Review and classification of Bio-inspired Algorithms and their Applications

Xumei Fan[1*], William Sayers[2], Shujun Zhang[2*], Zhiwu Han[3], Luquan Ren[3], Hassan Chizari[2]

[1]School of Management, Jilin University, Changchun 132200, China

[2]School of Computing and Technology, the University of Gloucestershire, Cheltenham, GL50 2RH, UK

[3]Key Laboratory of Bionics Engineering of Ministry of Education, Jilin University, Changchun 132200, China

Abstract

Scientists have long looked to nature and biology in order to understand and model solutions for complex real-world problems. The study of bionics bridges the functions, biological structures and functions and organizational principles found in nature with our modern technologies, numerous mathematical and metaheuristic algorithms have been developed along with the knowledge transferring process from the life forms to the human technologies. Output of bionics study includes not only physical products, but also various optimization computation methods that can be applied in different areas. Related algorithms can broadly be divided into four groups: (1)evolutionary based bio-inspired algorithms, (2) swarm intelligence-based bioinspired algorithms, (3) ecology-based bio-inspired algorithms and (4) multi-objective bioinspired algorithms. Bio-inspired algorithms such as neural network, ant colony algorithms, particle swarm optimization and others have been applied in almost every area of science, engineering and business management with a dramatic increase of number of relevant publications. This paper provides a systematic, pragmatic and comprehensive review of the latest developments in Evolutionary based Bio-Inspired Algorithms, Swarm Intelligence based Bio-Inspired Algorithms, Ecology based Bio-Inspired Algorithms and Multi-objective Bio-Inspired Algorithms.

Keywords: Bio-inspired, Optimization, Multi-objective optimization, Evolutionary Based Algorithms, Swarm Intelligence Based Algorithms, Ecology Based Bio-inspired Algorithms

1 Introduction

In recent decades, there has been an ever-increasing interest in nature-inspired algorithms[1–3]. The popularity increase in machine learning technology, with the development of convolutional neural networks and the rise in deep-learning technology as a result[4] have formed part of this. Alongside this has been the ever increasing amount of compute power that is readily available to individuals and small to

medium sized businesses[5,6].The pervasiveness of computing into everyday life combined with these impressive results from one area of bioinspired algorithms, and availability of compute power, have combined to drive interest in bioinspired algorithms in general upwards.

Previous review papers in this area are becoming dated[1,7,8] and new advancements in this field have been published[9,10] which may not have been covered by historical reviews. Additionally, previous review papers have tended to cover a smaller set of algorithms that are particularly applicable to the papers problem-area. Ever increasing core counts in consumer available computer hardware and the increasingly common availability of graphics processing units have made the field very fast-moving.

For all these reasons it is valuable and timely to have a systematic modern review, covering a broad range of nature-inspired algorithms.

Computers are increasingly capable of extreme parallelization, particularly with general purpose graphics processing unit programming, and specialized chips developed to undertake tensor and matrix processing. This plays very much to the strengths of most nature-inspired algorithms. It is likely that the field will experience further attention and the algorithms experience increased successes in coming years, and the first steps of this can already be seen to be taking place.

2 Classification of bio-inspired algorithms

There are many bio-inspired algorithms proposed, designed and developed based on various bio-inspirations. There are limited literatures being focused on the classification. The two most widely accepted categories are evolutional based algorithms and swarm based algorithms, inspired by the natural evolution and animals' collective behaviour, respectively. Based on that understanding, Binitha and Sathya classified the bio-inspired algorithms into three categories: evolutional-based algorithms, smarm-based algorithms and ecology-based algorithms[8]. Though these three categories cover the majority of the existing bio-inspired algorithms, there are some new algorithms that cannot be grouped into those three categories. So, in this paper, the bioinspired algorithms have been classified into four categories. They are evolutionary[11,12], swarm intelligence[13,14], ecologically based[5,16] algorithms and multi-objective[17–19]

variants of all these algorithms. Evolutionary algorithms also encompass artificial neural networks as the closest related category, despite artificial neural networks differing in some ways[20] from other evolutionary algorithms.

The classification of algorithms in this paper is based upon the source of algorithms biological inspiration. This classification method has the benefit that explanations of the functionality of the algorithms remains related and broadly consistent in terms of terminology used within a class of algorithms. The classifications used can be seen in Figure 1.

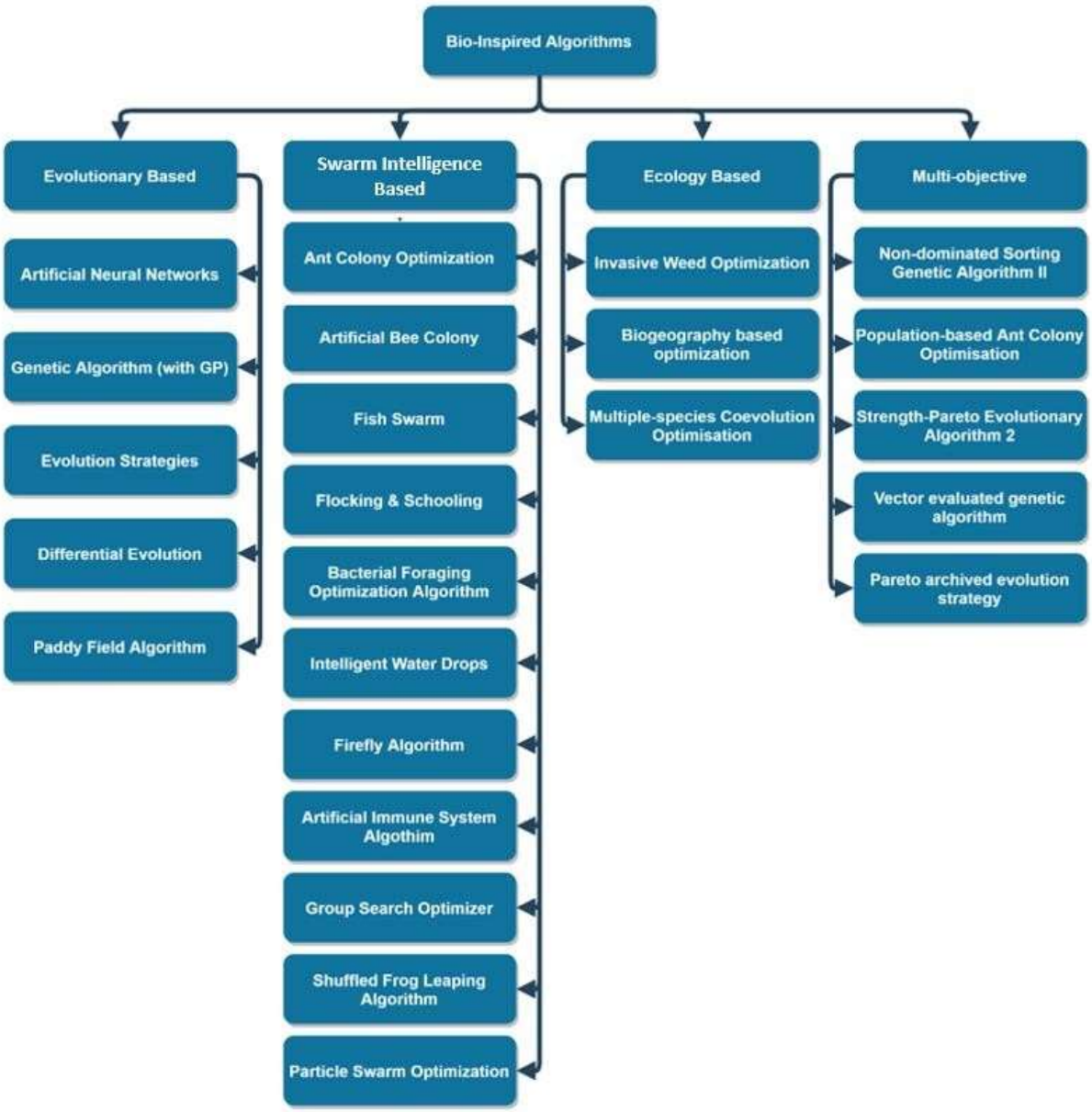

Figure 1 - Classification of Algorithms.

3 Evolutionary based bio-inspired algorithms

3.1 Artificial neural network and its application

3.1.1 Introduction to artificial neural networks

Artificial Neural Networks are a subset of machine learning, which attempt to model the biological brains structure and utilise various different mechanisms for learning, most commonly gradient descent based methods such as back-propagation [21,22]. They have the benefit of being very widely applicable and being proven as universal approximators [23,24] when structured as feed-forward networks. In early artificial neural network research, there was a focus on simple networks with a single hidden layer, for more modern applications specialised layers have been developed and increasingly deeper networks are being utilised. This is powered by advancements in computer hardware, in particular more common usage of graphics processing power for general computations, and the improvements in the software tools available, such as the tensor flow framework[25]. It is now common for neural networks to have several layers, depending on the intended application, see Figure 2 for an example of a feed forward neural network structure.

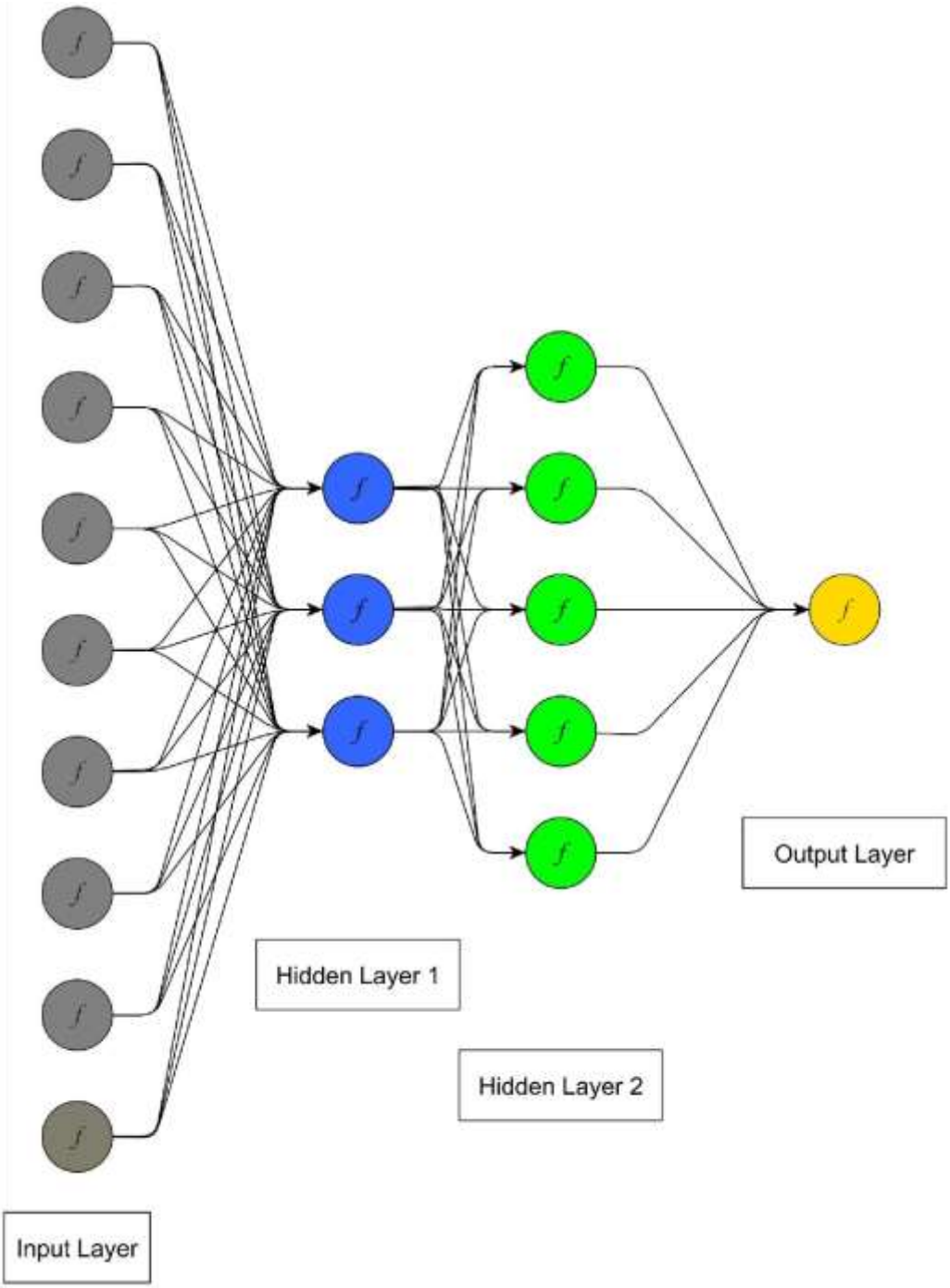

Figure 2 - An example feed forward artificial neural network structure.

3.1.2 Theory of artificial neural networks

Artificial neural networks or their precursors have been around since the 1940's when McCulloch and Pitts[26]published a paper on a mathematical model intended to emulate the function of a biological neuron. This demonstrated for the first time that networks of modelled neurons could perform computation.

Research continued, building on these ideas to develop the perceptron model[27] and learning rules to go along with this model. The downside of the

perceptron model and learning rule was a lack of a method to train a layered network[27], meaning that the network could not effectively classify non-linear problems. In the 1980's research in this area increased once more, particularly with the development and popularisation of back-propagation[21,22] as a learning mechanic for multi-layer networks which could solve non-linear problems.

More recently, computer hardware has progressed dramatically, allowing specialised deep neural networks to be trained in a reasonable amount of time and achieve impressive results

[4,28].

3.1.3 Artificial neural network structures

An artificial neural network is essentially a vectorised information processing algorithm. Due to their characteristics, which include good fault tolerance, generalisability and there are many different kinds of neural network algorithm[29] which are optimised for different applications.

Multilayer perceptron (MLP) networks[29–31] are one common approach, where perceptrons are layered input, hidden and output layers. The neurons of each layer in a MLP are connected to the neurons of the following layer but there is no interconnectivity at each layer. The data passes through the network and is processed in an easily visualised linear fashion as a result of this. This structure of network is often trained using back-propagation approaches[29,32].

Another structure is the radial basis function (RBF) network[30] which uses the RBF as its activation function. RBF networks are, similarly to MLP networks, structures in terms of input, hidden and output layers, where the input layer passes data to the hidden layer, then the hidden to the output. The structure differs from MLP networks in that the RBF is used as the activation function on the hidden layer, and the output layer node stake a weighted sum of activations from the hidden layer to produce its outputs, which (when used for classification) will often represent the likelihood of the data belonging to each category represented[29].

For applications where different inputs are not independent, recurrent structures such as the long short-term memory (LSTM) network are very popular[33,34]. These networks are based on a recurrent structure (where data is fed through the network back to previous layers in some form). This recurrent structure allows the network to track several steps in a computation, making it possible to classify or predict in cases where different samples to the network are not independent (i.e. predicting the next step in a time-series model, or the next word in a sentence).

In the field of image classification and processing[4,28] convolutional neural networks (CNN)[28] have become exceptionally powerful at image related tasks[35]. These CNN's consist of several specialised layers, applying convolutions and max-pooling operations to a loaded image, along with layers of more usual neurons, in this architecture referred to as "fully-connected layers". These networks perform at a very high level classifying spatially organised data such as images[35].

3.1.4 Applications of artificial neural networks

Artificial neural networks are very broadly applicable to a large range of problem domains, seeing particular success recently in image recognition and processing [28,35], and time-series analysis[33,34].

Impressively Arulkumaran et al.[33] managed to train a neural network based model to play real-time strategy games to a level where expert human opponents can be beaten. Neural networks have also seen success being used as meta-models within other optimization algorithms[17,36], as well as in the fields of natural language processing[37,38] and the construction of 3 dimensional scenes from collections of photographs[39,40].

3.1.5 Problems with artificial neural networks

Artificial neural networks are seeing a considerable amount of hype recently, and whilst they can solve a very wide-range of machine-learning problems, they may not always be the most efficient way to solve a given problem. Additionally, artificial neural network outputs are hard to analyse and identify the reason for a particular output[10] compared to other machine learning approaches. Finally, artificial neural

networks can often take a large amount of data to train well, which often must be cleaned and processed beforehand[28].

3.1.6 Further research

Recent efforts by various researchers in the field [10,28,33–35] show a very large amount of promise for artificial neural networks and their applications within various fields. They are swiftly becoming a major focus of research in machine learning techniques, due to their flexibility and the potential for very good performance to be achieved.

3.2 Genetic algorithm (with GP) and its applications

3.2.1 Introduction to genetic algorithms

Genetic algorithms are based loosely upon natural selection. The basis is that eventually organisms with useful traits will either supplant or exist alongside pre-existing organisms. These useful elements are initially produced by a mutation of existing elements, before being passed down parent to child. Genetic algorithms model this, usually by treating solutions to the problem as a "genome" which can then be mutated and recombined. Genetic algorithms are immensely popular optimization algorithms due to their suitability for non-linear, non-convex, multi-modal and discrete problems with which traditional gradient descent derived algorithms may perform poorly in comparison[12].

3.2.2 Theory of genetic algorithms and genetic programming

Genetic algorithms began to gain popularity as an effective and efficient optimization method in 1975. This was due to both the publication of "Adaptation in Natural and Artificial Systems"[41] and the thesis entitled "An analysis of the behaviour of a class of genetic adaptive systems" [42].Holland[41] presented the concept of adaptive algorithms utilising the concepts of mutation, selection and crossover, and De Jong[42] showed that genetic algorithms could perform exceptionally well on discontinuous and noisy data that is challenging for many other optimization techniques.

The process of a genetic algorithm can be separated into four distinct sub-processes: generation, selection, crossover and mutation. Generation involves building an initial population of potential solutions either by random creation or some other method. Selection is where the population is evaluated, and then a subset of that population is selected by one of many possible selection algorithms for the generation of a child population via the next stages of crossover and mutation. Crossover is the process of generating new chromosomes by combining aspects from previous solutions chosen by the selection algorithm via one of several possible crossover algorithms (single-point, multi-point, uniform, partially mapped crossover, etc.) in the hope of producing a "child" chromosome more fit than either of its "parent" chromosomes. Mutation involves introducing a chance of making random changes to the chromosomes, which helps to prevent premature convergence and allow a fuller exploration of the search space by including genes that were not present in the initial random population.

Genetic programming is strongly related to genetic algorithms and is the process of applying similar principles to program generation[43] with programs represented as a tree of operators and operands which can be recombined and mutated in order to optimise.

3.2.3 Genetic algorithms

Single-objective genetic algorithms all follow a similar broad approach but differ in terms of the approach to the individual steps of generation, selection, crossover, and mutation.

Generation is often entirely random, but can be either directed, or initialised by a previous run of the algorithm. This could allow for either a continuation of a previous optimization run, or for a partial optimization on a more general approach to be used to seed a new optimization with potentially good solutions[44]. This could also bias the algorithm towards local optima by not allowing a broader search of the problem-space. It is also possible to partially seed the initial population, to try to avoid this potential issue[44].

Selection is the process of identifying which solutions (genomes) will be used to generate new solutions for the next generation. Therefore, the method by which this is done has a significant impact on the convergence of the algorithm and how fast or slow this is. Most modern implementations use either truncation or tournament selection[12] as these are scaling invariant and inherently elitist, which has been shown[11] to enhance the effectiveness of the genetic algorithm.

3.2.4 Applications of genetic algorithms

Genetic algorithms have been applied to a huge number of real world and research problems, across a broad range of areas including engineering problems[10,12], classic optimization problems [44], and protein folding[45]. Due to their parallel nature, they offer a large variety of options for acceleration across multiple cores or machines and can take advantage of large amounts of compute power where it is available, as has been shown in recent research papers published[46,47]. Additionally, genetic algorithms can to some extent take advantage of the parallelisation capabilities of graphics processing units[47].

3.2.5 Problems with genetic algorithms

Genetic algorithms have no guarantee of finding the global optima with any combination of hyper-parameters. They can take a long time to converge when run with insufficient computing power as it is often necessary to have a good sized population and a large number of generations to achieve a good result[44]. The development of the fitness function for a genetic algorithm is also crucial as this guides the entire optimization process and if done without care, the algorithm could generate inefficient or infeasible solutions[10].

3.2.6 Further research

Some of the most promising research within the field of genetic algorithms involves their application to multi-objective problems, and the use of heuristic meta-models within the genetic algorithm as a means to increase speed of convergence without affecting the accuracy and the usefulness of the final solution[10].

3.3 Evolution strategies (ES) and its application

3.3.1 Introduction to ES

Evolution strategies come from the more general field of "Evolutionary Algorithms" which also encompasses Genetic Algorithms and Genetic Programming. Evolution strategies use mutation, recombination, and selection operators applied to a population of individuals. Each individual represents a single point in the search space being explored. The key distinction between genetic algorithms and evolutionary strategies is that evolutionary strategies evolve both characteristics and parameters[48].

3.3.2 Theory of ES

The theory of evolution strategies dates back to the 1960's at the Technical University of Berlin, where Rechenberg and Schwefel performed the initial research into this area[49-52]. At that time, the proposed application was to the automatic design and analysis of engineering experiments[48], rather than optimization of variables against objective functions. Flexible objects and systems were adjusted into their optimal states in noisy environments (i.e. wind tunnels). They were effective at this task, and were subsequently tested against various other similar tasks[53,54].

These experiments were undertaken using two main rules: firstly, all variables were randomly changed at a time and to a small degree; secondly if the new variable values did not reduce the optimality of the altered device, keep them, otherwise revert.

These rules can be recognised as an evolutionary algorithm, with a population of one original solution, and one child solution created by mutation, and an elitism mechanism in place so that the better of the two is retained and a new child generated from it.

3.3.3 Evolutionary strategy algorithms

The previously described approach is sometimes referred to as a two-membered or 1+1 evolutionary strategy (es)[8,55].

To help avoid local optima Rechenberg[49] proposed a multi-membered(μ+1) version of the

algorithm where multiple parents can participate in the generation of a child[55,56]. These were later developed into algorithms known as (μ+λ)-ES, as 'μ' parents produce 'λ' children[56]. The selection operator operates on the combined set of parents/children and the λ worst population members will be discarded at each iteration, so parents will survive in the population until they are superseded by superior children.

In order to avoid a number of issues inherent in the (μ+λ)-ES approach[56], Schwefel[57,58] investigated a variant of the (μ+λ)-ES where only the off-spring would under-go selection for parenthood and the parents would be therefore discarded from the population, known as (μ,λ)ES. Because of the limited life-span, inappropriate internal parameters can be forgotten. This can result in phases of recession but does avoid long stagnation due to less optimal parameters. This approach relies on a birth surplus (>) in order to keep the population size steady[48].

3.3.4 Applications of evolutionary strategies

Although they have not quite seen the explosive popularity of genetic algorithms, evolutionary strategies have seen many and varied applications from their initial application to hot water flashing nozzle optimization[53,54]. Throughout the 1970's and 80's they continue to be applied to similar continuous engineering problems[56-58].

ES continue to be used on optimization problems to this day, such as training artificial neural networks to predict host CPU utilization in cloud infrastructure[5],optimization of engineering and construction related problems[59], and process modelling and optimization[60].

3.3.5 Problems with evolutionary strategies

Evolutionary strategies have not seen the uptake that some other fields of optimization algorithm have done and are thus less well researched than some alternatives. Whilst they are generally found to be robust where they have been applied[5,57,58] it is likely that for a lot of problems researchers will be dealing with issues of application to their particular problem as well as investigating the performance of the optimization.

3.3.6 Further research

There are many interesting areas where evolutionary strategies could be potentially improved, largely drawing from other evolutionary algorithms and how they have been improved. Microsoft Research[61] has demonstrated a novel iterative procedure to optimise a surrogate objective function, which allows sample data to be reused over multiple epochs. They have demonstrated this approach produces better performance than standard ES in various test environments. In Salimans et al.[62]the authors investigate the use of an evolutionary strategy for the training of reinforcement learning algorithms. In this study the inherent parallelism of evolutionary strategies allows it to achieve competitive results with stochastic gradient descent and back-propagation methods more commonly used in this field. Both are promising areas which would warrant further investigation.

3.4 Other evolutionary based bio-inspired algorithms

There are many other evolutionary based bio-inspired algorithms. Due to the limitation of the paper length, some of the main algorithms are listed in Table 1.

Table 1 - Further evolutionary inspired algorithms.

Name of Algorithms	Explanation and references
Differential Evolution (DE)	A further evolutionary inspired algorithm, similar in principle to genetic algorithms and evolutionary strategies[106,107].
Paddy Field Algorithm (PFA)	The paddy field algorithm is inspired by the spread of rice in paddy fields[108].

4 Swarm intelligence based bio-inspired algorithms

4.1 Ant colony optimization (ACO) and its applications

4.1.1 Introduction to ACO

Ant colony optimization algorithms are based on the way an ant colony will send out workers to hunt randomly for food, who leave pheromone trails behind them. Where worker ants find food, they will then proceed back to the nest reinforcing their pheromone trail. If there are two paths to the food, initially ants will randomly pick a path, but over time the shorter path will be more often reinforced. Other ants will be attracted to follow stronger pheromone trail paths, and by this method the colony will swiftly find food sources, and get a large number of workers to them by the most efficient paths[63].

4.1.2 Theory of ACO

Ant colony optimization as an algorithm generally works via several artificial ants who will at each iteration plot a path through a problem space, composed of nodes (solutions) and edges between nodes, towards the objective. The artificial ants cannot revisit nodes they have previously visited, and each step of this path is selected via a stochastic mechanism that is biased by a "pheromone" variable associated with each edge that the artificial ants can read[63-65]. Several successfully applied ant colony optimization algorithms have been developed based on this general approach. Compared to many of the other bio-inspired algorithms covered within this review, ant-colony optimization approaches are a recent development in the field, the first having been created in the early nineteen nineties[66]. The general approach was then developed into a full meta-heuristic and progressed throughout the nineties and into the present decade

[63,65–67].

4.1.3 ACO algorithms

There are many ant-colony optimization algorithms based around the previously described principles, such as best-worst ant system (BWAS)[64]. In the best-worst ant colony optimization algorithm the best and worst performing solutions are both considered in order to reinforce edges contained in good solutions, and penalise edges contained in poor solutions, thus driving convergence. Additionally, this approach includes a restart mechanism for if the optimization gets stuck, and a mutation system which diversifies the solutions generated by adding randomness in such a way that different paths are encouraged at the later stages of the search process[64]. There are too many variants of ant colony optimization for everyone to be covered in this paper, but good overviews of existing algorithms can be found in review papers specific to ant colony optimization[63,68]. The nature of ant colony optimization means that it can potentially be used with many other search algorithms, which are appropriate to the problem it is being applied to, as well as having many ways in which its hyper-parameters can be managed as the algorithm iterates.

4.1.4 Applications of ACO

As ant colony optimization is employed as a meta-heuristic Ant colony optimization has been applied in several important areas. The first problem the algorithm was tested upon was the travelling salesman problem, which is known to be NP-hard and extremely challenging depending on the number of cities included[63,66]. Ant colony optimization has since then been applied to engineering problems[10,63], vehicle routing problems, machine learning related optimization, and bioinformatics problems, amongst others[63]. It has proven to be an effective and interesting meta-heuristic algorithm.

4.1.5 Problems with ACO

Ant colony optimization suffers from many of the same issues as genetic algorithms. Its effectiveness is there, when enough computing power or time is available to be utilised in solving a given problem. A sufficient number of iterations need to be completed to achieve convergence and the algorithm can be extremely complex to execute, especially when the underlying search strategy is a more complex one[63].

4.1.6 Further research

Ant colony optimization is very promising when applied to single objective optimization although it has been somewhat overtaken in application by more common genetic algorithms or simpler search strategies where compute power or time is limited. There have been some efforts to modify ant colony optimization to multi-objective paradigm and this is an interesting avenue that is perhaps deserving of further study[69].

4.2 Artificial bee colony (ABC)algorithm and its application

4.2.5 Introduction to ABC

The artificial bee colony algorithm is based loosely on the behaviour of bees in a hive hunting for food sources and communicating together. In the model, artificial bees are grouped into three categories, employed, onlookers, and scouts. Potential solutions to the optimization problem are “food sources” with the fitness of the solution modelled as “nectar quantity”. Scouts identify new food sources. Employed bees evaluate the quality of existing food sources, and onlookers get the information from employed bees about

food source quality of the source the employed bee checked. Onlookers then evaluate the food source against neighbours so that solutions can be discarded and replaced with scouted solutions[2,13].

4.2.6 Theory of ABC

The algorithm works broadly as follows, initially a food source (solution) is identified for each employed bee randomly. Each employed bee evaluates this food source and modifies it. If their new food source is more optimal than the previous, they forget the previous and remember the new. They then communicate this information to onlooker bees. Onlooker bees then choose a food source depending on fitness communicated by all employed bees and produce a modification. Again, if the modification is better, this replaces the original food source.

Abandoned sources are replaced by new sources by scouts[2,13].

4.2.7 ABC algorithms

As well as the initial artificial bee colony algorithm, several algorithms have been developed based around the same concepts, inspired by honey-bee foraging behaviour. For example the bees algorithm[70,71] has been developed which follows a very similar flow, but utilises a different mechanism for neighbourhood search, and a different simulation approach for inter-bee communication within the algorithm[71]. More recently, Khan &Maiti[72] proposed a version of the artificial bee colony algorithm modified to use swap-sequences in order to perform well on travelling salesman type problems, demonstrating performance that is roughly on-par with other approaches to solving problems in this domain.

4.2.8 Applications of ABC

Artificial bee colony algorithm has been shown[13,71] to perform on artificial test-cases with comparable performance to other state-of-the-art algorithms. On the travelling salesman benchmark problem it has been shown that with modification it can perform well[72]. Algorithms based on the artificial bee colony approach have also been applied with promising results to the problem of route optimization for network communications[73] as well as to colour image segmentation[74].

4.2.9 Problems with ABC

One of the major issues with the artificial bee colony algorithm is the lack of real-world test-cases where it has been applied and can demonstrate promising results or improvements in performance over alternative algorithms. Although it has been applied in a few areas[13,73,74] it does not have the same breadth of application as some other algorithms in this field. This means that researchers may spend some time dealing with issues of application to their specific problem, however it could also mean that the chance of getting novel results is heightened if this algorithm has not been applied to that problem before. Additionally, and in common with many swarm intelligence algorithms, computational time can be a limiting factor on its application.

4.2.10 Further research

Artificial bee colony algorithms are a promising area of research that has surfaced relatively recently in terms of swarm intelligence-based algorithms. Because of this there is considerable scope further research in various applications of these algorithms. One interesting field of application is in using artificial bee colony algorithms, with modification, for the purposes of data clustering[75,76], where they show promising abilities.

4.3 Fish swarm algorithm (FSA)and its application

4.3.9 Introduction to FSA

The artificial fish swarm algorithm is a swarm intelligence algorithm inspired by the way that fish will swarm together towards food sources. Each fish individually searches for the most food rich areas to move towards. Whilst doing this they also want to avoid overcrowded areas because the competition for food is too high. They also want to follow groupings which are not overcrowded, as these fish may have found a food-rich area they are grouping in. The combination of these factors results in a swarming behaviour which converges on the most food rich areas whilst also exploring around for other food rich areas[77].

4.3.10 Theory of FSA

The artificial fish swarm algorithm is based on having a number of artificial fish. These artificial fish are distributed throughout the problem-space, initially randomly. They then follow a number of specific

behaviours, “following”, “preying”, “swarming”, and “moving”. The fish will iterate through these behaviours starting with “following”, and proceeding in order towards “moving” every time an improved position is not found. At each behaviour, if that behaviour produces an improvement in position, that improvement is taken. If that behaviour does not produce an improvement in position after a given number of attempts, then the “movement” behaviour will be executed[14,78].

Whilst in the “following” behaviour, the fish will check its range of vision for the other fish with the best food availability (objective function evaluation). When it finds that fish, if it has a higher food availability than the current fish, and a smaller proportion of the total number of fish nearby to it, the current fish will move towards that fish. Otherwise, the current fish will transition into the “searching” behaviour[14,78].

The “preying” behaviour involves a fish identifying the centre position of nearby companion fish within its visual range. Once it has found that point, if it has a higher food availability than the current fish and that point is not overcrowded, the fish will move towards this point[14,78].

Whilst engaged in the “searching” behaviour, the fish will select a prospective new state within its range of vision, evaluate it, and then if it is better move towards that state. If it reaches a limit on number of evaluations to make, it will just move randomly[14,78].

In the “movement” behaviour, the fish will just select a random direction and move in this direction[14,78].

4.3.3 FSA algorithms

Following the same basic approach, there are a large number of artificial fish swarm algorithms[78,79].

These modifications of the original algorithm, which performs well as standard[79], come generally with trade-offs to be considered. For example, the “preying” behaviour can be modified to guide the fish towards the nearest well-performing position and the global best performing so far position[79] at the expense of added complexity to (and therefore increasing the computational load of) the algorithm. Other attempts look at building a form of gradient descent into the swarming behaviour, introducing crossover into the algorithm or introducing random restarts (with “leaping” behaviours) amongst other approaches[79].

4.3.4 Applications of FSA

Artificial fish swarm algorithm has been broadly applied across many fields. In structural damage detection[78] a modification of the artificial fish swarm algorithm has been shown to be able to locate structural damage effectively as well as identify the severity of the damage. Another modification of the fish swarm algorithm has been applied to reliability redundancy problems showing performance which is good in terms of computational accuracy and efficiency, demonstrating the ability to find solutions that are of a quality on-par with other heuristic algorithms[80]. Generally speaking artificial fish swarm algorithm derivatives show performance on-par with or improved upon industry standard optimization approaches, and computational impact similarly improved or comparable[14,78,79].

4.3.5 Problems with FSA

Similarly with other bio-inspired algorithms, the largest issue with application tends to come in terms of the computational time and effort required to produce high-quality solutions[10,79]. This is improved over an exhaustive search but remains an issue for adoption and thus a key competitive element when one algorithm in this field is evaluated against another.

4.3.6 Further research

A significant amount of research effort is being invested in artificial fish swarm algorithms recently. The algorithm performs well or on par compared to industry standard algorithms and fish behaviours can be swapped in and out or added to add complexity to the algorithm[78,80]. In common with other bio-inspired algorithms, the possibility of meta-heuristics in combination with optimization objective functions could allow for improved performance [10]. The addition of meta-heuristic objective functions needs to be undertaken carefully in order to avoid any loss of accuracy in the final solutions that could preclude the developed algorithm from, i.e. engineering applications where accuracy is crucial.

4.4 Others warm intelligence based bio-inspired algorithms

Some other notable swarm intelligence algorithms are listed in Table 2.

Table 2– Further swarm intelligence algorithms.

Name of Algorithms	Explanation and references
Flocking and Schooling in Birds and Its Application	Algorithms inspired by the flocking behaviour of birds on migratory flights, hunting for food, nesting areas or mates such as the bird mating optimizer[109].
Bacterial Foraging Optimization Algorithm (BFOA) and Its Application	An algorithm inspired by the foraging behaviours of bacteria [3].
Intelligent Water Drops Algorithm (IWD) and Its Application	An algorithm inspired by the manner in which water moves in seas, rivers and lakes, affected by the ease of the path offered to it[110,111].
Firefly Algorithm (FA) and Its Application	Algorithms based on the swarming behaviour of fireflies[112,113].
Artificial Immune System Algorithm (AISA) and Its application	Algorithms based on the manner in which immune systems respond to perceived threats to the system[114,115].
Group Search Optimizer (GSO) and Its Application	Group search optimizer is based on the searching behaviour of animals[116,117].
Shuffled Frog Leaping Algorithm (SFLA) and Its Application	Shuffled frog leaping algorithm is inspired by cultural transfer within a population, in this case of artificial frogs[118,119].
Particle Swarm Optimization (PSO) and Its Application	Particle swarm optimisation builds on a similar approach to genetic algorithms, although rather than moving individuals within the problem space, swarms are moved[120,121].

5 Ecology based bio-inspired algorithms

5.2 Invasive weed optimization (IWO) and its application

5.1.1 Introduction to IWO

IWO is based on the behaviour in nature of weeds spreading across a terrain or area. A number of weeds (problem solutions) are spread over the search area, much like weeds in nature spreading seeds and the offspring thriving or dying where they fall depending on the suitability of that area[15].

5.1.2 Theory of IWO

In IWO each weed represents a solution to the problem being optimised, and a population of weeds refers to the set of all weeds. The algorithm follows a series of steps iteratively. Initially, the population needs to be initialised, and weeds will be initialised within the search space with normally distributed random numbers with a mean of zero. Next, the higher a weeds fitness, the more seeds it will produce. This is governed by an equation[15] which takes into account current weed fitness, maximum and minimum fitness

in the current population, and a minimum and maximum number of weeds to be generated. The generated seeds will be normally distributed over the search space remaining close to the parent weed. When the number of weeds reaches a pre-set maximum number, weed production occurs as normal, but all weeds within the population are then ranked and weaker performing population members culled from the population[15].

5.1.3 IWO algorithms

Basak et al.[81]developed a differential invasive weed optimization algorithm (DIWO) which they applied to benchmark problems. This involved combining the difference vector-based mutation scheme of differential evolution with the invasive weed optimization algorithm. Comparing this combined algorithm to benchmark industry standard algorithms, demonstrated very strong performance of the algorithm, improving on both standard invasive weed optimization algorithm and the differential evolution algorithm for this specific problem. Rani et al.[82] developed a multi-objective specialisation of invasive weed optimization in order to apply their problem to optimal network reconfiguration. This involved using a non-domination sorting approach to model Pareto optimality between four objectives. A large number of other interesting customisations of IWO are present within literature[83-85].

5.1.4 Applications of IWO

Zhou et al.[15] applied the invasive weed optimization algorithm to attempt to solve the no-idle flow shop scheduling problem, showing a marked improvement over other industry standard algorithms against which it was benchmarked. Zhao et al.[85] have applied invasive weed optimization algorithm to the capacitated vehicle routing problem, with an addition of mutation and crossover to the algorithm to increase diversity in the population generated. Compared to other benchmark results the developed algorithm performs well for this problem. Invasive weed optimization algorithm and variations of it have also been applied to optimising heterogeneous wireless network issues[86] and economic dispatch of power systems[83]amongst others, demonstrating strong performances throughout.

5.1.5 Problems with IWO

IWO is a widely applicable and powerful meta-heuristic algorithm, which has been customised successfully for many different problem spaces[15,83-86]. The algorithm requires that hyper- parameters are initialised to appropriate values before the iterative process begins, and if they are not it is possible that the algorithm will converge to a local minimum or a never converge. Additionally, meta-heuristic algorithms such as IWO generally suffer from issues with the number of objective function calls taking place due to the intractable nature of the problem spaces to which these algorithms are generally applied, and IWO is no exception.

5.1.6 Further research

Multi-objective IWO is a promising space and the simplicity of the algorithm means that modifying the algorithm to work on optimising for Pareto optimality should result in simpler algorithms for this kind of optimization. Diversity could be preserved in these algorithms by adopting an approach similar to that used in NSGA-II[19]or other industry standard multi objective algorithms. The addition of meta-models could ease issues with objective function cost and further research into incorporating these within invasive weed optimization would be valuable.

5.2 Biogeography-based optimization (BBO) and its application

5.2.2 Introduction to BBO

Biogeography based optimization is inspired by the distribution of living things within a geographical area. Living things migrate between areas, new species arise to fill niches and habitats where resources are available but unexploited, and species become extinct when another species arises that can out-compete them in their habitat. The most optimal habitats have the largest range of species living within them. The mathematics that describe this process can be used as a basis for an optimization algorithm, which shares many features with other bio-inspired algorithms, and is applicable to many similar problems[16].

5.2.3 Theory of BBO

Biogeography based optimization begins with the initialisation of a population of solutions which are allowed and possible within the problem space. A number of hyperparameters are also defined at this stage

[16]. In BBO a "habitat" represents a solution to the problem being optimised. A number of these habitats are then initialised. For each habitat, the fitness (habitat suitability index) is calculated, and from that the number of species, immigration rate, and emigration rate are calculated. Immigration, emigration and mutation rates are then used to probabilistically modify each habitat. If an element of elitism is included in the implementation, then elite habitats will be exempt from modification. The algorithm then cycles back to calculating the fitness for each habitat and continues the cycle[16].

5.2.4 BBO Algorithms

Biogeography based algorithms have been customised to specifically work with many different problems. Goudos et al.[87]have adopted BBO algorithms and modified for a multi objective optimization approach for indoor wireless network planning. In order to accomplish this, they have formulated the indoor wireless network planning as a multi-objective optimization problem, i.e. to explore and find the Pareto front of the problem space. Lin[88]has combined biogeography based algorithms with heuristic approaches to construct hybrid algorithms for solving the job-shop scheduling problem. These promising approaches are only two of many[89-91] interesting modifications and customisations of BBO.

5.2.5 Applications of BBO

BBO has been applied to many problems, including outdoor wireless planning[87], scheduling problems[88], parameter estimation for solar and fuel cells[92], parameter estimation more generally for chaotic systems[91], emissions dispatch from fossil fuel power generation[90] and cloud computing[89].Generally, the algorithm is effective wherever applied and has been customisable to the necessary extent for broad application. It is a relatively recent algorithm and therefore the broadness of application is not quite as much as for mature algorithms in the field such as genetic algorithms, but it is impressive given its current lifespan.

5.2.6 Problems with BBO

Biogeography based algorithms suffer as all newer algorithms do from less depth of literature available in the field and less ready-made software packages to aid in the application of the algorithm. In general, with

population-based bio-inspired algorithms more generally, it can suffer from the requirement for a large number of evaluations of the objective functions[16]. BBO has several hyper-parameters which need tuning, which is a common trait in bio-inspired algorithms but does add complexity to the application of the algorithms.

5.2.7 Further research

Some of the issues listed above are already being looked into, the number of objective function evaluations could potentially be addressed by optimising the algorithm (to reduce the number of evaluations necessary),meta-modelling the objectives (to speed up some of the evaluations) or optimising the objective functions to increase performance of all of these. In practice, two or more of these are usually employed together by attempting to generate efficient

objective functions and tweaking hyper-parameters[87-90,92]. More applications of the algorithm generally would also be good to see, as this will inevitably lead to specific problems being encountered and hopefully overcome.

5.2.8 Further ecology based bio-inspired algorithms

Further ecology based bio-inspired algorithms can be found in Table 3.

Table 3 - Further ecology based bio-inspired algorithms.

Name of Algorithms	Explanation and references
Multi-species coevolution optimisation (PS^2O)	Algorithm based upon multi-species co-evolution within an environment represented by the problem-space[122].

6 Multi-objective bio-inspired algorithms

6.1 Non-dominated sorting genetic algorithm II (NGSA-II)

6.1.2 Introduction to NSGA-II

NSGA-I[19] is a multi-objective optimization algorithm[18] which generates an approximated Pareto-front based on a population of possible solutions to the optimization problem evaluated against a number of objectives in the range 1 to ‘n’ which are normally marked for maximisation or minimisation. The Pareto

front is a set of solutions where for each solution, the performance can only be improved in one of the objectives by reducing it in another objective. The Pareto front therefore dominates the hyper-volume of all other possible solutions within the solution space.

6.1.3 Theory of NSGA-II

NSGA-II achieves the development of an approximate Pareto front via an iterative approach[19]. First a population of a given size n is created, either randomly, or via some method for pre-seeding with good candidates. Then for each solution p, two values are calculated. The first is the domination count, which is the number of solutions which dominate p. The second is the set of all solutions that p dominates. All the solutions in the first non-dominated rank will therefore have a domination count of zero. For each of these solutions, each member of the dominated set is visited, and its dominating count reduced by one. If the dominating count of those sets is thus reduced to zero, it is added to the second non-dominated rank. This process continues until all solutions have been assigned to a rank.

Each solution is then assigned a fitness value equal to its rank (with 1 being the best and so on, assuming minimisation). Tournament selection, crossover and mutation operators can then be used to create a child population of size n. This created child population is then merged with the initially created parent population, to form a population of size 2n. This population can then be assigned non-dominating ranks in the same manner as previously described and ordered by these ranks.

Each solution will also have a "crowding distance" calculated[19], via a function which takes into account how close the nearest other solutions within the same rank are. For this measure, larger is better (more varied from other solutions within the same rank) and the outlier solutions are always assigned a "crowding distance" of infinity. Within their ranks, the individuals are then sorted by this crowding distance also, in reverse order.

The first 'n' individuals in this sorted population are then selected to form a new population of size 'n' consisting of the best ranks with a total membership <n, plus the most varied members of a final rank in order to make up a size of exactly n.

Once this is complete, binary tournament selection is again used, this time using the crowding comparison operator which takes into account both rank, and crowding distance. This allows the selection of parents of a new population of size n, at which point we will continue to iterate through combining the populations, ranking and scoring, then reducing to size n and cycling again, until a stopping condition is met.

6.1.3 NSGA-II algorithms

A number of algorithms have used NSGA-II as a base to build upon, including MOGAANN[36], LEMMO[93] and LEMMO-ANN[10,17]. Many customisations of NSGA-II based algorithms are based around meta-heuristics, niching, or other approaches to reduce the number of objective function evaluations[10,17,36,93,94]. Due to the nature of the algorithm, an extremely high number of objective function evaluations is often required, and the algorithm is often applied to complex real-world problems with objective functions that can involve extremely long run-times[10,17,36,93].

In keeping with other genetic algorithm approaches, NSGA-II is potentially very parallelisable and another approach that has been investigated is parallel running of objective function evaluations across multiple machines[95,96].

6.1.4 Applications of NSGA-II

NSGA-II has been applied extensively throughout industry, particularly in the engineering sector[10,17,36,95,96] where intractable multi-objective problems to which the algorithm is applicable abound. In the LEMMO algorithm a decision tree classifier is used within the NSGA-II algorithm as a feature identifier for characteristics of solutions which perform well or poorly.

At a high level the LEMMO application of NSGA-II works by performing a variable number of iterations of a standard NSGA-II[19] algorithm and storing data on which of the generated solutions are “good” solutions and which are “poor” solutions. It then uses this data to train a machine-learning algorithm to distinguish between good and bad functions and uses the outcome of this training in some fashion to generate new solutions, which, according to the trained machine-learning model, are “good” solutions. These are then integrated with the main population and the algorithm continues with further iterations

based on pure NSGA-II and further iterations of machine learning. The exact machine learning algorithm can be varied and the LEMMO approach has been shown to work well both with decision tree type algorithms [93,94,97] or with artificial neural networks[10,17]. Some further modifications are necessary to allow for artificial neural network training and a suitable method for generation of new solutions[10,17].

These algorithms using an NSGA-II based approached, combined with meta-heuristics in a LEMMO approach, have been applied to flood risk optimization[10], water distribution network optimization[17,93,94].

Other meta-heuristic based approaches have involved identifying the optimal placement for pressure sensors in a water distribution system, using a modified NSGA-II[19] approach and using artificial neural networks to estimate the model results[36], rather than classifying a solution as good or bad.

Outside of meta-heuristics, other approaches have involved parallelising the work-load for each iterations evaluation of objective functions, this approach has been applied with some success to real-options optimization for flood defence[95,96,98] but is reliant on having computing power available, or the ability to access cloud resources.

6.1.5 Problems with NSGA-II

A number of problems exist with NSGA-II, although they are generally common across most optimization algorithms within this multi-objective space. The first and most obvious is the potential runtime due to the number of objective function evaluations[10], this is the downside that the meta-heuristic modifications of the algorithm are intended to alleviate. A second downside is that with further insights into the data and the problem space, a more directed mutation approach could yield a significantly increased speed in convergence. Of course, this would also risk losing the benefit that the algorithm may identify solutions that would be nonclear to an analysis. The LEMMO approach is partly trying to alleviate this downside, by directing the generation of new solutions, using machine-learning approaches. A third downside is that an NSGA-II algorithm produces a Pareto[99] front, which is a range of solutions with varying scores in different objectives, that dominate all other solutions. A final solution needs to be selected from this front,

in order to be utilised, which can add complexity to the algorithm although it does allow for prioritisation of objectives, with the knowledge of what the trade-offs will be.

6.1.6 Further research

Research is already taking place, as mentioned above, into solving many of the problems associated with the NSGA-II algorithm by extending it or modifying the way individual steps are handled. A subsequent algorithm NSGA-III has been developed[100] but is largely focused at different types of problems, rather than being a direct replacement for/improvement on NSGA-II. Some of the more promising research is around meta-heuristics and parallelisation, as touched on in this paper, there is still considerable scope for further research here. Different machine learning techniques are suited for different kinds of problems, so implementing further modifications of LEMMO that use different machine learning approaches seems to have value. This isn't as simple as plugging in the new algorithms and testing, there is considerable extra work in integrating a new machine-learning technique.

6.2 Population-based ant-colony optimization

6.2.1 Introduction to population-based ant-colony optimization

Ant-colony optimization is a successful and useful single-objective bio-inspired algorithm [63–67]. It has therefore been widely modified to support multiple objectives and a pareto optimal approach. The primary method of applying ant-colony optimization to multi-objective optimization problems is referred to as P-ACO[101]and involves building upon the standard ACO process. It does this utilising matrices of pheromones, random weights, a concept of "lifespan" and a pheromone decoding scheme.

6.2.2 Theory of population-based ant-colony optimization

The process of ant colony optimization is split into two stages, construction and evaluation. These phases are managed by two data structures, a pheromone matrix and a solution archive. Initially, the solution archive will be empty, and initialised to minimum levels. After each iteration, a solution can be added to the solution archive. The pheromone matrix will then be updated to reflect this new solution, biased by specified values[101,102]. After a number of iterations, the maximum number of possible solutions will be

reached, at which stage every new solution replaces an existing solution, which then has its influence on the pheromone matrix removed. A number of strategies can be used to decide how to add and remove solutions from the solution archive[102,103]. The three main ones are as follows. An age-based strategy, where at each iteration the best solution found enters the solution archive, if the archive is full the new solution replaces the oldest solution within the archive. A quality-based strategy where solutions may only enter the solution archive if they are superior to all other solutions within the archive at the time. Finally, an elitist strategy which tracks the best solutions found within a run. Every time a solution is discovered that improves on the current best solution, the weights are updated based on this new solution[103].

6.2.3 Population-based ant-colony optimization algorithms

In population-based ant-colony optimization several different approaches are mentioned above, which can customise the algorithm to fit specific approaches. Additionally, the algorithm has been tweaked in many ways to fit specific problems. The optimization of research project portfolio's was one of the first applications of this algorithm[101] and involved an algorithm where feasibility and efficiency formed the objectives of the algorithm. Building on these early successes and more recently, attempts have been made to improve the application of P-ACO to multi-objective optimization[102].This is in an effort to apply these algorithms to civil-engineering based problems which are inherently multi-objective, and develop their applicability to such problems[102].

6.2.4 Applications of population-based ant-colony optimization algorithms

P-ACO has been applied across many domains. In the civil-engineering sector where large intractable problems are very common[102] such as the optimization of plane bar truss structure the algorithm saw promising results with some modification. In the original application of optimising research portfolio's, the algorithm out-performed alternative heuristic methods. The algorithm has been applied to single-objective problems such as the travelling salesman problem and the quadratic assignment problem with success also. In keeping with most other multi-objective optimization algorithms, it is possible to apply the technique to single objectives, it is just not often worth the additional complication of the algorithm.

6.2.5 Problems with population-based ant-colony optimization

Ant-colony optimization suffers from the complexity of the algorithm, which is a common theme of multi-objective optimization algorithms. This complexity means that simpler approaches are often preferred even when less accurate because they cost less in development time. Additionally, the algorithm is not as widely research as other multi-objective approaches, particularly genetic algorithm based multi-objective optimization approaches. This can add a further cost of research and development as there is less likely to be mostly developed solution you can tweak.

6.2.6 Further research

Further research in the area is currently largely around the improvement of the application of the algorithm to multi-objective optimization[102]. There is some interesting research taking place into optimising ant-colony optimization for running on massively parallel systems such as Apache Spark[104,105] which may also be applicable to multi-objective ant-colony optimization.

6.2 Further multi-objective bio-inspired algorithms

Further multi-objective Bio-inspired algorithms are shown in Table 4.

Table 4 - Further multi-objective bio-inspired algorithms

Name of Algorithms	Explanation and references
Strength-pareto evolutionary algorithm 2 (SPEA2)	This algorithm is based on the same non-domination pareto principle as NSGA-II using an evolutionary process. The fitness is calculated as a combination of the summed scores of the objectives (fitness), and the number of solutions dominated (strength)[123,124].

7 Conclusion

In conclusion ten bio-inspired algorithms have been reviewed in this paper. The first of these was artificial neural networks which attempt to mimic the learning mechanisms of the brain itself. Next genetic algorithms and genetic programming were discussed, as they are an extremely popular and well-known algorithm. Evolutionary strategies were then covered, particularly on the clear points where despite being an alternative evolutionary strategy they differ widely from genetic algorithms. Moving into swarm-intelligence based algorithms, ant colony optimization as a well-known and popular swarm intelligence

algorithm was described and the ingenious method of its functioning examined. The artificial bee colony with artificial bees communicating the location of optimal points in search space was then discussed, along with the fish swarm algorithm which models fish schools hunting food. Under ecology-based bio-inspired algorithms invasive weed optimization has been described with its effective approach to solution finding and biogeography-based optimization with the model of living creatures migrating between habitats of varying optimality. Finally, multi-objective optimization algorithms were explored, covering particularly NSGA-II with its non-dominated sorting approach to optimising to a Pareto front, and P-ACO and its specialisation of ant-colony optimization in a manner that allows for multi-objective optimization. In addition to these algorithms covered, a further thirteen algorithms have been briefly mentioned within the appropriate sections in order to direct readers towards literature covering these related algorithms. Hybridisation of nature inspired algorithms together in a similar way to that in which many algorithms combine solutions, appears to be a clear direction for iterative improvement of these algorithms and has been for some time. Additionally, utilising machine-learning models such as artificial neural networks to reduce objective function overhead whilst maintaining accuracy seems promising, although such solutions may well need to be bespoke for each kind of application problem.

References

[1] Zang H N, Zhang S J, Hapeshi K A. Review of Nature-Inspired Algorithms. Journal of Bionic Engineering, 2010, 7, S232–S237.

[2] Yang X S. Nature-Inspired Metaheuristic Algorithms. Luniver Press, Somerset, 2010, 1-5.

[3] Lindfield G, Penny J. Introduction to Nature-Inspired Optimization, Academic Press, London, United Kingdom, 2017, 101–117.

[4] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521, 436–444.

[5] Mason K, Duggan M, Barrett E, Duggan J, Howley E. Predicting host CPU utilization in the cloud using evolutionary neural networks. Future Generation Computer Systems, 2018, 86, 162–173.

[6] Hassan H A, Mohamed S A, Sheta W M. Scalability and communication performance of

HPC on Azure Cloud. Egyptian Informatics Journal, 2016, 17, 175–182.

[7] Abdul Khalid N E, Ariff N, Yahya S, Noor N. A Review of Bio-inspired Algorithms as Image Processing Techniques. Communications in Computer and Information Science, 2011,179, 660–673.

[8] Binitha S, Sathya S S. A survey of bio inspired optimization algorithms. International Journal of Soft Computing and Engineering, 2012, 2, 137–151.

[9] Chizari H, Lupu E, Thomas P. Randomness of Physiological Signals in Generation Cryptographic Key for Secure Communication Between Implantable Medical Devices Inside The Body And The Outside World. Living in the Internet of Things: Cybersecurity of the IoT 2018, 2018, 1–6.

[10] Sayers W. Artificial Intelligence Techniques for Flood Risk Management in Urban Environments. Thesis, University of Exeter, Exeter, UK, 2015.

[11] Bayer P, Finkel M. Evolutionary Algorithms for the Optimization of Advective Control of Contaminated Aquifer Zones. Water Resources Research, 2004, 40.

[12] Nicklow J, Reed P, Savić D, Dessalegne T, Harrell L, Chan-Hilton, A, Karamouz M, Minsker B, Ostfeld A, Singh A, ZechmanE. State of the Art for Genetic Algorithms and Beyond in Water Resources Planning and Management. Journal of Water Resources Planning and Management, 2010, 136, 412–432.

[13] Karaboga D. An idea based on honey bee swarm for numerical optimization. 2005.

[14] Zainal N, Zain A, SharifS. Overview of Artificial Fish Swarm Algorithm and its Applications in Industrial Problems. Applied Mechanics and Materials, 2015, 815, 253– 257.

[15] Zhou Y Q, Chen H, Zhou G. Invasive weed optimization algorithm for optimization no-idle flow shop scheduling problem. Neurocomputing, 2014, 137, 285–292.

[16] Simon D. Biogeography-Based Optimization. IEEE Transactions on Evolutionary Computation, 2008, 12, 702–713.

[17] Sayers W, Savic D, Kapelan Z. Performance of LEMMO with artificial neural networks for water systems optimisation. Urban Water Journal, 2019, 16, 21–32.

[18] Coello C A C. Twenty Years of Evolutionary Multiobjective Optimization: A Historical Overview of the Field. IEEE Computational Intelligence Magazine, 2005, 1, 28–36.

[19] Deb K, Pratap A, Agarwal S, MeyarivanT. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6, 182–197.

[20] Bishop C M. Neural Networks for Pattern Recognition. Oxford University Press, 1995, 77-161.

[21] Parker D B. Learning Logic. 1985.

[22] Werbos P. Beyond Regression: New Tools for Predictions and Analysis in the Behavioural Sciences. Harvard, 1974.

[23] Cybenko G. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems, 1989, 2, 303–314.

[24] Hornik K, Stinchcombe M, White H. Multilayer feedfordward networks are universal approximators. Neural Networks, 1989, 2, 359–366.

[25] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado G S, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, VanhouckeV, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015.

[26] McCulloch W, Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biology, 1943, 5, 115–133.

[27] Rosenblatt F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. Psychological Review, 1958, 65, 386–408.

[28] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional

Neural Networks. Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012, 1097–1105.

[29] Elsheikh A H, Sharshir S W, Elaziz M A, Kabeel A E, Guilan W, Haiou Z. Modeling of solar energy systems using artificial neural network: A comprehensive review. Solar Energy, 2019, 180, 622–639.

[30] Bagheri M, Mirbagheri S A, Ehteshami M, Bagheri Z. Modeling of a sequencing batch reactor treating municipal wastewater using multi-layer perceptron and radial basis function artificial neural networks. Process Safety and Environmental Protection, 2015, 93, 111– 123.

[31] Yusoff N I M, Alhamali D I, Ibrahim A N H, Rosyidi S A P, Hassan N A. Engineering characteristics of nanosilica/polymer-modified bitumen and predicting their rheological properties using multilayer perceptron neural network model. Construction and Building Materials, 2019, 204, 781–799.

[32] Whittington J C R, Bogacz R. Theories of Error Back-Propagation in the Brain. Trends in Cognitive Sciences, 2019, 23, 235–250.

[33] Arulkumaran K, Cully A, Togelius J. AlphaStar: An Evolutionary Computation Perspective. Proceedings of the Genetic and Evolutionary Computation Conference, Prague, 2019, 314-315.

[34] Tian Y, ZhangK, Li J, Lin X, Yang B. LSTM-based traffic flow prediction with missing data. Neurocomputing, 2018, 318, 297–305.

[35] Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, Chen T. Recent advances in convolutional neural networks. Pattern Recognition, 2018, 77, 354– 377.

[36] Behzadian K, Kapelan Z, Savić D A, Ardeshir A. Stochastic Sampling Design using Multiobjective Genetic Algorithm and Adaptive Neural Networks. Environmental Modelling & Software, 2009, 24, 530–541.

[37] Juhn Y, Liu H. Artificial intelligence approaches using natural language processing to advance EHR-based clinical research. Journal of Allergy and Clinical Immunology, 2020, 145, 463–469.

[38] Trappey A J C, Trappey C V, Wu JL, Wang J W C. Intelligent compilation of patent summaries using machine learning and natural language processing techniques. Advanced Engineering Informatics, 2020, 43.

[39] Dmitriev E A, Myasnikov V V. Possibility estimation of 3D scene reconstruction from multiple images. Proceedings of the International Conference onInformation Technology and Nanotechnology, Samara, Russia, 2019, 293–296.

[40] Gkioxari G, Malik J, Johnson J. Mesh R-CNN. Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 2019, 9785-9795.

[41] Holland J H. Adaptation in Natural and Artificial Systems. University of Michigan Press, 1975, 1-232.

[42] De Jong K A. An Analysis of the Behaviour of a Class of Genetic Adaptive Systems.Thesis, University of Michigan, USA, 1975.

[43] Koza J R.Genetic Programming. MIT Press, Massachusetts, USA, 1992, 73-191.

[44] Paul P V, Moganarangan N, Kumar S S, Raju R, Vengattaraman T, Dhavachelvan P. Performance analyses over population seeding techniques of the permutation-coded genetic algorithm: An empirical study based on traveling salesman problems. Applied Soft Computing, 2015, 32, 383–402.

[45] Islam M L, Shatabda S, Rashid M A, Khan M G M, Rahman M S. Protein structure prediction from inaccurate and sparse NMR data using an enhanced genetic algorithm. Computational Biology and Chemistry, 2019, 79, 6–15.

[46] Akopov A S, Beklaryan L A, Thakur M, Verma B D. Parallel multi-agent real-coded genetic algorithm for large-scale black-box single-objective optimisation. KnowledgeBased Systems, 2019, 174, 103–122.

[47] Luo J, Fujimura S, El Baz D, Plazolles B. GPU based parallel genetic algorithm for solving an energy efficient dynamic flexible flow shop scheduling problem. Journal of Parallel and Distributed Computing, 2019, 133, 244–257.

[48] Beyer HG, Schwefel HP. Evolution strategies - A comprehensive introduction. Natural Computing, 2002, 1, 3–52.

[49] Rechenberg I. Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinsipien der Biologischen Evolution. (In German)Frommann-Holzboog, Stuttgart, Germany, 1973, 1170.

[50] Rechenberg I. Cybernetic solution path of an experimental problem. Royal Aircraft Establishment Translation No. 1122, B. F. Toms, Trans., Ministry of Aviation, Royal Aircraft Establishment, 1965.

[51] Schwefel HP. Evolutionsstrategie und numerischeOptimierung. (In German)Thesis, Technische Universität Berlin, Berlin, Germany, 1975.

[52] Schwefel HP. Kybernetische Evolution als Strategie der Exprimentellen Forschung in der Strömungstechnik. (In German) Dissertation, TechnischeUnversitat Berlin, Berlin, Germany, 1965.

[53] Klockgether J, Schwefel H P. Two-phase nozzle and hollow core jet experiments. Proceedings of the 11th Symposium on Engineering Aspects of Magnetohydrodynamics, Pasadena, Californa,1970, 141-148.

[54] Schwefel HP. Projekt MHD-Staustrahlrohr: Experimentelle Optimierung einerZweiphasendüse, Teil I. (In German), 1968.

[55] Engelbrecht A P. Computational Intelligence, An Introduction. Wiley, 2007, 213-235.

[56] Bäck T, Hoffmeister F, Schwefel HP. A Survey of Evolution Strategies. Proceedings of the Fourth International Conference on Genetic Algorithms, San Diego, USA, 1991, 2–9.

[57] Schwefel H P. Numerical Optimization of Computer Models. Wiley, Chichester, UK, 1981, 1-330.

[58] Schwefel HP NumerischeOptimierung von Computer-Modellen mittels der Evolutionsstrategie. Birkhaeuser, Basel, Switzerland,1977,123-176..

[59] Lin Y, Yang Q, Guan G. Scantling optimization of FPSO internal turret area structure using RBF model and evolutionary strategy. Ocean Engineering, 2019, 191, 106562.

[60] Liu K, Zhang J. Nonlinear process modelling using echo state networks optimised by covariance matrix adaption evolutionary strategy. Computers & Chemical Engineering, 2020, 135, 106730.

[61] Liu G, Zhao L, Yang F, Bian J, Qin T, Yu N, Liu TY. Trust Region Evolution Strategies. Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, Honolulu, USA, 2019, 4252-4359.

[62] Salimans T, Ho J, Chen X, Sidor S, Sutskever I. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. Preprint at https://arxiv.org/abs/1703.03864, 2017.

[63] Dorigo M, Stützle T. Ant Colony Optimization: Overview and Recent Advances. Handbook of Metaheuristics, 2010, 146, 227–263.

[64] Cordon O, Viana I F de, Herrera F, Moreno L. A New ACO Model Integrating Evolutionary Computation Concepts: The Best-Worst Ant System, Proceedings of the 2nd International Workshop on Ant Algorithms, Brussels, Belgium,2000, 22-29.

[65] Dorigo M, Stutzle T. Ant Colony Optimization. MIT Press, Massachusetts, USA, 2004, 1-319.

[66] Dorigo M. Optimization, Learning and Natural Algorithms. Thesis, Politecno di Milano, Milan, Italy, 1992.

[67] Dorigo M, Maniezzo V, Colorni A. Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Man, Systems and Cybernetics - Part B, 1997, 26, 29–41.

[68] Mohan B C, Baskaran R. A survey: Ant Colony Optimization based recent research and implementation on several engineering domain. Expert Systems with Applications, 2012, 39, 4618–4627.

[69] López-Ibáñez M, Stutzle T. Automatic configuration of multi-objective ant colony optimization algorithms. Lecture Notes in Computer Science, 2010, 6234, 95–106.

[70] Pham D, Ghanbarzadeh A, Koç E, Otri S, Rahim S, Zaidi M. The Bees Algorithm Technical Note. Manufacturing Engineering Centre, Cardiff University, UK, 2005, 1–57.

[71] Pham D T, Castellani M. A comparative study of the Bees Algorithm as a tool for function optimisation. Cogent Engineering, 2015, 2, 1-28.

[72] Khan I, Maiti M K. A swap sequence based Artificial Bee Colony algorithm for Traveling Salesman Problem. Swarm and Evolutionary Computation, 2019, 44, 428–438.

[73] Ning J, Zhang C, Zhang B. A novel artificial bee colony algorithm for the QoS based multicast route optimization problem. Optik, 2016, 127, 2771–2779.

[74] Sağ T, Çunkaş M. Color image segmentation based on multi objective artificial bee colony optimization. Applied Soft Computing, 2015, 34, 389–401.

[75] Kumar A, Kumar D, Jarial S K. A Review on Artificial Bee Colony Algorithms and Their Applications to Data Clustering. Cybernetics and Information Technologies, 2017, 17, 3–28.

[76] Zou W, Zhu Y, Chen H, Sui X. A Clustering Approach Using Cooperative Artificial Bee Colony Algorithm. Discrete Dynamics in Nature and Society, 2010, 2010, 1-17.

[77] Li X L, Shao Z J, Qian J X. An optimizing method based on autonomous animats: fishswarm algorithm. Systems Engineering - Theory & Practice, 2002, 22, 32–38.

[78] Yu L, Li C. A Global Artificial Fish Swarm Algorithm for Structural Damage Detection. Advances in Structural Engineering, 2014, 17, 331–346.

[79] Neshat M, Adeli A, Sepidnam G, Sargolzaei M, Toosi A. A Review of Artificial Fish Swarm Optimization Methods and Applications. Int. Journal on Smart Sensing and Intelligent Systems, 2012, 5, 107–148.

[80] He Q, Hu X, Ren H, Zhang H. A novel artificial fish swarm algorithm for solving largescale reliability–redundancy application problem. ISA Transactions, 2015, 59, 105–113.

[81] Basak A, Maity D, Das S. A differential invasive weed optimization algorithm for improved global numerical optimization. Applied Mathematics and Computation, 2013, 219, 6645–6668.

[82] Rani D S, Subrahmanyam N, Sydulu M. Multi-Objective Invasive Weed Optimization – An application to optimal network reconfiguration in radial distribution systems. International Journal of Electrical Power & Energy Systems, 2015, 73, 932–942.

[83] Barisal A K, Prusty R C. Large scale economic dispatch of power systems using oppositional invasive weed optimization. Applied Soft Computing, 2015, 29, 122–137.

[84] Ghasemi M, Ghavidel S, Akbari E, Vahed A A. Solving non-linear, non-smooth and nonconvex optimal power flow problems using chaotic invasive weed optimization algorithms based on chaos. Energy, 2014, 73, 340–353.

[85] Zhao Y W, Leng L L, Qian Z Y, Wang W L. A Discrete Hybrid Invasive Weed Optimization Algorithm for the Capacitated Vehicle Routing Problem. Procedia Computer Science, 2016, 91, 978–987.

[86] Velmurugan T, Khara S, Nandakumar S, Saravanan B. Seamless Vertical Handoff using Invasive Weed Optimization (IWO) algorithm for heterogeneous wireless networks. Ain Shams Engineering Journal, 2016, 7, 101–111.

[87] Goudos S K, Plets D, Liu N, Martens L, Joseph W. A multi-objective approach to indoor wireless heterogeneous networks planning based on biogeography-based optimization. Computer Networks, 2015, 91, 564–576.

[88] Lin J. A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem. Knowledge-Based Systems, 2015, 78, 59–74.

[89] Kim SS, Byeon JH, Yu H, Liu H. Biogeography-based optimization for optimal job scheduling in cloud computing. Applied Mathematics and Computation, 2014, 247, 266– 280.

[90] Rajasomashekar S, Aravindhababu P. Biogeography based optimization technique for best compromise solution of economic emission dispatch. Swarm and Evolutionary Computation, 2012, 7, 47–57.

[91] Wang L, Xu Y. An effective hybrid biogeography-based optimization algorithm for parameter estimation of chaotic systems. Expert Systems with Applications, 2011, 38, 15103–15109.

[92] Niu Q, Zhang L T, Li K. A biogeography-based optimization algorithm with mutation strategies for model parameter estimation of solar and fuel cells. Energy Conversion and Management, 2014, 86, 1173–1185.

[93] Jourdan L, Corne D, Savic D, Walters G. Evolutionary Multi-Criterion Optimization, Springer, Berlin, Germany, 2005, 841–855.

[94] Jourdan L, Corne D, Savić D, Walters G. Hybridising Rule Induction and Multi-Objective Evolutionary Search for Optimising Water Distribution Systems. Proceedings of the Fourth International Conference on Hybrid Intelligent Systems, Kitakyushu, Japan, 2004, 434, 439.

[95] Woodward M, Kapelan Z, Gouldby B. Adaptive Flood Risk management under Climate Change Uncertainty using Real Options and Optimisation. Risk Analysis, 2013, 1, 75–92.

[96] Woodward M, Gouldby B, Kapelan Z, Hames, D. Multiobjective optimisation for improved management of flood risk. Journal of Water Resources Planning and Management (ASCE), 2013, 2, 201–215.

[97] di Pierro F, Khu ST, Savić D, Berardi L. Efficient multi-objective optimal design of water distribution networks on a budget of simulations using hybrid algorithms. Environmental Modelling & Software, 2009, 24, 202–213.

[98] Woodward M. The use of real options and multi-objective optimisation in flood risk management. Thesis, University of Exeter, Exeter, UK, 2012.

[99] Pareto V. Cours D'Economie Politique Vol. I & II. Lausanne, 1896.

[100] Deb K, Jain H. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. IEEE Transactions on Evolutionary Computation, 2014, 18, 577–601.

[101] Doerner K, J. Gutjahr W, Hartl R, Strauss C, Stummer C. Ant Colony Optimization in Multiobjective Portfolio Selection. Proceedings of the 4th Metaheuristics International Conference, Porto, Portugal, 2001, 243-248.

[102] Jing L, Zhuo-qun Z, Li-li Z, Kang-jie S. Multi-Objective Ant Colony Optimization Algorithm Based on Discrete Variables. IOP Conference Series: Earth and Environmental Science, 2018, 189, 042031.

[103] Oliveira S M, Hussin M S, Stuetzle T, Roli A, Dorigo M. A Detailed Analysis of the Population-based Ant Colony Optimization Algorithm for the TSP and the QAP.

Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, Dublin, Ireland, 2011, 13-14.

[104] Zaharia M, Xin R S, Wendell P, Das T, Armbrust M, Dave A, Meng X, Rosen J, Venkataraman S, Franklin M J, Ghodsi A, Gonzalez J, Shenker S, Stoica I. Apache Spark: A Unified Engine for Big Data Processing. Communications of the ACM, 2016, 59, 56–65.

[105] Gaifang D, Xueliang F, Honghui L, Pengfei X. Cooperative ant colony-genetic algorithm based on spark. Computers & Electrical Engineering, 2017, 60, 66–75.

[106] Poole D J, Allen C B. Constrained niching using differential evolution. Swarm and Evolutionary Computation, 2019, 44, 74–100.

[107] Tian M, Gao X. Differential evolution with neighborhood-based adaptive evolution mechanism for numerical optimization. Information Sciences, 2019, 478, 422–448.

[108] Kong X X, Chen Y L, Xie W, Wu X Y. A novel paddy field algorithm based on pattern search method. Proceedings of the 2012 IEEE International Conference on Information and Automation, Chengdu, China, 2012, 686–690.

[109] Askarzadeh A. Bird mating optimizer: An optimization algorithm inspired by bird mating strategies. Communications in Nonlinear Science and Numerical Simulation, 2014, 19, 1213–1228.

[110] Alijla B O, Wong LP, Lim C P, Khader A T, Al-Betar M A. A modified Intelligent Water Drops algorithm and its application to optimization problems. Expert Systems with Applications, 2014, 41, 6555–6569.

[111] Niu S H, Ong S K, Nee A Y C. An improved intelligent water drops algorithm for solving multi-objective job shop scheduling. Engineering Applications of Artificial Intelligence, 2013, 26, 2431–2442.

[112] Patle B K, Pandey A, Jagadeesh A, Parhi D R. Path planning in uncertain environment by using firefly algorithm. Defence Technology, 2018, 14, 691–701.

[113] Tighzert L, Fonlupt C, Mendil B. A set of new compact firefly algorithms. Swarm and Evolutionary Computation, 2018, 40, 92–115.

[114] Fernandes D A B, Freire M M, Fazendeiro P A P, Inácio P R M. Applications of artificial immune systems to computer security: A survey. Journal of Information Security and Applications, 2017, 35, 138–159.

[115] Ming L, Zhao J. Feature selection for chemical process fault diagnosis by artificial immune systems. Chinese Journal of Chemical Engineering, 2018, 26, 1599–1604.

[116] He S, Wu Q H, Saunders J. Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching Behaviour. Evolutionary Computation, IEEE Transactions on, 2009, 13, 973–990.

[117] Kang Q, Lan T, Yan Y, Wang L, Wu Q D. Group search optimizer based optimal location and capacity of distributed generations. Neurocomputing, 2012, 78, 55–63.

[118] Dash R, Dash R, Rautray R. An Evolutionary Framework based Microarray Gene Selection and Classification Approach using Binary Shuffled Frog Leaping Algorithm. Journal of King Saud University - Computer and Information Sciences, 2019.

[119] Eusuff M, Lansey K, Pasha F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Engineering Optimization, 2006, 38, 129–154.

[120] Lin A P, Sun W, Yu H S, Wu G H, Tang H W. Adaptive comprehensive learning particle swarm optimization with cooperative archive. Applied Soft Computing, 2019, 77, 533–546.

[121] Lin A P, Sun W, Yu H S, Wu G H, Tang H W. Global genetic learning particle swarm optimization with diversity enhancement by ring topology. Swarm and Evolutionary Computation, 2019, 44, 571–583.

[122] Chen H N, Zhu Y L. Optimization based on symbiotic multi-species coevolution. Applied Mathematics and Computation, 2008, 205, 47–60.

[123] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Proceedings of the Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Athens, Greece, 2002, 95–100.

[124] Gharari R, Poursalehi N, Abbasi M, Aghaie M. Implementation of Strength Pareto Evolutionary Algorithm II in the Multi objective Burnable Poison Placement Optimization of KWU Pressurized Water Reactor. Nuclear Engineering and Technology, 2016, 48, 1126–1139.

[125] Schaffer J D. Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms. Thesis, Vanderbilt University, Nashville, USA , 1984.

[126] Schaffer J D. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. 1985, 93–100.

[127] Knowles J D, Corne D W. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation, 2000, 8, 149–172.

[128] Knowles J, Corne D. Properties of an adaptive archiving algorithm for storing nondominated vectors. Evolutionary Computation, IEEE Transactions on, 2003, 7, 100–116.