Comput. Appl.* **2017**, *31*, 1879–1892. [CrossRef]
88. Bayati, M. Using cuckoo optimization algorithm and imperialist competitive algorithm to solve inverse kinematics problem for numerical control of robotic manipulators. *Proc. Inst. Mech. Eng. Part I J. Syst. Control. Eng.* **2015**, *229*, 375–387. [CrossRef]

89. Dhiman, G.; Oliva, D.; Kaur, A.; Singh, K.K.; Vimal, S.; Sharma, A.; Cengiz, K. BEPO: A novel binary emperor penguin optimizer for automatic feature selection. *Knowl. Based Syst.* **2020**, *211*, 106560. [CrossRef]
90. Kaur, H.; Rai, A.; Bhatia, S.S.; Dhiman, G. MOEPO: A novel Multi-objective Emperor Penguin Optimizer for global optimization: Special application in ranking of cloud service providers. *Eng. Appl. Artif. Intell.* **2020**, *96*, 104008. [CrossRef]
91. Baliarsingh, S.K.; Vipsita, S.; Muhammad, K.; Bakshi, S. Analysis of high-dimensional biomedical data using an evolutionary multi-objective emperor penguin optimizer. *Swarm Evol. Comput.* **2019**, *48*, 262–273. [CrossRef]
92. Dhiman, G. ESA: A hybrid bio-inspired metaheuristic optimization approach for engineering problems. *Eng. Comput.* **2019**, *37*, 323–353. [CrossRef]
93. Baliarsingh, S.K.; Ding, W.; Vipsita, S.; Bakshi, S. A memetic algorithm using emperor penguin and social engineering optimization for medical data classification. *Appl. Soft Comput.* **2019**, *85*, 105773. [CrossRef]
94. Xing, Z. An improved emperor penguin optimization based multilevel thresholding for color image segmentation. *Knowl. Based Syst.* **2020**, *194*, 105570. [CrossRef]
95. Harifi, S.; Mohammadzadeh, J.; Khalilian, M.; Ebrahimnejad, S. Hybrid-EPC: An Emperor Penguins Colony algorithm with crossover and mutation operators and its application in community detection. *Prog. Artif. Intell.* **2021**, *10*, 181–193. [CrossRef]
96. Harifi, S.; Khalilian, M.; Mohammadzadeh, J.; Ebrahimnejad, S. Optimization in solving inventory control problem using nature inspired Emperor Penguins Colony algorithm. *J. Intell. Manuf.* **2020**, *32*, 1361–1375. [CrossRef]
97. Harifi, S.; Khalilian, M.; Mohammadzadeh, J.; Ebrahimnejad, S. Optimizing a Neuro-Fuzzy System Based on Nature-Inspired Emperor Penguins Colony Optimization Algorithm. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 1110–1124. [CrossRef]
98. Chen, H.; Jiao, S.; Wang, M.; Heidari, A.A.; Zhao, X. Parameters identification of photovoltaic cells and modules using diversification-enriched Harris hawks optimization with chaotic drifts. *J. Clean. Prod.* **2019**, *244*, 118778. [CrossRef]
99. Zhang, Y.; Liu, R.; Wang, X.; Chen, H.; Li, C. Boosted binary Harris hawks optimizer and feature selection. *Eng. Comput.* **2021**, *37*, 3741–3770. [CrossRef]
100. Chen, H.; Heidari, A.A.; Chen, H.; Wang, M.; Pan, Z.; Gandomi, A.H. Multi-population differential evolution-assisted Harris hawks optimization: Framework and case studies. *Future Gener. Comput. Syst.* **2020**, *111*, 175–198. [CrossRef]
101. Essa, F.; Elaziz, M.A.; Elsheikh, A. An enhanced productivity prediction model of active solar still using artificial neural network and Harris Hawks optimizer. *Appl. Therm. Eng.* **2020**, *170*, 115020. [CrossRef]
102. Meng, T.; Pan, Q.-K.; Li, J.-Q.; Sang, H.-Y. An improved migrating birds optimization for an integrated lot-streaming flow shop scheduling problem. *Swarm Evol. Comput.* **2018**, *38*, 64–78. [CrossRef]
103. Segredo, E.; Lalla-Ruiz, E.; Hart, E.; Voß, S. On the performance of the hybridisation between migrating birds optimisation variants and differential evolution for large scale continuous problems. *Expert Syst. Appl.* **2018**, *102*, 126–142. [CrossRef]
104. Sioud, A.; Gagné, C. Enhanced migrating birds optimization algorithm for the permutation flow shop problem with sequence dependent setup times. *Eur. J. Oper. Res.* **2018**, *264*, 66–73. [CrossRef]
105. Zhang, B.; Pan, Q.-K.; Gao, L.; Zhang, X.-L.; Sang, H.-Y.; Li, J.-Q. An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming. *Appl. Soft Comput.* **2017**, *52*, 14–27. [CrossRef]
106. Gao, L.; Pan, Q.-K. A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem. *Inf. Sci.* **2016**, *372*, 655–676. [CrossRef]
107. Niroomand, S.; Hadi-Vencheh, A.; Şahin, R.; Vizvári, B. Modified migrating birds optimization algorithm for closed loop layout with exact distances in flexible manufacturing systems. *Expert Syst. Appl.* **2015**, *42*, 6586–6597. [CrossRef]
108. Pan, Q.-K.; Dong, Y. An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation. *Inf. Sci.* **2014**, *277*, 643–655. [CrossRef]
109. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm Evol. Comput.* **2019**, *44*, 148–175. [CrossRef]
110. Andrea, H.; Aranguren, I.; Oliva, D.; Abd Elaziz, M.; Cuevas, E. Efficient image segmentation through 2D histograms and an improved owl search algorithm. *Int. J. Mach. Learn. Cybern.* **2021**, *12*, 131–150.
111. El-Ashmawi, W.H.; Elminaam, D.S.A.; Nabil, A.M.; Eldesouky, E. A chaotic owl search algorithm based bilateral negotiation model. *Ain Shams Eng. J.* **2020**, *11*, 1163–1178. [CrossRef]
112. Mandal, A.K.; Sen, R.; Chakraborty, B. Binary owl search algorithm for feature subset selection. In Proceedings of the 2019 IEEE 10th International Conference on Awareness Science and Technology (iCAST), IEEE, Morioka, Japan, 23–25 October 2019.
113. Zhong, Y.; Wang, L.; Lin, M.; Zhang, H. Discrete pigeon-inspired optimization algorithm with Metropolis acceptance criterion for large-scale traveling salesman problem. *Swarm Evol. Comput.* **2019**, *48*, 134–144. [CrossRef]
114. Wang, H.; Zhang, Z.; Dai, Z.; Chen, J.; Zhu, X.; Du, W.; Cao, X. Heterogeneous pigeon-inspired optimization. *Sci. China Inf. Sci.* **2019**, *62*, 70205. [CrossRef]
115. Yang, Z.; Duan, H.; Fan, Y.; Deng, Y. Automatic Carrier Landing System multilayer parameter design based on Cauchy Mutation Pigeon-Inspired Optimization. *Aerosp. Sci. Technol.* **2018**, *79*, 518–530. [CrossRef]
116. Deng, Y.; Duan, H. Control parameter design for automatic carrier landing system via pigeon-inspired optimization. *Nonlinear Dyn.* **2016**, *85*, 97–106. [CrossRef]
117. Qiu, H.; Duan, H. Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. *Sci. China Ser. E Technol. Sci.* **2015**, *58*, 1915–1923. [CrossRef]

118. Zhang, B.; Duan, H. Predator-Prey Pigeon-Inspired Optimization for UAV Three-Dimensional Path Planning. In *Advances in Swarm Intelligence, Icsi 2014, Pt Ii*; Tan, Y., Shi, Y., Coello, C.A.C., Eds.; Springer: Cham, Switzerland, 2014; pp. 96–105.

119. Jiang, F.; He, J.; Zeng, Z. Pigeon-inspired optimization and extreme learning machine via wavelet packet analysis for predicting bulk commodity futures prices. *Sci. China Inf. Sci.* **2019**, *62*, 70204. [CrossRef]

120. Torabi, S.; Safi-Esfahani, F. Improved Raven Roosting Optimization algorithm (IRRO). *Swarm Evol. Comput.* **2018**, *40*, 144–154. [CrossRef]

121. Torabi, S.; Safi-Esfahani, F. A dynamic task scheduling framework based on chicken swarm and improved raven roosting optimization methods in cloud computing. *J. Supercomput.* **2018**, *74*, 2581–2626. [CrossRef]

122. Zhang, S.; Zhou, Y.; Luo, Q. A Complex-Valued Encoding Satin Bowerbird Optimization Algorithm for Global Optimization. *Evolving Systems* **2021**, *12*, 191–205. [CrossRef]

123. El-Hay, E.; El-Hameed, M.; El-Fergany, A. Steady-state and dynamic models of solid oxide fuel cells based on Satin Bowerbird Optimizer. *Int. J. Hydrogen Energy* **2018**, *43*, 14751–14761. [CrossRef]

124. Dhiman, G.; Singh, K.K.; Soni, M.; Nagar, A.; Dehghani, M.; Slowik, A.; Kaur, A.; Sharma, A.; Houssein, E.H.; Cengiz, K. MOSOA: A new multi-objective seagull optimization algorithm. *Expert Syst. Appl.* **2020**, *167*, 114150. [CrossRef]

125. Che, Y.; He, D. A Hybrid Whale Optimization with Seagull Algorithm for Global Optimization Problems. *Math. Probl. Eng.* **2021**, *2021*, 1–31.

126. Das, G.; Panda, R. Seagull-Cuckoo Search Algorithm for Function Optimization. In Proceedings of the 2021 6th International Conference for Convergence in Technology (I2CT), IEEE, Maharashtra, India, 2–4 April 2021.

127. Jia, H.; Xing, Z.; Song, W. A New Hybrid Seagull Optimization Algorithm for Feature Selection. *IEEE Access* **2019**, *7*, 49614–49631. [CrossRef]

128. Ali, H.H.; Fathy, A.; Kassem, A.M. Optimal model predictive control for LFC of multi-interconnected plants comprising renewable energy sources based on recent sooty terns approach. *Sustain. Energy Technol. Assess.* **2020**, *42*, 100844. [CrossRef]

129. Addi, N.S.; Abdullah, S.; Hamdan, A.R. Multi-population cooperative bat algorithm-based optimization of artificial neural network model. *Inf. Sci.* **2015**, *294*, 628–644.

130. Rekaby, A. Directed Artificial Bat Algorithm (DABA)-A new bio-inspired algorithm. In Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, Mysore, India, 22–25 August 2013.

131. Topal, A.O.; Altun, O. A novel meta-heuristic algorithm: Dynamic Virtual Bats Algorithm. *Inf. Sci.* **2016**, *354*, 222–235. [CrossRef]

132. Kaveh, A.; Farhoudi, N. A new optimization method: Dolphin echolocation. *Adv. Eng. Softw.* **2013**, *59*, 53–70. [CrossRef]

133. Ebrahimi, A.; Khamehchi, E. Sperm whale algorithm: An effective metaheuristic algorithm for production optimization problems. *J. Nat. Gas Sci. Eng.* **2016**, *29*, 211–222. [CrossRef]

134. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]

135. Gharehchopogh, F.S.; Gholizadeh, H. A comprehensive survey: Whale Optimization Algorithm and its applications. *Swarm Evol. Comput.* **2019**, *48*, 1–24. [CrossRef]

136. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]

137. KKumar, V.; Kumar, D. An astrophysics-inspired Grey wolf algorithm for numerical optimization and its application to engineering design problems. *Adv. Eng. Softw.* **2017**, *112*, 231–254. [CrossRef]

138. Fong, S.; Deb, S.; Yang, X.-S. A heuristic optimization method inspired by wolf preying behavior. *Neural Comput. Appl.* **2015**, *26*, 1725–1738. [CrossRef]

139. Bansal, J.C.; Sharma, H.; Jadon, S.S.; Clerc, M. Spider Monkey Optimization algorithm for numerical optimization. *Memetic Comput.* **2014**, *6*, 31–47. [CrossRef]

140. Khishe, M.; Mosavi, M.R. Chimp optimization algorithm. *Expert Syst. Appl.* **2020**, *149*, 113338. [CrossRef]

141. Abdollahzadeh, B.; Gharehchopogh, F.S.; Mirjalili, S. Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. *Int. J. Intell. Syst.* **2021**, *36*, 5887–5958. [CrossRef]

142. Dhiman, G.; Kumar, V. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Adv. Eng. Softw.* **2017**, *114*, 48–70. [CrossRef]

143. Rajakumar, B.R. The Lion's Algorithm: A New Nature-Inspired Search Algorithm. *Procedia Technol.* **2012**, *6*, 126–135. [CrossRef]

144. Yazdani, M.; Jolai, F. Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm. *J. Comput. Des. Eng.* **2015**, *3*, 24–36. [CrossRef]

145. Kaveh, A.; Mahjoubi, S. Lion Pride Optimization Algorithm: A meta-heuristic method for global optimization problems. *Sci. Iran.* **2018**, *25*, 3113–3132. [CrossRef]

146. I Mohammad, T.M.H.; Mohammad, H.B.; Shirzadi, M.T.M.H.; Bagheri, M.H. A novel meta-heuristic algorithm for numerical function optimization: Blind, naked mole-rats (BNMR) algorithm. *Sci. Res. Essays* **2012**, *7*, 3566–3583. [CrossRef]

147. Deb, S.; Fong, S.; Tian, Z. Elephant search algorithm for optimization problems. In Proceedings of the 2015 Tenth International Conference on Digital Information Management (ICDIM), IEEE, Jeju, Korea, 21–23 October 2015.

148. Wang, G.G.; Deb, S.; Coelho, L.D.S. Elephant Herding Optimization. In Proceedings of the 2015 3rd International Symposium on Computational and Business Intelligence, IEEE, Bali, Indonesia, 7–9 December 2015.

149. Wang, G.G.; Deb, S.; Gao, X.Z.; Coelho, L.D.S. A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. *Int. J. Bio-Inspired Comput.* **2016**, *8*, 394. [CrossRef]

150. Osaba, E.; Yang, X.-S.; Fister, I.; Del Ser, J.; Lopez-Garcia, P.; Vazquez-Pardavila, A.J. A Discrete and Improved Bat Algorithm for solving a medical goods distribution problem with pharmacological waste collection. *Swarm Evol. Comput.* **2019**, *44*, 273–286. [CrossRef]

151. Chakri, A.; Khelif, R.; Benouaret, M.; Yang, X.-S. New directional bat algorithm for continuous optimization problems. *Expert Syst. Appl.* **2017**, *69*, 159–175. [CrossRef]

152. Meng, X.-B.; Gao, X.; Liu, Y.; Zhang, H. A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization. *Expert Syst. Appl.* **2015**, *42*, 6350–6364. [CrossRef]

153. Yılmaz, S.; Kucuksille, E.U. A new modification approach on bat algorithm for solving optimization problems. *Appl. Soft Comput.* **2015**, *28*, 259–275. [CrossRef]

154. Fister, I., Jr.; Fister, D.; Yang, X.S. A hybrid bat algorithm. *arXiv* **2013**, arXiv:1303.6310.

155. Mirjalili, S.; Mirjalili, S.M.; Yang, X.S. Binary bat algorithm. *Neural Comput. Appl.* **2014**, *25*, 663–681. [CrossRef]

156. Hong, W.-C.; Li, M.-W.; Geng, J.; Zhang, Y. Novel chaotic bat algorithm for forecasting complex motion of floating platforms. *Appl. Math. Model.* **2019**, *72*, 425–443. [CrossRef]

157. Ahmadlou, M.; Karimi, M.; Alizadeh, S.; Shirzadi, A.; Parvinnejhad, D.; Shahabi, H.; Panahi, M. Flood susceptibility assessment using integration of adaptive network-based fuzzy inference system (ANFIS) and biogeography-based optimization (BBO) and BAT algorithms (BA). *Geocarto Int.* **2018**, *34*, 1252–1272. [CrossRef]

158. Dao, T.-K.; Pan, T.-S.; Nguyen, T.-T.; Pan, J.-S. Parallel bat algorithm for optimizing makespan in job shop scheduling problems. *J. Intell. Manuf.* **2015**, *29*, 451–462. [CrossRef]

159. Osaba, E.; Yang, X.-S.; Diaz, F.; Lopez-Garcia, P.; Carballedo, R. An improved discrete bat algorithm for symmetric and asymmetric Traveling Salesman Problems. *Eng. Appl. Artif. Intell.* **2016**, *48*, 59–71. [CrossRef]

160. Bahmani-Firouzi, B.; Azizipanah-Abarghooee, R. Optimal sizing of battery energy storage for micro-grid operation management using a new improved bat algorithm. *Int. J. Electr. Power Energy Syst.* **2014**, *56*, 42–54. [CrossRef]

161. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H.; Talatahari, S. Bat algorithm for constrained optimization tasks. *Neural Comput. Appl.* **2012**, *22*, 1239–1255. [CrossRef]

162. Hasançebi, O.; Teke, T.; Pekcan, O. A bat-inspired algorithm for structural optimization. *Comput. Struct.* **2013**, *128*, 77–90. [CrossRef]

163. Taherdangkoo, M.; Shirzadi, M.H.; Yazdi, M.; Bagheri, M.H. A robust clustering method based on blind, naked mole-rats (BNMR) algorithm. *Swarm Evol. Comput.* **2013**, *10*, 1–11. [CrossRef]

164. Kaur, M.; Kaur, R.; Singh, N.; Dhiman, G. SChoA: A newly fusion of sine and cosine with chimp optimization algorithm for HLS of datapaths in digital filters and engineering applications. *Eng. Comput.* **2021**, 1–29. [CrossRef]

165. Khishe, M.; Mosavi, M. Classification of underwater acoustical dataset using neural network trained by Chimp Optimization Algorithm. *Appl. Acoust.* **2019**, *157*, 107005. [CrossRef]

166. Kaveh, A.; Hosseini, P. A simplified dolphin echolocation optimization method for optimum design of trusses. *Iran Univ. Sci. Technol.* **2014**, *4*, 381–397.

167. Daryan, A.S.; Palizi, S.; Farhoudi, N. Optimization of plastic analysis of moment frames using modified dolphin echolocation algorithm. *Adv. Struct. Eng.* **2019**, *22*, 2504–2516. [CrossRef]

168. Gholizadeh, S.; Poorhoseini, H. Optimum design of steel frame structures by a modified dolphin echolocation algorithm. *Struct. Eng. Mech.* **2015**, *55*, 535–554. [CrossRef]

169. Lenin, K.; Reddy, B.R.; Kalavathi, M.S. Dolphin echolocation algorithm for solving optimal reactive power dispatch problem. *Int. J. Comput.* **2014**, *12*, 1–15.

170. Topal, A.O.; Yildiz, Y.E.; Ozkul, M. Improved Dynamic Virtual Bats Algorithm for Global Numerical Optimization. In Proceedings of the World Congress on Engineering and Computer Science, San Francisco, CA, USA, 25–27 October 2017.

171. Elhosseini, M.A.; El Sehiemy, R.A.; Rashwan, Y.I.; Gao, X. On the performance improvement of elephant herding optimization algorithm. *Knowl. Based Syst.* **2019**, *166*, 58–70. [CrossRef]

172. Jafari, M.; Salajegheh, E.; Salajegheh, J. An efficient hybrid of elephant herding optimization and cultural algorithm for optimal design of trusses. *Eng. Comput.* **2018**, *35*, 781–801. [CrossRef]

173. Sadouki, S.C.; Tari, A. Multi-objective and discrete Elephants Herding Optimization algorithm for QoS aware web service composition. *RAIRO Oper. Res.* **2019**, *53*, 445–459. [CrossRef]

174. Tuba, E.; Capor-Hrosik, R.; Alihodzic, A.; Jovanovic, R.; Tuba, M. Chaotic elephant herding optimization algorithm. In Proceedings of the 2018 IEEE 16th World Symposium on Applied Machine Intelligence and Informatics (SAMI); IEEE, Kosice and Herlany, Slovakia, 7–10 February 2018.

175. Xu, H.; Cao, Q.; Fu, H.; Fu, C.; Chen, H.; Su, J. Application of Support Vector Machine Model Based on an Improved Elephant Herding Optimization Algorithm in Network Intrusion Detection. In *International CCF Conference on Artificial Intelligence, Xuzhou, China, 22–23 August 2019*; Springer: Singapore, 2019.

176. Tuba, E.; Alihodzic, A.; Tuba, M. Multilevel image thresholding using elephant herding optimization algorithm. In Proceedings of the 2017 14th International Conference on Engineering of Modern Electric Systems (EMES), IEEE, Oradea, Romania,, 1–2 June 2017.

177. Tuba, E.; Ribic, I.; Capor-Hrosik, R.; Tuba, M. Support Vector Machine Optimized by Elephant Herding Algorithm for Erythemato-Squamous Diseases Detection. *Procedia Comput. Sci.* **2017**, *122*, 916–923. [CrossRef]

178. Pichpibul, T. Modified Elephant Search Algorithm for Distribution of Snack Food in Thailand. In Proceedings of the 2nd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence, ACM, Phuket, Thailand, 24–25 March 2018.

179. Tian, Z.; Fong, S.; Wong, R.; Millham, R. Elephant search algorithm on data clustering. In Proceedings of the 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), IEEE, Changsha, China, 13–15 August 2016.

180. Deb, S.; Tian, Z.; Fong, S.; Wong, R.; Millham, R.; Wong, K.K.L. Elephant search algorithm applied to data clustering. *Soft Comput.* **2018**, *22*, 6035–6046. [CrossRef]

181. Deb, S.; Fong, S.; Tian, Z.; Wong, R.K.; Mohammed, S.; Fiaidhi, J. Finding approximate solutions of NP-hard optimization and TSP problems using elephant search algorithm. *J. Supercomput.* **2016**, *72*, 3960–3992. [CrossRef]

182. Abdel-Basset, M.; El-Shahat, D.; El-Henawy, I.; de Albuquerque, V.H.C.; Mirjalili, S. A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. *Expert Syst. Appl.* **2020**, *139*, 112824. [CrossRef]

183. Gupta, S.; Deep, K. A novel Random Walk Grey Wolf Optimizer. *Swarm Evol. Comput.* **2019**, *44*, 101–112. [CrossRef]

184. Lu, C.; Gao, L.; Yi, J. Grey wolf optimizer with cellular topological structure. *Expert Syst. Appl.* **2018**, *107*, 89–114. [CrossRef]

185. Qais, M.H.; Hasanien, H.M.; Alghuwainem, S. Augmented grey wolf optimizer for grid-connected PMSG-based wind energy conversion systems. *Appl. Soft Comput.* **2018**, *69*, 504–515. [CrossRef]

186. Emary, E.; Zawbaa, H.M.; Hassanien, A.E. Binary grey wolf optimization approaches for feature selection. *Neurocomputing* **2016**, *172*, 371–381. [CrossRef]

187. Mirjalili, S.; Saremi, S.; Mirjalili, S.M.; Coelho, L.D.S. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Syst. Appl.* **2016**, *47*, 106–119. [CrossRef]

188. Mirjalili, S.; Aljarah, I.; Mafarja, M.; Heidari, A.A.; Faris, H. Grey Wolf optimizer: Theory, literature review, and application in computational fluid dynamics problems. In *Nature-Inspired Optimizers*; Springer: Cham, Switzerland, 2020; pp. 87–105.

189. Nahak, N.; Sahoo, S.R.; Mallick, R.K. Design of dual optimal UPFC based PI controller to damp low frequency oscillation in power system. In Proceedings of the Technologies for Smart-City Energy Security and Power (ICSESP), IEEE, Bhubaneswar, India, 28–30 March 2018.

190. Emary, E.; Zawbaa, H.M.; Grosan, C. Experienced Gray Wolf Optimization Through Reinforcement Learning and Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.* **2017**, *29*, 681–694. [CrossRef]

191. Mohanty, S.; Subudhi, B.; Ray, P.K. A New MPPT Design Using Grey Wolf Optimization Technique for Photovoltaic System Under Partial Shading Conditions. *IEEE Trans. Sustain. Energy* **2015**, *7*, 181–188. [CrossRef]

192. Rajakumar, B. Lion algorithm for standard and large scale bilinear system identification: A global optimization based on Lion's social behavior. In Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC); IEEE, Beijing, China, 6–11 July 2014.

193. Marichelvam, M.; Manimaran, P.; Geetha, M. Solving flexible job shop scheduling problems using a hybrid lion optimisation algorithm. *Int. J. Adv. Oper. Manag.* **2018**, *10*, 91–108. [CrossRef]

194. Paraskar, S.; Singh, D.K.; Tapre, P.C. Lion algorithm for generation rescheduling based congestion management in deregulated power system. In Proceedings of the 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), IEEE, Chennai, India, 1–2 August 2017.

195. Sowmiyasree, S.; Sumitra, P. Lion Optimization Algorithm Using Data Mining Classification and Clustering Models. *GSJ* **2018**, *6*, 219–226.

196. Kaveh, A.; Mahjoubi, S. Optimum Design of Double-layer Barrel Vaults by Lion Pride Optimization Algorithm and a Comparative Study. *Structures* **2018**, *13*, 213–229. [CrossRef]

197. Engy, E.; Ali, E.; Sally, E.-G. An optimized artificial neural network approach based on sperm whale optimization algorithm for predicting fertility quality. *Stud. Inform. Control.* **2018**, *27*, 349–358.

198. Sharma, N.; Kaur, A.; Sharma, H.; Sharma, A.; Bansal, J.C. Chaotic Spider Monkey Optimization Algorithm with Enhanced Learning. In *Soft Computing for Problem Solving*; Springer: Singapore, 2018; pp. 149–161.

199. Sharma, A.; Sharma, H.; Bhargava, A.; Sharma, N.; Bansal, J.C. Optimal power flow analysis using lévy flight spider monkey optimisation algorithm. *Int. J. Artif. Intell. Soft Comput.* **2017**, *5*, 320–352. [CrossRef]

200. Gupta, K.; Deep, K.; Bansal, J.C. Improving the Local Search Ability of Spider Monkey Optimization Algorithm Using Quadratic Approximation for Unconstrained Optimization. *Comput. Intell.* **2016**, *33*, 210–240. [CrossRef]

201. Sharma, A.; Sharma, A.; Panigrahi, B.K.; Kiran, D.; Kumar, R. Ageist Spider Monkey Optimization algorithm. *Swarm Evol. Comput.* **2016**, *28*, 58–77. [CrossRef]

202. Sharma, A.; Sharma, H.; Bhargava, A.; Sharma, N.; Bansal, J.C. Optimal placement and sizing of capacitor using Limaçon inspired spider monkey optimization algorithm. *Memetic Comput.* **2016**, *9*, 311–331. [CrossRef]

203. Singh, P.R.; Elaziz, M.A.; Xiong, S. Modified Spider Monkey Optimization based on Nelder–Mead method for global optimization. *Expert Syst. Appl.* **2018**, *110*, 264–289. [CrossRef]

204. Singh, U.; Salgotra, R.; Rattan, M. A Novel Binary Spider Monkey Optimization Algorithm for Thinning of Concentric Circular Antenna Arrays. *IETE J. Res.* **2016**, *62*, 736–744. [CrossRef]

205. Tripathy, D.; Sahu, B.K.; Patnaik, B.; Choudhury, N.D. Spider monkey optimization based fuzzy-2D-PID controller for load frequency control in two-area multi source interconnected power system. In Proceedings of the 2018 Technologies for Smart-City Energy Security and Power (ICSESP), IEEE, Bhubaneswar, India, 29–30 March 2018.

206. Ehteram, M.; Karami, H.; Farzin, S. Reducing Irrigation Deficiencies Based Optimizing Model for Multi-Reservoir Systems Utilizing Spider Monkey Algorithm. *Water Resour. Manag.* **2018**, *32*, 2315–2334. [CrossRef]

207. Cheruku, R.; Edla, D.R.; Kuppili, V. SM-RuleMiner: Spider monkey based rule miner using novel fitness function for diabetes classification. *Comput. Biol. Med.* **2017**, *81*, 79–92. [CrossRef] [PubMed]

208. Dhiman, G.; Kumar, V. Multi-objective spotted hyena optimizer: A Multi-objective optimization algorithm for engineering problems. *Knowl. Based Syst.* **2018**, *150*, 175–197. [CrossRef]

209. Dhiman, G.; Kaur, A. Spotted hyena optimizer for solving engineering design problems. In Proceedings of the 2017 International Conference on Machine Learning and Data Science (MLDS), IEEE, Noida, India, 14–15 December 2017.

210. Luo, Q.; Li, J.; Zhou, Y.; Liao, L. Using Spotted Hyena Optimizer for Training Feedforward Neural Networks. In Proceedings of the International Conference on Intelligent Computing, Wuhan, China, 15–18 August 2018.

211. Dhiman, G.; Kaur, A. Optimizing the Design of Airfoil and Optical Buffer Problems Using Spotted Hyena Optimizer. *Designs* **2018**, *2*, 28. [CrossRef]

212. Abdel-Basset, M.; Manogaran, G.; El-Shahat, D.; Mirjalili, S. A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem. *Futur. Gener. Comput. Syst.* **2018**, *85*, 129–145. [CrossRef]

213. Mafarja, M.M.; Mirjalili, S. Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. *Neurocomputing* **2017**, *260*, 302–312. [CrossRef]

214. Kumar, N.; Hussain, I.; Singh, B.; Panigrahi, B.K. MPPT in Dynamic Condition of Partially Shaded PV System by Using WODE Technique. *IEEE Trans. Sustain. Energy* **2017**, *8*, 1204–1214. [CrossRef]

215. Kaveh, A.; Ghazaan, M.I. Enhanced whale optimization algorithm for sizing optimization of skeletal structures. *Mech. Based Des. Struct. Mach.* **2016**, *45*, 345–362. [CrossRef]

216. Sun, W.Z.; Wang, J.S. Elman Neural Network Soft-Sensor Model of Conversion Velocity in Polymerization Process Optimized by Chaos Whale Optimization Algorithm. *IEEE Access* **2017**, *5*, 13062–13076. [CrossRef]

217. Medani, K.B.O.; Sayah, S.; Bekrar, A. Whale optimization algorithm based optimal reactive power dispatch: A case study of the Algerian power system. *Electr. Power Syst. Res.* **2018**, *163*, 696–705. [CrossRef]

218. Mehne, H.H.; Mirjalili, S. A parallel numerical method for solving optimal control problems based on whale optimization algorithm. *Knowl. Based Syst.* **2018**, *151*, 114–123. [CrossRef]

219. Tharwat, A.; Moemen, Y.S.; Hassanien, A.E. Classification of toxicity effects of biotransformed hepatic drugs using whale optimized support vector machines. *J. Biomed. Inform.* **2017**, *68*, 132–149. [CrossRef] [PubMed]

220. Dao, T.-K.; Pan, T.-S.; Pan, J.-S. A multi-objective optimal mobile robot path planning based on whale optimization algorithm. In Proceedings of the 2016 IEEE 13th International Conference on Signal Processing (ICSP); IEEE, Chengdu, China, 6–10 November 2016.

221. Hassanien, A.E.; Abd Elfattah, M.; Aboulenin, S.; Schaefer, G.; Zhu, S.Y.; Korovin, I. Historic handwritten manuscript binarisation using whale optimisation. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, Budapest, Hungary, 9–12 October 2016.

222. Shadravan, S.; Naji, H.; Bardsiri, V. The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. *Eng. Appl. Artif. Intell.* **2019**, *80*, 20–34. [CrossRef]

223. Kumar, N.; Singh, N.; Vidyarthi, D.P. Artificial lizard search optimization (ALSO): A novel nature-inspired meta-heuristic algorithm. *Soft Comput.* **2021**, *25*, 6179–6201. [CrossRef]

224. Braik, M.S. Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Syst. Appl.* **2021**, *174*, 114685. [CrossRef]

225. Boettcher, S.; Percus, A. Nature's way of optimizing. *Artif. Intell.* **2000**, *119*, 275–286. [CrossRef]

226. Oftadeh, R.; Mahjoob, M.; Shariatpanahi, M. A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search. *Comput. Math. Appl.* **2010**, *60*, 2087–2098. [CrossRef]

227. Civicioglu, P. Backtracking Search Optimization Algorithm for numerical optimization problems. *Appl. Math. Comput.* **2013**, *219*, 8121–8144. [CrossRef]

228. Haldar, V.; Chakraborty, N. A novel evolutionary technique based on electrolocation principle of elephant nose fish and shark: Fish electrolocation optimization. *Soft Comput.* **2016**, *21*, 3827–3848. [CrossRef]

229. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]

230. Mohseni, S.; Gholami, R.; Zarei, N.; Zadeh, A.R. Competition over resources: A new optimization algorithm based on animals behavioral ecology. In Proceedings of the 2014 International Conference on Intelligent Networking and Collaborative Systems (INCoS), Salerno, Italy, 10–12 September 2014.

231. Sharafi, Y.; Khanesar, M.A.; Teshnehlab, M. COOA: Competitive optimization algorithm. *Swarm Evol. Comput.* **2016**, *30*, 39–63. [CrossRef]

232. Atashpaz-Gargari, E.; Lucas, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007.

233. Simon, D. Biogeography-based optimization. *IEEE Trans. Evol. Comput.* **2008**, *12*, 702–713. [CrossRef]

234. Niu, Q.; Zhang, L.; Li, K. A biogeography-based optimization algorithm with mutation strategies for model parameter estimation of solar and fuel cells. *Energy Convers. Manag.* **2014**, *86*, 1173–1185. [CrossRef]

235. Li, X.; Zhang, J.; Yin, M. Animal migration optimization: An optimization algorithm inspired by animal migration behavior. *Neural Comput. Appl.* **2013**, *24*, 1867–1877. [CrossRef]

236. Lai, Z.; Feng, X.; Yu, H. An Improved Animal Migration Optimization Algorithm Based on Interactive Learning Behavior for High Dimensional Optimization Problem. In Proceedings of the 2019 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS), IEEE, Shenzhen, China, 9–11 May 2019.

237. Cao, Y.; Li, X.; Wang, J. Opposition-Based Animal Migration Optimization. *Math. Probl. Eng.* **2013**, *2013*, 1–7. [CrossRef]

238. Son, L.H.; Chiclana, F.; Kumar, R.; Mittal, M.; Khari, M.; Chatterjee, J.M.; Baik, S.W. ARM–AMO: An efficient association rule mining algorithm based on animal migration optimization. *Knowl. Based Syst.* **2018**, *154*, 68–80. [CrossRef]

239. Ma, M.; Luo, Q.; Zhou, Y.; Chen, X.; Li, L. An Improved Animal Migration Optimization Algorithm for Clustering Analysis. *Discret. Dyn. Nat. Soc.* **2015**, *2015*, 1–12. [CrossRef]

240. Morales, A.; Crawford, B.; Soto, R.; Lemus-Romani, J.; Astorga, G.; Salas-Fernández, A.; Rubio, J.M. Optimization of Bridges Reinforcement by Conversion to Tied Arch Using an Animal Migration Algorithm. In Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Graz, Austria, 9–11 July 2019.

241. Farshi, T.R. A multilevel image thresholding using the animal migration optimization algorithm. *Iran J. Comput. Sci.* **2018**, *2*, 9–22. [CrossRef]

242. Tsai, H.-C. Improving backtracking search algorithm with variable search strategies for continuous optimization. *Appl. Soft Comput.* **2019**, *80*, 567–578. [CrossRef]

243. Zhou, J.; Ye, H.; Ji, X.; Deng, W. An improved backtracking search algorithm for casting heat treatment charge plan problem. *J. Intell. Manuf.* **2017**, *30*, 1335–1350. [CrossRef]

244. Lin, J. Oppositional backtracking search optimization algorithm for parameter identification of hyperchaotic systems. *Nonlinear Dyn.* **2014**, *80*, 209–219. [CrossRef]

245. Chen, D.; Zou, F.; Lu, R.; Wang, P. Learning backtracking search optimisation algorithm and its application. *Inf. Sci.* **2017**, *376*, 71–94. [CrossRef]

246. Zhang, C.; Lin, Q.; Gao, L.; Li, X. Backtracking Search Algorithm with three constraint handling methods for constrained optimization problems. *Expert Syst. Appl.* **2015**, *42*, 7831–7845. [CrossRef]

247. Pourdaryaei, A.; Mokhlis, H.; Illias, H.A.; Kaboli, S.H.A.; Ahmad, S. Short-Term Electricity Price Forecasting via Hybrid Backtracking Search Algorithm and ANFIS Approach. *IEEE Access* **2019**, *7*, 77674–77691. [CrossRef]

248. Ma, H.; Fei, M.; Simon, D.; Chen, Z. Biogeography-based optimization in noisy environments. *Trans. Inst. Meas. Control.* **2014**, *37*, 190–204. [CrossRef]

249. Saremi, S.; Mirjalili, S.; Lewis, A. Biogeography-based optimisation with chaos. *Neural Comput. Appl.* **2014**, *25*, 1077–1097. [CrossRef]

250. Pham, B.T.; Nguyen, M.D.; Bui, K.-T.T.; Prakash, I.; Chapi, K.; Bui, D.T. A novel artificial intelligence approach based on Multi-layer Perceptron Neural Network and Biogeography-based Optimization for predicting coefficient of consolidation of soil. *Catena* **2018**, *173*, 302–311. [CrossRef]

251. Mendes, L.A.; Freire, R.Z.; Coelho, L.D.S.; Moraes, A.S. Minimizing computational cost and energy demand of building lighting systems: A real time experiment using a modified competition over resources algorithm. *Energy Build.* **2017**, *139*, 108–123. [CrossRef]

252. Bouchekara, H.R.; Nahas, M. Optimization of magnetic actuators using competition over resources algorithm. In *Progress in Electromagnetics Research Symposium-Fall (PIERS-FALL), Singapore, 19–22 November 2017*; IEEE: Singapore, 2017.

253. Kulluk, S. A novel hybrid algorithm combining hunting search with harmony search algorithm for training neural networks. *J. Oper. Res. Soc.* **2013**, *64*, 748–761. [CrossRef]

254. Doğan, E.; Erdal, F. Hunting search algorithm based design optimization of steel cellular beams. In Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation, New York, NY, USA, 6–10 July 2013.

255. Elaziz, M.A.; Ewees, A.A.; Yousri, D.; Alwerfali, H.S.N.; Awad, Q.A.; Lu, S.; Al-Qaness, M.A.A. An Improved Marine Predators Algorithm With Fuzzy Entropy for Multi-Level Thresholding: Real World Example of COVID-19 CT Image Segmentation. *IEEE Access* **2020**, *8*, 125306–125330. [CrossRef] [PubMed]

256. Zhong, K.; Luo, Q.; Zhou, Y.; Jiang, M. TLMPA: Teaching-learning-based Marine Predators algorithm. *AIMS Math.* **2021**, *6*, 1395–1442. [CrossRef]

257. Abdel-Basset, M.; Mohamed, R.; Chakrabortty, R.K.; Ryan, M.; Mirjalili, S. New binary marine predators optimization algorithms for 0–1 knapsack problems. *Comput. Ind. Eng.* **2020**, *151*, 106949. [CrossRef]

258. Ridha, H.M. Parameters extraction of single and double diodes photovoltaic models using Marine Predators Algorithm and Lambert W function. *Sol. Energy* **2020**, *209*, 674–693. [CrossRef]

259. Sayed, G.I.; Solyman, M.; Hassanien, A.E. A novel chaotic optimal foraging algorithm for unconstrained and constrained problems and its application in white blood cell segmentation. *Neural Comput. Appl.* **2018**, *31*, 7633–7664. [CrossRef]

260. Zhang, W.-B.; Zhu, G.-Y. Drilling Path Optimization by Optimal Foraging Algorithm. *IEEE Trans. Ind. Informatics* **2017**, *14*, 2847–2856. [CrossRef]

261. Sayed, G.I.; Soliman, M.; Hassanien, A.E. Modified optimal foraging algorithm for parameters optimization of support vector machine. In Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications, Cairo, Egypt, 22–24 February 2018.

262. Srivastava, S.; Sahana, S.K. *The Insects of Innovative Computational Intelligence*; Springer: Singapore, 2017.
263. Akay, B.; Karaboga, D. A modified Artificial Bee Colony algorithm for real-parameter optimization. *Inf. Sci.* **2012**, *192*, 120–142. [CrossRef]
264. Mirjalili, S. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Comput. Appl.* **2015**, *27*, 1053–1073. [CrossRef]
265. Cuevas, E.; Cienfuegos, M.; Zaldivar-Navarro, D.; Perez-Cisneros, M.A. A swarm optimization algorithm inspired in the behavior of the social-spider. *Expert Syst. Appl.* **2013**, *40*, 6374–6384. [CrossRef]
266. Häckel, S.; Dippold, P. The Bee Colony-inspired Algorithm (BCiA): A two-stage approach for solving the vehicle routing problem with time windows. In Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference, ACM, Montreal, Canada, 8–12 July 2009.
267. Rajasekhar, A.; Lynn, N.; Das, S.; Suganthan, P. Computing with the collective intelligence of honey bees—A survey. *Swarm Evol. Comput.* **2017**, *32*, 25–48. [CrossRef]
268. Diwold, K.; Beekman, M.; Middendorf, M. Honeybee optimisation–an overview and a new bee inspired optimisation scheme. In *Handbook of Swarm Intelligence, In Handbook of Swarm Intelligence*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 295–327.
269. Comellas, F.; Martínez-Navarro, J. Bumblebees: A multiagent combinatorial optimization algorithm inspired by social insect behaviour. In Proceedings of the 1st ACM/SIGEVO Summit on Genetic and Evolutionary Computation, GEC'09; ACM, Shanghai, China, 12–14 June 2009.
270. Marinakis, Y.; Marinaki, M.; Matsatsinis, N. A Bumble Bees Mating Optimization Algorithm for Global Unconstrained Optimization Problems. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Granada, Spain, 12–15 May 2010*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 305–318.
271. Shnerb, N.M.; Louzoun, Y.; Bettelheim, E.; Solomon, S. The importance of being discrete: Life always wins on the surface. *Proc. Natl. Acad. Sci. USA* **2000**, *97*, 10322–10324. [CrossRef] [PubMed]
272. Dorigo, M.; Maniezzo, V.; Colorni, A. *The Ant System: An Autocatalytic Optimizing Process*; Politecnico di Milano: Milan, Italy, 1991; pp. 1–21.
273. Zungeru, A.M.; Ang, L.-M.; Seng, K.P. Termite-hill: Performance optimized swarm intelligence based routing algorithm for wireless sensor networks. *J. Netw. Comput. Appl.* **2012**, *35*, 1901–1917. [CrossRef]
274. Das, K.N.; Singh, T.K. Drosophila Food-Search Optimization. *Appl. Math. Comput.* **2014**, *231*, 566–580. [CrossRef]
275. Abidin, Z.Z.; Arshad, M.R.; Ngah, U.K. A Simulation Based Fly Optimization Algorithm for Swarms of Mini Autonomous Surface Vehicles Application. Available online: http://nopr.niscair.res.in/handle/123456789/11731 (accessed on 3 January 2021).
276. Pan, W.-T. A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example. *Knowl. Based Syst.* **2012**, *26*, 69–74. [CrossRef]
277. Feng, X.; Lau, F.C.M.; Gao, D. *A New Bio-Inspired Approach to the Traveling Salesman Problem*; Springer: Berlin/Heidelberg, Germany, 2009.
278. Feng, X.; Lau, F.C.; Yu, H. A novel bio-inspired approach based on the behavior of mosquitoes. *Inf. Sci.* **2013**, *233*, 87–108. [CrossRef]
279. Wang, G.-G.; Deb, S.; Cui, Z. Monarch butterfly optimization. *Neural Computing and Applications* **2015**, *31*, 1995–2014. [CrossRef]
280. Bhattacharjee, K.K.; Sarmah, S.P. Monarch Migration Algorithm for optimization problems. In Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management;IEEE, Bali, Indonesia, 4–7 December 2016.
281. Kumar, A.; Misra, R.K.; Singh, D. Butterfly optimizer. In Proceedings of the 2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions, WCI 2015; IEEE, Kanpur, India, 14–17 December 2015.
282. Qi, X.; Zhu, Y.; Zhang, H. A new meta-heuristic butterfly-inspired algorithm. *J. Comput. Sci.* **2017**, *23*, 226–239. [CrossRef]
283. Arora, S.; Singh, S. Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* **2018**, *23*, 715–734. [CrossRef]
284. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl. Based Syst.* **2015**, *89*, 228–249. [CrossRef]
285. Mei, R.N.S.; Sulaiman, M.H.; Mustaffa, Z.; Daniyal, H. Optimal reactive power dispatch solution by loss minimization using moth-flame optimization technique. *Appl. Soft Comput.* **2017**, *59*, 210–222.
286. Mohamed, A.-A.A.; Mohamed, Y.S.; El-Gaafary, A.A.; Hemeida, A.M. Optimal power flow using moth swarm algorithm. *Electr. Power Syst. Res.* **2017**, *142*, 190–206. [CrossRef]
287. Wang, G.-G. Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Comput.* **2016**, *10*, 151–164. [CrossRef]
288. Chen, S. Locust Swarms-A new multi-optima search technique. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation. IEEE, Trondheim, Norway, 18–21 May 2009.
289. Cuevas, E.; Gonzalez, A.; Zaldívar, D.; Perez-Cisneros, M.A. An optimisation algorithm based on the behaviour of locust swarms. *Int. J. Bio-Inspired Comput.* **2015**, *7*, 402. [CrossRef]
290. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper Optimisation Algorithm: Theory and application. *Adv. Eng. Softw.* **2017**, *105*, 30–47. [CrossRef]
291. Mirjalili, S. The Ant Lion Optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98. [CrossRef]

292. Havens, T.C.; Spain, C.J.; Salmon, N.G.; Keller, J.M. Roach infestation optimization. In Proceedings of the 2008 IEEE Swarm Intelligence Symposium, St. Louis, MO, USA, 21–23 September 2008.

293. ZhaoHui, C.; HaiYan, T. Cockroach swarm optimization. In Proceedings of the 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, China, 16–19 April 2010.

294. Wu, S.-J.; Wu, C.-T. A bio-inspired optimization for inferring interactive networks: Cockroach swarm evolution. *Expert Syst. Appl.* **2015**, *42*, 3253–3267. [CrossRef]

295. Kallioras, N.A.; Lagaros, N.D.; Avtzis, D.N. Pity beetle algorithm – A new metaheuristic inspired by the behavior of bark beetles. *Adv. Eng. Softw.* **2018**, *121*, 147–166. [CrossRef]

296. Mirjalili, S.; Jangir, P.; Saremi, S. Multi-objective ant lion optimizer: A multi-objective optimization algorithm for solving engineering problems. *Appl. Intell.* **2016**, *46*, 79–95. [CrossRef]

297. Emary, E.; Zawbaa, H.M.; Hassanien, A.E. Binary ant lion approaches for feature selection. *Neurocomputing* **2016**, *213*, 54–65. [CrossRef]

298. Emary, E.; Zawbaa, H.M. Impact of Chaos Functions on Modern Swarm Optimizers. *PLoS ONE* **2016**, *11*, e0158738. [CrossRef] [PubMed]

299. Heidari, A.A.; Faris, H.; Mirjalili, S.; Aljarah, I.; Mafarja, M. Ant Lion optimizer: Theory, literature review, and application in multi-layer perceptron neural networks. In *Nature-Inspired Optimizers*; Springer: Cham, Switzerland, 2020; pp. 23–46.

300. Raju, M.; Saikia, L.C.; Sinha, N. Automatic generation control of a multi-area system using ant lion optimizer algorithm based PID plus second order derivative controller. *Int. J. Electr. Power Energy Syst.* **2016**, *80*, 52–63. [CrossRef]

301. Marinakis, Y.; Marinaki, M. Combinatorial neighborhood topology bumble bees mating optimization for the vehicle routing problem with stochastic demands. *Soft Comput.* **2014**, *19*, 353–373. [CrossRef]

302. Marinakis, Y.; Marinaki, M.; Migdalas, A. An Adaptive Bumble Bees Mating Optimization algorithm. *Appl. Soft Comput.* **2017**, *55*, 13–30. [CrossRef]

303. Marinaki, M.; Marinakis, Y. A bumble bees mating optimization algorithm for the feature selection problem. *Int. J. Mach. Learn. Cybern.* **2014**, *7*, 519–538. [CrossRef]

304. Kumar, A.; Maini, T.; Misra, R.K.; Singh, D. *Butterfly Constrained Optimizer for Constrained Optimization Problems*; Springer: Singapore, 2019.

305. Kumar, A.; Misra, R.K.; Singh, D. Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, Donostia, Spain, 5–8 June 2017.

306. Sharma, S.; Saha, A.K. m-MBOA: A novel butterfly optimization algorithm enhanced with mutualism scheme. *Soft Comput.* **2019**, *24*, 4809–4827. [CrossRef]

307. Li, G.; Shuang, F.; Zhao, P.; Le, C. An Improved Butterfly Optimization Algorithm for Engineering Design Problems Using the Cross-Entropy Method. *Symmetry* **2019**, *11*, 1049. [CrossRef]

308. Arora, S.; Anand, P. Learning automata-based butterfly optimization algorithm for engineering design problems. *Int. J. Comput. Mater. Sci. Eng.* **2018**, *7*, 1850021. [CrossRef]

309. Arora, S.; Anand, P. Binary butterfly optimization approaches for feature selection. *Expert Syst. Appl.* **2018**, *116*, 147–160. [CrossRef]

310. Aygül, K.; Cikan, M.; Demirdelen, T.; Tumay, M. Butterfly optimization algorithm based maximum power point tracking of photovoltaic systems under partial shading condition. *Energy Sources Part A Recovery Util. Environ. Eff.* **2019**, 1–19. [CrossRef]

311. Wang, Y.-J.; Sun, P. One-Way Pioneer Guide Pity Beetle Algorithm: A New Evolutionary Algorithm for Solving Global Optimization Problems. *IEEE Access* **2020**, *8*, 203270–203293. [CrossRef]

312. Priya, M.M.M.A.; Jawhar, D.S.J.; Geisa, D.J.M. Optimal Deep Belief Network with Opposition based Pity Beetle Algorithm for Lung Cancer Classification: A DBNOPBA Approach. *Comput. Methods Programs Biomed.* **2021**, *199*, 105902. [CrossRef]

313. KS, S.R.; Murugan, S. Memory based Hybrid Dragonfly Algorithm for numerical optimization problems. *Expert Syst. Appl.* **2017**, *83*, 63–78.

314. Sayed, G.I.; Tharwat, A.; Hassanien, A.E. Chaotic dragonfly algorithm: An improved metaheuristic algorithm for feature selection. *Appl. Intell.* **2018**, *49*, 188–205. [CrossRef]

315. Hariharan, M.; Sindhu, R.; Vijean, V.; Yazid, H.; Nadarajaw, T.; Yaacob, S.; Polat, K. Improved binary dragonfly optimization algorithm and wavelet packet based non-linear features for infant cry classification. *Comput. Methods Programs Biomed.* **2018**, *155*, 39–51. [CrossRef] [PubMed]

316. Mafarja, M.; Heidari, A.A.; Faris, H.; Mirjalili, S.; Aljarah, I. Dragonfly algorithm: Theory, literature review, and application in feature selection. In *Nature-Inspired Optimizers*; Springer: Cham, Switzerland, 2020; pp. 47–67.

317. El-Hay, E.A.; El-Hameed, M.A.; El-Fergany, A.A. Improved performance of PEM fuel cells stack feeding switched reluctance motor using multi-objective dragonfly optimizer. *Neural Comput. Appl.* **2018**, *31*, 6909–6924. [CrossRef]

318. Das, K.N.; Singh, T.K.; Baishnab, K.L. Parameter Optimization of Winner-Take-All Circuit for Attention Shift Using Drosophila Food-Search Optimization Algorithm. In *Proceedings of Fourth International Conference on Soft Computing for Problem Solving*; Springer: New Delhi, India, 2015. [CrossRef]

319. Fister, I.J.; Perc, M.; Kamal, S.M. A review of chaos-based firefly algorithms: Perspectives and research challenges. *Appl. Math. Comput.* **2015**, *252*, 155–165. [CrossRef]

320. Sahu, R.K.; Panda, S.; Pradhan, P.C. Design and analysis of hybrid firefly algorithm-pattern search based fuzzy PID controller for LFC of multi area power systems. *Int. J. Electr. Power Energy Syst.* **2015**, *69*, 200–212. [CrossRef]

321. Tahershamsi, A.; Kaveh, A.; Sheikholeslami, R.; Kazemzadeh Azad, S. An improved firefly algorithm with harmony search scheme for optimization of water distribution systems. *Sci. Iran.* **2014**, *21*, 1591–1607.

322. George, G.; Parthiban, L. Multi objective hybridized firefly algorithm with group search optimization for data clustering. In Proceedings of the 2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks, Kolkata, India, 20–22 November 2015.

323. Abd-Elazim, S.; Ali, E.S. Load frequency controller design of a two-area system composing of PV grid and thermal generator via firefly algorithm. *Neural Comput. Appl.* **2016**, *30*, 607–616. [CrossRef]

324. Dey, N.; Samanta, S.; Chakraborty, S.; Das, A.; Chaudhuri, S.S.; Suri, J.S. Firefly Algorithm for Optimization of Scaling Factors During Embedding of Manifold Medical Information: An Application in Ophthalmology Imaging. *J. Med Imaging Heal. Inform.* **2014**, *4*, 384–394. [CrossRef]

325. Sayadi, M.K.; Hafezalkotob, A.; Naini, S.G.J. Firefly-inspired algorithm for discrete optimization problems: An application to manufacturing cell formation. *J. Manuf. Syst.* **2013**, *32*, 78–84. [CrossRef]

326. Chen, H.; Li, S.; Heidari, A.A.; Wang, P.; Li, J.; Yang, Y.; Wang, M.; Huang, C. Efficient multi-population outpost fruit fly-driven optimizers: Framework and advances in support vector machines. *Expert Syst. Appl.* **2019**, *142*, 112999. [CrossRef]

327. Yuan, X.; Dai, X.; Zhao, J.; He, Q. On a novel multi-swarm fruit fly optimization algorithm and its application. *Appl. Math. Comput.* **2014**, *233*, 260–271. [CrossRef]

328. Ding, G.; Dong, F.; Zou, H. Fruit fly optimization algorithm based on a hybrid adaptive-cooperative learning and its application in multilevel image thresholding. *Appl. Soft Comput.* **2019**, *84*, 105704. [CrossRef]

329. Wang, L.; Shi, Y.; Liu, S. An improved fruit fly optimization algorithm and its application to joint replenishment problems. *Expert Syst. Appl.* **2015**, *42*, 4310–4323. [CrossRef]

330. Wu, L.; Zuo, C.; Zhang, H. A cloud model based fruit fly optimization algorithm. *Knowl. Based Syst.* **2015**, *89*, 603–617. [CrossRef]

331. Darwish, S.M.; Elmasry, A.; Ibrahim, S.H. Optimal Shortest Path in Mobile Ad-Hoc Network Based on Fruit Fly Optimization Algorithm. In Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications, Cairo, Egypt, 28–30 March 2019.

332. Mafarja, M.; Aljarah, I.; Faris, H.; Hammouri, A.I.; Al-Zoubi, A.M.; Mirjalili, S. Binary grasshopper optimisation algorithm approaches for feature selection problems. *Expert Syst. Appl.* **2018**, *117*, 267–286. [CrossRef]

333. Mirjalili, S.Z.; Mirjalili, S.; Saremi, S.; Faris, H.; Aljarah, I. Grasshopper optimization algorithm for multi-objective optimization problems. *Appl. Intell.* **2018**, *48*, 805–820. [CrossRef]

334. Arora, S.; Anand, P. Chaotic grasshopper optimization algorithm for global optimization. *Neural Comput. Appl.* **2018**, *31*, 4385–4405. [CrossRef]

335. Luo, J.; Chen, H.; Zhang, Q.; Xu, Y.; Huang, H.; Zhao, X. An improved grasshopper optimization algorithm with application to financial stress prediction. *Appl. Math. Model.* **2018**, *64*, 654–668. [CrossRef]

336. Aljarah, I.; Al-Zoubi, A.M.; Faris, H.; Hassonah, M.A.; Mirjalili, S.; Saadeh, H. Simultaneous Feature Selection and Support Vector Machine Optimization Using the Grasshopper Optimization Algorithm. *Cogn. Comput.* **2018**, *10*, 478–495. [CrossRef]

337. Shi, Y.; Li, Y.; Fan, J.; Wang, T.; Yin, T. A Novel Network Architecture of Decision-Making for Self-Driving Vehicles Based on Long Short-Term Memory and Grasshopper Optimization Algorithm. *IEEE Access* **2020**, *8*, 155429–155440. [CrossRef]

338. Cui, L.; Deng, J.; Wang, L.; Xu, M.; Zhang, Y. A novel locust swarm algorithm for the joint replenishment problem considering multiple discounts simultaneously. *Knowl. Based Syst.* **2016**, *111*, 51–62. [CrossRef]

339. Cuevas, E.; Zaldívar, D.; Perez-Cisneros, M. Automatic Segmentation by Using an Algorithm Based on the Behavior of Locust Swarms. In *Applications of Evolutionary Computation in Image Processing and Pattern Recognition*; Springer International Publishing: Cham, Switzerland, 2016; pp. 229–269.

340. Cuevas, E.; Gonzalez, A.; Fausto, F.; Zaldívar, D.; Perez-Cisneros, M.A. Multithreshold Segmentation by Using an Algorithm Based on the Behavior of Locust Swarms. *Math. Probl. Eng.* **2015**, *2015*, 1–25. [CrossRef]

341. Zervoudakis, K.; Tsafarakis, S. A mayfly optimization algorithm. *Comput. Ind. Eng.* **2020**, *145*, 106559. [CrossRef]

342. Bhattacharyya, T.; Chatterjee, B.; Singh, P.K.; Yoon, J.H.; Geem, Z.W.; Sarkar, R. Mayfly in Harmony: A New Hybrid Meta-Heuristic Feature Selection Algorithm. *IEEE Access* **2020**, *8*, 195929–195945. [CrossRef]

343. Ramasamy, K.; Ravichandran, C.S. Optimal design of renewable sources of PV /wind/ FC generation for power system reliability and cost using MA-RBFNN approach. *Int. J. Energy Res.* **2021**, *45*, 10946–10962. [CrossRef]

344. Yazdani, S.; Hadavandi, E. LMBO-DE: A linearized monarch butterfly optimization algorithm improved with differential evolution. *Soft Comput.* **2018**, *23*, 8029–8043. [CrossRef]

345. Wang, G.-G.; Deb, S.; Zhao, X.; Cui, Z. A new monarch butterfly optimization with an improved crossover operator. *Oper. Res.* **2016**, *18*, 731–755. [CrossRef]

346. Wang, G.-G.; Zhao, X.; Deb, S. A Novel Monarch Butterfly Optimization with Greedy Strategy and Self-Adaptive. *2015 Second. Int. Conf. Soft Comput. Mach. Intell.* **2015**, *2015*, 45–50.

347. Feng, Y.; Yang, J.; Wu, C.; Lu, M.; Zhao, X.-J. Solving 0–1 knapsack problems by chaotic monarch butterfly optimization algorithm with Gaussian mutation. *Memetic Comput.* **2016**, *10*, 135–150. [CrossRef]

348. Devikanniga, D.; Raj, R.J.S. Classification of osteoporosis by artificial neural network based on monarch butterfly optimisation algorithm. *Heal. Technol. Lett.* **2018**, *5*, 70–75. [CrossRef]

349. Chen, S.; Chen, R.; Gao, J. A Monarch Butterfly Optimization for the Dynamic Vehicle Routing Problem. *Algorithms* **2017**, *10*, 107. [CrossRef]

350. Zhu, Y.; Feng, X.; Yu, H. *Mosquito Host-Seeking Algorithm Based on Random Walk and Game of Life*; Springer International Publishing: Cham, Switzerland, 2018.

351. Xu, Y.; Chen, H.; Heidari, A.A.; Luo, J.; Zhang, Q.; Zhao, X.; Li, C. An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks. *Expert Syst. Appl.* **2019**, *129*, 135–155. [CrossRef]

352. Savsani, V.; Tawhid, M.A. Non-dominated sorting moth flame optimization (NS-MFO) for multi-objective problems. *Eng. Appl. Artif. Intell.* **2017**, *63*, 20–32. [CrossRef]

353. Wang, M.; Chen, H.; Yang, B.; Zhao, X.; Hu, L.; Cai, Z.; Huang, H.; Tong, C. Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. *Neurocomputing* **2017**, *267*, 69–84. [CrossRef]

354. Wu, Z.; Shen, D.; Shang, M.; Qi, S. Parameter Identification of Single-Phase Inverter Based on Improved Moth Flame Optimization Algorithm. *Electr. Power Components Syst.* **2019**, *47*, 456–469. [CrossRef]

355. Li, Z.; Zhou, Y.; Zhang, S.; Song, J. Lévy-Flight Moth-Flame Algorithm for Function Optimization and Engineering Design Problems. *Math. Probl. Eng.* **2016**, *2016*, 1–22. [CrossRef]

356. Mehne, S.H.H.; Mirjalili, S. Moth-Flame Optimization Algorithm: Theory, Literature Review, and Application in Optimal Nonlinear. *Nat. Inspired Optim. Theor. Lit. Rev. Appl.* **2020**, *810*, 143.

357. Luo, Q.; Yang, X.; Zhou, Y. Nature-inspired approach: An enhanced moth swarm algorithm for global optimization. *Math. Comput. Simul.* **2018**, *159*, 57–92. [CrossRef]

358. Shilaja, C.; Arunprasath, T. Optimal power flow using Moth Swarm Algorithm with Gravitational Search Algorithm considering wind power. *Future Gener. Comput. Syst.* **2019**, *98*, 708–715.

359. Duman, S. A Modified Moth Swarm Algorithm Based on an Arithmetic Crossover for Constrained Optimization and Optimal Power Flow Problems. *IEEE Access* **2018**, *6*, 45394–45416. [CrossRef]

360. Guvenc, U.; Duman, S.; Hınıslıoglu, Y. Chaotic moth swarm algorithm. In Proceedings of the 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA), IEEE, Gdynia, Poland, 3–5 July 2017.

361. Zhou, Y.; Yang, X.; Ling, Y.; Zhang, J. Meta-heuristic moth swarm algorithm for multilevel thresholding image segmentation. *Multimedia Tools Appl.* **2018**, *77*, 23699–23727. [CrossRef]

362. Fathy, A.; Elaziz, M.A.; Sayed, E.; Olabi, A.; Rezk, H. Optimal parameter identification of triple-junction photovoltaic panel based on enhanced moth search algorithm. *Energy* **2019**, *188*, 116025. [CrossRef]

363. Feng, Y.-H.; Wang, G.-G. Binary moth search algorithm for discounted {0-1} knapsack problem. *IEEE Access* **2018**, *6*, 10708–10719. [CrossRef]

364. Strumberger, I.; Bacanin, N. Modified Moth Search Algorithm for Global Optimization Problems. *Int. J. Comput.* **2018**, *3*, 44–48.

365. Strumberger, I.; Tuba, E.; Bacanin, N.; Beko, M.; Tuba, M. Hybridized moth search algorithm for constrained optimization problems. In Proceedings of the 2018 International Young Engineers Forum (YEF-ECE), IEEE, Costa da Caparica, Portugal, 4 May 2018.

366. Strumberger, I.; Sarac, M.; Markovic, D.; Bacanin, N. Moth Search Algorithm for Drone Placement Problem. *Int. J. Comput.* **2018**, *3*, 75–80.

367. Tsai, H.-C. Roach infestation optimization with friendship centers. *Eng. Appl. Artif. Intell.* **2015**, *39*, 109–119. [CrossRef]

368. Obagbuwa, I.C.; Adewumi, A.O. A modified roach infestation optimization. In Proceedings of the 2014 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, Honolulu, HI, USA, 21–24 May 2014.

369. Obagbuwa, I.C.; Adewumi, A.O.; Adebiyi, A.A. A dynamic step-size adaptation roach infestation optimization. In Proceedings of the 2014 IEEE International Advance Computing Conference, Gurgaon, India, 21–22 February 2014.

370. Kaveh, A.; Eslamlou, A.D. Water strider algorithm: A new metaheuristic and applications. *Structures* **2020**, *25*, 520–541. [CrossRef]

371. Kaveh, A.; Amirsoleimani, P.; Eslamlou, A.D.; Rahmani, P. Frequency-constrained optimization of large-scale dome-shaped trusses using chaotic water strider algorithm. *Struct.* **2021**, *32*, 1604–1618. [CrossRef]

372. Xu, Y.-P.; Ouyang, P.; Xing, S.-M.; Qi, L.-Y.; Khayatnezhad, M.; Jafari, H. Optimal structure design of a PV/FC HRES using amended Water Strider Algorithm. *Energy Rep.* **2021**, *7*, 2057–2067. [CrossRef]

373. Kaveh, A. Water Strider Optimization Algorithm and Its Enhancement. In *Advances in Metaheuristic Algorithms for Optimal Design of Structures*; Kaveh, A., Ed.; Springer International Publishing: Cham, Switzerland, 2021; pp. 783–848.

374. Luque-Chang, A.; Cuevas, E.; Fausto, F.; Zaldívar, D.; Pérez, M. Social Spider Optimization Algorithm: Modifications, Applications, and Perspectives. *Math. Probl. Eng.* **2018**, *2018*, 1–29. [CrossRef]

375. Yu, J.J.; Li, V.O. A social spider algorithm for global optimization. *Appl. Soft Comput.* **2015**, *30*, 614–627. [CrossRef]

376. James, J.; Li, V.O. Parameter sensitivity analysis of social spider algorithm. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC), IEEE, Sendai, Japan, 25–28 May 2015.

377. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4831–4845. [CrossRef]

378. Wang, G.-G.; Gandomi, A.H.; Alavi, A.H.; Gong, D. A comprehensive review of krill herd algorithm: Variants, hybrids and applications. *Artif. Intell. Rev.* **2017**, *51*, 119–148. [CrossRef]

379. Wang, G.G.; Deb, S.; Coelho, L.D.S. Earthworm optimization algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Int. J. Bio-Inspired Comput.* **2015**, *1*, 1. [CrossRef]

380. Kaur, S.; Awasthi, L.K.; Sangal, A.; Dhiman, G. Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103541. [CrossRef]

381. Javaid, N.; Ullah, I.; Zarin, S.S.; Kamal, M.; Omoniwa, B.; Mateen, A. *Differential-Evolution-Earthworm Hybrid Meta-heuristic Optimization Technique for Home Energy Management System in Smart Grid*; Springer International Publishing: Cham, Switzerland, 2019.

382. Faraz, S.H.; Ur Rehman, S.; Sarwar, M.A.; Ali, I.; Farooqi, M.; Javaid, N. *Comparison of BFA and EWA in Home Energy Management System Using RTP*; Springer International Publishing: Cham, Switzerland, 2018.

383. Ali, M.; Abid, S.; Ghafar, A.; Ayub, N.; Arshad, H.; Khan, S.; Javaid, N. *Earth Worm Optimization for Home Energy Management System in Smart Grid*; Springer International Publishing: Cham, Switzerland, 2018.

384. Wang, H.; Yi, J.-H. An improved optimization method based on krill herd and artificial bee colony with information exchange. *Memetic Comput.* **2017**, *10*, 177–198. [CrossRef]

385. Chansombat, S.; Musikapun, P.; Pongcharoen, P.; Hicks, C. A Hybrid Discrete Bat Algorithm with Krill Herd-based advanced planning and scheduling tool for the capital goods industry. *Int. J. Prod. Res.* **2018**, *57*, 6705–6726. [CrossRef]

386. Abdel-Basset, M.; Wang, G.-G.; Sangaiah, A.K.; Rushdy, E. Krill herd algorithm based on cuckoo search for solving engineering optimization problems. *Multimedia Tools Appl.* **2017**, *78*, 3861–3884. [CrossRef]

387. Abualigah, L.M.; Khader, A.T.; Hanandeh, E.S. Hybrid clustering analysis using improved krill herd algorithm. *Appl. Intell.* **2018**, *48*, 4047–4071. [CrossRef]

388. Asteris, P.G.; Nozhati, S.; Nikoo, M.; Cavaleri, L.; Nikoo, M. Krill herd algorithm-based neural network in structural seismic reliability evaluation. *Mech. Adv. Mater. Struct.* **2018**, *26*, 1146–1153. [CrossRef]

389. Das, S.R.; Kuhoo; Mishra, D.; Rout, M. An optimized feature reduction based currency forecasting model exploring the online sequential extreme learning machine and krill herd strategies. *Phys. A Stat. Mech. its Appl.* **2018**, *513*, 339–370. [CrossRef]

390. Nguyen, T.T.; Vo, D.N. Improved social spider optimization algorithm for optimal reactive power dispatch problem with different objectives. *Neural Comput. Appl.* **2019**, *32*, 5919–5950. [CrossRef]

391. El Aziz, M.A.; Hassanien, A.E. An improved social spider optimization algorithm based on rough sets for solving minimum number attribute reduction problem. *Neural Comput. Appl.* **2017**, *30*, 2441–2452. [CrossRef]

392. Cuevas, E.; Cienfuegos, M. A new algorithm inspired in the behavior of the social-spider for constrained optimization. *Expert Syst. Appl.* **2014**, *41*, 412–425. [CrossRef]

393. Mirjalili, S.Z.; Saremi, S.; Mirjalili, S.M. Designing evolutionary feedforward neural networks using social spider optimization algorithm. *Neural Comput. Appl.* **2015**, *26*, 1919–1928. [CrossRef]

394. Zhou, G.; Zhou, Y.; Zhao, R. Hybrid social spider optimization algorithm with differential mutation operator for the job-shop scheduling problem. *J. Ind. Manag. Optim.* **2021**, *17*, 533–548. [CrossRef]

395. Xavier, V.M.A.; Annadurai, S. Chaotic social spider algorithm for load balance aware task scheduling in cloud computing. *Clust. Comput.* **2018**, *22*, 287–297.

396. Elsayed, W.; Hegazy, Y.; Bendary, F.; El-Bages, M. Modified social spider algorithm for solving the economic dispatch problem. *Eng. Sci. Technol. Int. J.* **2016**, *19*, 1672–1681. [CrossRef]

397. Sung, H.-K.; Jung, N.-G.; Huang, S.-R.; Kim, J.-M. Application of Social Spider Algorithm to Optimize Train Energy. *J. Electr. Eng. Technol.* **2019**, *14*, 519–526. [CrossRef]

398. Yu, J.J.; Li, V.O. A social spider algorithm for solving the non-convex economic load dispatch problem. *Neurocomputing* **2016**, *171*, 955–965. [CrossRef]

399. Houssein, E.H.; Helmy, B.E.-D.; Elngar, A.A.; Abdelminaam, D.S.; Shaban, H. An Improved Tunicate Swarm Algorithm for Global Optimization and Image Segmentation. *IEEE Access* **2021**, *9*, 56066–56092. [CrossRef]

400. Fetouh, T.; Elsayed, A.M. Optimal Control and Operation of Fully Automated Distribution Networks Using Improved Tunicate Swarm Intelligent Algorithm. *IEEE Access* **2020**, *8*, 129689–129708. [CrossRef]

401. Chelliah, J.; Kader, N. Optimization for connectivity and coverage issue in target-based wireless sensor networks using an effective multiobjective hybrid tunicate and salp swarm optimizer. *Int. J. Commun. Syst.* **2020**, *34*, e4679.

402. Črepinšek, M.; Liu, S.H.; Mernik, M. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.* **2013**, *43*, 1–33. [CrossRef]

403. Cuevas, E.; Echavarría, A.; Zaldívar, D.; Pérez-Cisneros, M. A novel evolutionary algorithm inspired by the states of matter for template matching. *Expert Syst. Appl.* **2013**, *40*, 6359–6373. [CrossRef]

404. Corriveau, G.; Guilbault, R.; Tahan, A.; Sabourin, R. Review of phenotypic diversity formulations for diagnostic tool. *Appl. Soft Comput.* **2013**, *13*, 9–26. [CrossRef]

405. Shir, O.M. Niching in Evolutionary Algorithms. In *Handbook of Natural Computing*; Rozenberg, G., Bäck, T., Kok, J.N., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 1035–1069.

406. Preuss, M. Niching prospects. In *International Conference on Bioinspired Optimization Methods and Their Applications*; Filipic, B., Silic, J., Eds.; Josef Stefan Institute: Lublijana, Slovenia, 2006.

407. Silberholz, J.; Golden, B. Comparison of metaheuristics. In *Handbook of Metaheuristics*; Springer: Boston, MA, USA, 2010; pp. 625–640.

408. Rice, J.R. The Algorithm Selection Problem. *Adv. Comput.* **1976**, *15*, 65–118.

409. Misir, M.; Sebag, M. Alors: An algorithm recommender system. *Artif. Intell.* **2017**, *244*, 291–314. [CrossRef]

410. Bischl, B.; Kerschke, P.; Kotthoff, L.; Lindauer, M.; Malitsky, Y.; Fréchette, A.; Hoos, H.; Hutter, F.; Leyton-Brown, K.; Tierney, K.; et al. ASlib: A benchmark library for algorithm selection. *Artif. Intell.* **2016**, *237*, 41–58. [CrossRef]

411. LaTorre, A.; Molina, D.; Osaba, E.; Del Ser, J.; Herrera, F. Fairness in bio-inspired optimization research: A prescription of methodological guidelines for comparing meta-heuristics. *arXiv* **2020**, arXiv:2004.09969.

412. Naruei, I.; Keynia, F. A new optimization method based on COOT bird natural life model. *Expert Syst. Appl.* **2021**, *183*, 115352. [CrossRef]

413. Ewees, A.A.; Elaziz, M.A.; Houssein, E.H. Improved grasshopper optimization algorithm using opposition-based learning. *Expert Syst. Appl.* **2018**, *112*, 156–172. [CrossRef]

414. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Futur. Gener. Comput. Syst.* **2020**, *111*, 300–323. [CrossRef]

415. Sharma, T.K.; Abraham, A. Artificial bee colony with enhanced food locations for solving mechanical engineering design problems. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *11*, 267–290. [CrossRef]

416. Yang, X.-S.; Hosseini, S.S.S.; Gandomi, A. Firefly Algorithm for solving non-convex economic dispatch problems with valve loading effect. *Appl. Soft Comput.* **2012**, *12*, 1180–1186. [CrossRef]

417. Gupta, S.; Deep, K.; Moayedi, H.; Foong, L.K.; Assad, A. Sine cosine grey wolf optimizer to solve engineering design problems. *Eng. Comput.* **2021**, *37*, 3123–3149. [CrossRef]

418. Huang, F.-Z.; Wang, L.; He, Q. An effective co-evolutionary differential evolution for constrained optimization. *Appl. Math. Comput.* **2007**, *186*, 340–356. [CrossRef]

419. Mezura-Montes, E.; Hernández-Ocana, B. Bacterial foraging for engineering design problems: Preliminary results. In *Memorias del 4o Congreso Nacional de Computación Evolutiva (COMCEV'2008)*; Centro de Investigación en Matemáticas: Guanajuato, México, 2008.

420. He, Q.; Wang, L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Eng. Appl. Artif. Intell.* **2007**, *20*, 89–99. [CrossRef]

421. Bernardino, H.S.; Barbosa, H.J.; Lemonge, A.C.; Fonseca, L.G. A new hybrid AIS-GA for constrained optimization problems in mechanical engineering. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*; IEEE: Hong Kong, China, 2008.

422. Lee, K.S.; Geem, Z.W. A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Comput. Methods Appl. Mech. Eng.* **2005**, *194*, 3902–3933. [CrossRef]

423. Hedar, A.-R.; Fukushima, M. Derivative-Free Filter Simulated Annealing Method for Constrained Continuous Global Optimization. *J. Glob. Optim.* **2006**, *35*, 521–549. [CrossRef]

424. Houssein, E.H.; Saad, M.R.; Hashim, F.; Shaban, H.; Hassaballah, M. Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2020**, *94*, 103731. [CrossRef]

425. Hwang, S.-F.; He, R.-S. A hybrid real-parameter genetic algorithm for function optimization. *Adv. Eng. Informatics* **2006**, *20*, 7–21. [CrossRef]

426. Mahdavi, M.; Fesanghary, M.; Damangir, E. An improved harmony search algorithm for solving optimization problems. *Appl. Math. Comput.* **2007**, *188*, 1567–1579. [CrossRef]

427. Zhang, M.; Luo, W.; Wang, X. Differential evolution with dynamic stochastic selection for constrained optimization. *Inf. Sci.* **2008**, *178*, 3043–3074. [CrossRef]

428. Ragsdell, K.M.; Phillips, D.T. Optimal Design of a Class of Welded Structures Using Geometric Programming. *J. Eng. Ind.* **1976**, *98*, 1021–1025. [CrossRef]

429. Yapici, H.; Cetinkaya, N. A new meta-heuristic optimizer: Pathfinder algorithm. *Appl. Soft Comput.* **2019**, *78*, 545–568. [CrossRef]

430. Belegundu, A.D.; Arora, J.S. A study of mathematical programming methods for structural optimization. Part I: Theory. *Int. J. Numer. Methods Eng.* **1985**, *21*, 1583–1599. [CrossRef]

431. Arora, J.S. *Introduction to Optimum Design*; Elsevier: Cambridge, MA, USA, 2004.

432. Montague, M.; Aslam, J.A. Condorcet fusion for improved retrieval. In Proceedings of the Eleventh International Conference on Information and Knowledge Management, ACM, McLean, VA, USA, 4–9 December 2002.

433. Osaba, E.; Carballedo, R.; Diaz, F.; Onieva, E.; Masegosa, A.D.; Perallos, A. Good practice proposal for the implementation, presentation, and comparison of metaheuristics for solving routing problems. *Neurocomputing* **2018**, *271*, 2–8. [CrossRef]

434. Piotrowski, A.P.; Napiorkowski, M.J.; Napiorkowski, J.J.; Osuch, M.; Kundzewicz, Z.W. Are modern metaheuristics successful in calibrating simple conceptual rainfall–runoff models? *Hydrol. Sci. J.* **2017**, *62*, 606–625. [CrossRef]

435. Piotrowski, A.P.; Napiorkowski, J.J. Some metaheuristics should be simplified. *Inf. Sci.* **2018**, *427*, 32–62. [CrossRef]

436. Tzanetos, A.; Dounias, G. Nature inspired optimization algorithms or simply variations of metaheuristics? *Artif. Intell. Rev.* **2021**, *54*, 1841–1862. [CrossRef]

437. Lones, M.A. Mitigating Metaphors: A Comprehensible Guide to Recent Nature-Inspired Algorithms. *arXiv* **2019**, arXiv:1902.08001. [CrossRef]

438. Van Thieu, N. The State-of-the-art MEta-Heuristics Algorithms in PYthon (MEALPY). 2021. Available online: https://pypi.org/project/mealpy/ (accessed on 2 August 2021).

439. Molina, D.; Latorre, A.; Herrera, F. An Insight into Bio-inspired and Evolutionary Algorithms for Global Optimization: Review, Analysis, and Lessons Learnt over a Decade of Competitions. *Cogn. Comput.* **2018**, *10*, 517–544. [CrossRef]

440. Veček, N.; Črepinšek, M.; Mernik, M. On the influence of the number of algorithms, problems, and independent runs in the comparison of evolutionary algorithms. *Appl. Soft Comput.* **2017**, *54*, 23–45. [CrossRef]

441. Squillero, G.; Tonda, A. Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Inf. Sci.* **2016**, *329*, 782–799. [CrossRef]

442. Liu, H.-L.; Chen, L.; Deb, K.; Goodman, E. Investigating the Effect of Imbalance Between Convergence and Diversity in Evolutionary Multi-objective Algorithms. *IEEE Trans. Evol. Comput.* **2016**, *21*, 408–425. [CrossRef]

443. Wright, J.; Jordanov, I. Convergence properties of quantum evolutionary algorithms on high dimension problems. *Neurocomputing* **2019**, *326–327*, 82–99. [CrossRef]

444. Chen, Y.; He, J. Average Convergence Rate of Evolutionary Algorithms II: Continuous Optimization. *arXiv* **2018**, arXiv:1810.11672.

445. Shirakawa, S.; Nagao, T. Bag of local landscape features for fitness landscape analysis. *Soft Comput.* **2016**, *20*, 3787–3802. [CrossRef]

446. Yang, S.; Li, K.; Li, W.; Chen, W.; Chen, Y. Dynamic Fitness Landscape Analysis on Differential Evolution Algorithm. In *Bio-inspired Computing–Theories and Applications: 11th International Conference, BIC-TA 2016, Xi'an, China, 28–30 October 2016, Revised Selected Papers, Part II*; Gong, M., Pan, L., Song, T., Zhang, G., Eds.; Springer: Singapore, 2016; pp. 179–184.

447. Aleti, A.; Moser, I.; Grunske, L. Analysing the fitness landscape of search-based software testing problems. *Autom. Softw. Eng.* **2016**, *24*, 603–621. [CrossRef]

448. Liang, J.; Li, Y.; Qu, B.; Yu, K.; Hu, Y. *Mutation Strategy Selection Based on Fitness Landscape Analysis: A Preliminary Study*; Springer: Singapore, 2020.

449. Hussain, K.; Salleh, M.N.M.; Cheng, S.; Shi, Y. On the exploration and exploitation in popular swarm-based metaheuristic algorithms. *Neural Comput. Appl.* **2018**, *31*, 7665–7683. [CrossRef]

450. Chen, Y.; He, J. Exploitation and Exploration Analysis of Elitist Evolutionary Algorithms: A Case Study. *arXiv* **2020**, arXiv:2001.10932.

451. Morales-Castañeda, B.; Zaldivar, D.; Cuevas, E.; Fausto, F.; Rodríguez, A. A better balance in metaheuristic algorithms: Does it exist? *Swarm Evol. Comput.* **2020**, *54*, 100671. [CrossRef]