Expert Systems With Applications 59 (2016) 20–32

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Bio inspired computing – A review of algorithms and scope of applications

Arpan Kumar Kar*

Information Systems area, DMS, Indian Institute of Technology Delhi, Hauz Khas, Outer Ring Road, New Delhi 110016 India

ARTICLE INFO

Article history:
Received 6 October 2015
Revised 2 March 2016
Accepted 15 April 2016
Available online 16 April 2016

Keywords:
Bio-inspired computing
Artificial intelligence
Swarm intelligence
Intelligent algorithms
Metaheuristics
Literature review

ABSTRACT

With the explosion of data generation, getting optimal solutions to data driven problems is increasingly becoming a challenge, if not impossible. It is increasingly being recognised that applications of intelligent bio-inspired algorithms are necessary for addressing highly complex problems to provide working solutions in time, especially with dynamic problem definitions, fluctuations in constraints, incomplete or imperfect information and limited computation capacity. More and more such intelligent algorithms are thus being explored for solving different complex problems. While some studies are exploring the application of these algorithms in a novel context, other studies are incrementally improving the algorithm itself. However, the fast growth in the domain makes researchers unaware of the progresses across different approaches and hence awareness across algorithms is increasingly reducing, due to which the literature on bio-inspired computing is skewed towards few algorithms only (like neural networks, genetic algorithms, particle swarm and ant colony optimization). To address this concern, we identify the popularly used algorithms within the domain of bio-inspired algorithms and discuss their principles, developments and scope of application. Specifically, we have discussed the neural networks, genetic algorithm, particle swarm, ant colony optimization, artificial bee colony, bacterial foraging, cuckoo search, firefly, leaping frog, bat algorithm, flower pollination and artificial plant optimization algorithm. Further objectives which could be addressed by these twelve algorithms have also be identified and discussed. This review would pave the path for future studies to choose algorithms based on fitment. We have also identified other bio-inspired algorithms, where there are a lot of scope in theory development and applications, due to the absence of significant literature.

1. Introduction

The domain of bio-inspired computing is gradually getting prominence in the current times. As organizations and societies are gearing towards a digital era, there has been an explosion of data. This explosion of data is making it more and more challenging to extract meaningful information and gather knowledge by using standard algorithms, due to the increasing complexity of analysis. Finding the best solution increasingly becomes very difficult to identify, if not impossible, due to the very large and dynamic scope of solutions and complexity of computations. Often, the optimal solution for such a NP hard problem is a point in the n-dimensional hyperspace and identifying the solution is computationally very expensive or even not feasible in limited time. Therefore intelligent approaches are needed to identify suitable working solutions.

In this context, intelligent meta-heuristics algorithms can learn and provide a suitable working solution to very complex problems. Within meta-heuristics, bio-inspired computing is gradually gaining prominence since these algorithms are intelligent, can learn and adapt like biological organisms. These algorithms are drawing attention from the scientific community due to the increasing complexity of the problems, increasing range of potential solutions in multi-dimensional hyper-planes, dynamic nature of the problems and constraints, and challenges of incomplete, probabilistic and imperfect information for decision making. However, the fast developments in this domain are increasingly getting difficult to track, due to different algorithms which are being introduced very frequently. However, no study has attempted to identify these algorithms exhaustively, explore and compare their potential scope across different problem contexts.

In fact very few researchers are often familiar with the developments in the domain, where more and more new algorithms are gaining acceptance and prominence. Therefore, with limited visibility across algorithms, new researchers working in this domain tend

* Tel.: +919007782107.
E-mail address: arpan_kar@yahoo.co.in

http://dx.doi.org/10.1016/j.eswa.2016.04.018
0957-4174/

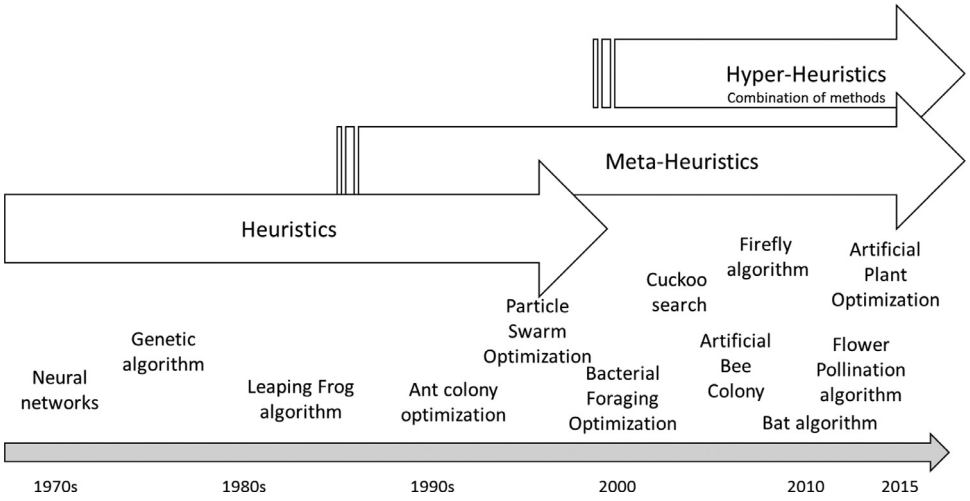

Fig. 1. Development focus of bio-inspired algorithms.

to focus on very limited and popular approaches, and therefore often "force-fit" algorithms rather than exploring the most suitable one, based on the problem statement, due to limited awareness. To address this gap, we review some of the popularly used bio-inspired algorithms as well as introduce the newly developed algorithms which have a huge potential for applications. Further to that, we also explore the potential scope of applications of the algorithms in specific domains, based on published scientific literature. While twelve of the slightly popular algorithms have been discussed, the scope of future research in other bio-inspired algorithms has been discussed. However, in depth discussion about the implementation (e.g. pseudocode, etc.) and enhancements in each algorithm is beyond the scope of the current article. Further, specific detailed citations of each application could not be provided, but we attempt to generalize whenever possible based on other focused reviews. Fig. 1 depicts a brief overview of the development of these meta-heuristics algorithms with the progress of time.

Some reviews of metaheuristics algorithms (Gogna & Tayal, 2013; Yang, 2011b) have been conducted, but these studies have focused mostly only genetic algorithm, ant colony optimization and neural networks as part of bio-inspired algorithms. Also such reviews are conducted in isolation, and do not provide an integrative insight across multiple algorithms and their future scope. The other algorithms these studies have focused on are nature inspired algorithms like tabu search and simulated annealing, but not only on bio-inspired algorithms, and thus have a different scope of discussion. No recent study has attempted to explore and consolidate the developments surrounding these newly developed algorithms within bio-inspired computing. Probably this is due to the recency of development of some of these algorithms, as indicated in Fig. 1. This study therefore provides a lot of insight for scholars who are attempting to explore the domain, and based on their problem formulation, they would be able to select a suitable algorithm for further exploration in real life problems in business organizations, society, and government.

The subsequent sections are subdivided in the following: first we explore the different types of popularly used algorithms. Subsequently we explore the applications of these algorithms in specific context. Then based on the applications and scope of the algorithms, we try to provide insights on the potential applications for future research directions. We do not attempt to explore the detailed algorithms, scope or performance centric issues for the current study.

2. Research methodology

This research was conducted in two phases. In the first phase, the objective was identifying the algorithms itself. In the next phase, after the identification of the algorithms, we attempted to identify studies which had implemented these algorithms, to different problems and domains.

While the classic algorithms like neural networks, genetic algorithm, particle swarm and ant colony optimization are well known and has a lot of literature surrounding their enhancements and applications, a bigger challenge was to identify the more recent developments in bio-inspired computing. Further some algorithms failed to be adopted and used by the scientific community at large, despite having made a strong novel contribution long ago (e.g. wasp algorithm, Theraulaz, Goss, Gervet, & Deneubourg, 1991; shark algorithm, Hersovici et al., 1998), as compared to the popularity of other algorithms introduced in the era. Identifying different algorithms itself was a major challenge since many of these algorithms are in a very nascent stage of development and often these have been published as conference proceedings or book chapters. One possible reason for such sources of publication could be the time taken by peer reviewed journals for publishing such research, and the required theoretical rigour in validating approaches. So for identifying the recent developments, we restricted our search in Scopus and Google Scholar directory, using specific keywords like bio-inspired algorithms, heuristics, meta-heuristics, hyper-heuristics and nature inspired algorithms. The objective was to identify recently published conference proceedings, edited books and journal articles in the broad area of bio-inspired computing. From these sources, we were able to identify the recent algorithms which have been developed, and subsequently we proceeded to the next stage. Also some of these publications had references to specific algorithms, which were not available from our keyword specific search. In the next stage we started exploring the literature surrounding each of these algorithms to understand them in greater detail. Further such literature highlighted the benefits and limitations of these algorithms. Also the review of literature enlightened us with the potential scope of applications for some of these algorithms. However, we restricted our search to algorithms which had over fifteen published applications so that the scope can be truly generalizable. Many other algorithms were identified like the amoeba based algorithm (Zhang et al., 2013), bean optimization algorithm (Zhang, Sun, Mei, & Wang, 2010), individual bird based algorithms based on doves (Su, Su, & Zhao, 2009),

and eagles (Yang & Deb, 2010); individual insect based algorithms like fruit fly (Pan, 2012), wasp (Theraulaz et al., 1991), and glowworm (Krishnanand & Ghose, 2005); individual animal based algorithms like monkey (Mucherino & Seref, 2007), shark (Hersovici et al., 1998), wolf (Liu, Yan, Liu, & Wu, 2011) and lion (Yazdani & Jolai, 2015). These algorithms could not be explored for their scope of applications across domains, due to lack of extensive application specific studies. In the subsequent section, we would highlight the different algorithms which were identified by our review and describe them in brief, before exploring their scope of application in different problem domains.

The current study has been adopted using a narrative literature review approach. With the given focus of our current study, meta-analysis or meta-synthesis of literature was out of the scope, since the scope of how these algorithms were used in different studies and domains, leaves little scope of statistical analysis within a common base. This review of literature could have adopted an approach of systematic review while listing searches and numbers of studies which were identified for each algorithm. However, while actually trying to identify the studies, problems were faced in terms of identifying potential algorithms, since a lot of times, the search terms were not able to identify the algorithms, due to wide disparity among keywords and title descriptors. Then in these cases, the algorithms needed to be identified using cross-referencing mechanisms from published literature in the domain of Bio-inspired computing. While the approach adopted was less systematic, the current research methodology ensured greater coverage of studies. One of the limitation of this research methodology is while choosing articles across different databases where indexing rules were different, articles have been included from both academic and practitioner focussed publications. Selection and screening of articles had to be done manually after reading them for their scope, especially the introduction, discussion and conclusion sections. Another limitation is that there is a chance of missing out on identifying recently developed algorithms, especially those on which very few studies have been reported.

3. Review of algorithms

This section is subdivided into independent reviews of multiple algorithms. All of these bio inspired algorithms like neural network, genetic algorithm or swarm intelligence, try to replicate the way biological organisms and sub-organism entities (like neurons and bacteria) operate to achieve high level of efficiency, even if sometimes the actual optimal solution is not achieved. Now it is important to understand that for a single objective optimization problem, the optimal solution can often be a single point in the solution space, while for bi-objective optimization, the Pareto front forms a curve, and for tri-objective problems, it becomes a surface. The complexity of finding a solution thus increases non-linearly, with the increase of dimension for such NP hard problems. This is where heuristics and meta-heuristics contribute. A particular focus in such exploration is in the domain of swarm intelligence and similar algorithms (Kennedy, 1997; Kennedy & Eberhart, 1995). Swarm intelligence focuses on artificially recreating the concept of naturally intelligent decentralized organisms whereby the collective intelligence of the group is more than the sum of individual intelligence. Based on this principle, algorithms such as ant colony optimization, bird flocking, bacteria foraging and fish schooling were identified for providing solutions in complex systems. Further many of these algorithms are efficient in providing solutions in the multi-objective domain where a set of suitable non-dominated solutions are often usable instead of the actual optimal front, if these solutions could be found with lower computational complexity. Most approaches for solving multi-objective problems convert the objectives into a single objective with some

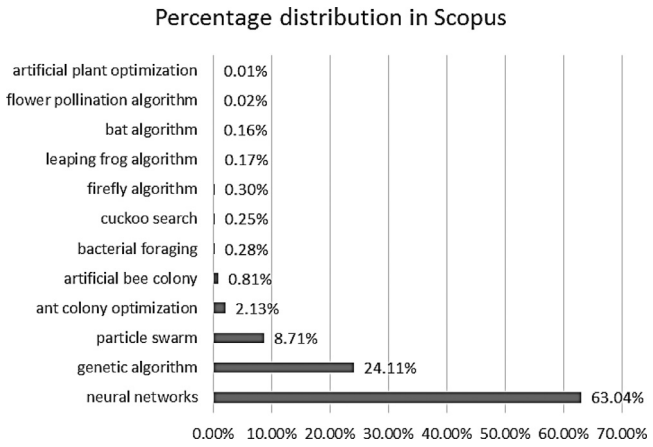

Fig. 2. Search results in Scopus with algorithm names in title.

prioritization method, which affects performance. However, some of these recently developed bio-inspired algorithms truly support multiple objective problems.

The exploration in Scopus highlighted that not the same amount of literature is available in these algorithms. In fact, the usage of these algorithms, have been highly skewed. Fig. 2 highlights the percentage wise results of a search using Scopus, where the algorithm is explicitly mentioned in the article title.

A search of these algorithms in Scopus database highlights the dominant contributors, dominant subject areas and publication volume till February, 2016, for these algorithms, as indicated in Table 1.

While there may be other very important contributions and in other subject areas, this table is reported solely on the basis of the algorithm name being present in the title or subject terms, in Scopus database.

However, despite such advances, scholars have lesser knowledge about the developments across algorithms. Not much literature has focused on providing insights of these algorithms and their scope of applications across other disciplines and subject areas. This is the gap what we try to address by reviewing the literature surrounding bio-inspired algorithms. We highlight the fundamentals and developments in theory within such bio-inspired algorithms in the following sub-sections.

3.1. Neural networks

Neural Networks (Grossberg, 1988) are often defined as adaptive non-linear data processing algorithms that combines multiple processing units connected in a network in different layers. These networks are characterized by being self-adapting, self-organizing and with the ability to learn based on inputs and feedbacks from the ecosystem within which it is operating. The feedback could be positive or negative depending on accuracy of results. These neural networks try to replicate the way the neurons in any intelligent organism (like the human neurons) are coded to take inputs. The network acts like a black box that operates on these inputs and provides outputs. The digression of the output from the desired result is sent back as feedback to improve the processing model of the network.

While there are different approaches for implementing neural networks, probably the simplest implementation is that of a perceptron network. In the perceptron network, there is a feedback to improve upon the output and there is often a single layer that provides the internal operations. Perceptron networks can be used both for linear and non-linear systems (Sadegh, 1993). Further such a network could also have multiple inputs, multiple

Table 1
Scopus search results for the algorithms reviewed.

Algorithms	Dominant contributors	Dominant subject areas	Articles	Conference
Neural networks	Cao, J; Melin, P; Wang, J; Oh, SK; Pedrycz, W.	Engineering, Computer Science, Mathematics, Physics and Astronomy, Materials Science	62,490	53,651
Genetic algorithm	Gen, M; Goldberg, DE; Chakraborti, N; Sakawa, M, Castillo, O.	Engineering, Computer Science, Mathematics, Physics and Astronomy, Materials Science	22,939	22,112
Particle swarm	Sun, J.; Engelbrecht, AP; Xu, W; Abraham, A; Zeng, J.	Computer science, Engineering, Mathematics, Energy, Physics and Astronomy	8386	7907
Ant colony optimization	Blum, C; Zhang, J; Dorigo, M; Stutzle, T; Kaveh, A.	Computer science, Engineering, Mathematics, Decision Sciences, Social Sciences	1858	2114
Artificial bee colony	Karaboga, D; Pant, M; Ozturk, C; Vega-Rodriguez, MA; Akay, B.	Computer science, Engineering, Mathematics, Energy, Decision sciences	866	633
Bacterial foraging	Niu, B; Abraham, A; Chen, H; Zhu, Y; Das, S.	Computer science, Engineering, Mathematics, Energy, Physics and Astronomy	284	237
Cuckoo search	Yang, XS; Deb, S; khan, A; Rehman, MZ; Zhou, Y.	Computer science, Engineering, Mathematics, Energy, Environmental Sciences	292	162
Firefly algorithm	Yang, XS; Shareef, H; Mohamed, A; Tuba, M; Horng, MH.	Computer science, Engineering, Mathematics, Energy, Physics and Astronomy	315	232
Leaping frog algorithm	Hosseinian, SH; Chen, MR; Li, X; Zhao, L; Wang, L.	Engineering, Computer Science, Mathematics, Energy, Social Science	199	117
Bat algorithm	Yang, XS; Zhou, Y; Tsai, PW; Nguyen, TT; Dao, TK	Computer science, Engineering, Mathematics, Energy, Materials science	169	113
Flower pollination algorithm	Abdelaziz, AY; Ali, ES; Yang, XS; Abd Elazim, SM; Dubey, HM	Computer science, Engineering, Mathematics, Energy, Neuroscience	21	15
Artificial plant optimization	Cui, Z; Shi, Z; Zeng, J; Liu, D; Cai, X.	Engineering, Physics and Astronomy, Mathematics, Computer science, Materials science	16	8

layers and multiple outputs (Bounds, Lloyd, Mathew, & Waddell, 1988). Also development in neural networks has seen applications of probabilistic and approximation based algorithms to accommodate imprecise or incomplete information to improve outcome (Hornik, 1991; Specht, 1990). Also, the way information as processed was segregated into linear and non-linear neural networks in the way the individual information processing units (nodes) operated within the network (Grossberg, 1988; Oja, 1992). Neural networks have further been extended for auto associative networks, iterative auto-associative networks, bidirectional associative memory and allied algorithms (Fausett, 1994). Further recent literature (Kar, 2013; Schmidhuber, 2015) highlights how deep, shallow, unsupervised, supervised and reinforcement based learning approaches are used to train the networks and how different levels of network nodes have been introduced and used over the years. It is interesting to note that neural networks can be combined with other algorithms, based on the needs of the problem, to provide improved predicting capabilities to the system (Kar, 2015; Schmidhuber, 2015).

3.2. Genetic algorithm

Genetic algorithm (Holland, 1975) was introduced to mimic the way nature uses computational techniques to obtain suitable working solutions while creating future generations in biological organisms. It is an evolutionary search heuristic that mimics the process of natural selection (Darwin, 1859) and uses nature inspired operators to identify good working solutions. Ever since its conceptualization, it has been significantly used to solve a variety of single and multi-objective problems that are combinatorial and non-deterministic in nature (Aytug, Khouja, & Vergara, 2003; Dimopoulos & Zalzala, 2000).

For using this algorithm, a problem solution is defined in terms of the fitness function where the fitness of the potential solution is an indicator of its suitability. This fitness may be computed from a set of integers, vectors, matrices, linked lists or other data structure based on how the problem is tackled. Fitness could be either a maximization or a minimization function, based on the objective of the problem. For using genetic algorithms, four basic operators were defined in literature (Srinivas & Patnaik, 1994; Colin & Jonathan, 2002; Reeves, 2003), namely inheritance, cross-over, reproduction, and mutation. For using these operators, for each new potential solution to be produced, a pair of pre-optimized solutions is selected. By using the operators like crossover and reproduction, a "child" solution is developed, where the new solution retains many of the positive characteristics of its "parents" while reducing the less useful characteristics. However in the mutation operator, a specific fitness driver may be abruptly changed to enhance the fitness of the child solution significantly. This is often done to avoid local optimality and challenges associated with intermediate levels (Mitchell, Forrest, & Holland, 1992). It is important to note that genetic algorithms often fail to address very complex high dimensional, multi-modal problems where fitness function evaluation becomes computationally very complex or due to very high scale of iterations (Goldberg, 2006; Reeves, 2003). Performance of genetic algorithms, in terms of accuracy and time complexity, reduce significantly in such problem domains.

3.3. Ant colony optimization algorithm

Ant colony optimization (Dorigo & Birattari, 2010; Dorigo, Birattari, & &Stützle, 2006) is a search algorithm, for solving combinatorial optimization problems. It is based on the in-direct communication of simple agents (called ants here) foraging for information, mediated by artificial trails (called pheromones). The trails serve as a distributed numerical information for the agents to construct solutions based on probabilistic search experience. The results are obtained in a reasonably decent amount of search time.

In this algorithm, the solution is often attempted in a sequence of iterative steps (Bououden, Chadli, & Karimi, 2015; Dorigo & Blum, 2005; Ghasab, Khamis, Mohammad, & Fariman, 2015; Hong, Tung, Wang, Wu, & Wu, 2012; Mandloi and Bhatia, 2015). Candidate solutions are identifies from a sample of solutions using a parametric probability distribution. For doing so, a group of ants are selected and a pool of decision variable is defined in the problem. The ants select the design variables for creating the candidate solutions. As the ants explore the candidate solutions, a local updation of the solution is done based on its suitability. These candidate solutions are used to modify the value of the trails based on the local updation in such a way, that the higher quality solutions are selected in the subsequent sampling for candidate solutions by

the same group of ants. The random candidate solutions created in the initial phase therefore paves the path for an optimal solution.

3.4. Particle swarm optimization algorithm

Particle swarm optimization (Shi & Eberhart, 1998, 1999) is based on the collective group behaviour of organisms such as fish schooling, insect swarming or birds flocking, whereby the group attempts to meet the collective objective of the group based on the feedback from the other members. A swarm is a large number of decentralized, homogenous agents that interact locally among themselves and their environment, whereby a global (often optimal) solution is strived to be achieved. Particle swarm optimization is used for problems where the function to be optimized is discontinuous, non-differentiable with too many non-linearly related parameters (Floreano & Mattiussi, 2008). These algorithms operates in a sequence of few iterative steps defined on the behaviour of the organism it emulates (Couceiro & Ghamisi, 2016). Each particle or member in the swarm (say a bird or fish) tries to sense a potential solution at any point of time. It communicates a signal proportional to the suitability of the candidate solution to the other particles in the swarm. Each swarm particle or member can therefore sense the strength of the signal communicated by the other members, and thus the suit-ability of the candidate solution based on a fitness function (Gandomi, Yang, & Alavi, 2013). When a particle or member tries to focus on a more suitable candidate solution from among the locally available candidate solutions, based on different learning mechanisms (Yang et al, 2013; Zhan, Zhang, Li, & Shi, 2011), a new movement direction is identified along with an inertial influence to gradually guide the particles towards an optimal solution where-ever possible. Since real numbers are used to break local optimality of candidate solutions, this algorithm provides a simpler complexity of implementation. Especially noteworthy in such swarm algorithms is the role of Levy flights and Levy walks, whereby the behaviour of such swarm members that move instantaneously between successive sites is captured and used for estimating direction of convergence of candidate solutions (Shlesinger & Klafter, 1986). Further, the extension of chaotic swarm optimization (Hong, 2009; Liu, Wang, Jin, Tang, & Huang, 2005) was introduced to enhance the performance of particle swarm approaches, by introducing adaptive inertia weight factors to efficiently balance the exploration and exploitation abilities.

3.5. Artificial bee colony algorithm

The artificial bee colony algorithm (Gao & Liu, 2012; Karaboga, 2005) is a bio-inspired optimization algorithm which searches for an optimal numerical solution among a large number of alternatives. This approach is based on the collective foraging behaviour of the honey bees. The behaviour of honey bees based on the communication, task allocation, nest site selection, reproduction, mating, floral foraging, and pheromone laying and navigation behaviours has been mimicked in this algorithm (Karaboga, Gorkemli, Ozturk, & Karaboga, 2014).

In this algorithm, first, potential food sources are identified from the population, which are initialized by the bees. Employed bees search for new food sources with a random stimulus initially. Subsequently, once a food source is identified (a candidate solution), the suitability (fitness) of the same is identified and computed. In the next phase, if a new food source is subsequently discovered (a new candidate solution) by "employed bees" with a greater suitability, the new source is adopted else the new one is rejected. "Employed bees" share the fitness information with the onlooker bees who choose their food source the probability of the food occurring (derived as a ratio of the fitness function of a source to that of the sum of fitness functions of all the sources). If bees are unable to improve the fitness of the food source, their solutions are rejected. Extensions of this algorithm have emulated the characteristics of queen bee behaviour, their dance and communication, their foraging behaviour, their mating and reproduction behaviour, their pheromone laying behaviour and navigation behaviour.

3.6. Bacterial foraging optimization algorithm

Bacterial foraging optimization algorithm (Biswas, Dasgupta, Das, & Abraham, 2007; Passino, 2002) was introduced as an extension of how natural selection tends to eliminate organisms with poor abilities to locate and ingest food for survival. Such foraging for food may happen at the individual organism level or at the social level. Foraging theory is based on the assumption that individual or groups of organisms search for and obtain food in a way that maximizes their energy intake per unit time spent on the search for food. Deviation from the objective may be due to fighting, reproducing, migrating or other activities based on external or environment factors (Passino, 2012). Further changes in the amount of availability of food (gradient of decline) in a particular location (say the amount of grass in a field, or the number of fruits in a tree) may signal the organisms to search for new food. This algorithm could be efficiently used to find potential suitable solutions by the application of operators like chemotaxis, swarming, reproduction, and elimination-dispersal (Biswas et al., 2007; Das, Biswas, Dasgupta, & Abraham, 2009). An agent is nominated as a bacterium, and it starts exploring for a local suitable solution. Then by using these operators, the bacteria try to locate the global optimum. The operator chemotaxis highlights two types of movement, namely swimming for some time and tumbling, while trying to find an optimal solution in a random direction. Swarming is the operator when a group of bacteria (agents) move up the gradient curve for better access to food based on information. The reproduction operator is used to identify and replicate the more efficient agents (those who have a higher value on the objective function, maximum energy per unit time spent) and re-move the inefficient ones. The elimination and dispersal operator accommodates changes in the environment, due to which a group of bacteria are eliminated or moved to a different location (gradient). While the algorithm is easy to comprehend and implement, poor convergence capability has been noted for complex optimization problems.

3.7. Leaping frog algorithm

The Leaping frog algorithm (Snyman, 1982; Snyman, 2000) was introduced as a search algorithm for minimization or maximization problems for identifying pre-dominantly local optimum (often in NP Hard problems). It combines the benefits of memetic algorithms and social behaviour based algorithms. The function in this algorithm need not be explicitly given but the gradient vector needs explicit definition (Snyman, 2000). While it is predominantly used for unconstrained problems, it has been used for constrained problems also (Fang & Wang, 2012; Holm & Botha, 1999). This method is especially suitable for problems where the objective function is affected by local inaccuracies, discontinuities and noise in the data. An extension is the shuffled leap from algorithm (Eusuff & Lansey, 2003) which allows for the exchange of information between local searches to move toward a global optimum. The extension of shuffled frog-leaping algorithm (Eusuff, Lansey, & Pasha, 2006) combines the benefits of genetic-based memetic algorithm and the social behaviour-based swarm optimization algorithms. These algorithms operate in sequential and iterative steps predominantly (Fang & Wang, 2012; Li, Luo, Chen, & Wang, 2012; Niknam, rasoulNarimani, Jabbari, & Malekpour, 2011), which again is inspired by the way frogs hunt for food in nature. An initial

population is formed by randomly generated virtual frogs, where each frog is represented by a vector. First, the whole population is partitioned into subsets called memeplexes. Each of these memeplexes has different sets of frogs. In each memeplex, a subset of the memeplex called sub-memeplex is constructed based on probability distributions. Within each sub-memeplex, the worst frog will leap towards a food source according to its own experience as well as the feedback from the best frog in that memeplex. Based on the leap, if the new positions are better that the old position, the frog repeats the process, else a new frog is generated randomly. After all the frogs in the whole population is shuffled, the better subpopulations are selected and the new population is divided into new memeplexes again and the process is repeated till problem termination conditions are satisfied.

3.8. Cuckoo search

Cuckoo search (Gandomi, Yang, & Alavi, 2013; Yang & Deb, 2009; Yang & Deb, 2013) was developed for addressing single or multi-objective problems under complex nonlinear constraints. For many global optimization problems with a single or multiple objectives, if the functions are highly nonlinear, achieving global optimality is not easy. Again, many real-world problems are often NP-hard, which means there is no known efficient algorithm which can be used for a given problem. Cuckoo search can address some of such challenging problems in providing a workable solution, which need not always be the globally optimal solution. In this algorithm, the breeding behaviour of cuckoos is replicated. Cuckoos lay their eggs in the nests of other birds and remove the eggs of the other bird to ensure higher probability of hatching of the cuckoo eggs. There are three basic types of brood parasitism which are imitated by this algorithm, namely intra-specific brood parasitism, co-operative breeding, and nest takeover (Yang & Deb, 2009, 2010b). Each cuckoo (agent) lays one or multiple eggs (candidate solution) at a time, depending on whether the problem conceptualized has a single or multiple objective. The nests may be identified through a Levy flight. The cuckoo puts its egg(s) in randomly chosen nest belonging to another bird. The best nests with higher quality of eggs (optimality of candidate solutions as defined by an objective function), will carry over to the next generations. Since the number of available host nests is fixed, there is a probability that the egg laid by a cuckoo is discovered by the host parent. In this case, the host parent can either throw the egg away or abandon the nest, and build a completely new nest with a potential to become a future nest. Every time a new set of candidate solutions are generated, a Levy flight is performed. Eggs of two or more nests may be mixed and redistributed for attempting to identify a globally optimal solution through induced diversity of the eggs in a nest.

3.9. Firefly algorithm

Firefly algorithms (Gandomi, Yang, & Alavi, 2011; Lukasik & Zak, 2009; Yang, 2009; Yang, Hosseini, & Gandomi, 2012) were introduced to address NP hard problems with non-convex objective functions which have equality and inequality based constraints. Firefly algorithms can deal with multi-modal functions more efficiently than other swarm algorithms (Yang, 2009; Yang, 2010b). Firefly algorithm employs a population based search and thus candidate solutions benefit from building blocks from very different solutions. Thus the mechanism facilitates good learning for parameter training to balance exploration versus exploitation. This algorithm is based on the natural behaviour of the fireflies which involves bioluminescence signalling to other fireflies and for deterring predators. These fireflies exhibit characteristics for swarm intelligence through self organizing and decentralized decision making. The bioluminescence signalling not only is an instrument for foraging for food, but also enables courtship signals for reproduction. Brightness of the flash is an indicator of fitness of the male firefly. However, for the standard algorithm, all fireflies are considered to be unisex and the attractiveness of the firefly is a function of light intensity, which again is the indicator of the fitness of a potential "candidate solution". An initial population of fireflies is created. After this initialization, one of the parameters for fitness is modified, and subsequently the fitness is evaluated for each firefly in the population. Subsequently, the fireflies may be ranked and best individuals of a solution may be taken forward for the next round of evaluation. The iteration may be controlled by the number of computations decided in advance. Further another positive aspect of the firefly algorithm is its ability to be used with other algorithms (hybrid approaches) to improve outcome (Rahmani & MirHassani, 2014; Verma, Aggarwal, & Patodi, 2016).

3.10. Bat algorithm

The bat algorithm (Yang, 2010a; Yang, 2011a) is one of the recently developed algorithm which uses the echo based location determining behaviour of bats to solve both single objective and multi-objective optimization problems in the continuous solution domain. This process, called echolocation, is used to refer to the way bats use echoes of sounds emitted by them to navigate their surroundings. By using this process bats can find the location of objects and prey, even in the dark. As per the algorithm, bats use echolocation to identify distance of objects or food. The bats can adjust their flight velocity as well as the frequency and loudness of their cry while searching for prey. Subsequently vector algebra is used for solving the problem iteratively. With each iteration, the loudness and the rate of pulse emission (frequency) needs to be updated such that the rate of emission increases and the loudness decreases once a bat identifies a potential prey. While most of the application of the bat algorithm is in the continuous problem domain, a binary version of the algorithm was proposed to address discrete decision making (Mirjalili, Mirjalili, & Yang, 2014). Further chaotic version of this algorithm (Gandomi & Yang, 2014; Gandomi et al., 2013) was developed to increase its global search mobility for robust global optimization. Studies have also been initiated to combine this algorithm with the classic bio-inspired algorithms like neural networks (Jaddi, Abdullah, & Hamdan, 2015).

3.11. Flower pollination algorithm

The flower pollination algorithm (Yang, 2012; Yang *et al.*, 2013) is based on the mechanisms of pollination of flowers. This was developed for applications in the domain of global optimization problems with multiple diverse criteria and multiple objectives. Pollination is a process whereby flowers spread their pollens, the active unit of reproduction, to another plant, for the purpose of germination (reproduction). This process requires the support of some agents namely pollinators. Typically, most flowers pollinate biotically using some insects or animals. However, some plants also pollinate abiotically using agents like wind and water. Pollination (Yang, 2012) can be achieved by self pollination or cross pollination, based on whether the process involves flowers of the same plant or flowers of different plants. Biotic and cross pollination are considered global pollination (optimization) processes while abiotic or self pollination is local pollination (optimization). The algorithm assumes that a solution is equivalent to a pollen or a flower with a single pollen. The pollination process would try to stimulate the reproduction of the fittest (candidate solution, estimated as a vector). The step size in subsequent iteration is

dependent on the strength of the pollination. For a global pollination, levy flights may be used in-between for the imitation of the movement of a pollinator, which may be an insect or a bird (Yang, Karamanoglu, & He, 2014). Else, from the uniform distribution, random candidate solution vectors may be chosen for pollination and evaluated. If the new solutions are better, they are retained, and the population pool is updated, else they are discarded. Iteratively this is continued till the best solution is identified. Further there may be additional challenging issues while implementing this algorithm such as time complexity, in-homogeneity and multi-dimensionality, which are addressed by appropriate methods.

3.12. Artificial plant optimization algorithm

The artificial plant optimization algorithm (Cui & Cai, 2013; Yang & Karamanoglu, 2013) was developed to address global optimization problems. This is especially suitable for problems which are non-differential, multimodal, and high-dimensional in nature. This algorithm emulates the growing phenomenon of a plant and how it enables photosynthesis for promoting growth and creation of food. The search space of the problem domain is mapped as the environment in which the plant survives. Provisions for sustenance are resources like air (oxygen, carbon dioxide), water, nutrients, and sunlight, and some of these resources (like air and water) could be considered inexhaustible and uniformly available while the others (light) could be varying for different branches of the same plant. The process is initialized by selecting a random set of branches from the plant, which act as candidate solutions. The fitness of these branches is calculated subsequently. Subsequently operators like photosynthesis and phototrophism are used on this sample. Photosynthesis is used to measure the efficiency of energy production of the branch using models like rectangular hyperbolic model, non-rectangular hyperbolic model, updated rectangular hyperbolic model, parabola model, straight line model and exponential curve models (Ye & Yu, 2007). Phototrophism is used to understand the direction of growth towards the light source, and is an indicator of the candidate solution moving towards an optimal solution, based on problem dimensionality and potential direction of convergence. Phototrophism also accommodates the influences of other candidate solutions / branches and is thus computationally complex, though the updation may be uniform. Further small probability of random influences may also be used to avoid optima in a uniform distribution.

4. Bio inspired algorithms – An overview of applications

In this section, we briefly describe the scope of the problems where the specific bio-inspired algorithms have been used and the nature of the outcome which has been achieved. However, specific in-text citation has been avoided since that would enhance the size of the domain review article too much, and affect the readability also. Fig. 3 depicts the scale at which complexity of problems has increased and how these algorithms have been used to address them with the progress of time.

Given the concerns surrounding the complexity of problems, especially those which were NP hard combinatorial problems for which searching for a pareto-optimal solution becomes extremely complex, algorithms focusing on meta-heuristics and hyper-heuristics started getting prominence. In the subsequent section we highlight some of the objectives of the application domains which have been addressed through the implementation of these bio-inspired algorithms, either in isolation or in association with other algorithms (hybrid approaches and hyper-heuristics).

4.1. Neural networks

Neural networks (Craven & Shavlik, 1997; Fausett, 1994; Lampinen & Vehtari, 2001) have been used extensively for generation of association rules, classification problems, detection problems, pattern recognition, non-linear regression, feature selection, missing data prediction, time-series prediction, data normalization, principal component analysis and probabilistic prediction. They have also been used for single objective and multi objective problems in the continuous domain. Further, rule extraction algorithms of neural networks can further be used with both supervised and unsupervised learning across multi-level optimization problems. Multi-level deep learning in neural networks have been introduced for non-linear feedforward networks, but recently, deep learning has been implemented in recurrent neural networks for applications such as language and speech modelling.

4.2. Genetic algorithm

In recent times, genetic algorithms (Goldberg, 2006; Grefenstette, 2013) have been used for searching among alternatives, maximization / minimization problems like the traveling salesman problem, sorting problems, multi-objective decision making, multi-criteria decision making and constrained optimization problems. Further a lot of application especially in the industrial engineering domain (Aytug et al., 2003; Reeves, 2003) has used genetic algorithms for network (path) analysis, job scheduling, supplier selection and project selection. Further domains like intrusion detection, parallel computation problems, dispatch problems, navigation and load balancing problems. Not much exploration has however happened for problems with multiple levels for such optimization, although pareto multi-objective optimization has been done using generic algorithms. However, challenges have also been faced by users in formulating the fitness function for understanding the fitness of candidate solutions, for higher dimensional multi-modal problems. This becomes more challenging especially if the solution space is highly scalable.

4.3. Ant colony optimization

The ant colony optimization algorithm (Liao, Socha, Montes de Oca, Stutzle, & Dorigo, 2014; Romdhane et al., 2013; Liao, Stützle, deOca, & Dorigo, 2014; Ghasab et al, 2015; Mandloi & Bhatia, 2015) is used to tackle mixed variable optimization problems and continuous optimization problems for selection, searching and optimization based problems. Many of such exploration have also happened to solve problems which are NP Hard, especially having multiple objectives on multiple levels and also in the continuous problem domain. For example, this algorithm has been used for job scheduling problems, data compression by extending c-means on images, environmental/economic dispatch problems, parameter estimation in dynamic systems, gaming theory, feature selection, congestion control, medical decision making, satellite control, social graph mining, target tracking, and signal processing problems.

4.4. Particle swarm algorithm

Reviews of swarm intelligence (Chakraborty & Kar, 2016) highlight that there are insect based approaches, animal based approaches and bird based approaches. Applications of the swarm based algorithm (Kennedy, 1997, Kennedy, 2011; Kennedy & Eberhart, 1995; Martens, Baesens & Fawcett, 2011) could be in multi-criteria decision problems, searching, constraint based optimization problems, deterministic optimization problems, scheduling problems, thresh-holding and maximization or minimization problems.

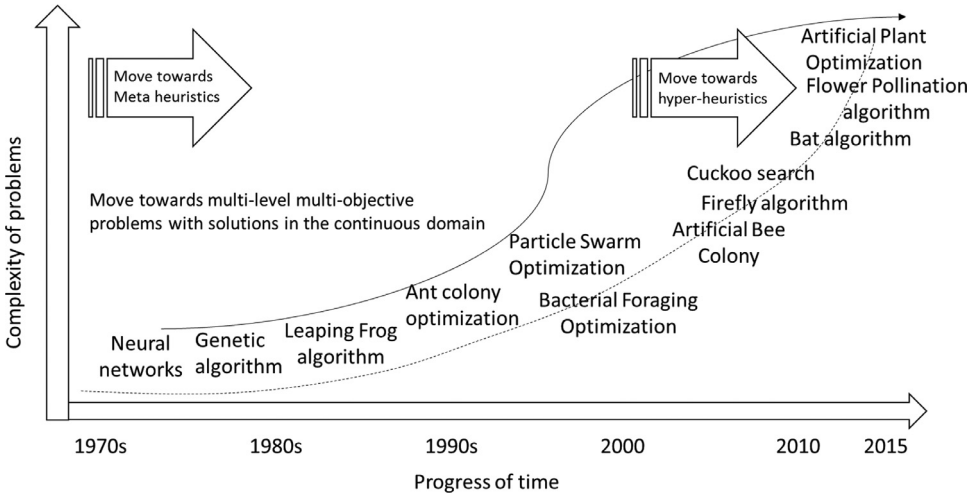

Fig. 3. Paradigm of evolution of algorithms with increase of complexity of problems.

These would typically be extremely complex problems with multi-dimensional multi-objective nature, with potential candidate solutions in a hyper-plane.

4.5. Artificial bee colony algorithm

The artificial bee colony algorithm has predominantly been used in literature (Karaboga & Basturk, 2008; Karboga & Akay, 2009) as a single objective numerical value optimizer. Further it has been used for searching, routing problem, assignment problem, task allocation problem and maximization or minimization problems and multilevel thresh-holding. Further it can be used for collective decision making for multi-criteria selection problems. Further these problem domains would be characterized by evolutionary computation needs with high scalability and differential evolution of solution space. For this algorithm, while the intensification process is controlled by both stochastic and greedy selection based approaches, diversification of suitable optimal solution is controlled by random selection based approaches. It has thus been used for unconstrained optimizations problems primarily but has also been used for constrained optimization problems. Further it can address multi-dimensional numeric problems, both single and multi-objective optimization problems, discrete and continuous optimization problems, evolutionary and differential evolution problems. The performance of this algorithm (Karaboga & Basturk, 2008; Karaboga & Ozturk, 2011) is very good in terms of the local and the global optimization due to the selection schemes employed and the neighbour production mechanism used. Consequently, the simulation results show that this algorithm is flexible, simple to use and robust, and can be used efficiently in the optimization of multimodal and multi-variable problems.

4.6. Bacterial foraging optimization

This bacterial foraging optimization algorithm (Das, Dasgupta, Biswas, Abraham & Konar, 2009; Dasgupta et al., 2009; Passino, 2012) has been used in literature for non-linear characteristics of multi-objective problems, dynamic resource allocation, independent component analysis, filtering multiple potential solutions, pattern recognition and job scheduling. It provides decent results in non-gradient optimization problems with multi-objective functions. While the computation complexity of the algorithm facilitates easy implementation, poor convergence capability has been noted for complex optimization problems although for less complex problems, it outperforms algorithms like genetic algorithm and particle swarm algorithms, based on convergence speed and final accuracy.

4.7. Leaping frog algorithm

The leaping frog algorithm (Elbeltagi, Hegazy, & Grierson, 2007; Rahimi-Vahed & Mirzaei, 2007; Snyman, 2000) has been used in different objectives like combinatorial, optimization problems, network design problems, job scheduling problems, thresh-holding problems, net-work scaling problems, cost minimization problems, permutation based searching problems and resource constrained problems. The shuffled leaping frog as an extension highlights better convergence in outcome in different applications in complex gradient problem domains, as compared to the original algorithm. As per literature (Eusuff et al., 2006; Li et al., 2012), the quality of outcome of the algorithm rises with the increase in the frog count in the population as well as the frog count in a sub-complex, but at the cost of number of function evaluation required to find the solution, which affects time complexity. It was also noted that the quality of outcome is more dependent on the number of memeplexes than to the number of frogs in a memeplex. In this algorithm, except when a random point is generated, the search will remain within the range of the set of feasible points. Thus, if all boundary points are not included in the initial pool of potential solutions, movement to the boundary may become difficult. Although this scheme is successful in terms of convergence, it has high complexity of computation and complexity.

4.8. Cuckoo search algorithm

Cuckoo search (Bhandari et al., 2014; Kar, 2014; Walia & Kapoor, 2014; Yang & Deb, 2014; Araghi, Khosravi, & Creighton , 2015; Gotmare, Patidar, & George, 2015; Kumar & Rawat, 2015) has been used for multi-criteria heuristic search problems, multi-objective heuristic problems, maximization of multiple opposite objectives, optimization among designs, gradient based optimization, gradient free optimization, multi-objective scheduling problems, multi-objective allocation problems, phase equilibrium problems, reliability optimization problems, path identification for network analysis and knapsack problems. Cuckoo search would be the a good method to use in situations where objective function evaluations are computationally expensive, since it performs very well at high numbers of iteration and performs comparably well elsewhere.

4.9. Firefly algorithm

Applications of firefly algorithm has been observed (Fister, Yang, & Brest, 2013; Long, Meesad, & Unger, 2015; Verma *et al.*, 2015; Kavousi-Fard et al., 2014; Mishra, Agarwal, Sharma, & Bedi, 2014) for solving problems with multi-modal functions, classification problems, NP hard problems with equality and/or inequality based constraints, continuous and discrete search based problems, combinatorial optimization problems, parallel computational problems and multi-objective search problems. The firefly algorithm has also been used with other methods like Levy flights, cellular learning automata, rough set theory and neural networks to form hybrid approaches. Chaos has been introduced in the firefly algorithm (Gandomi, Yang, Talatahari, & Alavi, 2013; Verma et al., 2016) so as to increase its global search mobility for robust global optimization. The results of introducing chaos reveal that there is significant improvement due to the utilization of deterministic chaotic signals in place of constant values. This algorithm can provide a good balance of exploitation and exploration (Yang & He, 2013b), since it requires far fewer function evaluations, thereby reducing computational complexity.

4.10. Bat algorithm

The bat algorithm has been used in existing literature (Meng, Gao, Liu, & Zhang, 2015; Rodrigues et al., 2014; Svečko & Kusić, 2015; Yang & He, 2013a) for multi-objective optimization, constrained optimization search, combinatorial optimization and scheduling, inverse parameter estimation, classification, clustering, vector matching and multi-valued systems based optimization. The bat algorithm is especially suitable for complex high-dimensional problems where convergence often is a challenge like in structural design optimization problems and chaotic multi-objective problems (Yang & Gandomi, 2012). The performance of the bat algorithm for constrained optimization tasks has been reported to be better than other bio-inspired computing approaches like genetic algorithms and particle swarm optimization (Gandomi, Yun, Yang, & Talatahari 2013).

4.11. Flower pollination algorithm

The flower pollination algorithm (Bekdaş, Nigdeli, & Yang, 2015, Nigdeli, Bekdaş, & Yang, 2016, Yang, 2012) can be used for global optimization problems with multiple diverse and opposite criteria with additional challenges surrounding nature of the hyper-surface depending on the number of dimensions of the multi-objective problem. Further the algorithm may be used to solve large integer programming problems, local and global search problems, high complexity convergence problems, and even non-linear Levy flight based design problems. The algorithm has been used in areas like civil engineering, structural engineering, energy management, emission control, feature selection, bundle sizing, electromagnetics, and large linear programming problems.

4.12. Artificial plant optimization algorithm

The artificial plant optimization algorithm (Cui, Yang, & Shi, 2012; Yu, Cui, & Zhang, 2013) could be used in global optimization problems in the continuous problem domain like protein folding, number sequence estimation, wireless sensor networks and other bio-informatics problem of similar nature and scope. It can be used for addressing problems with multiple objectives with unconstraint multi-modal benchmarks, where there may be numerous candidate solutions where an improved sub-optimal solution may also be suitable for the application domain.

Further after reviewing the applications of the algorithms, we attempt to summarize the scope of application and the objective which can be fulfilled by these algorithms based on evidences in existing literature, to provide a snapshot of their potential. This has been illustrated through Table 2.

Based on the scope of how these algorithms have been developed and applied in different context, we attempt to provide a direction on how they can be further explored in the subsequent section.

5. Implication of reviews of the algorithms

The review of the algorithms made us realize that the presence of literature surrounding these algorithms is extremely skewed and there is a need for literature in some of the less dominant algorithms. Further, we realized after the review that all the different algorithms could further be classified into four classes, based on the work that has been done. The classes have different scope, in terms of applications. The algorithms have been classified into four quadrants, as illustrated in Table 3. The implication of being present in any of the quadrants in Table 3 has been explained subsequently.

In the above categorization, we group algorithms based on our review of literature, on how they may be explored in the emerging times. Quadrant 1 reflects the zone of theory development, where we see a lot of scope is present in terms of both incremental development of the algorithms and comparative analysis among the algorithms. It would be extremely interesting to add the outcome of such research in the existing body of literature in intelligent systems and meta-heuristics. Further, the lack of literature in these algorithms highlight huge potential for research scholars working in this domain to explore how these algorithms may be improved by introducing novel improvements and dimensions (for example, chaos, uncertainty or constraints).

Quadrant 2 captures the zone of applications. These algorithms are somewhat mature in terms of theory development. It would be really interesting to revisit these algorithms and apply them in different disciplines. Areas like industrial engineering, information systems, financial management and supply chain management, have witnessed a lot of exploration of algorithms like genetic engineering, neural networks and ant colony optimization. So these could be domains which may adopt the algorithms under quadrant 2 and apply them in novel engineering and business contexts. The literature surrounding applications of such expert systems, intelligent systems and meta-heuristics, would highly benefit from such studies, and thus would draw attention from journals which publish studies in the domain. Further, the application of hybrid algorithms and hyper-heuristics involving these algorithms would be of interest to the academic community.

Quadrant 3 captures the algorithms, which were introduced, but somehow failed to capture the interest of the research community. While these algorithms have had studies both in terms of theory development and applications, not much inertia was carried in terms of using these algorithms across different studies. Maybe challenges were faced by the academic communities to implement the algorithms across different problems. It may be noteworthy to try and mix these algorithms with other theories and explore the quality of outcome. It may be of interest to combine these algorithms with theories like fuzzy sets, rough sets, chaotic systems, and other multivalued logic systems, and report their theoretical foundations. Also exploration of usage of such algorithms to novel problem domains would also be useful to the academic research community.

Quadrant 4 captures algorithms which have been clearly adopted by a huge group of scientific community. Too many studies have implemented these algorithms for different applications.

Table 2
Problem description and the solution objectives for selecting an algorithm.

SN	Algorithm	Problem domain description	Potential solution objectives (keywords)
1	Neural network	Single dimension non-linear problems, probabilistic problems, decision boundaries available, discrete solutions required	Association rules, pattern classification, regression, feature selection, missing data prediction, sequence mining, data reduction, probabilistic prediction, Bayesian and deep learning, feature detection, speech and image recognition, synchronization, control of non-linear systems, switching networks,
2	Genetic algorithm	Single or multi-objective problems, evaluation function of solutions known, well defined static solution space with low scalability. Performance drops for higher dimensionality.	Search, maximization or minimization, sorting, multi-criteria selection, job allocation, process scheduling, network analysis, anomaly detection, parallel computation, prioritization, classification, network path routing, layout planning, anomaly detection, signal coordination, anomaly detection, sorting, structural systems.
3	Ant colony optimization	Multi-variable and mixed variable problems, continuous optimization problems, NP hard combinatorial problems. Performance drops for higher dimensionality.	Network analysis, travelling salesman problem, scheduling, routing, clustering, data compression, environmental and economic dispatch problems, routing, data reconciliation, parameter estimation, gaming theory, objective tracking, demand forecasting, layout design, continuous optimization, timing optimization, resource consumption optimization.
4	Particle swarm	Population based multidimensional multi-objective problems, with potential candidate solutions in a hyper-plane. High scalability of problem space.	Distributed resource management, search, location identification, resource allocation, regulation, chaotic systems, oscillatory systems, global optimization, path optimization, adaptive learning, job scheduling, thresh-holding, network training, minimization, maximization, migration.
5	Artificial bee colony	Unconstrained and constrained optimization problems, evolutionary and differential evolution problems, multi-dimensional and multi-modal problems	Numerical function optimization, multilevel thresh-holding, network routing, allocation / assignment, test suite optimization, search, bench-marking, probability distribution, feature selection, single and multi-objective optimization, discrete and continuous optimization.
6	Bacterial foraging optimization	Distributed optimization problems, non-linear multi-objective problems with low complexity, gradient based domains.	Multi-optimal function optimization, numerical optimization, global optimization, gradient based search, linear combiners, forecasting models, minimization and maximization, load forecasting and compensation, portfolio value prediction, clustering, load dispatch and calibration.
7	Leaping frog	Gradient based continuous optimization problems with dynamic paths to a local optima. Function definition not needed achieving the solution.	Combinatorial problems, optimization problems, network design problems, job scheduling problems, thresh-holding problems, net-work scaling problems, cost minimization problems, permutation based searching problems and resource constrained problems
8	Cuckoo search	Single or multi-objective problems under complex nonlinear constraints. NP Hard problems where suboptimal solutions may also be practically usable.	Search problems, multi-objective problems, optimization among designs, gradient based optimization, gradient free optimization, multi-objective scheduling, multi-objective allocation, phase equilibrium problems, reliability optimization, path identification for network analysis, knapsack problems.
9	Firefly algorithm	NP hard problems with non-convex objective functions which have equality and inequality based constraints. Deal with multi-modal functions well.	Dispatch problems, job scheduling, chaotic problems, structural optimization, continuous optimization, vector quantization, clustering, price forecasting, discrete optimization, load forecasting, network analysis, travelling salesman problem, non-linear optimization, dynamic environment problems.
10	Bat algorithm	Nonlinear and multimodal problems. Especially suitable for complex high-dimensional problems where convergence is a challenge.	Structural design optimization, multi-objective optimisation, numerical optimization problems, network path analysis, multi-constrained operations, adaptive learning problems, environmental/economic dispatch, scheduling, effort estimation, classification, vector matching and association rule mining.
11	Flower pollination	Continuous optimization problems, single objective and multi-objective optimization. Functions could have convex, non-convex, discontinuous or Pareto fronts.	Control in multi-machine systems, feature selection, structure optimization, data reduction, array synthesis, classification, search, multi-criteria selection, chaotic systems, electro-magnetics, large scale linear programming, energy management, structural engineering.
12	Artificial plant optimization	Global optimization problems with multiple criteria. Additional challenges surrounding nature of the hyper-surface.	Protein analysis, network configuration simulation analysis, coverage optimization, telecom sensor networks, molecular structure analysis.

The scope of finding unexplored applications which may interest the scientific community may be a daunting task. Further, a lot of studies have actually explored the development of theories in these algorithms, and unless substantial new findings emerge out of new approaches within these algorithms, the methodological implications would be of lesser interest to the scientific community. However, it is interesting to note that since these algorithms have been highly used by the scientific community, these are ready for adoption in the practice. The algorithms can be readily adopted for industrial applications since most of their pseudo-codes and algorithmic logic has consolidated and standardised over the period of scientific exploration. Being tried and tested approaches, the applications of these algorithms would have lower time for implementation and lesser issues due to lack of body of scientific knowledge.

6. Concluding discussion

While these algorithms have witnessed a lot of attention from decision scientists in recent years, the understanding within the domain is far from being mature. Except for a few algorithms like genetic algorithms, artificial bee colony, particle swarm optimization and neural networks, literature presents a lot of debate on the convergence and stability of these algorithms. The focus of this paper is not to highlight how these algorithms may be used for solving real life problem, but to provide initial food for thought for understanding the scope and objective for these algorithms.

Some reviews of metaheuristics (Gogna & Tayal, 2013; Yang, 2011) have focused on exploring some of these classic algorithms, but mostly algorithms like genetic algorithm, ant colony optimization and neural networks have been captured in those reviews of

Table 3
Categorization of algorithms based on scope for researchers.

Quadrant 1: Zone of theory development	Quadrant 2: Zone of applications
• Amoeba (Zhang et al., 2013) • Artificial plant optimization (Cui & Cai, 2013) • Bean optimization (Zhang et al., 2010) • Dove (Su et al., 2009) • Eagle (Yang & Deb, 2010) • Fruit fly (Pan, 2012) • Glow-worm (Krishnanand & Ghose, 2005) • Grey wolf algorithm (Mirjalili, Mirjalili, & Yang, 2014) • Krill-herd (Gandomi & Alavi, 2012) • Lion (Yazdani & Jolai, 2015) • Monkey (Mucherino & Seref, 2007) • Wolf (Liu et al., 2011)	• Bacterial foraging (Passino, 2002) • Bat algorithm (Yang, 2010) • Bee colony (Karaboga, 2005) • Cuckoo search (Yang & Deb, 2009) • Firefly algorithm (Yang, 2009) • Flower pollination (Yang, 2012)
Quadrant 3: Zone of rediscovery	**Quadrant 4: Zone of commercialization**
• Leaping Frog (Snyman, 1982) • Shark (Hersovici et al., 1998) • Wasp (Theraulaz et al., 1991)	• Ant colony optimization (Dorigo et al., 2006) • Genetic algorithm (Holland, 1975) • Neural networks (Grossberg, 1988) • Particle swarm (Shi & Eberhart, 1999)

literature. Reviews of swarm intelligence have also been somewhat sparse. Though there has been some reviews of some of the new algorithms in isolation, it does not provide the reader an overview of the potential of similar other algorithms which can be used when the nature of the problems are similar. Therefore, such reviews may be useful, once the scholar is somewhat versed with the knowledge of the presence of different types of these algorithms. Our study also directs the reader towards such review papers of specific algorithms which may be useful for developing a further understanding on scope of application and the development of theories.

The preliminary understanding of these algorithms would also be useful for practitioners, especially those who are working in the domain of information or data science. Depending on the nature of their business problem and the complications associated with the objectives, constraints and computational space, they may use this as an initial reading point to explore many of these bio-inspired computing algorithms for actual applications in real life problems in the social, organizational or governance space.

7. Implications for academics and practice

It is interesting to note that many of these bio-inspired algorithms have had their roots in the engineering and pure science based domains. However, there is a huge potential to use these methods across different problem domains as has been seen, as methods and their associated theories become more mature. Such a diverse application has been witnessed in the domain of business and management, especially for methods like neural networks, genetic algorithms and ant colony optimization. In particular, a lot of these algorithms have been used in production management, information systems management and supply chain management. Further a lot of the applications of these matured algorithms have been witnessed in the domain of information systems, focusing on decisions at the operational level, tactical level, functional level strategic decisions, business level strategic decisions and corporate level strategic decisions across domains like agriculture, urban planning, military, healthcare, education, governance and other application industries (Bahrammirzaee, 2010; Eom & Kim, 2006). These algorithms (e.g. neural networks, genetic algorithms and ant colony optimization) could be readily adopted now for industry level applications and solution development.

However, not much exploration has happened in the true application space for the rest of these algorithms, probably due to the recency of some of these developments or due to the lack of availability of pseudocodes which can be used directly. So in the near future, it would be exciting to find academic studies highlighting the application of algorithms like artificial bee colony algorithm, bacterial foraging algorithm, firefly algorithm, leaping frog algorithm, bat algorithm, flower pollination algorithm and artificial plant optimization algorithm in actual problem contexts surrounding enterprise operations, government applications and social application domains, as has been seen in some of the mature algorithms. Indeed such exploration would also enhance the scope of enrichment of the algorithms itself through developments in theoretical understanding. Further newer and newer algorithms are rapidly getting introduced in the domain of bio inspired algorithm. Many of the less explored algorithms like the amoeba based algorithm (Zhang et al., 2013), bean optimization algorithm (Zhang et al., 2010), individual bird based algorithms based on doves and eagles (Su et al., 2009; Yang & Deb, 2010), individual insect based algorithms like fruit fly, wasp and glow-worm (Krishnanand & Ghose, 2005; Pan, 2012; Theraulaz et al., 1991), individual animal based algorithms like monkey, shark, wolf and lions (Hersovici et al., 1998; Liu et al., 2011; Mucherino & Seref, 2007; Yazdani & Jolai, 2015) could be explored, both for developing the algorithm itself and also for exploring their scope of applications across domains. Also, the scholars exploring any of the algorithms could start by exploring the recent publications of the dominant contributors, who have been identified in Table 1.

However, since very few studies have been conducted using some of these algorithms, methodological research would also be very useful for these algorithms prior to exploring the scope of applications. For example, the lion algorithm (Yazdani & Jolai, 2015) has been recently introduced for addressing complex optimization problems, based on the mating and territorial behaviour of lions. Similarly, the krill-herd algorithm (Gandomi & Alavi, 2012) explores the foraging behaviour of the marine krills. Further the grey wolf algorithm (Mirjalili, Mirjalili & Lewis, 2014) mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. However, for our discussion in this study, we have chosen algorithms which have slightly larger set of applications, so that their scope can be outlined through a review of literature. Such a review covering some of these algorithms is not possible due to the presence of very few application based studies, in specific domains. It would be interesting to focus on studies which contribute to the theory development within these algorithms and also compare these newly emerging algorithms with the existing ones and highlight the differences in performances.

References

Araghi, S., Khosravi, A., & Creighton, D. (2015). Intelligent cuckoo search optimized traffic signal controllers for multi-intersection network. *Expert Systems with Applications, 42*(9), 4422–4431.

Aytug, H., Khouja, M., & Vergara, F. E. (2003). Use of genetic algorithms to solve production and operations management problems: A review. *International Journal of Production Re-search, 41*(17), 3955–4009.

Bahrammirzaee, A. (2010). A comparative survey of artificial intelligence applications in finance: Artificial neural networks, expert system and hybrid intelligent systems. *Neural Computing and Applications, 19*(8), 1165–1195.

Bekdaş, G., Nigdeli, S. M., & Yang, X. S. (2015). Sizing optimization of truss structures using flower pollination algorithm. *Applied Soft Computing, 37*, 322–331.

Bhandari, A. K., Singh, V. K., Kumar, A., & Singh, G. K. (2014). Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. *Expert Systems with Applications, 41*(7), 3538–3560.

Biswas, A., Dasgupta, S., Das, S., & Abraham, A. (2007). Synergy of PSO and bacterial foraging optimization—A comparative study on numerical benchmarks. In *Innovations in hybrid intelligent systems* (pp. 255–263). Berlin Heidelberg: Springer.

Bounds, D. G., Lloyd, P. J., Mathew, B., & Waddell, G. (1988). A multilayer perceptron network for the diagnosis of low back pain. In *Neural Networks, 1988., IEEE International Conference on* (pp. 481–489).

Bououden, S., Chadli, M., & Karimi, H. R. (2015). An ant colony optimization-based fuzzy predictive control approach for nonlinear processes. *Information Sciences, 299*, 143–158.

Chakraborty, A., & Kar, A.K. (2016). Swarm intelligence: A review of algorithms. Forthcoming in Modeling and Optimization in Science and Technologies.

Colin, R. R., & Jonathan, E. R. (2002). *Genetic algorithms-Principles and perspectives, A guide to GA Theory*. Kluwer Academic Publishers, Springer.

Couceiro, M., & Ghamisi, P. (2016). Particle swarm optimization. In *Fractional Order Darwinian Particle Swarm Optimization* (pp. 1–10). Springer International Publishing.

Craven, M. W., & &Shavlik, J. W. (1997). Using neural networks for data mining. *Future generation computer systems, 13*(2), 211–229.

Cui, Z., & Cai, X. (2013). Artificial plant optimization algorithm. In *Swarm intelligence and bio-inspired computation: Theory and applications* (pp. 351–365).

Cui, Z., Yang, H., & Shi, Z. (2012). Using artificial plant optimization algorithm to solve coverage problem in WSN. *Sensor Letters, 10*(8), 1666–1675.

Darwin, C. (1859). *On the origins of species by means of natural selection*. London: Murray.

Das, S., Biswas, A., Dasgupta, S., & Abraham, A. (2009). Bacterial foraging optimization algorithm: Theoretical foundations, analysis, and applications. In *Foundations of computational intelligence volume 3* (pp. 23–55). Berlin Heidelberg: Springer.

Das, S., Dasgupta, S., Biswas, A., Abraham, A., & Konar, A. (2009). On stability of the chemotactic dynamics in bacterial-foraging optimization algorithm. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 39*(3), 670–679.

Dasgupta, S., Das, S., Abraham, A., & Biswas, A. (2009). Adaptive computational chemotaxis in bacterial foraging optimization: An analysis. *IEEE Transactions on Evolutionary Computation, 13*(4), 919–941.

Dimopoulos, C., & &Zalzala, A. (2000). Recent developments in evolutionary computation for manufacturing optimization: Problems, solutions, and comparisons. *IEEE Transactions on Evolutionary Computation, 4*(2), 93–113.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science, 344*(2), 243–278.

Dorigo, M., & Birattari, M. (2010). Ant colony optimization. In *Encyclopaedia of machine learning* (pp. 36–39). Springer US.

Dorigo, M., Birattari, M., & &Stützle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine, 1*(4), 28–39.

Elbeltagi, E., Hegazy, T., & Grierson, D. (2007). A modified shuffled frog-leaping optimization algorithm: Applications to project management. *Structure and Infrastructure Engineering, 3*(1), 53–60.

Eom, S., & Kim, E. (2006). A survey of decision support system applications (1995–2001). *Journal of the Operational Research Society, 57*(11), 1264–1278.

Eusuff, M. M., & Lansey, K. E. (2003). Optimization of water distribution network design using the shuffled frog leaping algorithm. *Journal of Water Resources Planning and Management, 129*(3), 210–225.

Eusuff, M., Lansey, K., & Pasha, F. (2006). Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. *Engineering Optimization, 38*(2), 129–154.

Fang, C., & Wang, L. (2012). An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem. *Computers & Operations Research, 39*(5), 890–901.

Fausett, L. (1994). *Fundamentals of neural networks: Architectures, algorithms, and applications*. Upper Saddle River, New Jersey: Prentice-Hall, Inc.

Fister, I., Yang, X. S., & Brest, J. (2013). A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation, 13*, 34–46.

Floreano, D., & Mattiussi, C. (2008). *Bio-inspired artificial intelligence: Theories, methods, and technologies*. Cambridge: MIT Press.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation, 17*(12), 4831–4845.

Gandomi, A. H., & Yang, X. S. (2014). Chaotic bat algorithm. *Journal of Computational Science, 5*(2), 224–232.

Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2011). Mixed variable structural optimization using firefly algorithm. *Computers & Structures, 89*(23), 2325–2336.

Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with computers, 29*(1), 17–35.

Gandomi, A. H., Yang, X. S., Alavi, A. H., & Talatahari, S. (2013). Bat algorithm for constrained optimization tasks. *Neural Computing and Applications, 22*(6), 1239–1255.

Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H. (2013). Firefly algorithm with chaos. *Communications in Nonlinear Science and Numerical Simulation, 18*(1), 89–98.

Gandomi, A. H., Yun, G. J., Yang, X. S., & Talatahari, S. (2013). Chaos-enhanced accelerated particle swarm optimization. *Communications in Nonlinear Science and Numerical Simulation, 18*(2), 327–340.

Gao, W. F., & Liu, S. Y. (2012). A modified artificial bee colony algorithm. *Computers & Operations Research, 39*(3), 687–697.

Ghasab, M. A. J., Khamis, S., Mohammad, F., & Fariman, H. J. (2015). Feature decision-making ant colony optimization system for an automated recognition of plant species. *Expert Systems with Applications, 42*(5), 2361–2370.

Gogna, A., & &Tayal, A. (2013). Metaheuristics: Review and application. *Journal of Experimental & Theoretical Artificial Intelligence, 25*(4), 503–526.

Goldberg, D. E. (2006). *Genetic algorithms*. India: Pearson Education.

Gotmare, A., Patidar, R., & George, N. V. (2015). Nonlinear system identification using a cuckoo search optimized adaptive Hammerstein model. *Expert Systems with Applications, 42*(5), 2538–2546.

Grefenstette, J. J. (2013). *Genetic algorithms and their applications: Proceedings of the second international conference on genetic algorithms*. Hillsdale, New Jersey: Psychology Press.

Grossberg, S. (1988). Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural networks, 1*(1), 17–61.

Hersovici, M., Jacovi, M., Maarek, Y. S., Pelleg, D., Shtalhaim, M., & Ur, S. (1998). The sharksearch algorithm. An application: Tailored web site mapping. *Computer Networks and ISDN Systems, 30*, 317–326.

Holland, J.H. (1975). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence University of Michigan Press.

Holm, J. E., & Botha, E. C. (1999). Leap-frog is a robust algorithm for training neural net-works. *Network: Computation in Neural Systems, 10*(1), 1–13.

Hong, T. P., Tung, Y. F., Wang, S. L., Wu, Y. L., & Wu, M. T. (2012). A multi-level ant–colony mining algorithm for membership functions. *Information Sciences, 182*(1), 3–14.

Hong, W. C. (2009). Chaotic particle swarm optimization algorithm in a support vector regression electric load forecasting model. *Energy Conversion and Management, 50*(1), 105–117.

Hornik, K. (1991). Approximation capabilities of multilayer feed-forward networks. *Neural networks, 4*(2), 251–257.

Jaddi, N. S., Abdullah, S., & Hamdan, A. R. (2015). Multi-population cooperative bat algorithm-based optimization of artificial neural network model. *Information Sciences, 294*, 628–644.

Kar, A. K. (2013). Using artificial neural networks and analytic hierarchy process for the supplier selection problem. In *2013 IEEE international conference on signal processing, computing and control* (pp. 1–6).

Kar, A. K. (2014). A decision support system for website selection for internet based advertising and promotions. In *Emerging Trends in Computing and Communication* (pp. 453–457).

Kar, A. K. (2015). A hybrid group decision support system for supplier selection using analytic hierarchy process, fuzzy set theory and neural network. *Journal of Computational Science, 6*, 23–33.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200). Technical report-tr06, Erciyes University, Computer Engineering Department.

Karaboga, D., & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied soft computing, 8*(1), 687–697.

Karaboga, D., & Ozturk, C. (2011). A novel clustering approach: Artificial bee colony (ABC) algorithm. *Applied soft computing, 11*(1), 652–657.

Karaboga, D., & Akay, B. (2009). A survey: Algorithms simulating bee swarm intelligence. *Artificial Intelligence Review, 31*(1-4), 61–85.

Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2014). A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review, 42*(1), 21–57.

Kavousi-Fard, A., Samet, H., & Marzbani, F. (2014). A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting. *Expert systems with applications, 41*(13), 6047–6056.

Kennedy, J. (1997). The particle swarm: Social adaptation of knowledge. In *Proceedings of the IEEE international conference on evolutionary computation* (pp. 303–308).

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Network* (pp. 1942–1948).

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopaedia of machine learning* (pp. 760–766). New York: Springer US.

Krishnanand, K. N., & Ghose, D. (2005). Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In *IEEE Swarm intelligence symposium* (pp. 84–91).

Kumar, M., & Rawat, T. K. (2015). Optimal design of FIR fractional order differentiator using cuckoo search algorithm. *Expert Systems with Applications, 42*(7), 3433–3449.

Lampinen, J., & &Vehtari, A. (2001). Bayesian approach for neural networks—review and case studies. *Neural networks, 14*(3), 257–274.

Li, X., Luo, J., Chen, M. R., & Wang, N. (2012). An improved shuffled frog-leaping algorithm with extremal optimisation for continuous optimisation. *Information Sciences, 192*, 143–151.

Liao, T., Socha, K., Montes de Oca, M., Stutzle, T., & Dorigo, M. (2014). Ant colony optimization for mixed-variable optimization problems. *IEEE Transactions on Evolutionary Computation, 18*(4), 503–518.

Liao, T., Stützle, T., deOca, M. A. M., & Dorigo, M. (2014). A unified ant colony optimization algorithm for continuous optimization. *European Journal of Operational Research, 234*(3), 597–609.

Liu, B., Wang, L., Jin, Y. H., Tang, F., & Huang, D. X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons & Fractals, 25*(5), 1261–1271.

Liu, C., Yan, X., Liu, C., & Wu, H. (2011). The wolf colony algorithm and its application. *Chinese Journal of Electronics, 20*, 212–216.

Long, N. C., Meesad, P., & Unger, H. (2015). A highly accurate firefly based algorithm for heart disease prediction. *Expert Systems with Applications, 42*(21), 8221–8231.

Lukasik, S., & &Zak, S. (2009). Firefly algorithm for continuous constrained optimization tasks. In *Computational collective intelligence. Semantic web, social networks and multiagent systems* (pp. 97–106). Berlin: Springer.

Mandloi, M., & Bhatia, V. (2015). Congestion control based ant colony optimization algorithm for large MIMO detection. *Expert Systems with Applications, 42*(7), 3662–3669.

Martens, D., Baesens, B., & Fawcett, T. (2011). Editorial survey: swarm intelligence for data mining. *Machine Learning, 82*(1), 1–42.

Meng, X. B., Gao, X. Z., Liu, Y., & Zhang, H. (2015). A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization. *Expert Systems with Applications, 42*(17), 6350–6364.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Mirjalili, S., Mirjalili, S. M., & Yang, X. S. (2014). Binary bat algorithm. *Neural Computing and Applications, 25*(3-4), 663–681.

Mishra, A., Agarwal, C., Sharma, A., & Bedi, P. (2014). Optimized gray-scale image watermarking using DWT–SVD and Firefly Algorithm. *Expert Systems with Applications, 41*(17), 7858–7867.

Mitchell, M., Forrest, S., & Holland, J. H. (1992, December). The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proceedings of the first European conference on artificial life* (pp. 245–254). Cambridge: The MIT Press.

Mucherino, A., & Seref, O. (2007). Monkey search: A novel metaheuristic search for global optimization. *AIP Conference Proceedings, 953*, 162–173.

Nigdeli, S. M., Bekdaş, G., & Yang, X. S. (2016). Application of the flower pollination algorithm in structural engineering. In *Metaheuristics and optimization in civil engineering* (pp. 25–42). Berlin: Springer.

Niknam, T., rasoulNarimani, M., Jabbari, M., & Malekpour, A. R. (2011). A modified shuffle frog leaping algorithm for multi-objective optimal power flow. *Energy, 36*(11), 6420–6432.

Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Networks, 5*(6), 927–935.

Pan, W.-T. (2012). A new fruit fly optimization algorithm: Taking the financial distress model as an example. *Knowledge-Based Systems, 26*, 69–74.

Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *Control Systems, IEEE, 22*(3), 52–67.

Passino, K. M. (2012). Bacterial foraging optimization. In *Innovations and Developments of Swarm Intelligence Applications* (pp. 219–233).

Rahimi-Vahed, A., & Mirzaei, A. H. (2007). A hybrid multi-objective shuffled frog-leaping algorithm for a mixed-model assembly line sequencing problem. *Computers & Industrial Engineering, 53*(4), 642–666.

Rahmani, A., & MirHassani, S. A. (2014). A hybrid Firefly-Genetic Algorithm for the capacitated facility location problem. *Information Sciences, 283*, 70–78.

Reeves, C. (2003). Genetic algorithms. In *Handbook of Metaheuristics: Vol. 57* (pp. 55–82).

Rodrigues, D., Pereira, L. A., Nakamura, R. Y., Costa, K. A., Yang, X. S., Souza, A. N., & Papa, J. P. (2014). A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications, 41*(5), 2250–2258.

Romdhane, L. B., Chaabani, Y., & Zardi, H.MARS Research Group. (2013). A robust ant colony optimization-based algorithm for community mining in large scale oriented social graphs. *Expert Systems with Applications, 40*(14), 5709–5718.

Sadegh, N. (1993). A perceptron network for functional identification and control of nonlinear systems. *IEEE Transactions on Neural Networks, 4*(6), 982–988.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, 61*, 85–117.

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Evolutionary programming VII* (pp. 591–600). Berlin: Springer.

Shi, Y., & Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Proceedings of the 1999 IEEE congress on evolutionary computation* (pp. 1945–1950).

Shlesinger, M. F., & &Klafter, J. (1986). Lévy walks versus Lévy flights. In *On growth and form* (pp. 279–283). Netherlands: Springer.

Snyman, J. A. (1982). A new and dynamic method for unconstrained minimization. *Applied Mathematical Modelling, 6*(6), 449–462.

Snyman, J. A. (2000). The LFOPC leap-frog algorithm for constrained optimization. *Computers & Mathematics with Applications, 40*(8), 1085–1096.

Specht, D. F. (1990). Probabilistic neural networks. *Neural networks, 3*(1), 109–118.

Srinivas, M., & Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics, 24*(4), 656–667.

Su, M.-C., Su, S.-Y., & Zhao, Y.-X. (2009). A swarm-inspired projection algorithm. *Pattern Recognition, 42*, 2764–2786.

Svečko, R., & Kusić, D. (2015). Feedforward neural network position control of a piezoelectric actuator based on a BAT search algorithm. *Expert Systems with Applications, 42*(13), 5416–5423.

Theraulaz, G., Goss, S., Gervet, J., & Deneubourg, J. L. (1991). Task differentiation in polistes wasps colonies: A model for self-organizing groups of robots. In *First international conference on simulation of adaptive behavior* (pp. 346–355). Cambridge: MIT Press.

Verma, O. P., Aggarwal, D., & Patodi, T. (2016). Opposition and dimensional based modified firefly algorithm. *Expert Systems with Applications, 44*, 168–176.

Walia, G. S., & Kapoor, R. (2014). Intelligent video target tracking using an evolutionary particle filter based upon improved cuckoo search. *Expert Systems with Applications, 41*(14), 6315–6326.

Yang, X. S. (2009). Firefly algorithms for multimodal optimization. In *Stochastic algorithms: Foundations and applications* (pp. 169–178). Berlin Heidelberg: Springer.

Yang, X. S. (2010). Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation, 2*(2), 78–84.

Yang, X. S. (2011). Review of meta-heuristics and generalised evolutionary walk algorithm. *International Journal of Bio-Inspired Computation, 3*(2), 77–84.

Yang, X. S. (2012). Flower pollination algorithm for global optimization. In *Unconventional computation and natural computation* (pp. 240–249). Berlin Heidelberg: Springer.

Yang, X. S., & Deb, S. (2009, December). Cuckoo search via Lévy flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on* (pp. 210–214). IEEE.

Yang, X. S., & Deb, S. (2010). Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization. In J. R. Gonzalez (Ed.), *Nature inspired cooperative strategies for optimization (NISCO 2010), SCI 284* (pp. 101–111). Berlin: Springer.

Yang, X. S., & Deb, S. (2010). Engineering optimisation by cuckoo search. *International Journal of Mathematical Modelling and Numerical Optimisation, 1*(4), 330–343.

Yang, X. S., & Deb, S. (2013). Multiobjective cuckoo search for design optimization. *Computers & Operations Research, 40*(6), 1616–1624.

Yang, X. S., & Deb, S. (2014). Cuckoo search: Recent advances and applications. *Neural Computing and Applications, 24*(1), 169–174.

Yang, X. S., & He, X. (2013a). Bat algorithm: Literature review and applications. *International Journal of Bio-Inspired Computation, 5*(3), 141–149.

Yang, X. S., & He, X. (2013b). Firefly algorithm: Recent advances and applications. *International Journal of Swarm Intelligence, 1*(1), 36–50.

Yang, X. S., & Gandomi, A. H. (2012). Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations, 29*(5), 464–483.

Yang, X. S., & Karamanoglu, M. (2013). Swarm Intelligence and Bio-Inspired Computation: An Overview. *Swarm Intelligence and Bio-inspired Computation-Theory and Applications* (pp. 3–23). Elsevier, Waltham, USA.

Yang, X.S., Cui, Z., Xiao, R., Gandomi, A.H., & Karamanoglu, M. ((2013).Eds.). Swarm intelligence and bio-inspired computation: Theory and applications, Elsevier, Waltham USA.

Yang, X. S., Hosseini, S. S. S., & Gandomi, A. H. (2012). Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect. *Applied Soft Computing, 12*(3), 1180–1186.

Yang, X. S., Karamanoglu, M., & He, X. (2014). Flower pollination algorithm: A novel approach for multi-objective optimization. *Engineering Optimization, 46*(9), 1222–1237.

Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65–74). Berlin: Springer.

Yang, X. S. (2011). Bat algorithm for multi-objective optimization. *International Journal of Bio-Inspired Computation, 3*(5), 267–274.

Yazdani, M., & Jolai, F. (2015). Lion optimization algorithm (LOA): A nature-inspired metaheuristic algorithm. *Journal of Computational Design and Engineering, 3*(1), 24–36.

Ye, Z. P., & Yu, Q. (2007). Comparison of a new model of light response of photosynthesis with traditional models. *Journal-Shenyang Agricultural University, 38*(6), 771–784.

Yu, B., Cui, Z., & Zhang, G. (2013). Artificial plant optimization algorithm with correlation branches. *Journal of Bioinformatics and Intelligent Control, 2*(2), 146–155.

Zhan, Z. H., Zhang, J., Li, Y., & Shi, Y. H. (2011). Orthogonal learning particle swarm optimization. *IEEE Transactions on Evolutionary Computation, 15*(6), 832–847.

Zhang, X., Huang, S., Hu, Y., Zhang, Y., Mahadevan, S., & Deng, Y. (2013). Solving 0–1 knapsack problems based on amoeboid organism algorithm. *Applied Mathematics and Computation, 219*, 9959–9970.

Zhang, X., Sun, B., Mei, T., & Wang, R. (2010). Post-disaster restoration based on fuzzy preference relation and bean optimization algorithm. In *IEEE youth conference on information computing and telecommunications* (pp. 271–274).