Evolving Systems (2023) 14:141–156
https://doi.org/10.1007/s12530-022-09432-6

REVIEW

Nature inspired optimization algorithms: a comprehensive overview

Ankur Kumar[1] · **Mohammad Nadeem**[1] · **Haider Banka**[2]

Received: 11 June 2021 / Accepted: 28 February 2022 / Published online: 19 March 2022

Abstract
Nature performs complex tasks in a simple yet efficient way. Natural processes may seem straightforward from outside but are composed of several inherently complicated sub-processes. Inspired from nature, several Nature Inspired Optimization Algorithms (NIOAs) have been developed in recent years. The family of NIOAs is expanding rapidly. Therefore, the set of NIOAs became quite large and selecting an appropriate NIOA is a tedious job. Since each one of the algorithms offers something novel, the similarities and differences among them are necessary to be established so that the selection of an algorithm for a particular problem becomes relatively easy. Moreover, a problem needs to be mapped in a NIOA, requiring understanding of fundamental components of NIOAs. Tuning parameters and algorithm operators another important concern in NIOAs that need be addressed carefully for better performance of the algorithm. Our work distinguishes NIOAs on the basis of various criteria and discusses the building blocks of various algorithms to achieve aforementioned objectives. The purpose of present study is to analyze major concepts related to NIOAs such as fundamentals of NIOAs, comparison among them, advancements, etc. In order to explain the usage of components of NIOA, an illustrative example is also presented.

Keywords Soft computing · Evolutionary computation · Optimization · Nature inspired algorithm

1 Introduction

Optimization has been an integral part of a decision-making process (Vercellis 2009). It is a process of finding the best possible solution of a given problem. Most of the real-world optimization problems such as Travelling Salesman Problem (TSP) are difficult to solve (Dorigo and Gambardella 1997). Exhaustive search can provide the optimum solution but is impractical in such scenarios (Woeginger 2003). Moreover, optimization problems are becoming more and more complex such as timetabling and scheduling problems due to increasing number of variables, dimensions, time complexity, space complexity, etc Lewis (2008), Rinnooy Kan (2012). These problems are considered NP-hard. The NP-hard problems cannot be solved through traditional methods and in such cases NIOAs come into effect (Yang 2010). The application areas of NIOAs cover intricate domains such as image processing (Duarte et al. 2006; Hammouche et al. 2010; Jalaleddin Mousavirad and Ebrahimpour-Komleh 2017; Cuevas and Sossa 2013; Jino Ramson et al. 2019), control systems and auto tuning (Angelov and Guthke 1997; Diogo Pereira Puchta et al. 2019; Serdar et al. 2021; Rodríguez-Molina et al. 2020; Pritesh et al. 2021) , wireless sensor networks (Pinto Alex et al. 2014; Tsai et al. 2015; PraveenKumar et al. 2019; Gupta and Jha 2018; Kaur and Mahajan 2018), traffic management in transportation systems (Zhao et al. 2011; Mohammad Reza et al. 2018; DelSer et al. 2019), intrusion detection (Shafi and Abbass 2007; Costa et al. 2015; Behdad et al. 2012), data analytics (Cheng et al. 2013; Chou and Ngo 2016; Iqbal et al. 2018; Mohammadi et al. 2020), energy system (Verma et al. 2018; Gamarra and Guerrero 2015; Bo-Yang et al. 2018), healthcare systems (Sivakumar and Marcus 2012; Gálvez et al. 2018; Tsai et al. 2015), social media network assessment (Hussain and Cambria 2018; Gonzalez-Pardo et al. 2017; Bello-Orgaz et al. 2017), manufacturing industry automation (Wari and Zhu 2016; Gen et al. 2017; Diez-Olivan et al. 2019), cyber crime and malware detection (Choraś and Kozik 2018; Dilek et al. 2015; Afifi

✉ Mohammad Nadeem
nadeem.amu@gmail.com; mnadeem.cs@amu.ac.in

[1] Department of Computer Science, Aligarh Muslim University, Aligarh, UP 202002, India

[2] Department of Computer Science and Engineering, Indian Institute of Technology (ISM), Dhanbad, Jharkhand 826004, India

et al. 2016; Cui et al. 2018) and many more (Angelov and Buswell 2003; Casey and Damper 2010; Gogna and Tayal 2013; Yang and He 2016; Yang 2020).

Nature has always inspired human beings to long for the unreachable. The natural phenomenon inspires humans to solve problems as nature does in an efficient and systematic way. Therefore, humans mimic nature to develop problem solving techniques and apply them different domains. It has been seen in real-life situations that we are often interested in an eqrly 'good' solution rather than a delayed 'best' solution. Keeping this observation in mind, many Nature Inspired Optimization Algorithms (NIOAs) are designed to provide the solution within acceptable time-limits. They are a subclass of metaheuristic algorithms that mimic natural processes such as natural selection, group movement, food foraging, physical laws, etc Yang (2014). These algorithms are of general purpose and do not require detailed knowledge of the problem to be solved. Algorithms inspired from nature provide a good framework to search good solutions when search direction is not clear (Talbi 2009).

There are a number of nature inspired optimization algorithms available in literature and are somewhat different from each other. Since they are large in number, studying and analyzing each one of them for an individual is difficult. Some algorithms employ biological concepts, other follow laws of physics and chemistry. Some algorithms are based on single solution and others require a set of solutions. Some algorithms are stochastic in nature while others are deterministic. However, there are some common concepts that are applicable to all such as encoding, tuning parameters, operators etc. Therefore, a researcher who is not very familiar with NIOAs is always in a dilemma which algorithm should be preferred.

There are a few notable works in the literature which present a classification of NIOAs. Most of them categorized NIOAs as evolutionary algorithm and swarm intelligence based algorithm. Some of the researchers grouped NIOAs based on the source of inspiration. Fister Jr et al. (2013) put the algorithms into four categories on the basis of source of inspiration such as swarm based, bio inspired but not swarm based, physics/chemistry based, and other algorithms. Siddique and Adeli (2015) went one step further and focussed on two aspects while making comparisons i.e. source of inspiration and nature of algorithms (deterministic and stochastic). Boussaid and others. Boussaïd et al. (2013) divided NIOAs into single and population based methods to investigate the similarities and differences among them. A survey of metaheuristics for high-dimensional problems was presented in Mahdavi et al. (2015). Hussain et al. (2019) provided an exhaustive survey of metaheuristics considering research works of over three decades. A detailed assessment of biology based optimization algorithms, its present situation and future prospects are discussed in Del Ser et al. (2019). Yang (2020) outlined a list of challenges and open issues in the domain of NIOA that are yet to be addressed.

Though NIOAs faced criticism for exhibiting similar behavior (Sörensen 2015), their contribution in solving large set of problems can not be denied. Given the significance of NIOAs in the variety of domains, there is need of a study which provides a comprehensive overview of NIOAs and tries to cover all the major components related to the algorithm. Our work tries to fill this research gap and present a brief yet complete overview of NIOAs. Present work does not only compare NIOAs but also assembles necessary information such as building blocks of NIOAs, advancements in NIOAs and future enchantments at one place. Fundamental concepts that are instrumental in the working of NIOAs like exploration, exploitation, encoding, elitism, interpretation of results etc. are discussed. Apart from source of inspiration, two more criteria are employed to distinguish among NIOAs. One of them is the number of solutions an algorithm evaluates in each iteration. Third criterion is whether a given algorithm is deterministic or stochastic in nature. Each algorithm has a number of control parameters that decide its convergence rate or performance. Bad selection of these parameters results in sub-optimal solutions. Moreover, NIOA relies on a set of operators to reach to the optimum values. A list of tuning parameters and operators is also presented as each algorithm employ distinct ways to achieve solutions. New advancements in NIOAs such as parallelism and hybridization are also included in the study. Some open directions are also suggested for researchers to follow. 29 Popular and original (without considering the variations of algorithm) single objective NIOAs are considered in the present work. Abbreviations used in the study are presented in Table 1.

The rest of the paper is organized as follows. Section 2 contains the fundamental concepts of NIOAs. Section 3 describes the classification and structure of NIOAs. An illustration is presented in Sect. 4 to clarify how the components of a NIOA are used to solve a problem. Section 5 contains the further advancements of NIOAs. Few future directions are suggested in Sect. 6. Section 7 presents concluding remarks.

2 Components of nature inspired optimization algorithm

Though there are many NIOAs, they follow a set of broader concepts in their own ways. In this section, the common notions used in NIOAs are discussed.

2.1 Exploration

As the name suggests, exploration refers to look for unknown. The primary concern with solving the NP-Hard

Table 1 Commonly used abbreviations and their full forms

Abbreviation	Full form	Abbreviation	Full form
ABC	Artificial Bee Colony	HS	Harmony Search
ACO	Ant Colony Optimization	KHA	Krill Herd Algorithm
BA	Bat Algorithm	LOA	Lion Optimization Algorithm
BBO	Biogeography Based Optimization	MSA	Monkey Search Algorithm
BFO	Bacterial Foraging Optimization	NIOA	Nature Inspired Optimization Algorithm
CFO	Central Force Optimization	NP	Non-deterministic Polynomial time
CLSA	Clonal Selection Algorithm	OIO	Optics Inspired Optimization
CRO	Chemical Reaction Optimization	PFA	Paddy Field Algorithm
CSA	Cuckoo Search Algorithm	PSO	Particle Swarm Optimization
DE	Differential Evolution	RFD	River Formation Dynamics
DPO	Dolphin Pod Optimization	SA	Simulated Annealing
EMO	Electromagnetism Optimization	SFLA	Shuffled Frog Leaping Algorithm
FA	Firefly Algorithm	SOA	Spiral Optimization Algorithm
FPA	Flower Pollination Algorithm	SSO	Social Spider Optimization
GA	Genetic Algorithm	TS	Tabu Search
GSA	Gravitational Search Algorithm		

optimization problems is the gigantic size of search space which leads to the unacceptable time complexity for any conventional algorithm. NIOAs offer probing only potential solution areas of search space thus reducing the time complexity. Therefore, exploration is seen as the process of reaching out to feasible unvisited area of search space (Yang et al. 2014). It also helps an algorithm to escape from sub-optimum solutions. Different NIOAs utilize different mechanisms to implement exploration that improves the quality of obtained solution. In fact, the performance of an algorithm is directly related with the amount of emphasis given to the exploration process. Well executed exploration, especially in the initial iterations, improves the performance of algorithm significantly.

2.2 Exploitation

Exploitation process is another very important building block of an NIOA and often seen in contrast with exploration. It refers to the ability of an algorithm to utilize the local knowledge to generate better solutions. Since solutions improve with each iteration, exploitation process tries to find better individuals confined in a certain region of search space with the help of existing best solutions (Yang et al. 2014). It can be seen as searching in the neighbourhood of already obtained solutions to find the maxima or minima within a small area. To reach to a global optimum a systematic coordination between exploration and exploitation strategies is needed. The efficiency of an NIOA depends heavily on the interplay of these two processes.

2.3 Encoding

NIOAs follow natural phenomenon that are not easy to be imitated. For an algorithm to work efficiently, a solution should be represented in a certain way. Encoding is the process of representing a solution into problem and algorithm specific form Yang (2014). Depending upon the requirement of problem, a solution could be encoded in the form of binary string, ordered set of numbers, unordered set of numbers, permutation of numbers etc Talbi (2009). For example, in genetic algorithm the set of individuals are often represented as binary strings. A list of commonly used encodings is presented in Table 2.

2.4 Generation of new solutions

Most algorithms produce new solutions in two ways, 1) randomly 2) using algorithmic operators. Random generation of solution(s) is often carried out in the initial stage of the algorithm. Algorithmic operators select existing solution(s) and update them in several stages of the optimization process. The new solutions with improved values are accepted and are used further to generate even better solutions in later stages. However, certain restrictions should be ensured during both processes such as, the produced solutions must follow the encoding technique of the algorithm and must reside in feasible search space.

2.5 Elitism

NIOAs are iterative in nature. New solutions, produced with each iteration, may be better than the existing ones or not.

Table 2 Commonly used encodings of a solution in NIOAs

Encoding	Example	Remarks
Binary string	10011010	The binary string is first converted into equivalent real value to check the goodness of the solution. The same encoding may also represent presence (1) or absence of (0) of an attribute.
Set of integers	(34,70)	Here a solution consists of two variables which are integers
Set of real values	(5.78,0.223,9.01)	Example refers to a solution represented by a set of three real values. Each value represents individual variable of the problem.
Ordered set of integers	(7,4,9,1,5)	Represents a path in a graph, $V_7 \rightarrow V_4 \rightarrow V_9 \rightarrow V_1 \rightarrow V_5$
2D Matrix	$\begin{pmatrix} 1 & 0 & 2 & 1 \\ 2 & 1 & 1 & 0 \\ 2 & 0 & 1 & 1 \end{pmatrix}$	Represents a schedule. For example, rows may represent nurses of a hospital and columns may represent days of a week. Entries of the matrix may represent morning shift (1), evening shift (2) or off-day (0)

Elitism is the process of preserving high-quality solutions so that they are not lost over iterations. They are usually stored separately. Elite solutions can participate in the generation of new solutions or not depending upon the structure of the algorithm. The benefit of using elitist model is that the quality of the solution can only upgrade as the number of iterations increase.

2.6 Stopping criteria

Since optimum solution is unknown in real world problems, a large number of iterations are performed to reach to it. Therefore, an obvious need of defining the stopping criteria is there after reaching to convergence. Various techniques have been suggested in literature to deal with the issue. Commonly used criteria are: setting the number of maximum iterations and amount of change in best solution over iterations. More sophisticate standards are also suggested. One has to be careful while selecting a particular criterion as it can affect the performance of the algorithm (Fernández-Vargas et al. 2016).

Apart from the above-mentioned criteria to stop the execution of algorithm, researchers also use other rules depending upon the optimization problem at hand. Rather than maximum number of generations, maximum CPU time in seconds or maximum number of objective function evaluations can also be mentioned to stall the execution. These rules are referred as exhaustion-based criteria (Zielinski and Laur 2008). If the best solution is already known as in the case of benchmark functions, error-based stopping rules can be used, given as the variation between the best result generated by algorithm and the already known value of global optimum. An algorithm can also be stopped if the quality of newly generated solutions is insignificant i.e. a tolerance limit is specified and relative change in the best solution is monitored. If the change is less than the specified limit for a given number of generations or given amount of time, the algorithm is forced to stop.

The criteria discussed so far are used very often in NIOA literature but more complex rules are also there (Gogna and Tayal 2013). The movement of solutions can be considered as a potential metric to stall the algorithm. In such scenarios, the movement of decision variables and/or objective function values is monitored and if their shift is marginal over iterations/time, the algorithm is stopped. It has been observed that the solutions in population-based NIOA tend to converge around global optimum. Therefore, the distribution of the solutions should be confined to a small region over iterations/time and hence, can be used as a stopping criteria. More than one rules can also be combined to set a stopping criteria.

2.7 Interpretation of result

A large number of NIOAs are stochastic in nature i.e. they randomly pick the solution from search space. The result may fall in local optima as different algorithms follow different search paths to find results even for the same set of parameters. Therefore, the results obtained through them must be interpreted correctly. If the optimum solution of the given problem is unknown, one can never guarantee the obtained result from NIOA as best since the algorithm does not consider the whole search space. One popular way to deal with this issue is to run the algorithm many times, compare the results of all instances and select the best out of them.

2.8 Objective function

Objective function represents the function to be optimized. When a solution is produced, it is essential for the algorithm to check how good or bad it is. The goodness of a solution is given by its corresponding objective function value. It

could be in the form of a function or not, depending upon the problem under consideration. It helps the algorithm in decision making such as which solution should be carried forward into the next iteration or whether a particular operator should be applied on it.

With this discussion it is clear that NIOAs are similar and try to achieve the above-mentioned concepts using various phenomenon. The flowchart of the generic steps of an NIOA are indicated in Fig. 1.

3 Classification and structure of NIOAs

In the section, various categorization criteria are discussed, and structure of individual algorithm is highlighted with the help of corresponding control parameters and operators. Classification of NIOAs is presented graphically in Fig. 2.

3.1 Classification criteria

A study of 29 NIOAs was carried out to discern the similarities and differences among them. From that, three criteria are identified for categorization and the same are discussed below.

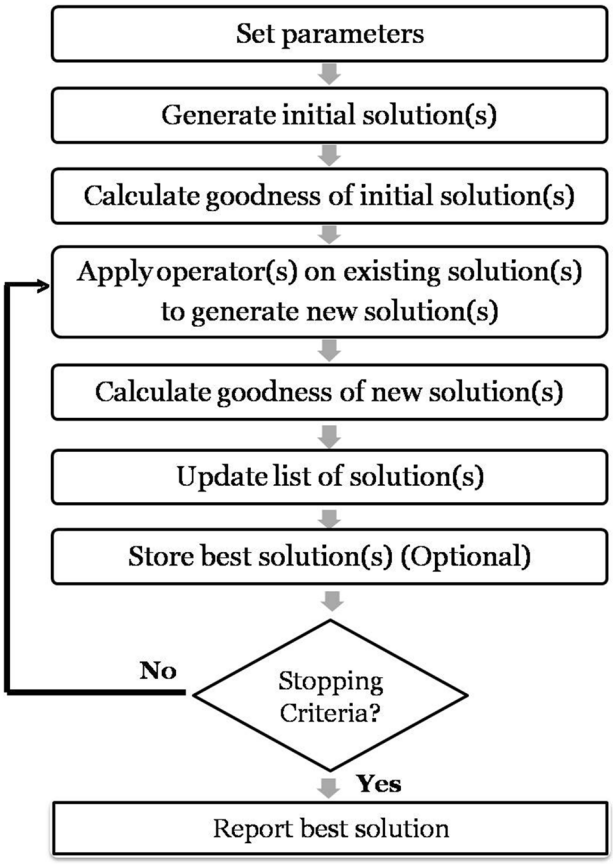

Fig. 1 Generic steps in a nature inspired optimization algorithm

3.1.1 Source of inspiration

Grouping on the basis of inspiration-source has been done elsewhere as well. It indicates a particular domain a NIOA imitates. Commonly followed domains are:

- Biology: Most of the algorithms are inspired from biological process. The algorithms such as are genetic algorithm, ant colony optimization, particle swarm optimization, firefly algorithm, cuckoo search algorithm, bat algorithm, differential evolution, dolphin pod optimization, flower pollination, etc are bio-inspired.
- Physics: Some algorithms mimic Physics related phenomenon. Algorithms which fall in this category are gravitational search algorithm, simulated annealing, central force optimization, river formation dynamics, electro-magnetism optimization, etc.
- Others: In our study, there is one chemistry-based algorithm which is chemical reaction optimization, one algorithm has taken inspiration from music i.e. harmony search and tabu search is based on social rituals.

3.1.2 Size of solution set

An algorithm evaluates either one solution or a set of solutions in one iteration. On the criteria, Based on the number of solutions used algorithms are divided into two types:

- Single solution-based: As discussed above, an NIOA that considers only one solution in each iteration fall in this group. Such algorithm moves through a trajectory to reach optimal solution.
- Multiple solution-based: An algorithm that operates on a set of potential solutions to move towards optimal goals simultaneously is assigned in this category. Such an algorithm is also called population based algorithm. Most of NIOAs belong to this class.

3.1.3 Nature of algorithm

The nature of algorithm can be understood as the process of calculating new solution. It can broadly be divided into two:

- Stochastic: If a NIOA involves any random component in its overall calculation to produce new solutions, it is termed as stochastic or randomized NIOA. One of the characteristic of stochastic algorithm is that given a set of input parameters, the intermediate calculations and final output are difficult to predict beforehand. Therefore, results yielded in one run of the algorithm can

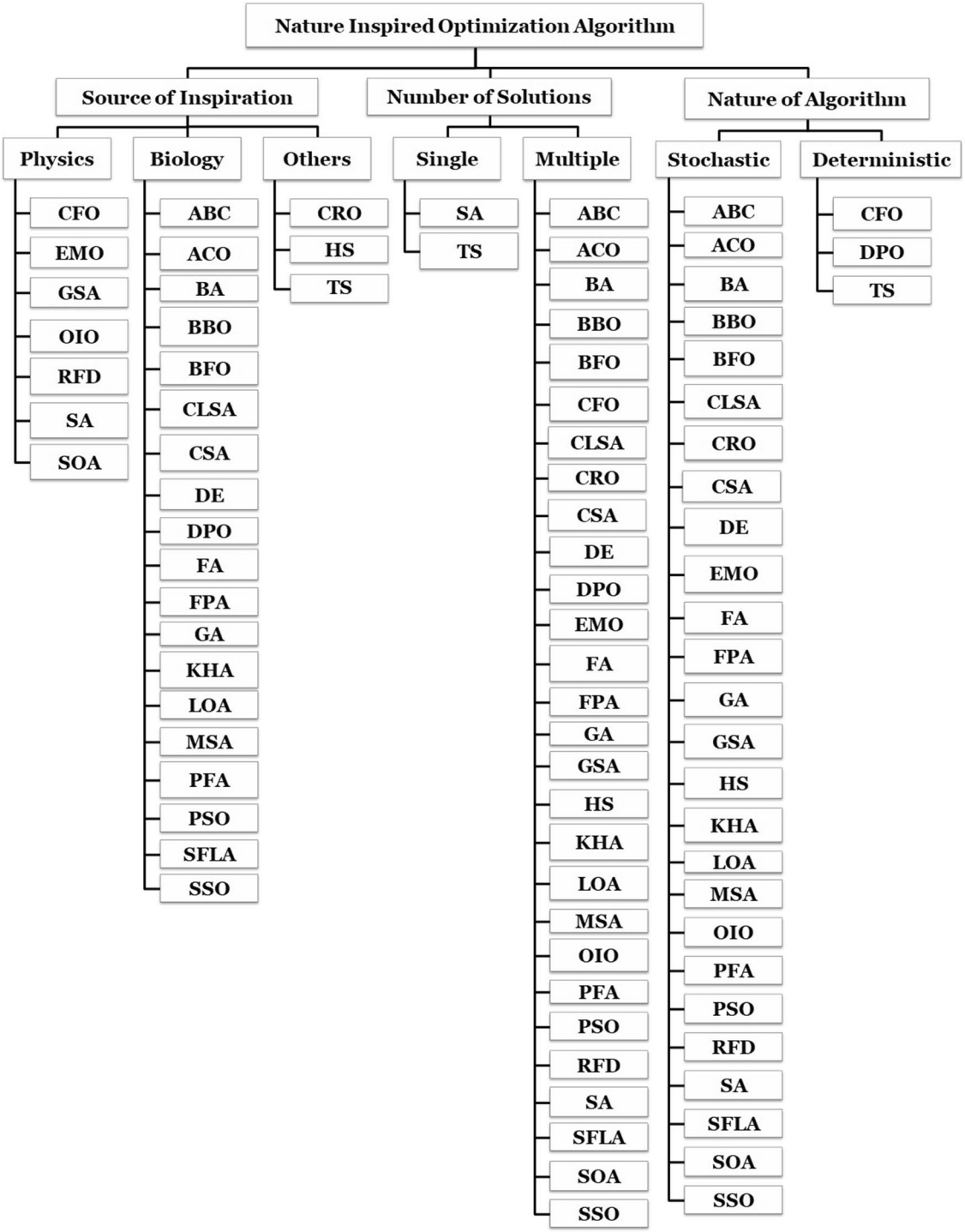

Fig. 2 Classification hierarchy of nature inspired optimization algorithms

be different from another run. Most of nature-inspired algorithms are stochastic.

- Deterministic: In contrast to stochastic NIOAs, we will call those NIOAs deterministic that do not use random numbers for generating further solutions. Therefore, intermediate results and end solution are relatively easy to estimate.

Based on above-mentioned factors, NIOAs are grouped, and the results are presented in Table 3.

Table 3 Classification of nature inspired optimization algorithms based on three criteria

S. no.	Algorithm	Source of inspiration			Number of solutions		Nature of algorithm	
		Biology	Physics	Others	Single	Multiple	Stochastic	Deterministic
1.	ABC Karaboga and Basturk (2007)	✓	✗	✗	✗	✓	✓	✗
2.	ACO Dorigo et al. (2006)	✓	✗	✗	✗	✓	✓	✗
3.	BA Yang and Gandomi (2012)	✓	✗	✗	✗	✓	✓	✗
4.	BBO Simon (2008)	✓	✗	✗	✗	✓	✓	✗
5.	BFO Das et al. (2009)	✓	✗	✗	✗	✓	✓	✗
6.	CFO Formato (2008)	✗	✓	✗	✗	✓	✗	✓
7.	CLSA De Castro and Von Zuben (2000)	✓	✗	✗	✗	✓	✓	✗
8.	CRO Lam and Li (2012)	✗	✗	✓	✗	✓	✓	✗
9.	CSA Xin-She and Suash (2009)	✓	✗	✗	✗	✓	✓	✗
10.	DE Storn and Price (1997)	✓	✗	✗	✗	✓	✓	✗
11.	DPO Serani and Diez (2017)	✓	✗	✗	✗	✓	✗	✓
12.	EMO İlker and Shu-Chering (2003)	✗	✓	✗	✗	✓	✓	✗
13.	FA Yang et al. (2008)	✓	✗	✗	✗	✓	✓	✗
14.	FPA Yang (2012)	✓	✗	✗	✗	✓	✓	✗
15.	GA Goldberg (2006)	✓	✗	✗	✗	✓	✓	✗
16.	GSA Rashedi et al. (2009)	✗	✓	✗	✗	✓	✓	✗
17.	HS ZongWoo et al. (2001)	✗	✗	✓	✗	✓	✓	✗
18.	KHA Amir and Amir (2012)	✓	✗	✗	✗	✓	✓	✗
19.	LOA Yazdani and Jolai (2016)	✓	✗	✗	✗	✓	✓	✗
20.	MSA Mucherino and Seref (2007)	✓	✗	✗	✗	✓	✓	✗
21.	OIO AliHusseinzadeh (2015)	✗	✓	✗	✗	✓	✓	✗
22.	PFA Premaratne et al. (2009)	✓	✗	✗	✗	✓	✓	✗
23.	PSO Kennedy and Eberhart (1995)	✓	✗	✗	✗	✓	✓	✗
24.	RFD Rabanal et al. (2007)	✗	✓	✗	✗	✓	✓	✗
25.	SA Kirkpatrick (1983)	✗	✓	✗	✓	✗	✓	✗
26.	SFLA Eusuff et al. (2006)	✓	✗	✗	✗	✓	✓	✗
27.	SOA Tamura and Yasuda (2011)	✗	✓	✗	✗	✓	✓	✗
28.	SSO James and Li (2015)	✓	✗	✗	✗	✓	✓	✗
29.	TS Glover and Laguna (1998)	✓	✗	✗	✓	✗	✗	✓

3.2 Structure of an NIOA

As we discussed earlier, NIOAs are similar and try to achieve a balanced mix of exploration and exploitation processes. However, the way in which one algorithm achieves the same may be different from another NIOA. Parameters and operators are the main factors that govern the structure of an NIOA and its ability to find the optimum solution. A list of NIOAs with corresponding parameters and operators is prepared and presented in Table 4.

3.2.1 Tuning parameters

Each NIOA has parameters the values of which are initialized before starting the optimization process. Values of these parameters are fixed at initial stage but may or may not vary while proceeding to further stages. The amount of change, if any, is usually minimal. Initial setting of parameters has tremendous effect on the performance of the algorithm (Pellegrini and Birattari 2006). In fact, if the algorithm does not give desired results, the values of parameters are modified and algorithm is run again. Usually, a NIOA having few parameters is preferred.

3.2.2 Primary operators

As discussed in earlier sections, new solutions are generated from existing ones by applying algorithmic operators. A NIOA can have many such operations. An operator tries to find improved results in the neighbourhood of existing solution and helps the algorithm to converge to the optimal solution. In fact, they are the realization of exploitation process. A good operator has a positive impact on the efficiency of the algorithm.

Table 4 List of tuning parameters and operators of NIOAs

S. no.	Algorithm	Tuning parameters	Operators
1.	ABC	limit or abandonment criteria	food source selection
		number of employed, onlooker and scout bees	food source position update
2.	ACO	amount of pheromone	pheromone updation
		pheromone evaporation rate	ant path construction
3.	BA	pulse of loudness	frequency update
		pulse rate	velocity update
			position update
4.	BBO	immigration rate	habitat modification
		emigration rate	mutation
		maximum mutation rate	
5.	BBO	chemotactic step	chemotaxis
		swimming length	swarming
		reproduction step	reproduction
		elimination-dispersal event	elimination-dispersal
		elimination-dispersal probability	
		size of the step for tumble	
6.	CFO	gravitational constant	probe acceleration update
		two free parameters α and β	ant path construction
7.	CLSA	number of best antibodies	selection
		number of replaced antibodies	cloning
			hypermutation
8.	CRO	kinetic energy loss rate	on-wall ineffective collision
		initial kinetic energy	decomposition
		molecular collision number	inter-molecular ineffective collision
		central energy buffer	synthesis
		two free parameters α and β	
9.	CSA	nest abandon fraction	cuckoo updation through levy flight
		levy flight step size	
10.	DE	mutation coefficient	mutation
		crossover probability	crossover
			selection
11.	DPO	weight for the attraction force	dolphin velocity update
		integration time step	dolphin position update
		pod dynamics parameter	
		food attraction force tuning	
12.	EMO	local search iteration number	local search operation
		local search parameter	point movement operation
13.	FA	absorption coefficient	position update of the firefly
		randomization parameter	
		attractiveness constant	
14.	FPA	strength of the pollination	global flower pollination
		switch probability	local flower pollination
15.	GA	crossover rate	selection
		crossover rate	crossover
			mutation
16.	GSA	gravitational constant	acceleration calculation
		a free parameter	agent velocity update
			agent position update
17.	HS	harmony memory considering rate	harmony improvisation
		pitch adjusting rate	harmony pitch adjustment

Table 4 (continued)

S. no.	Algorithm	Tuning parameters	Operators
18.	KHA	maximum induced speed	motion induced by other krill
		maximum diffusion speed	foraging motion
		inertia weights	physical diffusion
		crossover probability	crossover
		mutation probability	mutation
19.	LOA	number of prides	hunting
		sex rate	movement towards safe place
		percent of nomad lions	roaming
		mating probability	mating
		roaming percent	defence
		immigration rate	migration
		mutation probability	
20.	MSA	step length of climb process	climb process
		eyesight of monkey	watch-jump process
		somersault interval	somersault process
21.	OIO	number of changes	mirror type determination
		a free parameter K	correction of spherical aberration
			new image generation
			aggregation of images
22.	PFA	maximum seeds per plant	selection
		dispersion spread	seeding
		neighborhood function radius	pollination
			dispersion
23.	PSO	inertia weight	particle velocity update
		cognitive acceleration coefficient	particle position update
		social acceleration coefficient	
24.	RFD	erosion parameter	drop movement
		block drop parameter	erosion
		not climb factor	depositing sediments
		two free parameters ω and δ	
25.	SA	temperature	annealing process
26.	SFLA	number of memeplexes	construction of memeplex and submemeplex
		number of frogs in a memeplex	improving the position of worst frog
		number of frogs in a submemeplex	upgradation the memeplex
		maximum number of steps in one memeplex	shuffling of memeplex
		maximum step size adopted by an infected frog	
27.	SOA	composite rotation matrix	search point updation
		step rate	center updation
28.	SSO	attenuation rate	vibration generation
		probability of changing mask	mask changing
		probability of value in dimension mask to be one	random walk calculation
29.	TS	size of tabu list(s)	generation of list of new solutions
			computation of next best solution
			updation of tabu list
			updation of aspiration criteria

In the present study, only main parameters and operators are considered. For example, some parameters are common to each algorithm such as size of solution set (or population size), dimensionality of a solution and number of iterations. Similarly, operator like goodness/fitness calculation is implemented in each NIOA. We did not consider such parameters and operators.

4 An illustrative example

In order to explain the usage of components of NIOA, an example problem is taken and Gravitational Search Algorithm (GSA) is applied on it.

4.1 GSA

As mentioned in the previous section, GSA is a population-based stochastic algorithm and is inspired from the laws of gravitation. It starts by assigning initial random positions to search agents. Each agent is characterized by its position and mass (active gravitational mass, passive gravitational mass or inertial mass). The mass of agent is calculated by the objective function to be optimized and changes with each iteration. Change in mass affects the gravitational force F that affects the acceleration applied on the agent. Acceleration decides the new velocity of the agent, leading to update in its position. This cycle continues till some stopping criteria is met. Next, basic GSA is outlined and description of each step is given:

Step I: **Initialization** N agents are assigned random position initially. The position of ith agent in n dimensional space is defined as:

$$X_i = (x_i^1, x_i^2, \ldots, x_i^d, \ldots, x_i^n) \tag{1}$$

where x_i^d signifies dth dimension of ith agent.

Step II: **Fitness evaluation** Calculate fitness of each agent according to its position vector and based on the optimization function under consideration.

Step III: **Best and worst fitness computation** The next step is to calculate best and worst fitness among all agents at current iteration t depending upon type of optimization function.

Maximization problem:

$$best(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t) \tag{2}$$

$$worst(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t) \tag{3}$$

Minimization problem:

$$best(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t) \tag{4}$$

$$worst(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t) \tag{5}$$

where best(t) and worst(t) denote the best and worst fitness among all agents at time t and $fit_j(t)$ represents the fitness of jth agent at current iteration t.

Step IV: **Calculation of gravitational constant** The gravitational constant has the initial value G_0 and will decay with each iteration to control the search accuracy i.e. G is a function of G_0 and time t:

$$G(t) = h(G_0, t) \tag{6}$$

Step V: **Calculation of agents' Masses** From computational point of view:

$$M_{ai} = M_{pi} = M_{ii} = M_i, \quad i = 1, 2, \ldots, N, \tag{7}$$

And value of M_i is calculated as follows:

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \tag{8}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{9}$$

where M_{ai}, M_{pi} and M_{ii} denote ith agent's active gravitational mass, passive gravitational mass and inertial mass respectively.

Step VI: **Calculation of force in different directions** The magnitude of force acting on mass i from mass j in the direction of d at time t is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \epsilon}(x_j^d(t) - x_i^d(t)) \tag{10}$$

where ϵ is a small constant and distance R_{ij} is the euclidian distance between agents i and j,

$$R_{ij}(t) = \|X_i(t), X_j(t)\|_2 \tag{11}$$

The total force exerted from other agents on agent i in a dimension d is given by:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t) \tag{12}$$

where $rand_j$ is a random number in the interval [0, 1] and provides stochastic characteristic to GSA. To improve the performance of algorithm, instead of

calculating forces by all the agents, we can restrict only to the *Kbest* agents and Eq. (12) can be modified as:

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i}^{N} rand_j F_{ij}^d(t) \tag{13}$$

Kbest is a function of time and has initial value K_0 which decreases with time. Therefore, at the beginning, all agents will exert the force and as the number of iterations increase, *Kbest* decreases linearly. Subsequently, there will be only one agent in the end applying force to the other agents.

Step VII: **Calculation of acceleration** Acceleration of ith agent in dth dimension at iteration t is computed as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{14}$$

Step VIII: **Updation of velocity and position** Since velocity of an agent at next iteration is equal to the sum of the fraction of its current velocity and acceleration, position and velocity of corresponding agent is calculated for $(t+1)$ iteration as:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{15}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{16}$$

where $rand_i$ is a random number in the interval [0, 1]

Step IX: **Stopping criteria** Repeat steps II to VIII until the stopping criteria is met.

Apart from agent population 'N' and number of iterations 'T', the algorithm uses two tunable parameters: the gravitational constant 'G' (mentioned in Step:IV) and a free parameter. A good set of these value decides the performance of the algorithm and may vary from one problem to the another.

4.2 Mapping of problem into GSA framework

Each problem is unique and need to be mapped into algorithmic framework before it can be solved. In this section, an example problem is mapped into GSA structure.

1. **Objective function:** Let us say, we want to minimize the function $f(x) = x_1^2 + x_2^2$, where $x_1, x_2 \epsilon [-50, 50]$. Here, f(x) is called the objective function that is to be optimized. Sometimes the optimization problem at hand is theoretical in nature. In such cases, an optimization function need to be designed that can capture the essence of the problem in a mathematical form.
2. **Encoding:** For this problem, a potential solution can be encoded as an ordered set of values (x_i, x_j). For different values of (x_i, x_j), the corresponding values of objective function f can be calculated, referred as fitness/goodness values for a given set (x_i, x_j). These fitness values help GSA calculate algorithmic metaphors such as mass, force and acceleration.
3. **New solutions:** The Eqs. (15) and (16) help the algorithm generate new improved solutions at each iteration once mass, force and acceleration values are calculated for existing solutions. GSA may produce new solution vectors that are beyond the range $[-50, 50]$. In such cases, those solutions are brought back in the specified range (Cantú et al. 2021).
4. **Stopping criteria:** Step-IX of GSA does not mention any stopping criteria to stop the algorithm. For the given problem, simple halting rules such as maximum number of iterations or less variation in best goodness value can be used. We have taken maximum number of iterations as the stopping criterion (mentioned in the next section).
5. **Elitism:** GSA is elitist in nature by default because the algorithm keeps track of the best solution generated after each iteration (Step-III) and report it in the end.
6. **Exploration and exploitation:** As discussed in earlier sections, exploration refers to the ability of an algorithm to explore the unexplored regions of the problem search space. To achieve the purpose, a component of randomness is often added in the algorithm (Blum and Roli 2003). In the case of GSA also, random numbers are introduced in Eqs. (12) and (15). The value of random number determines the area of search space to be explored by the algorithm. On the other hand, exploitation means using the information at hand about the current good solutions i.e. to look for better solutions in the neighbourhood of already known solutions. The Eq. (16) helps GSA search the local space around solutions found so far.

4.3 Hands-on calculations

To understand how the mapped problem is solved by GSA, the results of various steps of the algorithm are given below for one iteration.

Optimization problem: Minimize: $f(x) = x_1^2 + x_2^2$ where $x_1, x_2 \epsilon [-50, 50]$

Setting parameters: $N = 5$ (Population size), $T = 100$ (Max. number of iterations) and $G = G_0 e^{-\alpha t/T}$ where $G_0 = 100$ and $\alpha = 20$

Initialization with five agents randomly: $A_1 = (-5, 0)$, $A_2 = (0, 2)$, $A_3 = (-2, 3)$, $A_4 = (5, 5)$ and $A_5 = (0, 4)$

Initial fitness calculation: using the given optimization function, the fitness values are calculated as $f_{A1}(1) = 25$, $f_{A2}(1) = 4$, $f_{A3}(1) = 13$, $f_{A4}(1) = 50$ and $f_{A5}(1) = 16$

Therefore, $Best(1) = 4$ and $Worst(1) = 50$.

Calculating gravitational constant: Since, $G = G_0 e^{-\alpha t/T}$

$G = 100 \times e^{-20\times 1/100}$, $t = 1$ as it is the first iteration

$G = 83.33$

Calculating mass: Now, $m_{A1}(1) = \dfrac{f_{A1}(1) - Worst(1)}{Best(1) - Worst(1)}$

$m_{A1} = (25 - 50)/(4 - 50) = 0.543$

Similarly, $m_{A2}(1) = 1$, $m_{A3}(1) = 0.80$, $m_{A4}(1) = 0$ and $m_{A5}(1) = 0.76$

Using Eq. (9), $M_{A1}(1) = 0.176$, $M_{A2}(1) = 0.325$, $M_{A3}(1) = 0.260$, $M_{A4}(1) = 0$ and $M_{A5}(1) = 0.237$

Calculating force: Using Eq. (10), $F^1_{A1A2}(1) = \dfrac{83 \times 0.176 \times 0.325}{5.3 + 0.1} \times (0 + 5) = \dfrac{23.738}{5.4} = 4.39$

Here, 5.3 is the Euclidean distance between A_1 and A_2.

Similarly, $F^1_{A1A3}(1) = 0.268$, $F^1_{A1A4}(1) = 0$ and $F^1_{A1A5}(1) = 2.70$

Now, the force acting on the agent i due to all agents (Eq. 12),

$$F^1_{A1}(1) = \sum_{j=1, j\neq i}^{N} rand_j F^1_{A1Aj}(1)$$

$F^1_{A1}(1) = 0.5 \times (4.39 + 2.68 + 0 + 2.70) == 4.885$

Calculating acceleration: Using Eq. (14), $a^1_{A1}(1) = \dfrac{F^1_{A1}(1)}{M_{A1}(1)}$

$a^1_{A1}(1) = \dfrac{4.885}{0.176} = 27.75$

Calculating updated velocity: updated velocity in (t+1) th iteration (Eq. 15),

$v^1_{A1}(2) = rand \times v^1_{A1}(1) + a^1_{A1}(1)$

$v^1_{A1}(2) = 0.5 \times 0 + 27.75 = 27.75$

Calculating updated position: using Eq. (16),

$X^1_{A1}(2) = X^1_{A1}(1) + V^1_{A1}(2)$

$X^1_{A1}(2) = -5 + 27.75 = 22.75$

Similarly, the second dimension of agent A_1 ($X^2_{A1}(2)$) and other updated agent positions such as $(X^1_{A2}(2), X^2_{A2}(2))$, $(X^1_{A3}(2), X^2_{A3}(2))$, $(X^1_{A4}(2), X^2_{A4}(2))$ and $(X^1_{A5}(2), X^2_{A5}(2))$ can be calculated.

5 Advancements

Over time, new ideas made its way into NIOA literature and became integral part of it. They enhanced the problem-solving power of algorithms. In this section, two such concepts are discussed.

5.1 Parallelism

Real life optimization problems are computation and memory intensive. Even NIOAs require much time to produce satisfactory results in such cases which is unacceptable in many real-time applications. With the advancements in technology, multi-processor and multi-core platforms came into existence that leads us to the development of parallel computing. Any algorithm is composed of many steps that may not depend on each other. In parallel computation, the independent components of an algorithm are calculated on different processors and their results are combined afterwards, making it faster than sequential processing (Alba et al. 2013; Krömer et al. 2014). This approach is particularly helpful in searching large search space and producing improved quality of solutions in relatively less time (Narasimhan 2009). There are various ways in which parallelism can be achieved in NIOAs. In the simplest form, one problem can be solved independently by many NIOAs on different processors and the best result can be taken after comparing individual results. In a more complex setting, NIOAs can share information with each other on the occurrence of specific event such as after completion of n iterations. Independent components of single NIOA can also be distributed on processors. For example, population based NIOA requires calculation of fitness values of a given population in each iteration. Such population can be equally distributed among processors to simultaneously compute their fitness values. Apart from that, operators of an NIOA may be unrelated with each other such as crossover and mutation operators of GA and thus can be parallelized. Sometimes computation of objective and constraint functions is time consuming and can be divided into partial sets which are then assigned to available processors. There are numerous other ways in which simultaneous execution of the components of a single NIOA or multiple NIOAs can be planned (Alba et al. 2005). It is worth noticing that parallelism can alter the behaviour of NIOAs and a careful analysis of distribution of responsibilities among processors is necessary.

5.2 Hybridization

Hybridization is a popular concept in science and engineering that amalgamates two or more individual methods into one. The degree of amalgamation may depend on many criteria. Hybrid NIOA is a result of combining an NIOA with another NIOA, Mathematical Programming, Machine Learning or other related fields (Christian et al. 2011). The field

of hybridization in NIOA is very vast because algorithms can be combined in enormous ways. Uniting two methods could either be done for the overall problem-solving process or induced midway on the occurrence of specific event. For example, the initial solution of a single solution NIOA can be generated using a population based NIOA since the performance of single solution NIOA depends on the initial solution (Li et al. 2002). Similarly, the best solution obtained through a population based NIOA can be enhanced further if it is supplied as the initial solution in a single solution NIOA. Another very important concern in NIOA is the appropriate selection of tuning parameters which can be resolved with help of hybridization. These parameters can be modified as the iterations increase and the amount of change can be calculated with the help of machine learning techniques (Birattari and Kacprzyk 2009). Components of an NIOA can also be optimized using this approach. For example, simulated annealing can be used to effectively perform crossover and mutation operations in GA (Adler 1993). In some cases, data is given rather than the objective function. To calculate the fitness of a solution in such scenarios, supervised learning techniques are incorporated with NIOA. Sometimes, large search space is partitioned into distinct groups using unsupervised learning techniques and each sub-space is solved separately using NIOAs (Kennedy 2000). With the abovementioned discussion, it is clear that hybridization has enormous applications in optimization using NIOA.

6 Future directions

Though optimization algorithms inspired from nature are efficient, a lot more is yet to be achieved. Many open research areas have been suggested in literature (Yang 2020; Del Ser et al. 2019; Boussaïd et al. 2013; Mahdavi et al. 2015) . More efficient and less complex NIOAs are always welcome. An algorithm should give better results preferably in less computational time. Complexity also refers to the ease of problem mapping into corresponding NIOA. An algorithm with lesser number of operators and tunable parameters is considered less sophisticated. Selecting appropriate NIOA for a given problem is a tedious task and the domain of hyper-heuristics has emerged to deal with the issue. Hyper-heuristics refers to the techniques to pick the most suitable algorithm out of many for a problem (EdmundK et al. 2013). A detailed categorization of NIOAs would be helpful such as which algorithms are more suitable for integer programming or mixed integer programming problems. Classification can also be done on the basis of appropriateness of NIOAs for particular applications. For example, ACO and RFD were primarily designed to find the shortest path in a complex graph (Dorigo and Di Caro 1999; Rabanal et al. 2007). Such groupings will definitely ease up the job of researchers. In addition, the impact of control parameters of an NIOA can be studied thoroughly and new ways of getting their suitable values can be suggested. Future studies can also focus on the convergence proofs of NIOAs. They help us analyze an algorithm mathematically and provide its theoretical foundations. They also give insights regarding convergence, best and worst-case scenarios, significance of individual parameters etc. Only a few such studies are carried out so far in this area (Eiben et al. 1990; Vincent et al. 1994; Tamura and Yasuda 2017). It has been observed that NIOAs yield promising results while solving low dimensional optimization problems. Their performance degrades drastically when dimensionality of a problem becomes large, requiring special attention of researchers on scalability (Mahdavi et al. 2015). One issue with NIOAs is the generation of duplicate solutions. As iterations increase, a few solutions are repeated more often than others and unseen region of search space is not explored. Alam et al. (2016) suggested an algorithm which tries to reduce the number of duplicate solutions. Future works can focus on developing similar algorithms or on integrating such components in existing NIOAs. Moreover, hybridization and parallel models have opened a vast ocean of research directions.

The above-mentioned discussion primarily focus on new directions of research in the domain of NIOA. However, there is still room for improvement in the existing algorithms and their components. The litrature of NIOA is almost empty with the theoratical and mathematical proofs and foundations of the algorithm. Such analysis help us understand how exactly an algorithm works and gives is insights regrading its convergence criterion, scalability and robustness. Each algorithm has parameters that need to be set to optimal values to yield good results. Usually, the parameter space is significantly large and researchers rely on previous studies or rule-of-thumbs to fix the their values such as 0.8 is a well accepted value of crossover probability in GA. Improved parameter tuning methods with proper justifications form another promising research area. Desigining new benchmark problems and fucntions for different categories of optimizations problems is also required so that novel and existing algorithms can be thoroughly tested on them to idetify their strenghts and weaknesses. Developing new performance measures to assess the effectiveness of an algorithm is needed as well. Application oriented algorithms to deal with large-scale real-life scenarios are also welcome.

7 Conclusion

NIOA provides a simple and effective tool to solve complex real-world optimization problems within acceptable time limits. Selecting an algorithm requires a thorough understanding of its fundamentals, control parameters, operators and the problem at hand. Application of an algorithm is a multi-step process. Each sub-process needs to be carefully designed and adopted. Present work tried to address these issues and discussed relevant concepts related to NIOAs such as components, classification, structure and advancements. Some open research areas are also suggested. NIOAs are difficult to compare (Wolpert and Macready 1997) and hence categorize, yet better classification is essential. Apart from that, new algorithms are appreciated. Hopefully, researchers will ponder more on nature's phenomenon and we will witness more powerful and less complex algorithms in future studies.

References

Adler D (1993) Genetic algorithms and simulated annealing: a marriage proposal. In: IEEE international conference on neural networks, pp 1104–1109, IEEE

Afifi F, Anuar NB, Shamshirband S, Choo K-KR (2016) Dyhap: dynamic hybrid anfis-pso approach for predicting mobile malwared. PLoS One 11(9)

Alam M, Chatterjee S, Banka H (2016) A novel parallel search technique for optimization. In: 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), pp 259–263, IEEE

Alba E, Luque G, Nesmachnow S (2013) Parallel metaheuristics: recent advances and new trends. Int Trans Oper Res 20(1):1–48

Alba E, Talbi EG, Luque G, Melab N (2005) Metaheuristics and parallelism. Parallel metaheuristics: a new class of algorithms. Wiley, pp 79–104

Ali Husseinzadeh K (2015) A new metaheuristic for optimization: optics inspired optimization (oio). Comput Oper Res 55:99–125

Amir HG, Amir HA (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845

Angelov PP, Buswell RA (2003) Automatic generation of fuzzy rule-based models from data by genetic algorithms. Inf Sci 150(1–2):17–31

Angelov P, Guthke R (1997) A genetic-algorithm-based approach to optimization of bioprocesses described by fuzzy rules. Bioprocess Eng 16(5):299–303

Behdad M, Barone L, Bennamoun M, French T (2012) Nature-inspired techniques in the context of fraud detection. IEEE Trans Syst Man Cybern Part C Appl Rev 42(6):1273–1290

Bello-Orgaz G, Hernandez-Castro J, Camacho D (2017) Detecting discussion communities on vaccination in twitter. Futur Gener Comput Syst 66:125–136

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv (CSUR) 35(3):268–308

Boussaïd I, Lepagnot J, Siarry P (2013) A survey on optimization metaheuristics. Inf Sci 237:82–117

Bo-Yang Q, Zhu YS, Jiao YC, Wu MY, PonnuthuraiN S, JingJ L (2018) A survey on multi-objective evolutionary algorithms for the solution of the environmental/economic dispatch problems. Swarm Evol Comput 38:1–11

Cantú VH, Azzaro-Pantel C, Ponsich A (2021) Constraint-handling techniques within differential evolution for solving process engineering problems. Appl Soft Comput 108:107442

Casey MC, Damper RI (2010) Special issue on biologically-inspired information fusion. Inf Fusion 11(1):2–3

Cheng S, Shi Y, Qin Q, Bai R (2013) Swarm intelligence in big data analytics. In International Conference on Intelligent Data engineering and automated learning, pp 417–426, Springer, New York

Choraś M, Kozik R (2018) Machine learning techniques for threat modeling and detection. In: Security and Resilience in Intelligent Data-Centric Systems and Communication Networks, pp 179–192. Elsevier

Chou J-S, Ngo N-T (2016) Smart grid data analytics framework for increasing energy savings in residential buildings. Autom Constr 72:247–257

Christian B, Jakob P, Raidl Günther R, Andrea R (2011) Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 11(6):4135–4151

Costa KAP, Pereira LAM, Nakamura RYM, Pereira CR, Papa JP, Falcão AX (2015) A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks. Inf Sci 294:95–108

Cuevas E, Sossa H et al (2013) A comparison of nature inspired algorithms for multi-threshold image segmentation. Expert Syst Appl 40(4):1213–1219

Cui Z, Xue F, Cai X, Cao Y, Wang G, Chen J (2018) Detection of malicious code variants based on deep learning. IEEE Trans Industr Inf 14(7):3187–3196

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. Springer, New York

De Castro LN, Von Zuben FJ (2000) The clonal selection algorithm with engineering applications. In Proceedings of GECCO, volume 2000, pp 36–39

Del Ser J, Osaba E, Molina D, Yang X-S, Salcedo-Sanz S, Camacho D, Das S, Suganthan PN, Coello CA, Francisco H (2019) Bio-inspired computation Where we stand and what's next. Swarm Evolut Comput 48:220–250

DelSer J, Osaba E, Sanchez-Medina JJ, Fister I (2019) Bioinspired computational intelligence and transportation systems: a long road ahead. IEEE Trans Intell Transp Syst 21(2):466–495

Diez-Olivan A, DelSer J, Galar D, Sierra B (2019) Data fusion and machine learning for industrial prognosis Trends and perspectives towards industry 40. Inf Fusion 50:92–111

Dilek S, Çakır H, Aydın M (2015) Applications of artificial intelligence techniques to combating cyber crimes: a review. arXiv: 1502.03552

Diogo Pereira Puchta E, Siqueira HV, dos SantosKaster M (2019) Optimization tools based on metaheuristics for performance enhancement in a gaussian adaptive pid controller. IEEE Trans Cybern 50(3):1185–1194

Dorigo M, Gambardella LM (1997) Ant colonies for the travelling salesman problem. Biosystems 43(2):73–81

Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Magn 1(4):28–39

Dorigo M, Di Caro G (1999) Ant colony optimization: a new metaheuristic. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406), vol 2, pp 1470–1477, IEEE

Duarte A, Sánchez Á, Fernández F, Montemayor AS (2006) Improving image segmentation quality through effective region merging using a hierarchical social metaheuristic. Pattern Recogn Lett 27(11):1239–1251

EdmundK B, Michel G, Matthew H, Graham K, Gabriela O, Özcan E, Qu R (2013) Hyper-heuristics: a survey of the state of the art. J Oper Res Soc 64(12):1695–1724

Eiben AE, Aarts EHL, Van Hee KM (1990) Global convergence of genetic algorithms: a markov chain analysis. In: International Conference on Parallel Problem Solving from Nature, pp 3–12, Springer, New York

Eusuff M, Lansey K, Pasha F (2006) Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Eng Optim 38(2):129–154

Fernández-Vargas JA, Bonilla-Petriciolet A, Rangaiah GP, Fateen S-EK (2016) Performance analysis of stopping criteria of population-based metaheuristics for global optimization in phase equilibrium calculations and modeling. Fluid Phase Equilib 427:104–125

Fister Jr I, Yang X-S, Fister I, Brest J, Fister D (2013) A brief review of nature-inspired algorithms for optimization. arXiv:1307.4186

Formato RA (2008) Central force optimization: a new nature inspired computational framework for multidimensional search and optimization. In Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), pp 221–238. Springer, New York

Gálvez A, Fister I, Osaba E, Del Ser J, Iglesias A (2018) Automatic fitting of feature points for border detection of skin lesions in medical images with bat algorithm. In: International Symposium on Intelligent and Distributed Computing, pp 357–368. Springer

Gamarra C, Guerrero JM (2015) Computational optimization techniques applied to microgrids planning: a review. Renew Sustain Energy Rev 48:413–424

Gen M, Zhang W, Lin L, Yun YS (2017) Recent advances in hybrid evolutionary algorithms for multiobjective manufacturing scheduling. Comput Ind Eng 112:616–633

Glover F, Laguna M (1998) Tabu search. In: Handbook of combinatorial optimization, pp 2093–2229, Springer, New York

Gogna A, Tayal A (2013) Metaheuristics: review and application. J Exp Theor Artif Intell 25(4):503–526

Goldberg DE (2006) Genetic algorithms. Pearson Education India

Gonzalez-Pardo A, Jung JJ, Camacho D (2017) Aco-based clustering for ego network analysis. Futur Gener Comput Syst 66:160–170

Gupta GP, Jha S (2018) Integrated clustering and routing protocol for wireless sensor networks using cuckoo and harmony search based metaheuristic techniques. Eng Appl Artif Intell 68:101–109

Hammouche K, Diaf M, Siarry P (2010) A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem. Eng Appl Artif Intell 23(5):676–688

Hussain A, Cambria E (2018) Semi-supervised learning for big social data analysis. Neurocomputing 275:1662–1673

Hussain K, Salleh MNM, Cheng S, Shi Y (2019) Metaheuristic research: a comprehensive survey. Artif Intell Rev 52(4):2191–2233

İlker BŞ, Shu-Chering F (2003) An electromagnetism-like mechanism for global optimization. J Global Optim 25(3):263–282

Iqbal R, Doctor F, More B, Mahmud S, Yousuf U (2018) Big data analytics: computational intelligence techniques and application areas. Technological Forecasting and Social Change, pp 119253

Jalaleddin Mousavirad S, Ebrahimpour-Komleh H (2017) Multilevel image thresholding using entropy of histogram and recently developed population-based metaheuristic algorithms. Evol Intel 10(1–2):45–75

James JQ, Li VOK (2015) A social spider algorithm for global optimization. Appl Soft Comput 30:614–627

Jino Ramson SR, Lova Raju K, Vishnu S, Anagnostopoulos T (2019) Nature inspired optimization techniques for image processing-a short review. In Nature inspired optimization techniques for image processing-a short review. In. Springer, New York, pp 113–145

Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Global Optim 39(3):459–471

Kaur S, Mahajan R (2018) Hybrid meta-heuristic optimization based energy efficient protocol for wireless sensor networks. Egypt Inf J 19(3):145–150

Kennedy J (2000) Stereotyping: improving particle swarm performance with cluster analysis. In: Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512), vol 2, pp 1507–1512, IEEE

Kennedy James, Eberhart Russell (1995) Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks, vol 4, pp 1942–1948. IEEE

Kirkpatrick S, Daniel Gelatt C, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680

Krömer P, Platoš J, Snášel V (2014) Nature-inspired meta-heuristics on modern gpus: state of the art and brief survey of selected algorithms. Int J Parallel Prog 42(5):681–709

Lam AYS, Li VOK (2012) Chemical reaction optimization: a tutorial. Mem Comput 4(1):3–17

Lewis R (2008) A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum 30(1):167–190

Li WD, Ong SK, Nee AYC (2002) Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts. Int J Prod Res 40(8):1899–1922

Mahdavi S, Shiri ME, Rahnamayan S (2015) Metaheuristics in large-scale global continues optimization: A survey. Inf Sci 295:407–428

Mauro B, Janusz K (2009) Tuning metaheuristics: a machine learning perspective, vol 197, Springer, New York

Mohammadi FG, Amini MH, Arabnia HR (2020) Applications of nature-inspired algorithms for dimension reduction: Enabling efficient data analytics. In: Optimization, Learning, and Control for Interdependent Complex Networks, pp 67–84, Springer, New York

MohammadReza Jabbarpour, Houman Zarrabi, RashidHafeez Khokhar, Shahaboddin Shamshirband (2018) Kim-Kwang Raymond Choo. Applications of computational intelligence in vehicle traffic congestion problem a survey. Soft Comput 22(7):2299–2320

Mucherino A, Seref O (2007) Monkey search: a novel metaheuristic search for global optimization. In: AIP conference proceedings, vol 953, pp 162–173. American Institute of Physics

Narasimhan H (2009) Parallel artificial bee colony (pabc) algorithm. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), pp 306–311, IEEE

Pellegrini P, Birattari M (2006) The relevance of tuning the parameters of metaheuristics. In: Technical Report. Technical report, IRIDIA, Université Libre de Bruxelles

Pinto Alex R, Carlos M, Araújo G, Francisco V, Paulo P (2014) An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms. Inf Fusion 15:90–101

PraveenKumar D, Tarachand A, SekharaRao AC (2019) Machine learning algorithms for wireless sensor networks: a survey. Inf Fusion 49:1–25

Premaratne U, Samarabandu J, Sidhu T (2009) A new biologically inspired optimization algorithm. In: 2009 international conference on industrial and information systems (ICIIS), pp 279–284, IEEE

Pritesh S, Ravi S, Kulkarni AJ, Patrick S (2021) Metaheuristic algorithms in industry 4. 0. CRC Press, New York
Rabanal P, Rodríguez I, Rubio F (2007) Using river formation dynamics to design heuristic algorithms. In: International conference on unconventional computation, pp 163–177, Springer, New York
Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248
Rinnooy Kan AHG (2012) Machine scheduling problems: classification, complexity and computations. Springer, New York
Rodríguez-Molina A, Mezura-Montes E, Villarreal-Cervantes MG, Aldape-Pérez M (2020) Multi-objective meta-heuristic optimization in intelligent control: a survey on the controller tuning problem. Appl Soft Comput 93
Serani A, Diez M (2017) Dolphin pod optimization. In: International Workshop on Machine Learning, Optimization, and Big Data, pp 50–62. Springer, New York
Serdar U, Melih NS, Gebrail B (2021) Novel metaheuristic-based tuning of pid controllers for seismic structures and verification of robustness. J Build Eng 33
Shafi K, Abbass HA (2007) Biologically-inspired complex adaptive systems approaches to network intrusion detection. Inf Secur Tech Rep 12(4):209–217
Siddique N, Adeli H (2015) Nature inspired computing: an overview and some future directions. Cogn Comput 7(6):706–714
Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
Sivakumar R, Marcus K (2012) Diagnose breast cancer through mammograms using eabco algorithm. Int J Eng Technol 4(5):302–307
Sörensen K (2015) Metaheuristics-the metaphor exposed. Int Trans Oper Res 22(1):3–18
Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
Talbi E-G (2009) Metaheuristics: from design to implementation, vol 74. Wiley, Amsterdam
Tamura K, Yasuda K (2011) Primary study of spiral dynamics inspired optimization. IEEJ Trans Electr Electron Eng 6(S1):S98–S100
Tamura K, Yasuda K (2017) The spiral optimization algorithm: Convergence conditions and settings. IEEE Trans Syst Man Cybern Syst
Tsai C-W, Tsai P-W, Pan J-S, Chao H-C (2015) Metaheuristics for the deployment problem of wsn: a review. Microprocess Microsyst 39(8):1305–1317
Vercellis C (2009) Business intelligence: data mining and optimization for decision making. Wiley, Amsterdam
Verma P, Sanyal K, Srinivasan D, Swarup KS, Mehta R (2018) Computational intelligence techniques in smart grid planning and operation: a survey. In: 2018 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia), pp 891–896. IEEE
Vincent G, Mirko K, Rasson JP (1994) Simulated annealing: A proof of convergence. IEEE Trans Pattern Anal Mach Intell 16(6):652–656
Wari E, Zhu W (2016) A survey on metaheuristics for optimization in food manufacturing industry. Appl Soft Comput 46:328–343
Woeginger Gerhard J (2003) Exact algorithms for np-hard problems: a survey. Springer, New York, pp 185–207
Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
Xin-She Y, Suash D (2009) Cuckoo search via lévy flights. In: 2009 World congress on nature & biologically inspired computing (NaBIC), pp 210–214. IEEE
Yang X-S, Deb S, Fong S (2014) Metaheuristic algorithms: optimal balance of intensification and diversification. Appl Math Inf Sci 8(3):977
Yang X-S, Gandomi AH (2012) Bat algorithm: a novel approach for global engineering optimization. Engineering computations
Yang X-S, He X (2016) Nature-inspired optimization algorithms in engineering: overview and applications. In Nature-inspired computation in engineering, pp 1–20, Springer, New York
Yang Xin-She (2012) Flower pollination algorithm for global optimization. In: International conference on unconventional computing and natural computation, pp 240–249, Springer, New York
Yang X-S (2020) Nature-inspired optimization algorithms: challenges and open problems. J Comput Sci 46
Yang X-S et al (2008) Firefly algorithm. Nat-Inspired Metaheuristic Algorithms 20:79–90
Yang X-S (2010) Nature-inspired metaheuristic algorithms. Luniver Press, London
Yang X-S (2014) Nature-inspired optimization algorithms. Elsevier, Amsterdam
Yazdani M, Jolai F (2016) Lion optimization algorithm (loa): a nature-inspired metaheuristic algorithm. J Comput Des Eng 3(1):24–36
Zhao D, Dai Y, Zhang Z (2011) Computational intelligence in urban traffic signal control: A survey. IEEE Trans Syst Man Cybern Part C Appl Rev 42(4):485–494
Zielinski K, Laur R (2008) Stopping criteria for differential evolution in constrained single-objective optimization. In *Advances in differential evolution*, pp 111–138, Springer, New York
ZongWoo G, Joong HK, Gobichettipalayam Vasudevan L (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68