Expert Systems With Applications 193 (2022) 116468

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A new optimization algorithm inspired by the quest for the evolution of human society: Human felicity algorithm

Mohammad Verij kazemi [a,*], Elham Fazeli Veysari [b]

[a] Institute of Higher Education Pooyandegan Danesh., chalous, Mazandaran, Iran
[b] Faculty Of Management and Accounting, Islamic Azad University, Tonekabon Branch, Tonkabon, Mazandaran, Iran

ARTICLE INFO

Keywords:
Elite
Disciple
Human felicity
Optimization algorithm
Revolution

ABSTRACT

The present paper introduces a new evolutionary algorithm to solve optimization issues called the human felicity algorithm (HFA). The main idea of the HFA has been inspired the efforts of human society to become felicity. Because the change in human felicity is possible, only through a change in human thought, in the proposed algorithm, the objective function is human felicity in society and the search space optimization is the human thought in society. Like other optimization algorithms, the human felicity algorithm starts solving the problem by an initial random population. Then the population is divided into three categories of elites, disciples, and ordinary people. Afterward, people in human society change their thoughts by 3 ways: 1- being influenced by elites 2- personal experiences 3- sudden and drastic changes of thoughts for various reasons, such as war and famine. To test the capability of the proposed algorithm, the standard test functions of IEEE (Institute of Electrical and Electronics Engineers) Congress on Evolutionary Computation (CEC) 2014, CEC 2019, and CEC2020 with different dimensions have been used. In addition, to evaluate the ability of the proposed algorithm to solve various real-world complex engineering problems, 4 standard classic engineering optimization problems are examined and compared. To test the ability of the proposed algorithm in hyper-parameters tuning, the five-database classification was simulated using the Support Vector Machines (SVM). The simulation of the proposed algorithm on different test functions shows its high ability to solve complex problems with many local optimal points. For statistical analysis, powerful nonparametric methods such as Friedman and Nemenyi post-Hoc test for statistical analysis have been used. Comparing the HFA with some other optimization algorithms demonstrates its better performance in solving complicated multidimensional problems with the lower iteration.

1. Introduction

Exact optimization algorithms are not able to find a suitable solution in solving optimization problems with a high-dimensional search space. In these problems, the search space grows exponentially with the size of the problem thus; comprehensive search is not practical. Population-based optimization algorithms find near-optimal solutions to the difficult optimization problems by motivation from nature hence; the population is moved towards better solution areas of the search space (Karaboga & Akay, 2009). Two important classes of population-based optimization algorithms are evolutionary algorithms (Bäck & Schwefel, 1993) and swarm intelligence-based algorithms (Bonabeau, Theraulaz, & Dorigo, 1999). Although Genetic Algorithm (GA) (Goldberg & Holland, 1988), particle swarm optimization (PSO) (Eberhart & Kennedy, 1995), Genetic Programming (GP) (Koza & Koza, 1992; Koza, 1994), Differential Evolution (DE) algorithm (Storn & Price, 1997), and Evolutionary Programming (EP) (Fogel, 2006) are popular evolutionary algorithms, GA and PSO are the most widely used one in the literature. Swarm-based algorithms (Cheng, He, Jin, & Yao, 2018; Dhal, Ray, Das, & Das, 2019) are inspired by the behavior of some social organisms, such as ants, termites, birds, and fishes. Self-organization and decentralized control are striking features of swarm-based systems that, such as in nature, lead to emergent behavior. Emergent behavior is a feature that emerges through local interactions between system components and it is not achievable by any of the system components acting alone. Several various optimization algorithms have been proposed based on the nature of inspiration.

In recent years, scientists and researchers have proposed many optimization algorithms. Competitive swarm optimizer (CSO) for large-scale optimization is fundamentally inspired by the particle swarm

* Corresponding author.
E-mail addresses: m.v.kazemi62@gmail.com (M. Verij kazemi), elham.fveisari@gmail.com (E. Fazeli Veysari).

https://doi.org/10.1016/j.eswa.2021.116468
Received 3 June 2021; Received in revised form 6 December 2021; Accepted 25 December 2021
Available online 5 January 2022

optimization but is conceptually different (Cheng & Jin, 2014). Chaotic cuckoo search (CCS) optimization method is proposed by incorporating chaotic theory into cuckoo search algorithm (Wang, Deb, Gandomi, Zhang, & Alavi, 2016). Ageist Spider Monkey Optimization (ASMO) is motivated by the intelligent behavior of spider monkeys, based on the fact that not all monkeys in the population are the same. They belong to different age groups and have different levels of activity (Sharma, Sharma, Panigrahi, Kiran, & Kumar, 2016). Galactic Swarm Optimization (GSO) is imitated the motion of stars, galaxies and superclusters of galaxies under the influence of gravity (Muthiah-Nakarajan & Noel, 2016). Variable neighborhood bat algorithm (VNBA) was introduced by incorporating variable neighborhood search (VNS) and Bat algorithm (BA) algorithms (Wang, Lu, & Zhao, 2016). Polar Bear Optimization (PBO) algorithm is mimicked from the way polar bears hunt to survive in harsh arctic conditions (Połap, 2017). Salp Swarm Algorithm (SSA) is based on the swarming mechanism of salps (Mirjalili et al., 2017). Red fox optimization (RFO) is inspired on mathematical model of red fox live and hunting behavior (Połap, & Woźniak, 2021). CSO algorithm was improved and upgraded in (Mohapatra, Das, & Roy, 2017). Farmland fertility Algorithm (FFA) is inspired by farmland fertility in nature that is presented in (Shayanfar & Gharehchopogh, 2018). The Coyote Optimization Algorithm (COA) is metaheuristic for optimization that is inspired on the canis latrans species (Pierezan & Coelho, 2018). Supply-demand-based optimization (SDO) is motivated by the supply–demand mechanism in economics (Zhao, Wang, & Zhang, 2019). Fitness-dependent optimizer (FDO) is a metaheuristic algorithm that mimics the reproduction behavior of the bee swarm in finding better hives. (Abdullah & Ahmed, 2019). Poor and rich optimization (PRO) is motivated by the efforts of the two groups of the poor and the rich to achieve wealth and improve their economic situation (Moosavi & Bardsiri, 2019). Owl Optimization Algorithm (OOA) is imitated by the decoy behavior of the owls when any kind of danger is detected near the nests (Segundo, Mariani, & Coelho, 2019). Seagull Optimization Algorithm (SOA) is motivated by the migration and attacking behaviors of a seagull in nature (Dhiman & Kumar, 2019). A novel population-based evolutionary meta-heuristic algorithm is introduced, which imitates the Find-Fix-Finish-Exploit-Analyze (F3EA) targeting process. It considers the surface of the objective function as the battlefield and executes (F3EA) steps in an iterative manner (Kashan, Tavakkoli-Moghaddam, & Gen, 2019). Ludo Game-based Swarm Intelligence (LGSI) algorithm is simulated the rules of playing the game Ludo using two or four players to perform the update process for different swarm intelligent behaviors (Singh, Abd Elaziz, & Xiong, 2019). The artificial electric field algorithm (AEFA) is inspired by the Coulomb's law of electrostatic force, the concept of charge is extended to fitness value of the population in an innovative way (Yadav, 2019). The squirrel search algorithm (SSA) is imitated the dynamic foraging behavior of southern flying squirrels and their efficient way of locomotion known as gliding (Jain, Singh, & Rani, 2019). Falcon Optimization Algorithm (FOA) is a robust and powerful stochastic population-based algorithm that needs the adjustment of few parameters for its three-stage movement decision. FOA is mimicked the hunting behavior of falcons is presented (de Vasconcelos Segundo, Mariani, & dos Santos Coelho, 2019a; de Vasconcelos Segundo, Mariani, & dos Santos Coelho, 2019b). Pathfinder Algorithm (PFA) is inspired by collective movement of animal group and mimics the leadership hierarchy of swarms to find best food area or prey (Yapici & Cetinkaya, 2019). The Barnacles Mating Optimizer (BMO) is mimicked the mating behavior of barnacles in nature for solving optimization problems (Sulaiman, Mustaffa, Saari, & Daniyal, 2020). Search and Rescue (SAR) optimization algorithm is imitated the explorations behavior of humans during search and rescue operations (Shabani, Asgarian, Salido, & Gharebaghi, 2020). The Lévy flight distribution (LFD) algorithm is inspired by the Lévy flight random walk for exploring large search spaces (Houssein, Saad, Hashim, Shaban, & Hassaballah, 2020). Slime mould algorithm (SMA), is imitated the oscillation mode of slime mould in nature (Li, Chen, Wang, Heidari, & Mirjalili, 2020). Student psychology based optimization (SPBO) is based on the psychology of the students who are trying to give more effort to improve their performance in the examination up to the level for becoming the best student in the class (Das, Mukherjee, & Das, 2020). Wingsuit Flying Search (WFS) is motivated by the popular extreme sport – wingsuit flying. The algorithm mimics the intention of a flier to land at the lowest possible point of the Earth surface within their range (Covic & Lacevic, 2020). Political Optimizer (PO) is motivated by the multi-phased process of politics such as constituency allocation, party switching, election campaign, inter-party election, and parliamentary affairs (Askari, Younas, & Saeed, 2020). Aquila Optimizer (AO) is motivated by the Aquila's behaviors in nature during the process of catching the prey (Abualigah et al., 2021). The Equilibrium Optimizer (EO) is inspired by control volume mass balance models used to estimate both dynamic and equilibrium states (Faramarzi, Heidarinejad, Stephens, & Mirjalili, 2020). Learner performance based behavior algorithm (LPB) is inspired from the process of accepting graduated learners from high school in different departments at university (Rahman & Rashid, 2021). The Sine Cosine Algorithm (SCA) is based a mathematical model based on sine and cosine functions (Gabis, Meraihi, Mirjalili, & Ramdane-Cherif, 2021; Mirjalili, 2016). The Honey Badger Algorithm (HBA) is inspired from the intelligent foraging behavior of honey badger, to mathematically develop an efficient search strategy for solving optimization problems (Hashim, Houssein, Hussain, Mabrouk, & Al-Atabany, 2022). Several random and adaptive variables are integrated to this algorithm to emphasize exploration and exploitation of the search space in different milestones of optimization. In (Dhal et al., 2019), a good classification of nature-inspired optimization algorithms is presented and 132 of them are mentioned. Also in (Bonyadi & Michalewicz, 2017; Slowik & Kwasnicka, 2020), Evolutionary algorithms and their applications to engineering problems are reviewed. In References (Carrasco, García, Rueda, Das, & Herrera, 2020; Derrac, García, Hui, Suganthan, & Herrera, 2014; Eftimov & Korošec, 2019), provide useful statistical methods for comparing meta-heuristic and heuristic optimization algorithms. In these articles, good comparisons have been made using statistical methods between different algorithms.

A new optimization method has been presented in this paper inspired by the collective intelligence of the human to obtain felicity. The main goal of the human being in life as the most intelligent creature in the world is attaining felicity and inner tranquility. This concept has been ambiguous and complicated for thousands of years since human creation. However, each school of thought to the world and holy prophet has interpreted this concept in a different way. The common expression in all of these methods is the point that the way to happiness and felicity passes through changing attitudes and thoughts about the world. Inspired by this topic, the human felicity algorithm (HFA) is presented in this article. In this regard, the dimensions of the optimization problem are the dimensions of society's intellectual space. The ultimate goal of optimization is to find the best state of mind of people in society to achieve a sense of well-being.

Most evolutionary optimization algorithms are inspired by the performance of organisms such as bees, frogs, fireflies, bats, wolves, or natural phenomena such as the law of gravity, the water cycle in nature. These very limited evolutionary optimization algorithms are inspired by human life. Among these algorithms, none have been inspired by the promotion and optimization of spirituality and facility within man. Many religious, philosophical, spiritual and even martial arts schools have found varied, effective and efficient methods to promote facility human beings. Therefore, in this article, by examining the evolution and promotion of facility human beings throughout history, the HFA algorithm has been designed and proposed. To evaluate the performance of HFA algorithm, this algorithm has been compared with seven evolutionary optimization algorithms. Various standard functions have been used for comparison. To discover the differences between the algorithms, Friedman and Nemenyi post-Hoc test, which are the most powerful and most suitable nonparametric methods, have been used.

Classical engineering optimization problems and golden standard results from analytical methods have also been used to compare the performance of algorithms. The simulation results inform the proper performance of HFA compared to other simulated algorithms.

The remainder of this article is organized as follows. Section 2 the purpose of humans in life. In other words, the goal function of human life is studied throughout history. Section 3 deals with the description of the reason why population perspectives change regarding the existence of different schools of thought in society. This type of change occurs in three phases: 1) perspective change of the society regarding personal experiences, 2) perspective change due to the effect of people on one another, and 3) Sudden change in public opinion due to unknown factors or special circumstances such as war. The artificial HFA has been described in section 4 based on the three phases mentioned and the simulation results of the proposed algorithm in exchange for different test functions have been shown in section 5. The simulation results state that HFA in complicated multidimensional functions has more appropriate performance than other algorithms.

2. Ower what is human looking for in life?

The ultimate goal of human beings in life, science development, inventions, and explorations is to attain felicity and perfection. The way to achieve evolution and felicity has always been considered by philosophers and elites in human societies. Throughout history, elites and prophets have prescribed different things to help people attain their happiness and felicity. The emergence of different schools of thought to the world, religions, different ways to achieve perfection, and the current social condition indicate that the best and ultimate prescription to attain felicity that is acceptable to all and applicable to all cultures and races have not yet been found. Although it is obvious that people with different opinions may attain the same sense of inner peace and tranquility.

The presence of different solutions presented by schools of thought and the evolutionary trend of human society states that human beings have not found more suitable ways to attain felicity compared with cavemen. If the feeling of felicity and happiness for human beings has not increased throughout history, it must be admitted that human beings must correct their path. Science and knowledge that does not give a human a better sense of living and does not get him any closer to felicity do not have a great value. There have been a lot of distressed wealthy people and elites who have made great damages to humans and have not used their knowledge, talent, and materialistic and intellectual wealth to make humanity perfect.

Although the concept of happiness and felicity is still vague and complicated, what is apparent is that felicity belongs to a society that has more peace and tranquility. It can be said that wealth and power do not count as happiness and felicity because although they bring welfare, they are not peacemakers. “Felicity is a choice. Peace is a state of mind. Both are free”. People who are equal in terms of financial, social, political, family and conditions do not have the same perception of felicity, because felicity is an inner feeling. That is why it all depends on who we are, not who we own.

The concept of inner being is a feeling of felicity that is a function of the attitude towards different aspects of life, which has been expressed by most philosophers and elders of different schools. For example, Maurice Maeterlinck says,“ The blossom of felicity is hidden in a dark, silent and deep place which is very close to us, but we hardly ever pass that place and that is our own heart.” Moreover, Lenau believes that “most people seek felicity as if they are looking for their hats on their heads”. Pythagoras also says, “We should get familiar with our soul and seek felicity in the depth of our heart and soul”.

3. The human solution to achieve felicity throughout history

3.1. The emergence of elites in society and their influence on the general population

Throughout history, every human being has had a different orientation and perspective on the world from others. Meanwhile, the ones who have found a greater sense of felicity and inner peace have been known as sages, intellectual leaders of a special school, elites, or philosophers in that society. Some people in that society start looking for their lost gem as the disciples of the school in hope of attaining a sense of inner peace and perfection. Most of these biased disciples find following the instructions presented by their school of thought to the world as the only way to felicity and see them independent of being affected by and learning from other school’s instructions. However, their perspectives and attitudes to life are similar to those of teachers who are the same elites of society. In fact, a school of thought influences them. Therefore, the followers of a special school of thought have similar and often equal attitudes in life. On the other hand, all people in a society do not coordinate their thoughts as disciples with just one particular elite and just one particular school, although they may be influenced by that school of thought to the world. Furthermore, most people are usually affected by the thoughts of some elites. The wiser an elite is, the more followers and disciples ha has and the greater effect he can have on general people. It is also clear that there is a greater sense of felicity and happiness in society people where there are different attitudes and thoughts. According to Walter Lippmann “When all think the same, then no one is thinking”.

3.2. Seeking felicity based on personal experiences

In addition to being influenced by elites, every member of society changes his thoughts to the world based on his personal experiences in the hope of attaining peace. These personal searches, which are not influenced by others, guarantee the concept that all should not think the same so as not to inhibit the society from being dynamic. According to Ibn-e-Sina,“ some are so engaged with the past scientific heritage that they do not have the chance to refer to their mind and if they do, they not willing to fix their blunders. All of these changes may not cause a better condition, but they ultimately and constantly increase the chance to make the society and the individuals approach their exalted goals, so one should not be afraid of using his personal experiences and making mistakes in his decisions; instead, he should consider risk-taking and evolution and be interested in finding an exalted direction as something sacred. People who are afraid of taking risks rarely enjoy the sweet success of winning.” Edison also believes that “Many of life’s failures are people who did not realize how close they were to success when they gave up. Albert Einstein also says, “A person who made a mistake never tried anything new”.

3.3. Revolution and evolution in the thoughts to the world of some people in society

Furthermore, to the aforementioned changes and the point that there is always a significant relationship between perspectives before and after changes, some people’s minds may suddenly and quite randomly change and evolve. Conditions like war, famine, natural disasters, political and social changes, loss of a dear one can influence the rate of a sudden change in people’s opinions. For many geniuses, drastic emotional changes have taken place in a short period that has changed their outlook. Their worldview and sense of well-being and peace may also increase dramatically.

4. Human felicity algorithm

Regarding the points mentioned in section 3 about the three categories of people in society, if we consider the evolutionary trend of that

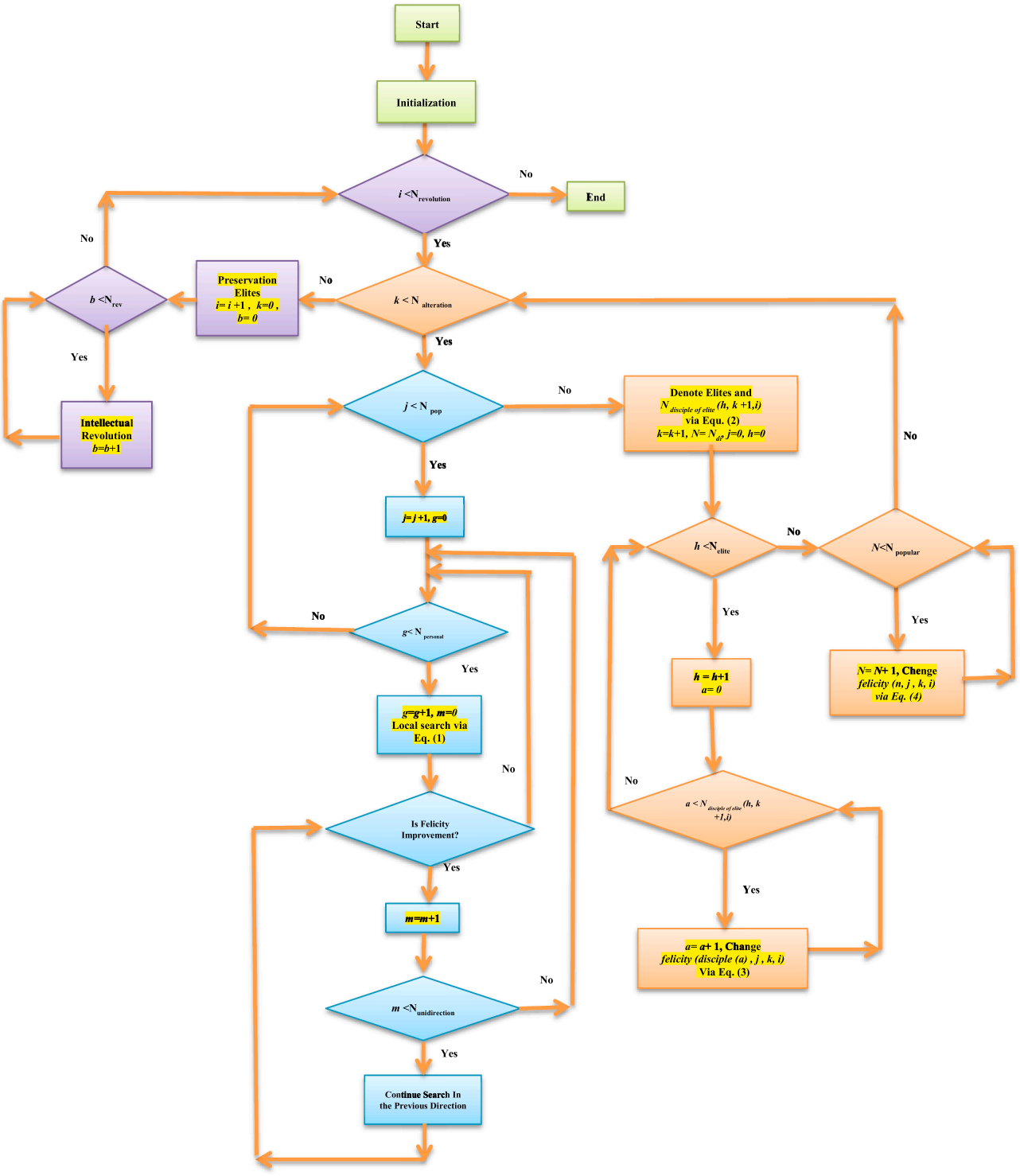

Fig. 1. Flowchart of HFA.

society throughout history, we can generally say that every individual has an idea and starts trying to attain peace based on his personal experiences which in turn makes elites appear in the society.

Section 3 discusses three ways to change people's attitudes. By observation the evolution of society throughout history, it can say that people's thinking changes based on what they have learned, personal experiences, and spiritual revolutions. This change in thinking and worldview change a person's sense of happiness and felicity. Elites teach limited people as disciples. One or more elites influence other members of society, and in addition, each person's perspective changes based on personal experiences. Meanwhile, the emergence of new elites in society is possible. Ideas and schools of thought to the world have always been discarded due to the emergence of new elites and schools of thought and the provision of more appropriate solutions. The trend of personal experiences, elite appearance, and social change by the influence of elites continues continuously. At some point in time, for reasons such as war and natural disasters, part of the existing thoughts in society undergo sudden changes, and then the same process and trend goes on and on.

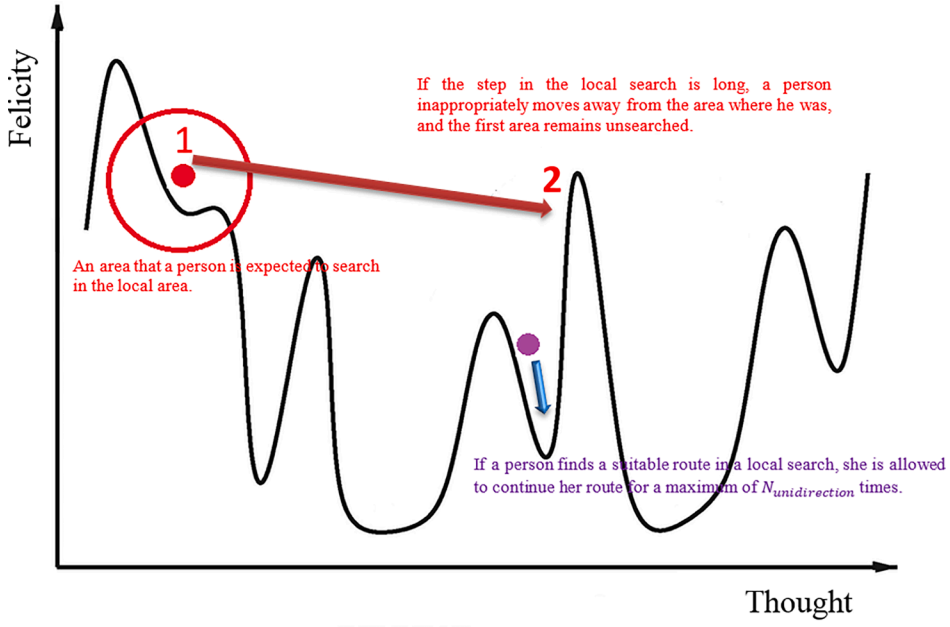

Fig. 2. The concept of local search in the HFA.

The correct application of the way society evolves makes the algorithm even more complicated, so some of these simplifications have been considered:

1) The amount of an individual's felicity is measurable regarding his perspectives towards different dimensions of life. In fact, in each society, it is completely obvious what parameters individuals' felicity depends on and if the rate of each parameter is clear, then felicity is measurable. However, the maximum rate and the amount of each parameter are not clear.
2) Individuals in society are only influenced by elites and others are not influential in changing their perspectives.
3) Society is aware of the amount of felicity for all individuals.
4) After mortality, birth, and immigration, the population always remains constant.

Npop is the number of people in society and if felicity depends on P parameter, first everyone randomly belongs to a P dimensional environment of thoughts. Three events occur in society: 1) local search, 2) elite determination and intellectual movement toward them, and 3) sudden changes in individual perspectives. These three events are shown by j, k, and i, respectively. $felicity(n,j,k,i)$ is the felicity of n^{th} person in society in his j^{th} local search, k^{th} movement, and i^{th} revolution. The flowchart of the proposed algorithm has been shown in Fig. 1.

4.1. Local search (thought change based on personal experiences)

Each individual has an attitude towards each parameter in society. He changes one or more dimensions of it to attain more felicity. These changes are random and based on Eq. (1). As pointed in section 212, these changes in thoughts are not based on getting influenced by elites.

$$felicity(n,j+1,k,i) = felicity(n,j,k,i) + s(n)*Đ¤ \quad (1)$$

In which $s(n)$ is the n^{th} individual's step length and Φ is a random vector. In fact, $s(n)$ determines the amount of changes, and Φ shows the direction. $s(n)$ is an important parameter. Assigning a large or very small size to this parameter makes it difficult to find the optimal point. This is shown in Fig. 2. Suppose a person's position is in position 1 in Fig. 2. This person is represented by a solid red circle. If the size $s(n)$ is too large, this person goes from point 1 to point 2 in the first step of the local search. This causes the area near position 1 to be left without a thorough search. Therefore, this step is not appropriate. If a small $s(n)$ is selected, the area that each person is searching for is very small. Ideally, it is expected to search the area marked with a red circle. If the radius of this circle is large, an accurate search will not take place in this area, and if the radius of this circle is small, the appropriate area will not be searched. If changing attitudes based on personal experiences heads to higher tranquility, people in the society would follow it in the same direction. The number of movements in this direction cannot of course be unlimited. The maximum number of movements in one direction can be $N_{unidirection}$ and the maximum number of local searches is $N_{personal}$. This is explained in Fig. 2. Consider a person in a solid purple circle position. There are two paths for this person, one is the path to the summit on the left and one is the path to the valley on the right. If this point moves in the desired direction (moving in the valley path), the proposed algorithm allows him to move the $N_{unidirection}$ step in this desired direction. The flowchart of the local search has been shown in Fig. 3.

4.2. Changing society thoughts to the world influenced by elites

As mentioned before, elites who attain higher felicity influence population thoughts; however, a dynamic society should have more than one elite. Also, some people in the community may not be influenced by the elites of the community. The maximum frequency of changing population thoughts influenced by elites is $N_{alteration}$.

The HFA for a society with 15 populations is symbolically shown in the Fig. 4. In this Figure, the thought are considered two-dimensional. Every member of society has an idea in this space. Because Socrates, Plato, and Aristotle achieved a greater sense of Felicity, they were chosen as the elites of society. There are 5 disciplines in society. Aristotle has one disciple and Socrates and Plato each have two disciplines. As can be seen in Fig. 4, the disciple is influenced by one elite and is close to the elite in terms of thinking. There are also 7 ordinary people in this society whose thoughts are influenced by one, two or three elites. Arrows point to move people's thoughts. The way disciplines move towards the elites is symbolically shown in Fig. 5.

4.2.1. Determining an elite

It is not appropriate to just look at people's current situation in society to determine the elites. Experiences and the trend of people's

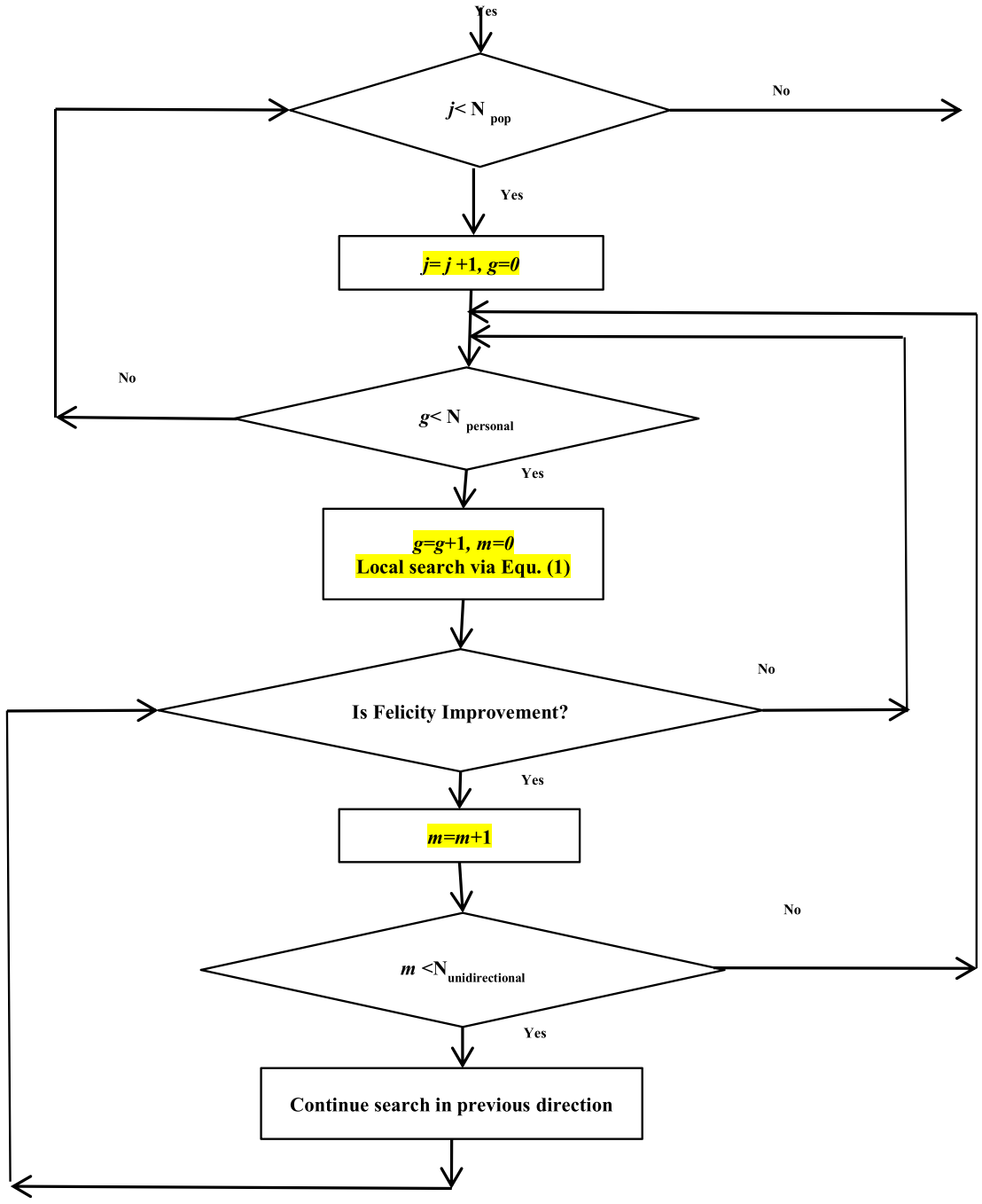

Fig. 3. Flowchart of local search in HFA.

evolution should also be considered in determining elites, but very long in the past should not be considered as a criterion in his determination because an individual's perspectives change through times. Thus, to choose an elite, a near pastime or present can be suitable criteria. In the proposed algorithm, the average level of felicity in the $N_{calculatelite}$ previous local search is the criterion for determining the elites of the community. The number of elites in society is shown by N_{elite} from which 5–10% of the whole population would be a suitable choice.

4.2.2. Training disciples in schools of thought to the world

Each elite has a considerable influence on the thoughts of some people in society so that their disciples start changing their thoughts rapidly and coordinate themselves with him. These people as disciples of a school of thought see the world from the angle of the elites of their school since they are trained by them. Elite with a better status has more disciples.

The total number of disciples is $N_{disciple}$. Allocating 25–35% of the total population to this number can be a suitable configuration in most optimization problems. After ranking the felicity of the people in the society, the number of disciples of each elite is determined by Eq. (2).

$$N_{\text{discipleofelite}}(h,k+1,i)=\frac{\frac{1}{\sum_{j=1}^{N_{personal}} felicity(h,j,k,i)}}{\sum_{n=1}^{N_{elite}}\frac{1}{\sum_{j=1}^{N_{personal}} felicity(h,j,k,i)}}*N_{disciple},\ h\le N_{elite} \tag{2}$$

In which $N_{\text{discipleofelite}}(h,k+1,i)$ are the number of the disciples of the h^{th} elite in the k + 1th movement and the i^{th} thought revolution. As can be seen in Eq. (2), the criterion for selecting elite is not just its current position. The past and recent history of individuals in society are

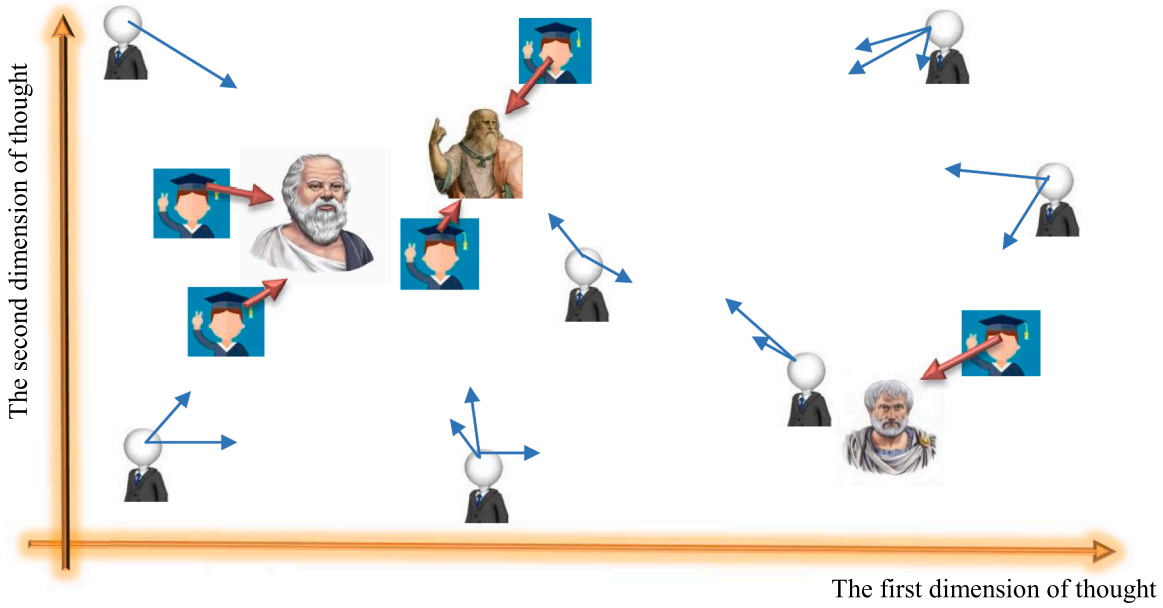

Fig. 4. HFA for a symbolic society.

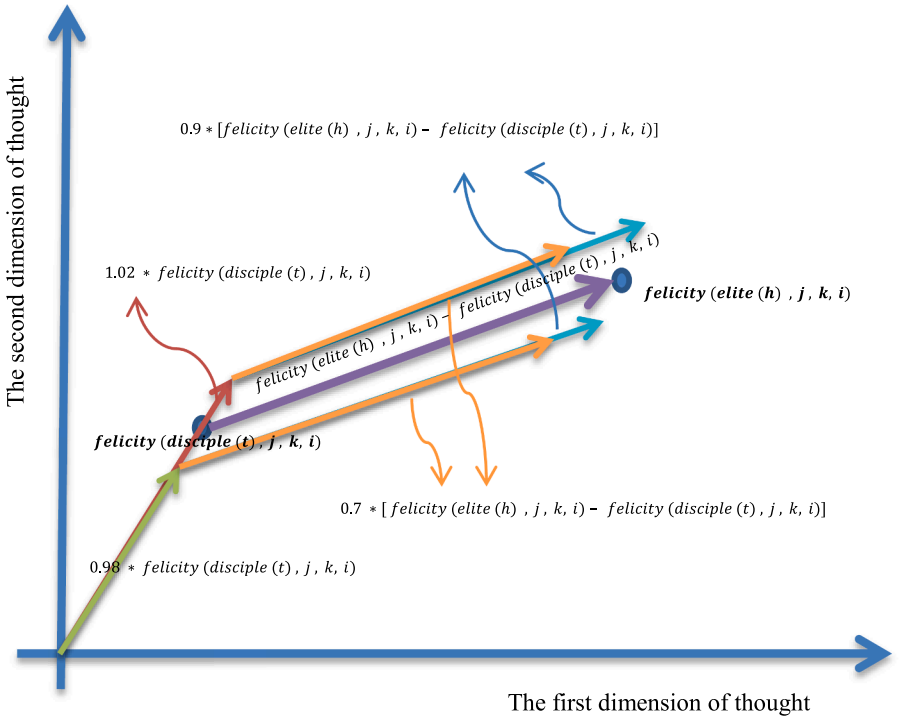

Fig. 5. Change the disciples of the felicity influenced by the elite.

influential in determining the elite. In the HFA for selecting the elite, the sum of a person's positions in $N_{personal}$ local search in the previous movement is important. In other words, an individual from society is considered an elite who has less $\sum_{j=1}^{N_{personal}} felicity(h,j,k,i)$. Since elite with less $\sum_{j=1}^{N_{personal}} felicity(h,j,k,i)$ should have more disciples, Eq. (2) is suggested to allocate the number of disciples per elite. To better understand, let us give an example. Suppose that for an optimization problem, the number of people in the community is 50, the number of elites is 3 and the number of disciples is 7. Assume $N_{personal}$ is 4 in this example. Suppose we are in movement 10, revolution 2. For movement 11, we selected the elites by calculating $\sum_{j=1}^{4} felicity(h,j,10,2)$ for each section of the community. We are now in the process of assigning the number of disciples to each elite. Suppose in local searches the value $felicity(h,j,k,i)$ was obtained for elite 1 values of 1, 1.1, 1.2, 0.9, for elite 2 values of 1.2, 1.1, 1.3, 1.2 and for the third elite values of 1.4, 0.9, 1 and 1.6.

According to Eq. (2) to the first elite number $\frac{\frac{1}{4.2}}{\frac{1}{4.2}+\frac{1}{4.8}+\frac{1}{4.9}} * 7 = 2.56$

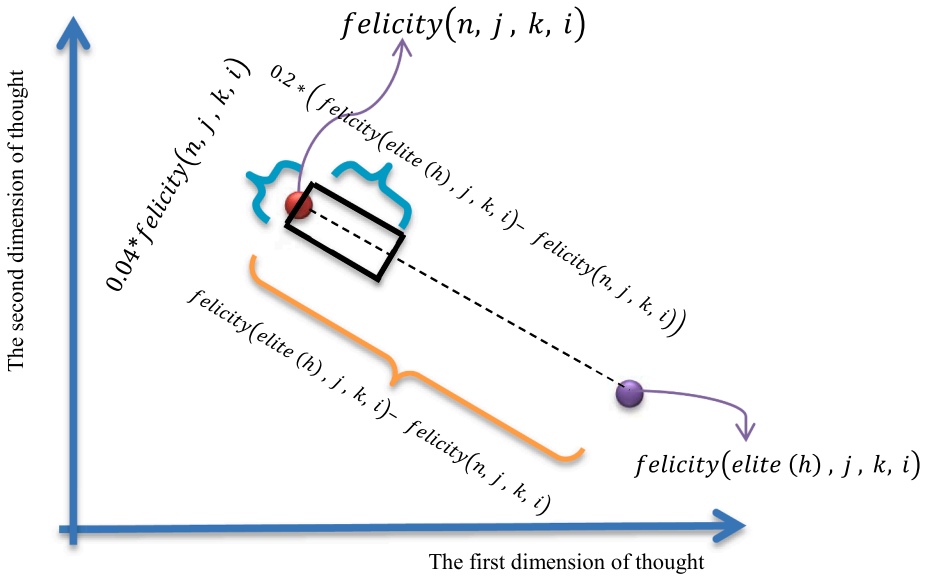

Fig. 6. The effect of α on the location of ordinary people in society when updating thoughts.

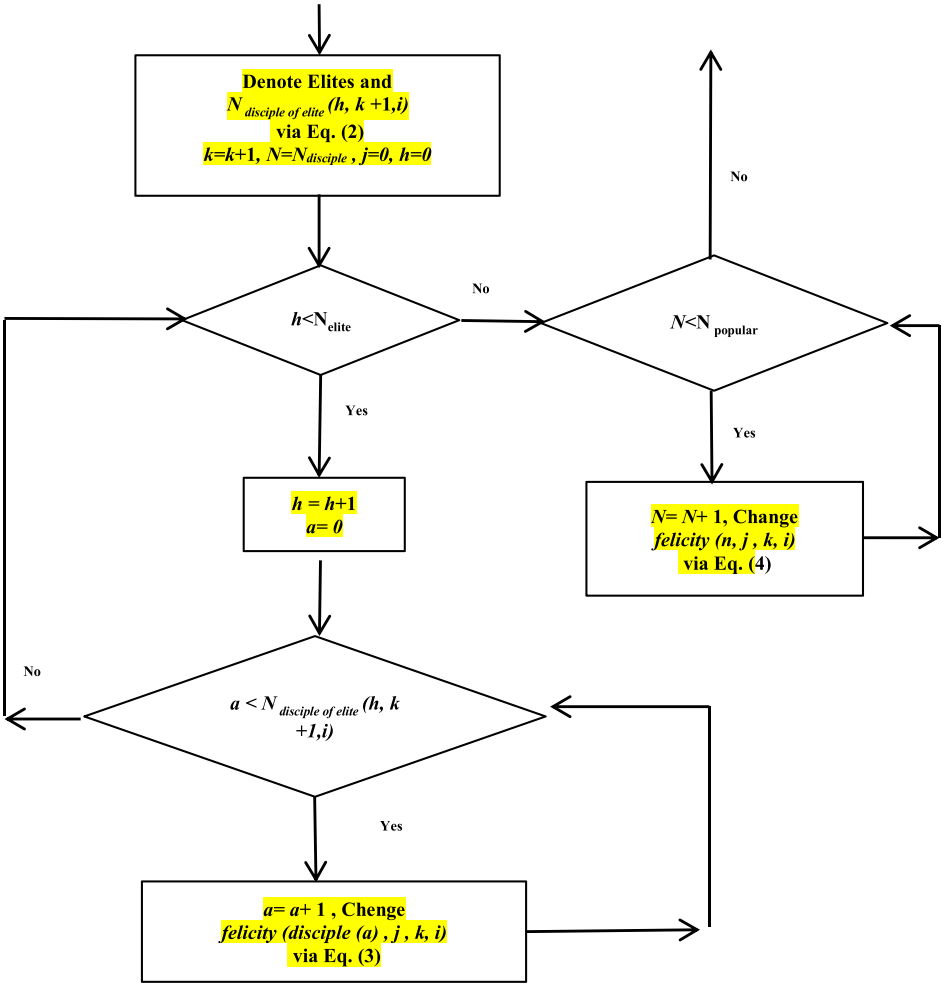

Fig. 7. Flowchart of changing people influenced by the elite.

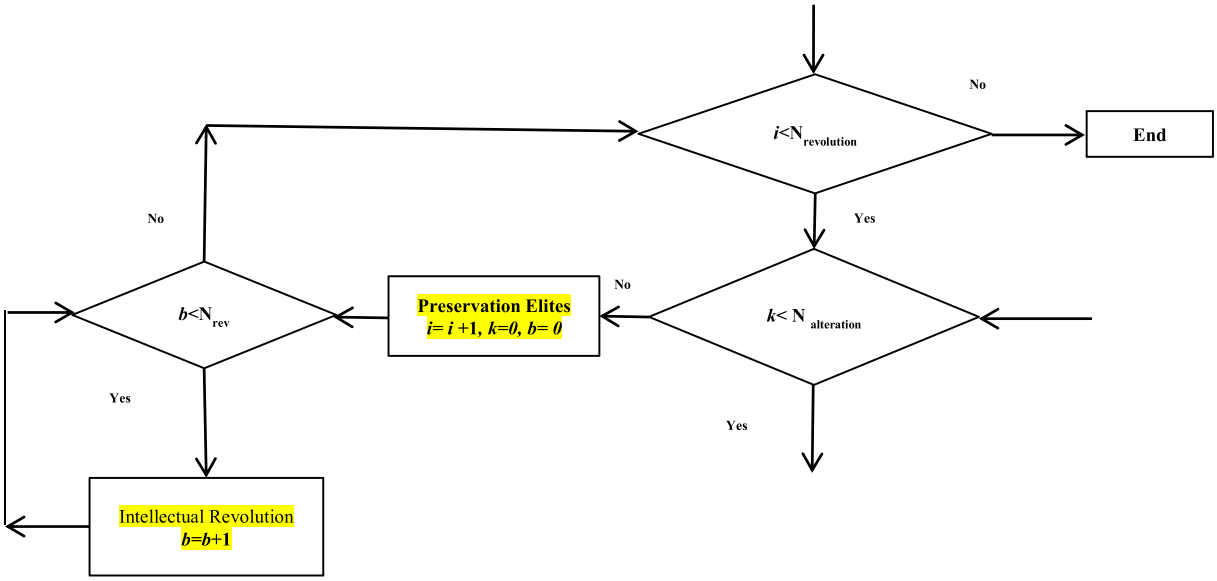

Fig. 8. Flowchart of intellectual revolution.

disciples, to the second elite number $\frac{\frac{1}{4.8}}{\frac{1}{4.2}+\frac{1}{4.8}+\frac{1}{4.9}}*7 = 2.24$ disciples, to the third elite number $\frac{\frac{1}{4.9}}{\frac{1}{4.2}+\frac{1}{4.8}+\frac{1}{4.9}}*7 = 2.19$ disciples is assigned. Since the number of disciples must be an integer, we assign 3, 2 and 2 disciples to the first to third elites, respectively.

The changes in the attitude of the disciple under the influence by elites are shown in Eq. (3).

$$felicity(disciple(t),j,k+1,i) = w*felicity(disciple(t),j,k,i) + c*(felicity(elite(h),j,k,i) - felicity(disciple(t),j,k,i)) \quad (3)$$

in which $elite(h)$ is the h^{th} elite and $disciple(t)$ is the t^{th} disciple of the elite. $felicity(elite(h),j,k,i)$ are the *felicity* of the h^{th} elite in j^{th} local search, k^{th} movement, and i^{th} revolution.$felicity(disciple(t),j,k,i)$ is the felicity of the t^{th} disciple of elite in j^{th} local search, k^{th} movement, and i^{th} revolution. To escape the local minima, the values c and w are randomly selected. It should be noted that the closer the values of c, w are to the number one, the more the disciple will look like an elite. If in Eq.3, $w = c = 1$, then $felicity(disciple(t),j,k+1,i) = felicity(elite(h),j,k,i)$, in other words, the disciple thinks exactly like his elite their felicities are equal. If $w = 1$, $c \neq 1$, the intellectual location of the t^{th} disciple changes along the connection line between the h^{th} elite and the t^{th} disciple, but for the t^{th} disciple to be able to cover more space around the h^{th} elite, as in Fig. 5, $0.98 <$ w less than 1.02 and $0.7 <$ c less than 0.9 are chosen randomly (the ranges of changes wandc can change in different problems).

4.2.3. Changing general people's thoughts influenced by elites

Most people in a society do not see themselves as disciples of a special school of thought, thus they do not see the world from those elites' perspectives. However, they may gain a new perspective on the world influenced by the thoughts and ideas of one or more elites. According to the above, the new location of ordinary people in society after moving to the elite is calculated from Eq. 4.$felicity(n,j,k+1,i) = \alpha*felicity(n,j,k,i) + \beta*(felicity(elite(h),j,k,i) - felicity(n,j,k,i))$(4)

If the n^{th} individual in society is only influenced by the h^{th} elite, his changed perspectives can be calculated by Eq. 4. In Eq. 4,$0.98 < \alpha < 1.02$ and $0 < \beta < 0.2$ is a special random for the h^{th} elite.

To escape the local minima, the values α, β are randomly selected. The larger β, the more an individual in society becomes like an elite. Since we do not want ordinary people in society to look like elites, we consider the limit $0 < \beta < 0.2$. If $\alpha = 1$, the new location of ordinary people in the community is between a straight line from person to elite. In order to allow people to search in more areas, we consider $0.98 < \alpha < 1.02$. This point is shown in Fig. 6. In this figure, the elite is shown in purple and an ordinary person in society is shown in red. If $\alpha =$ 1, the person can only move on the dashed line shown in Fig. 6. If, in addition to the above limit for β, α is selected, the new position of the individual may be placed anywhere inside the black rectangle. We must note that if α goes too far away from the number 1, the person loses his previous experience and the place of his previous thoughts, and in practice the probability of reaching the optimal point decreases. In addition to the stated challenge, by moving α away from one, the possibility of people going out of range is greatly increased. Because each dimension usually has an allowable range, when updating the position of individuals in the community, selecting very small or very large values for α increase the likelihood of going out of range.

If one's thoughts are influenced by some elites, Eq. 4 is used once for each elite's thoughts influencing him. It should always be noted that the elite is more felicity, the more influential he will be on people. The flowchart and the way people's thoughts are influenced by elites in society are presented in Fig. 7. A person in a society may be influenced by several elites, in which case for each person according to the number of elites influencing him, Eq. 4 should be used.

The most important difference between general people and disciple of a school of thought to the world is that generally people are not strongly influenced by a school of thought or elite, and they are less influential than the elite and have high inertia when they change their thoughts. Not all members of society are directly influenced by elites and schools of thought to the world. The number of people in the community who are influenced by the elite is $N_{popular}$.

Table 1
The pseudocode of HFA.

```
Initialize all parameters (Npop , Nelite , Ndisciple , Npopular , Nrev , s (n) ,
Npersonal , Nunidirection , Nalteration , Nrevolution)
Assign each person a Felicity at random
  for Nrevolution
    for Nalteration
      Identify the Nelite elite
      Count the number of disciples in each elite according to Eq. (2)
      Change the Felicity of disciples according to Eq. (3)
      Change the Felicity of Npopular people according to Eq. 4
        for Npersonal
          Local search for each person with s (n) according to Eq. (1)
          for Nunidirection
            if Felicity increases then continue the previous direction
            end if
          end for
        end for
      end for
The mutation for Nrev number of people who have the least Felicity
end for
```

4.3. Thought revolution

Based on the reasons like mortality, war, the death of loved ones, and special living conditions, there may be a severe intellectual and spiritual transformation in some people in the society, which can be the source of good changes in the society. Due to some other reasons, some people's attitudes may also experience great change. What occurs in all the above changes is in the formation of a new attitude in society regardless of the previous attitudes. In HFA, the attitude of N_{rev} is removed from the society and N_{rev} new attitude is added to it. Since an intelligent society prevents the destruction of its elites, in HFA, all the people except the elites can experience thought revolution. The number of people in the society who experience intellectual revolution at any stage is N_{rev}. The maximum frequency of a revolution by public opinions is $N_{\text{revolution}}$. The flowchart of the revolution in society thoughts is presented in Fig. 8.

Deep intellectual transformations occur in special circumstances, but the influence of the elite and the search for perfection based on personal experience always occur. In the proposed algorithm, after $N_{\text{alteration}}$ times of influence of elites and N_{pop} times of local search, N_{rev} intellectual revolution takes place.

According to the contents of this section, to better understand the proposed algorithm, the pseudocode of the HFA is presented in Table 1.

5. Validation and comparison HFA with some other optimization algorithms

It is relatively straightforward to implement the HFA in any programming language. For the ease of visualization, we have implemented it using Matlab software for various test functions. The IEEE Congress on Evolutionary Computation (CEC) introduces benchmark objective functions each year to measure and compare the ability of evolutionary optimization algorithms to find the optimal minimum point. This set of functions is usually known by the year as standard functions introduced by the IEEE Congress on Evolutionary Computation, for example CEC 2014. These benchmark functions are standard and can well measure the various features of optimization algorithms such as the ability to escape the local minimum, the speed of the algorithm, the convergence of the algorithm. In order to better evaluate the performance of the proposed algorithm in solving problems, in the first step CEC 2014 benchmark test function (Liang, Qu, & Suganthan, 2013) and in the second step, the CEC 2019 test function (Price, Awad, Ali, & Suganthan, 2018) and in the third step CEC 2020 Benchmark Test Functions (Yue et al., 2019) have been used. Convergence analysis in the proposed optimization algorithm for CEC 2019 test function has also been performed. Also, in this section, to evaluate the ability of the proposed algorithm to solve classic engineering optimization problems, 4 standard optimization problems are examined and compared. Also, in all three steps, for accurate comparison of the proposed algorithm, the outputs of the HFA are compared with the outputs of other optimization algorithms. Particle swarm optimization (PSO), chaotic cuckoo search (CCS) , Ageist Spider Monkey Optimization (ASMO), elephant herding optimization (EHO), variable neighborhood bat (VNBA), Red fox optimization (RFO), and Coyote Optimization Algorithm (COA) have been selected for comparison with the proposed algorithm due to their novelty, good performance and high citations to researcher's scientific publications in recent years. In the following, Friedman and Nemenyi post-Hoc test (Carrasco et al., 2020; Derrac et al., 2014) which is a suitable method for comparing the average ranking of different groups, is used to rank the above seven algorithms.

Tuning the parameters of optimization algorithms is very important. If the parameters of the optimization algorithms are not set to the appropriate value, the possibility of achieving the optimal solution is greatly reduced. In addition, to test the power of an optimization algorithm in finding the optimal point and adjusting its parameters to the appropriate value. In this paper, valid articles and trial and error have been used to set the parameters of optimization algorithms. For Efficient tuning of PSO parameters, methods of articles (Isiet & Gadala, 2020; Mashayekhi, Harati, & Estekanchi, 2019; Wang, Geng, & Qiao, 2014) have been utilized. For proper adjustment of CCS parameters, points

Table 2
Parameters of optimization algorithms.

PSO	$c1 = c2 = 1.5$, $w = 1$,
CCS	$N = 10$, $p_a = .025$, $\beta = 1.5$
ASMO	$GLL = 20$, $LLL = 500$, $SS = 32$, $MG = 4$
EHO	$n_c = 8$; $\beta = 0.1$, $\alpha = 0.3$, $\gamma = 0.05$
VNBA	$A = 0.5$, $r = 0.5$, $\varepsilon = 0.001$
RFO	$a = 0.2$,$1 = \varnothing_0$
COA	Number of coyotes in each pack = 10

Table 3
Summary of the CEC 2014 test functions.

Types	No	Functions	Ideal Fitness
	F01	Rotated High Conditioned Elliptic Function	100
	F02	Rotated Bent Cigar Function	200
	F03	Rotated Discus Function	300
	F04	Shifted and Rotated Rosenbrock's Function	400
	F05	Shifted and Rotated Ackley's Function	500
	F06	Shifted and Rotated Weierstrass Function	600
	F07	Shifted and Rotated Griewank's Function	700
	F08	Shifted Rastrigin's Function	800
	F09	Shifted and Rotated Rastrigin's Function	900
	F10	Shifted Schwefel's Function	1000
	F11	Shifted and Rotated Schwefel's Function	1100
	F12	Shifted and Rotated Katsuura Function	1200
	F13	Shifted and Rotated HappyCat Function	1300
	F14	Shifted and Rotated HGBat Function	1400
	F15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
	F16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
	F17	Hybrid Function 1	1700
	F18	Hybrid Function 2	1800
	F19	Hybrid Function 3	1900
	F20	Hybrid Function 4	2000
	F21	Hybrid Function 5	2100
	F22	Hybrid Function 6	2200
	F23	Composition Function 1	2300
	F24	Composition Function 2	2400
	F25	Composition Function 3	2500
	F26	Composition Function 4	2600
	F27	Composition Function 5	2700
	F28	Composition Function 6	2800
	F29	Composition Function 7	2900
	F30	Composition Function 8	3000

Table 4
Mean and SD value algorithms for CEC 2014 benchmark functions of 30D.

FN	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	3.47E+07	1.69E+07	2.93E+09	8.68E+08	1.20E+08	3.55E+07	4.83E+08	1.30E+08	1.21E+08	4.42E+07	1.93E+08	4.94E+07	3.34E+07	1.63E+05	1.03E+08	1.30E+06
F2	6.75E+06	2.56E+06	9.48E+10	6.53E+09	1.95E+09	2.81E+08	3.82E+10	7.67E+09	2.81E+08	1.24E+08	1.35E+10	2.58E+09	2.14E+09	6.91E+05	2.06E+09	6.91E+05
F3	2.10E+04	1.23E+04	1.01E+07	1.32E+07	2.45E+05	7.65E+04	5.64E+04	6.17E+03	2.63E+04	1.64E+04	8.73E+04	6.27E+03	1.66E+05	1.79E+02	1.78E+05	2.09E+02
F4	5.40E+02	3.96E+01	2.43E+04	3.94E+03	7.11E+02	3.56E+01	5.45E+03	1.17E+03	6.29E+02	7.41E+01	2.18E+03	4.07E+02	4.88E+02	2.52E-01	3.98E+02	8.54E-01
F5	5.16E+02	7.79E-02	5.03E+02	6.33E-02	5.11E+02	9.73E-02	5.05E+02	5.80E-02	5.15E+02	5.43E-02	4.99E+02	4.59E-02	4.98E+02	3.88E-04	4.88E+02	3.62E-03
F6	6.25E+02	3.01E+00	6.04E+02	2.00E+00	6.22E+02	2.79E+00	6.18E+02	1.81E+00	6.16E+02	2.24E+00	6.00E+02	1.63E+00	5.99E+02	1.36E-02	4.37E+02	1.65E+00
F7	7.02E+02	2.32E+00	1.66E+03	1.28E+02	7.04E+02	3.15E+00	9.89E+02	4.51E+01	7.02E+02	1.18E+00	7.01E+02	1.60E+01	7.00E+02	1.12E-02	7.01E+02	1.01E+01
F8	9.04E+02	1.73E+01	1.16E+03	2.51E+01	1.06E+03	1.96E+01	1.05E+03	1.93E+01	8.15E+02	3.41E+00	8.02E+02	1.24E+01	8.01E+02	6.82E-02	4.21E+02	1.18E+00
F9	1.13E+03	3.17E+01	1.33E+03	3.95E+01	1.17E+03	1.84E+01	1.21E+03	3.04E+01	1.03E+03	2.69E+01	9.03E+02	2.09E+01	9.02E+02	1.17E+02	9.02E+02	2.28E+02
F10	1.54E+03	3.82E+02	8.06E+03	3.28E+02	9.01E+03	4.41E+02	7.69E+03	3.53E+02	1.20E+03	7.82E+01	3.18E+03	2.48E+02	1.00E+03	1.37E+03	1.87E+03	1.82E+03
F11	4.96E+03	7.91E+02	8.83E+03	5.25E+02	9.80E+03	4.16E+02	7.96E+03	3.49E+02	4.96E+03	5.89E+02	4.43E+03	3.85E+02	1.10E+03	3.03E+00	4.98E+03	5.21E+02
F12	1.21E+03	3.27E-01	1.21E+03	3.95E-01	1.21E+03	8.09E-01	1.21E+03	3.10E-01	1.21E+03	1.37E-01	1.21E+03	2.15E-01	1.21E+03	2.16E-01	1.21E+03	1.56E-01
F13	1.31E+03	6.20E-01	1.31E+03	1.57E+00	1.31E+03	1.97E-01	1.31E+03	2.11E-01	1.31E+03	1.08E-01	1.31E+03	2.80E-01	1.31E+03	1.56E-01	1.31E+03	2.42E-01
F14	1.42E+03	1.25E+01	1.67E+04	4.59E+01	1.41E+03	1.34E+00	1.47E+03	1.04E+01	1.41E+03	1.63E-01	1.41E+03	7.71E+00	1.41E+03	2.36E-01	1.41E+03	8.74E-01
F15	1.51E+03	8.97E+00	1.81E+06	8.34E+05	1.53E+03	2.18E+01	2.89E+05	1.14E+05	1.52E+03	2.46E+01	1.09E+05	3.85E+04	3.40E+04	9.34E+00	3.84E+04	2.28E+00
F16	1.62E+03	5.09E-01	1.61E+03	2.60E-01	1.61E+03	2.06E-01	1.61E+03	2.44E-01	1.61E+03	5.08E-01	1.61E+03	2.54E-01	1.61E+03	2.06E-03	1.61E+03	8.51E-03
F17	8.22E+06	5.05E+06	3.82E+08	1.73E+08	8.77E+06	3.78E+06	1.15E+07	4.10E+06	1.47E+07	6.02E+06	9.00E+06	3.09E+06	6.53E+06	2.51E+04	6.41E+06	5.54E+05
F18	5.15E+05	1.92E+05	1.21E+10	3.86E+09	9.30E+07	2.91E+07	3.83E+08	1.45E+08	1.24E+07	6.87E+06	2.03E+08	4.90E+07	2.02E+08	6.14E+03	2.18E+08	8.27E+04
F19	1.96E+03	3.56E+01	2.73E+03	3.70E+02	1.90E+03	4.57E+00	2.09E+03	4.75E+01	1.94E+03	4.05E+01	1.91E+03	2.80E+01	1.91E+03	1.36E-01	2.54E+03	2.05E+01
F20	4.37E+04	2.14E+04	5.99E+06	1.25E+07	7.12E+05	6.42E+05	2.28E+04	5.46E+03	3.70E+04	2.13E+04	5.87E+04	9.43E+03	9.87E+03	1.16E+03	8.81E+04	5.85E+03
F21	1.37E+06	1.11E+06	1.85E+08	9.29E+07	4.34E+06	2.07E+06	2.74E+06	1.22E+06	3.67E+06	1.77E+06	2.51E+06	7.88E+05	2.05E+05	8.37E+03	2.51E+05	3.51E+06
F22	2.85E+03	2.48E+02	3.43E+06	3.84E+06	3.37E+03	1.73E+02	3.27E+03	1.97E+02	2.91E+03	1.48E+02	2.28E+04	1.50E+02	5.60E+04	9.62E-00	8.21E+04	4.58E+02
F23	2.64E+03	3.53E+01	3.73E+03	4.91E+02	2.60E+03	7.32E+00	2.43E+03	1.82E-02	2.63E+03	2.88E+01	2.32E+03	1.19E+01	2.32E+03	1.20E-01	2.63E+03	1.19E+01
F24	2.65E+03	1.11E+01	2.53E+03	6.83E+00	2.61E+03	1.85E+00	2.52E+03	5.54E-03	2.60E+03	1.94E+01	2.42E+03	3.75E+00	2.42E+03	5.45E-01	2.41E+03	3.21E-01
F25	2.70E+03	5.71E+00	2.66E+03	5.95E+01	2.67E+03	4.88E+00	2.62E+03	1.90E-04	2.69E+03	5.08E+00	2.52E+03	1.93E+00	2.52E+03	2.65E-00	5.29E+03	2.62E+00
F26	2.68E+03	3.19E+01	2.79E+03	1.06E+02	2.65E+03	1.58E-01	2.63E+03	1.58E+00	2.67E+03	1.14E-01	2.62E+03	1.13E+01	2.62E+03	5.41E-00	2.62E+03	1.13E+01
F27	4.34E+03	1.02E+02	7.07E+03	8.01E+02	4.17E+03	9.74E+01	2.81E+03	3.37E-03	4.23E+03	9.60E+01	2.72E+03	3.45E+01	2.72E+03	4.99E-01	2.72E+03	3.65E-01
F28	4.33E+03	2.41E+02	1.41E+04	1.91E+03	4.16E+03	1.17E+02	2.91E+03	5.99E-03	4.46E+03	3.50E+02	2.82E+03	8.14E+01	2.82E+03	1.19E+00	2.82E+03	1.06E+00
F29	8.16E+03	2.26E+03	1.10E+09	2.92E+08	1.81E+06	3.15E+06	3.62E+05	2.60E+04	3.11E+06	4.11E+06	6.79E+06	1.37E+04	1.80E+07	1.23E+02	1.26E+07	1.23E+02
F30	1.22E+04	3.09E+03	2.35E+07	1.48E+07	8.86E+04	3.12E+04	5.41E+05	6.55E+04	4.92E+04	2.00E+04	3.29E+05	2.22E+05	3.91E+05	9.19E+02	3.88E+05	2.57E+02
	5	**3**	**2**	**0**	**2**	**1**	**1**	**4**	**2**	**1**	**1**	**0**	**13**	**17**	**4**	**4**
	16	**20**	**21**	**23**	**15**	**17**	**18**	**20**	**20**	**25**	**22**	**21**	**0**	**0**	**17**	**19**

expressed in (Liu, Liu, Wang, & Zou, 2018; Wang, Deb, Gandomi, Zhang, & Alavi, 2014) have been reviewed and used. In order to achieve optimal EHO values, the parameters of this algorithm have been regulated take into account the points expressed in (Elhosseini, El Sehiemy, Rashwan, & Gao, 2019; Li et al., 2019, 2020). To adjust the parameters of the RFO algorithm in appropriate values, useful points in (Połap & Woźniak, 2021; Zhang et al., 2021) have been used. In articles (Bangyal, Ahmad, Rauf, & Pervaiz, 2018; Li, Wang, & Alavi, 2020; Wang et al., 2016) appropriate tips for tuning the parameters of the VNBA are stated. Also for the proper performance of the COA algorithm in order to set the parameters of the optimization algorithm have been used the references (Pierezan & Coelho, 2018; Tong, Zhu, Pierezan, Xu, & dos Santos Coelho, 2021; Yuan, Wang, Wang, & Yildizbasi, 2020).

According to the above, the parameters of different algorithms are

Table 5
The Friedman test based on the results of Table 4.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
Friedman test based on mean values algorithms	3.45	4.99	4.58	4.24	3.88	3.98	2.12	3.44
Final rank algorithms based on mean values	3	8	7	6	4	5	1	2
Friedman test based on SD values algorithms	4.21	4.22	4.60	3.89	3.95	3.84	3.23	3.94
Final rank algorithms based on SD values	6	7	8	3	5	2	1	4

Table 6
Statistical analysis using Non-parametric Nemenyi test for CEC 2014 benchmark functions of 30D, where '1′ indicates significant difference and '0′ otherwise.

	PSO	CCS	ASMO	EHO	VNBA	RFO	HFA	COA
PSO	NAN	0	0	0	0	0	1	0
CCS	0	NAN	0	0	0	0	1	1
ASMO	0	0	NAN	0	0	0	1	0
EHO	0	0	0	NAN	0	0	1	0
VNBA	0	0	0	0	NAN	0	1	0
RFO	0	0	0	0	0	NAN	1	0
HFA	1	1	1	1	1	1	NAN	0
COA	0	1	0	0	0	0	0	NAN

set. The parameters that are constant throughout this article are shown in Table 2. Some parameters are not the same as the initial population in the different parts of Section 5(these particles have different names in different algorithms, for example in the VNBA algorithm are called bats, in the EHO algorithm are called elephants and in the RFO algorithm are called fox), not listed in Table 2. In each part, the initial population are equal for all algorithms and the condition for ending the program is the number of iterations, which is equal for all algorithms.

5.1. CEC 2014 benchmark test functions

IEEE CEC 2014 Benchmark test suite contains 30 single objective test functions that represent real optimization problems. These test functions are divided into four different groups: unimodal functions (F01-F03), simple multimodal (F04-F16) functions, hybrid functions (F17-F22), and composition functions (F23-F30). More details about the benchmark functions can be shown in Table 3.

All seven algorithms are evaluated on 30D, 50D, and 100D Benchmark functions. All algorithms for each optimization function are run 30 times independently in each case and the mean values and standard deviation (SD) of each algorithm in each case are shown in the following tables. In this section, the maximum number of generations for each optimization algorithm is set to 7500.

5.1.1. $D = 30$

All algorithms for each optimization function are run 30 times. The average and standard deviation of the optimal solution are calculated in the last iteration. The mean and standard deviation values of each optimization algorithm for CEC 2014 benchmark functions with 30D are shown in table 4.

The numbers in the penultimate row indicate the number of functions with the best mean or standard deviation for each optimization algorithm. It also can be seen from the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 5, 2, 2, 1, 2, 1, 16, and 4 times optimal mean results from 30 benchmark functions, respectively. In addition, it can be seen in the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 3, 0, 1, 4, 1, 0, 17, and 4 times optimal SD results from 30 benchmark functions, respectively. The last row shows a comparison between the current strategy and the HFA on benchmark functions. According to the last line, the mean of HFA for 16, 21, 15, 18, 20, 22, and 17 benchmark functions was less than the mean of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. Also, the SD of HFA for 20, 23, 17, 20, 25, 21, and 19 benchmark functions was less than the SD of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively.

Table 5 shows the ranking of the algorithms in Table 4 based on the Friedman test. According to mean, HFA obtains the best rank, COA ranks 2, and the following are PSO, VNBA, RFO, EHO, ASMO, and CCS. In addition, According to SD, HFA obtains the best rank, RFO ranks 2, and the following are EHO, COA, VNBA, PSO, CCS, and ASMO. In the Friedman test, the p-value was less than 0.001.

The Nemenyi post-test detects whether there is a significant difference between the two algorithms. If the difference between the two algorithms' average ranking values exceeds the Critical Difference (CD), the hypothesis that the two algorithms perform equally is rejected with the appropriate level of confidence. The mathematical description of CD can be found in Eq. (5) (Nemenyi, 1963).

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{5}$$

where q_α is the critical value for multiple non-parametric comparisons, which can be obtained from a fixed critical table. k is the number of comparison algorithms and N is the number of test functions. Therefore, in the 5.1 section, it is N = 30 and k = 8. The Non-parametric Nemenyi test for CEC 2014 benchmark functions of 30D is reported in table 6. In this table, '1′ indicates a significant difference and '0′ otherwise. Carefully in this table, it can be understood that there is a significant difference between the HFA and other algorithms. There are also significant differences between algorithms COA and CCS. In this paper, According to the mean values algorithms in each simulation, the Nemenyi post-test was performed.

5.1.2. $D = 50$

The mean and standard deviation values in the last iteration for CEC 2014 benchmark functions with 50D are shown in Table 7.

The numbers in the penultimate row indicate the number of functions with the best mean or standard deviation for each optimization algorithm. It also can be seen from the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 3, 1, 1, 1, 3, 2, 16, and 3 times optimal mean results from 30 benchmark functions, respectively. In addition, it can be seen in the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 3, 3, 4, 2, 2, 2, 10, and 4 times optimal SD results from 30 benchmark functions, respectively. The last row shows a comparison between the current strategy and the HFA on 30 benchmark functions. According to the last line, the mean of HFA for 23, 24, 22, 24, 19, 22, and 19 benchmark functions was less than the mean of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. Also, the SD of HFA for 17, 26, 17, 15, 16, 18, and 18 benchmark functions was less than the SD of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively.

Table 8 shows the ranking of the algorithms in table 7 based on the Friedman test. According to mean, HFA obtains the best rank, VNBA ranks 2, and the following are RFO, COA, ASMO, PSO, CCS, and EHO. Also, According to SD, HFA obtains the best rank, EHO ranks 2, and the following are COA, RFO, CCS, ASMO, PSO, and VNBA. In the Friedman test, the p-value was less than 0.003.

Based on the Nemenyi test, the performance of the two algorithms is significantly different if the corresponding average ranks differ by at least the CD. Statistical analysis using the Non-parametric Nemenyi test for CEC 2014 benchmark functions of 50D is shown in table 9. Carefully in this table, it can be understood that there is a significant difference

Table 7
Mean and SD value algorithms for CEC 2014 benchmark functions of 50D.

FN	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	4.92E + 07	9.19E + 06	1.42E + 10	3.58E + 09	1.99E + 08	4.04E + 07	1.43E + 09	1.81E + 08	1.29E + 08	4.33E + 07	3.68E + 08	1.04E + 08	3.73E + 07	1.22E + 08	3.06E + 07	1.22E + 08
F2	1.32E + 07	1.74E + 06	2.36E + 11	1.90E + 10	4.37E + 09	7.14E + 08	1.49E + 11	2.60E + 10	5.69E + 08	1.67E + 08	3.72E + 10	9.10E + 09	3.28E + 09	1.27E + 09	4.59E + 10	8.58E + 09
F3	6.23E + 04	1.24E + 04	1.40E + 07	7.67E + 06	4.66E + 05	9.45E + 04	1.06E + 05	7.01E + 03	4.80E + 04	1.26E + 04	1.10E + 05	9.01E + 04	2.11E + 05	2.51E + 05	8.96E + 04	8.85E + 04
F4	8.75E + 02	4.65E + 01	7.53E + 04	1.00E + 04	9.20E + 02	7.65E + 01	2.95E + 04	5.62E + 03	7.79E + 02	9.25E + 01	7.61E + 03	2.04E + 03	5.53E + 02	4.69E + 02	8.97E + 03	8.85E + 02
F5	7.82E + 02	5.96E-02	5.23E + 02	3.12E-02	5.22E + 02	7.56E-02	6.25E + 02	4.24E-02	5.21E + 02	5.70E-02	5.19E + 02	3.53E-02	5.18E + 02	3.90E-01	8.218E + 02	8.560E + 00
F6	9.81E + 02	3.17E + 00	6.51E + 02	2.23E + 00	6.41E + 02	7.40E + 00	8.00E + 02	2.99E + 00	6.40E + 02	4.44E + 00	6.24E + 02	2.13E + 00	6.23E + 02	2.44E-01	6.21E + 02	2.04E-01
F7	1.09E + 03	8.76E + 00	3.92E + 03	4.93E + 02	7.30E + 02	6.46E + 00	2.28E + 03	2.63E + 02	7.30E + 02	1.53E + 00	8.53E + 02	9.82E + 01	7.28E + 02	2.29E + 01	3.56E + 03	8.58E + 01
F8	1.47E + 03	3.07E + 01	1.60E + 03	3.74E + 01	1.26E + 03	3.97E + 01	1.64E + 03	3.36E + 01	8.37E + 02	5.07E + 00	8.34E + 02	2.24E + 01	8.33E + 02	3.00E + 00	8.28E + 02	8.32E + 01
F9	1.97E + 03	3.82E + 01	1.85E + 03	6.13E + 01	1.40E + 03	3.32E + 01	1.91E + 03	4.00E + 01	1.16E + 03	3.35E + 01	9.89E + 02	2.74E + 01	9.38E + 02	4.17E + 00	1.85E + 03	3.89E + 01
F10	4.23E + 03	4.80E + 02	1.45E + 04	5.10E + 02	1.60E + 04	5.77E + 02	1.72E + 04	6.11E + 02	1.41E + 03	1.20E + 02	5.49E + 03	3.79E + 02	1.04E + 03	4.66E + 01	1.85E + 04	9.54E + 02
F11	1.39E + 04	8.91E + 02	1.47E + 04	3.43E + 02	1.65E + 04	6.27E + 02	1.75E + 04	5.06E + 02	8.04E + 03	6.37E + 02	8.05E + 03	4.83E + 02	1.15E + 03	5.11E + 01	1.75E + 04	4.23E + 01
F12	1.80E + 03	2.07E-01	1.26E + 03	5.69E-01	1.26E + 03	8.35E-01	1.44E + 03	3.82E-01	1.26E + 03	1.65E-01	1.25E + 03	2.08E-01	1.25E + 03	3.44E + 00	1.25E + 03	1.32E + 01
F13	1.95E + 03	5.25E-01	1.36E + 03	5.46E-01	1.36E + 03	1.61E-01	1.57E + 03	6.72E-01	1.36E + 03	1.24E-01	1.36E + 03	4.16E-01	1.36E + 03	5.07E-01	1.36E + 03	5.07E-01
F14	2.16E + 03	1.01E + 01	1.92E + 03	6.50E + 01	1.47E + 03	2.67E + 00	2.09E + 03	5.51E + 01	1.47E + 03	2.97E-01	1.46E + 03	2.30E + 01	1.46E + 03	3.80E + 00	1.46E + 03	8.25E + 00
F15	2.31E + 03	1.00E + 01	8.05E + 07	3.08E + 07	1.69E + 03	1.20E + 02	4.13E + 06	2.99E + 06	1.60E + 03	1.03E + 02	1.03E + 06	1.36E + 06	4.19E + 04	1.08E + 06	1.86E + 05	7.98E + 05
F16	2.43E + 03	4.51E-01	1.68E + 03	2.03E-01	1.68E + 03	2.84E-01	1.94E + 03	2.32E-01	1.68E + 03	4.12E-01	1.67E + 03	2.36E-01	1.67E + 03	2.62E-02	1.84E + 03	8.25E-02
F17	1.97E + 07	5.84E + 06	2.33E + 09	7.99E + 08	3.03E + 07	1.08E + 07	1.19E + 08	2.66E + 07	3.85E + 07	1.74E + 07	3.52E + 07	1.98E + 07	8.93E + 06	2.69E + 07	3.52E + 07	4.25E + 07
F18	1.52E + 06	2.19E + 05	2.75E + 10	6.15E + 09	2.52E + 08	5.23E + 07	5.24E + 09	5.10E + 08	4.99E + 07	2.95E + 07	1.31E + 09	2.41E + 08	2.97E + 08	2.13E + 08	8.75E + 07	5.65E + 08
F19	3.02E + 03	3.47E + 01	9.41E + 03	1.73E + 03	1.96E + 03	6.73E + 00	2.89E + 03	8.63E + 01	1.99E + 03	1.70E + 01	1.99E + 03	6.02E + 01	1.99E + 03	5.96E + 01	1.99E + 03	4.26E + 01
F20	1.20E + 05	3.59E + 04	2.16E + 07	2.65E + 07	1.27E + 06	8.33E + 05	3.22E + 04	5.51E + 03	5.61E + 04	1.53E + 04	7.31E + 04	3.02E + 05	1.19E + 04	8.65E + 05	2.39E + 04	6.23E + 05
F21	1.56E + 07	5.81E + 06	3.41E + 08	1.34E + 08	1.48E + 07	5.47E + 06	9.23E + 06	1.86E + 06	2.03E + 07	6.76E + 06	6.45E + 06	4.09E + 06	2.62E + 05	4.60E + 06	4.55E + 05	9.56E + 06
F22	5.49E + 03	3.57E + 02	8.16E + 06	4.70E + 06	4.85E + 03	2.92E + 02	7.82E + 03	9.59E + 02	4.11E + 03	3.65E + 02	2.79E + 04	5.16E + 04	6.97E + 04	1.53E + 05	6.85E + 04	8.21E + 05
F23	4.04E + 03	2.41E + 01	5.94E + 03	8.80E + 02	2.67E + 03	1.23E + 01	3.00E + 03	2.39E-02	2.70E + 03	2.38E + 01	2.41E + 03	1.79E + 01	2.41E + 03	2.94E + 01	2.42E + 03	3.84E + 01
F24	4.14E + 03	1.02E + 01	2.92E + 03	4.14E + 01	2.68E + 03	4.68E + 00	3.12E + 03	1.16E-02	2.69E + 03	3.71E + 00	2.51E + 03	3.96E + 00	2.51E + 03	1.66E + 01	2.88E + 03	1.54E + 01
F25	4.11E + 03	9.03E + 00	2.95E + 03	5.05E + 01	2.70E + 03	8.29E + 00	3.24E + 03	4.61E-04	2.73E + 03	5.74E + 00	2.62E + 03	3.65E + 00	2.62E + 03	1.92E-01	2.85E + 03	2.50E-01
F26	4.28E + 03	9.10E + 01	3.31E + 03	1.98E + 02	2.73E + 03	1.98E-01	3.29E + 03	2.90E + 01	2.78E + 03	4.47E + 01	2.72E + 03	4.33E + 01	2.72E + 03	9.96E + 00	2.72E + 03	9.05E + 00
F27	6.57E + 03	1.01E + 02	7.30E + 03	8.01E + 02	4.17E + 03	9.74E + 01	3.48E + 03	3.37E-03	4.27E + 03	9.60E + 01	2.83E + 03	4.34E + 01	2.83E + 03	2.92E + 01	2.83E + 03	2.23E + 02
F28	7.95E + 03	5.21E + 02	2.33E + 04	2.47E + 03	5.49E + 03	3.02E + 02	3.60E + 03	2.57E-02	5.29E + 03	5.98E + 02	2.97E + 03	2.06E + 02	2.93E + 03	9.64E + 01	3.21E + 03	3.05E + 02
F29	2.45E + 06	8.65E + 06	6.14E + 09	1.35E + 09	5.79E + 07	7.52E + 07	1.12E + 06	2.53E + 04	3.18E + 07	1.42E + 07	9.30E + 06	1.77E + 07	2.51E + 07	4.44E + 07	5.96E + 07	8.94E + 07
F30	5.16E + 04	1.31E + 04	1.75E + 08	4.32E + 07	2.75E + 05	9.49E + 04	7.05E + 05	1.12E + 03	7.22E + 04	3.29E + 04	4.25E + 05	4.75E + 05	5.06E + 05	1.41E + 06	5.01E + 05	1.41E + 05
	3	**3**	**1**	**3**	**1**	**4**	**1**	**2**	**3**	**2**	**2**	**2**	**16**	**10**	**3**	**4**
	23	**17**	**24**	**26**	**22**	**17**	**24**	**15**	**19**	**16**	**22**	**18**	**0**	**0**	**19**	**18**

Table 8
Friedman test based on the results of Table 7.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
Friedman test based on mean values algorithms	4.70	5.28	3.86	5.13	3.69	3.86	2.48	3.96
Final rank algorithms based on mean values	6	8	5	7	2	3	1	4
Friedman test based on SD values algorithms	4.24	3.99	4.12	3.86	4.36	3.95	3.24	3.88
Final rank algorithms based on SD values	7	5	6	2	8	4	1	3

Table 9
Statistical analysis using Non-parametric Nemenyi test for CEC 2014 benchmark functions of 50D, where '1' indicates significant difference and '0' otherwise.

	PSO	CCS	ASMO	EHO	VNBA	RFO	HFA	COA
PSO	NAN	0	0	0	0	0	1	0
CCS	0	NAN	0	0	0	0	1	0
ASMO	0	0	NAN	0	0	0	1	0
EHO	0	0	0	NAN	0	0	1	0
VNBA	0	0	0	0	NAN	0	0	0
RFO	0	0	0	0	0	NAN	1	0
HFA	1	1	1	1	0	1	NAN	1
COA	0	0	0	0	0	0	1	NAN

between the HFA and other algorithms except for VNBA. The other seven algorithms are not significantly different.

5.1.3. $D = 100$

Like the previous two sections, all algorithms for each optimization function are run 30 times. The mean and standard deviation values in the last iteration for CEC 2014 benchmark functions with 100D are shown in Table 10.

The numbers in the penultimate row indicate the number of functions with the best mean or standard deviation for each optimization algorithm. It also can be seen from the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 2, 1, 2, 3, 3, 1, 15, and 3 times optimal mean results from 30 benchmark functions, respectively. In addition it can be seen in the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 4, 1, 3, 2, 3, 1, 11, and 4 times optimal SD results from benchmark functions, respectively. The last line of the table presents the number of benchmark functions that, after applying the HFA algorithm to them, their mean and standard deviation have become more favorable than applying other optimization algorithms to them. According to the last line, the mean of HFA for 22, 27, 20, 23, 18, 24, and 18 benchmark functions was less than the mean of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. Also, the SD of HFA for 19, 20, 20, 23, 22, 22, and 15 benchmark functions was less than the SD of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively.

Table 11 shows the ranking of the algorithms in table 10 based on the Friedman test. According to mean, HFA obtains the best rank, COA ranks 2, and the following are ASMO, VNBA, RFO, EHO, PSO, and CCS. In addition, According to SD, HFA obtains the best rank, VNBA ranks 2, and the following are PSO, COA, RFO, ASMO, EHO, and CCS. In the Friedman test, the p-value was less than 0.005.

After calculating the cd according to Eq. (5), with the help the Non-parametric Nemenyi test, a decision is made on the difference or no significant difference in the performance of the algorithms. Accordingly, if the center distance of the two algorithms is less than the CD value, the performance of the two algorithms is not significantly different, and if the center distance of the two algorithms is greater than the CD value, the two algorithms are significantly different. Statistical analysis using the Non-parametric Nemenyi test for CEC 2014 benchmark functions of 100D is presented in Table 12. According to the information in this table, it is clear that there is a significant difference between the HFA and other algorithms except for COA. The other seven algorithms are not significantly different.

Table 13 shows the best values of HFA in IEEE CEC 2014 functions in 30D, 50D, and 100D. The column of the group represents the grouping of all problems. The column of Rate demonstrates the ratio between the best values obtained and all the problems in this group. As seen in table 9, HFA performs well in solving the first group of unimodal functions, the second group of simple multimodal functions, and the third group of hybrid functions. For unimodal functions, simple multimodal functions, and hybrid functions, HFA can find the optimal solution for most problems, and the ratio of finding the optimal solution is 88%, 82%, and 87%, respectively. On the other hand, the performance of the HFA in the fourth group of composition functions is not as good as the previous group functions. In 60% of the functions of the fourth group, the best answer belongs to the HFA.

According to Tables 4–10, HFA performs much better than other algorithms in IEEE CEC 2014 functions in 30D, 50D, and 100D. In order for HFA to achieve an appropriate response in solving optimization problems, it is recommended that the following points be observed in setting the parameters:

1- The initial population (N_{pop}) should be selected according to the dimensions and complexity of the problem (in this section, the initial population is considered for all the same algorithms)
2- The number of elites (N_{elite}) depends on the complexity of the optimization problem, but the choice of more than 3 elites in the problems is not suggested.
3- The number of the disciple ($N_{disciple}$) depends on the optimization problem and the number of elites. It is suggested to consider 2 points: 1- About 20% to 30% of the initial population should be allocated to disciples. 2- The number of the disciple should be considered about 1 to 3 times that of the elite. The minimum number obtained from the above two constraints should be allocated as the number of the disciple.
4- It is suggested that the number of local searches ($N_{personal}$) is selected from 2 to 4. Assigning large numbers to this parameter slows down the HFA and usually does not have much effect on improving the HFA.
5- It is suggested to determine the number of steps in the desired direction ($N_{unidirection}$) in the local search 2 to 4 steps. Assigning large numbers usually does not have much effect on improving the HFA.

The step length in local search) s (n)) depends on the size of the search space and the complexity of the issues. This parameter in the IEEE CEC 2014 functions was considered 0.05 for the elites and 1 for the rest of the community.

The number of times assigned to the revolution ($N_{revolution}$) depends entirely on the size and complexity of the problem, and no specific number can be suggested.

It is suggested that the number of people who suffer from revolution (N_{rev}) be equal to 60–70% of the initial population.

The number of cycles of movement of society towards the elites ($N_{alteration}$) is completely dependent on the complexity of the problem. This parameter in the solution of IEEE CEC 2014 functions is assigned equal to 20.

Complexity Analysis:

Complexity makes a comparison of the running-time of different algorithms. The number of function evaluations is considered the most time consuming operation in relation to other operations. In HFA, the movement of people in society towards the elite, computational complexity is equal to : $\mathcal{O}^*(N-N_{elite})^*D^*f_{eval}$. $N, D, and f_{eval}$ are number of people in the society , number of variables (dimension), and number of function evaluations , respectively. Computational complexity becomes $\mathcal{O}^*D^*N^*N_{pesrenal}$ in local search. As a result, the mathematical complexity of the proposed algorithm is $\mathcal{O}^*N_{\text{revolution}}[(N-N_{elite})^*D^*f_{eval}+D^*N^*N_{pesrenal}]$. The running time complexity is strongly influenced by the $N_{\text{revolution}}$. As a result, $N_{\text{revolution}}$ should not be selected large, although large selection of this parameter has no effect on improving the performance of the HFA. It is obvious though, the improvement of computation precision in HFA algorithm usually needs the sacrifice of time complexity. The characteristics of the computer on which the algorithms were run are:

MATLAB R2014a, PC with core i3-6100 (3.70 GHz) CPU, 4 GB RAM, and win 7 OS. The computational complexity of HFA algorithm is calculated as described in (Liang, Qu, & Suganthan, 2013). The calculated times and algorithm complexity for CEC 2014 benchmark functions are presented in Table 14.

Table 10
Mean and SD value algorithms for CEC 2014 benchmark functions of 100D.

FN	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	1.39E + 09	4.49E + 07	6.85E + 11	1.97E + 09	4.22E + 10	4.14E + 08	1.38E + 09	2.28E + 09	6.99E + 09	3.82E + 08	3.31E + 09	1.33E + 11	1.21E + 12	6.52E + 05	6.58E + 09	8.63E + 8
F2	5.20E + 08	1.07E + 07	5.86E + 12	2.89E + 11	5.79E + 12	1.63E + 10	1.15E + 10	7.96E + 11	1.02E + 12	2.04E + 09	9.80E + 10	5.00E + 11	2.63E + 11	1.69E + 07	5.08E + 11	1.61E + 07
F3	3.69E + 06	1.14E + 05	1.93E + 08	1.16E + 07	2.01E + 06	1.05E + 06	8.75E + 05	7.17E + 04	1.38E + 06	8.66E + 04	8.84E + 06	4.03E + 07	2.45E + 06	2.54E + 03	8.84E + 06	1.48E + 05
F4	2.83E + 04	4.90E + 02	2.34E + 06	9.23E + 04	1.60E + 06	1.48E + 03	9.65E + 03	2.43E + 05	2.66E + 05	1.04E + 03	1.19E + 04	2.30E + 05	1.95E + 03	6.28E + 06	2.83E + 03	1.42E + 03
F5	2.37E + 04	4.10E-01	5.11E + 03	2.44E-01	7.73E + 03	5.28E-01	5.27E + 03	2.79E-01	3.70E + 03	5.39E-01	5.11E + 03	1.38E-01	9.68E + 02	2.82E + 00	2.73E + 03	8.95E-01
F6	3.08E + 04	3.00E + 01	7.09E + 03	2.51E + 01	1.04E + 04	1.76E + 02	6.65E + 03	4.43E + 01	4.88E + 03	7.91E + 01	6.60E + 03	2.23E + 01	1.16E + 03	3.16E + 01	2.94E + 04	1.86E + 02
F7	3.36E + 04	2.98E + 02	9.30E + 04	5.43E + 03	5.25E + 04	1.19E + 02	7.15E + 03	1.38E + 04	1.25E + 04	1.80E + 01	7.58E + 03	1.71E + 04	1.36E + 03	3.37E + 03	1.36E + 03	3.24E + 03
F8	4.81E + 04	4.89E + 02	2.20E + 04	3.66E + 02	2.58E + 04	7.23E + 02	8.60E + 03	5.26E + 02	9.49E + 03	6.79E + 01	1.51E + 04	5.02E + 02	1.55E + 03	9.50E + 01	2.36E + 03	9.20E + 01
F9	6.84E + 04	4.15E + 02	2.57E + 04	3.24E + 02	3.00E + 04	5.39E + 02	1.31E + 04	4.73E + 02	1.22E + 04	3.74E + 02	1.68E + 04	8.56E + 02	1.75E + 03	9.40E + 01	5.72E + 03	8.41E + 01
F10	2.32E + 05	5.43E + 03	2.62E + 05	5.22E + 03	3.83E + 05	6.80E + 03	1.66E + 04	9.52E + 03	9.48E + 04	1.65E + 03	2.83E + 05	7.12E + 03	1.94E + 03	1.00E + 02	3.62E + 05	4.32E + 03
F11	7.78E + 05	9.03E + 03	2.46E + 05	5.45E + 03	3.85E + 05	8.52E + 03	1.30E + 05	6.61E + 03	1.46E + 05	6.19E + 03	2.77E + 05	2.02E + 03	2.14E + 03	6.21E + 02	2.64E + 03	8.31E + 03
F12	5.45E + 04	1.18E + 01	1.18E + 04	1.80E + 01	1.78E + 04	7.76E + 01	1.21E + 04	4.23E + 01	8.53E + 03	1.80E + 01	1.19E + 04	7.37E + 01	2.33E + 03	3.96E + 01	8.21E + 03	2.68E + 01
F13	5.91E + 04	4.01E + 00	1.29E + 04	5.56E + 00	1.96E + 04	1.18E + 00	1.31E + 04	1.92E + 01	9.30E + 03	1.28E + 00	1.27E + 04	1.71E + 00	2.53E + 03	1.19E + 01	5.31E + 03	5.21E + 01
F14	6.59E + 04	7.35E + 01	2.21E + 04	6.17E + 02	2.96E + 04	4.76E + 01	1.41E + 04	2.62E + 03	1.19E + 04	4.86E + 00	1.39E + 04	8.27E + 02	2.72E + 03	4.42E + 03	2.25E + 03	3.56E + 03
F15	7.05E + 04	1.00E + 02	3.57E + 10	4.30E + 08	5.90E + 08	5.91E + 03	1.68E + 04	7.03E + 08	9.74E + 07	3.89E + 03	1.86E + 04	1.03E + 10	3.53E + 05	8.98E + 10	3.17E + 08	5.30E + 08
F16	7.41E + 04	3.59E + 00	1.60E + 04	1.98E + 00	2.42E + 04	3.53E + 00	1.65E + 04	1.98E + 00	1.16E + 04	3.01E + 00	1.60E + 04	1.43E + 00	3.11E + 03	1.20E + 00	3.56E + 03	1.20E + 00
F17	9.40E + 08	6.08E + 07	1.42E + 11	1.14E + 09	1.23E + 10	2.76E + 08	1.01E + 09	1.56E + 09	1.37E + 09	4.54E + 08	1.05E + 09	3.33E + 10	2.19E + 08	2.08E + 06	2.19E + 08	5.98E + 06
F18	8.92E + 07	2.24E + 06	6.25E + 11	1.07E + 10	7.18E + 11	8.47E + 08	2.01E + 09	1.61E + 10	8.51E + 10	1.14E + 09	6.82E + 09	8.80E + 10	1.47E + 10	5.33E + 08	1.22E + 10	5.63E + 08
F19	9.27E + 04	3.04E + 02	3.24E + 05	1.16E + 03	4.01E + 04	8.93E + 01	2.04E + 04	1.41E + 03	1.64E + 04	6.44E + 01	2.02E + 04	7.28E + 04	3.70E + 03	1.88E + 02	2.43E + 04	4.28E + 03
F20	6.55E + 06	5.41E + 05	7.76E + 08	8.72E + 07	4.54E + 05	9.73E + 06	8.50E + 05	5.00E + 04	9.11E + 05	9.96E + 04	2.28E + 07	5.05E + 08	2.59E + 04	4.63E + 04	2.84E + 04	2.61E + 05
F21	3.56E + 09	2.74E + 08	6.29E + 09	1.91E + 08	3.11E + 08	1.30E + 08	1.12E + 09	2.54E + 07	1.66E + 08	2.32E + 08	5.04E + 08	1.75E + 09	3.18E + 06	1.82E + 05	2.29E + 07	1.66E + 07
F22	2.11E + 05	4.61E + 03	1.94E + 08	1.59E + 08	1.87E + 05	4.43E + 03	5.80E + 04	4.20E + 04	3.41E + 05	8.14E + 03	6.98E + 04	5.18E + 07	6.49E + 05	1.58E + 08	3.40E + 04	2.91E + 06
F23	1.23E + 05	1.48E + 02	9.46E + 04	2.40E + 02	3.71E + 04	1.84E + 02	2.77E + 04	2.81E + 01	1.86E + 04	1.76E + 02	2.74E + 04	1.42E + 04	4.48E + 03	5.17E + 01	4.48E + 03	2.36E + 01
F24	1.29E + 05	8.43E + 01	3.36E + 04	3.75E + 01	3.86E + 04	1.07E + 02	2.78E + 04	2.18E + 01	1.95E + 04	6.39E + 00	2.75E + 04	2.26E + 03	4.67E + 03	3.65E + 01	1.83E + 04	8.43E + 01
F25	1.25E + 05	1.28E + 02	3.26E + 04	6.21E + 01	4.01E + 04	1.27E + 02	2.77E + 04	1.01E + 00	1.94E + 04	5.84E + 01	2.72E + 04	3.86E + 02	4.87E + 03	1.01E-01	8.15E + 04	5.28E + 02
F26	1.36E + 05	2.34E + 03	3.92E + 04	1.50E + 03	4.11E + 04	2.24E + 00	2.89E + 04	4.79E + 03	2.06E + 04	1.58E + 05	2.65E + 04	3.33E + 03	5.06E + 03	1.32E + 01	1.36E + 04	4.34E + 03
F27	1.99E + 05	8.99E + 02	7.53E + 04	4.91E + 02	4.31E + 04	8.77E + 02	4.31E + 04	3.03E-02	2.69E + 04	8.64E + 02	4.17E + 04	7.21E + 03	5.26E + 03	1.23E + 04	8.29E + 03	8.99E + 02
F28	2.92E + 05	1.02E + 04	3.85E + 05	4.69E + 03	4.45E + 04	7.03E + 03	6.28E + 04	9.91E-01	3.48E + 04	9.18E + 03	7.25E + 04	2.86E + 04	5.45E + 03	5.60E-01	2.92E + 03	8.23E + 01
F29	1.47E + 10	2.99E + 11	3.41E + 11	2.06E + 11	3.46E + 07	1.62E + 10	3.25E + 09	2.22E + 10	1.27E + 08	4.39E + 08	1.85E + 10	5.64E + 10	7.60E + 08	1.15E + 07	7.22E + 08	1.15E + 07
F30	4.37E + 06	5.02E + 05	1.30E + 10	9.16E + 06	9.18E + 06	2.60E + 06	1.06E + 06	1.71E + 06	5.47E + 06	4.86E + 05	8.56E + 06	1.14E + 09	9.52E + 06	1.56E + 08	9.45E + 06	8.52E + 06
	2	**4**	**1**	**1**	**2**	**3**	**3**	**2**	**3**	**3**	**1**	**1**	**15**	**11**	**3**	**4**
	22	**19**	**27**	**20**	**20**	**20**	**23**	**23**	**18**	**22**	**24**	**22**	**0**	**0**	**18**	**15**

Table 11
Friedman test based on the results of Table 10.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
Friedman test based on mean values algorithms	5.23	5.20	3.71	5.12	3.88	3.91	2.05	3.68
Final rank algorithms based on mean values	8	7	3	6	4	5	1	2
Friedman test based on SD values algorithms	3.56	5.19	4.21	4.46	3.14	4.21	2.68	3.95
Final rank algorithms based on SD values	3	8	6	7	2	5	1	4

Table 12
Statistical analysis using Non-parametric Nemenyi test for CEC 2014 benchmark functions of 100D, where '1' indicates significant difference and '0' otherwise.

	PSO	CCS	ASMO	EHO	VNBA	RFO	HFA	COA
PSO	NAN	0	0	0	0	0	1	0
CCS	0	NAN	0	0	0	0	1	0
ASMO	0	0	NAN	0	0	0	1	0
EHO	0	0	0	NAN	0	0	1	0
VNBA	0	0	0	0	NAN	0	1	0
RFO	0	0	0	0	0	NAN	1	0
HFA	1	1	1	1	1	1	NAN	0
COA	0	0	0	0	0	0	0	NAN

5.2. CEC-C06 2019 benchmark test functions

A group of 10 trendy CEC benchmark test functions is utilized, CEC-C06 2019 benchmark was improved by professor Suganthan and his team for single-objective optimization problem, the test functions are made mention as "The 100-Digit Challenge", which are intended to be utilized in annual optimization competition. Functions CEC01 to CEC03 are not rotated and shifted, whereas functions CEC04 to CEC10 are rotated and shifted. The parameter setting was outlined by the CEC benchmark developer, as functions CEC04 to CEC10 were set as a 10-dimensional minimization problem in [−100, 100] boundary range, however, CEC01 to CEC03 have different dimensions as shown among the in Table 15. All the CEC functions are scalable and each global optimum of those functions was united towards point 1 (Abdullah & Ahmed, 2019; Liang, Qu, & Suganthan, 2013). The test functions are solved fifty times utilizing eighty search agents over five hundred iterations. All algorithms for each CEC-C06 2019 benchmark test function are run 30 times. The mean and standard deviation values in the last iteration for benchmark functions are shown in table 16.

In order to better analyze and obtain a more appropriate view, as in tables 4, 7, and 10, the penultimate row of table 15 demonstrates the number of times that the mean or standard deviation of one algorithm is the best among all algorithms. In addition, the last row of table 12 represents the number of times that the mean and standard deviation of the proposed algorithm was less than other algorithms. It also can be seen from the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 1, 0, 1, 1, 1, 1,5, and 1 times optimal mean results from 10 benchmark functions, respectively. In addition, it can be seen in the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 0, 1, 0, 0, 2, 1, 5, and 1 times optimal SD results from benchmark functions, respectively. The last row shows a comparison between the current strategy and the HFA on benchmark functions. According to the last line, the mean of HFA for 7, 7, 6, 5, 7, 8, and 7 benchmark functions was less than the mean of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. In addition, the SD of HFA for 6, 7, 6, 6, 5, 7, and 6 benchmark functions was less than the SD of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. Based on the above points, it can be concluded that the proposed algorithm has a better performance in finding the optimal points of each CEC-C06 2019 test function than other algorithms.

For a better and more accurate conclusion, as in tables 5, 8, and 11, the Friedman test was used to rank the non-parametric parameters of table 15, the results of which are presented in table 13. According to mean, HFA obtains the best rank, EHO ranks 2, and the following are COA, ASMO, VNBA, RFO, CCS, and PSO. In addition, According to SD, HFA obtains the best rank, COA ranks 2, and the following are VNBA, PSO, EHO, ASMO, RFO, and CCS. In the Friedman test, the p-value was less than 0.002.

Non-parametric Nemenyi test was used to better analyze the data in Table 16 and complete the Friedman test presented in Table 17. In this section, the number of test functions is 10 and the number of algorithms that are compared with each other is 8, so by placing in Eq.5, the value of CD = 1.23 is calculated. In the Non-parametric Nemenyi test, the

Table 13
The statistical analysis of the best values of HFA.

Group	Problem	30D	50D	100D	Rate
Unimodal functions	**F1**	2.64E + 05	1.62E + 05	6.33E + 06	89%
	F2	2.00E + 02	2.15E + 02	2.42E + 02	
	F3	3.00E + 02	3.60E + 02	2.36E + 03	
Simple multimodal functions	**F4**	4.41E + 02	4.59E + 02	5.01E + 02	87%
	F5	5.07E + 02	5.17E + 02	8.07E + 02	
	F6	6.04E + 02	6.10E + 02	6.23E + 02	
	F7	7.00E + 02	7.00E + 02	7.00E + 02	
	F8	8.11E + 02	8.21E + 02	8.42E + 02	
	F9	9.11E + 02	9.20E + 02	9.40E + 02	
	F10	1.75E + 03	2.59E + 03	6.66E + 03	
	F11	2.05E + 03	3.17E + 03	9.07E + 03	
	F12	1.20E + 03	1.20E + 03	1.20E + 03	
	F13	1.30E + 03	1.30E + 03	1.30E + 03	
	F14	1.40E + 03	1.40E + 03	1.40E + 03	
	F15	1.50E + 03	1.51E + 03	1.52E + 03	
	F16	1.60E + 03	1.61E + 03	1.61E + 03	
Hybrid functions	**F17**	2.73E + 03	1.23E + 05	2.65E + 06	94%
	F18	1.85E + 02	2.09E + 03	1.45E + 03	
	F19	1.90E + 03	1.92E + 03	1.95E + 03	
	F20	2.07E + 03	2.15E + 03	2.32E + 03	
	F21	1.12E + 04	6.74E + 04	3.02E + 06	
	F22	2.27E + 03	2.42E + 03	2.82E + 03	
Composition functions	**F23**	2.31E + 03	2.42E + 03	2.44E + 03	75%
	F24	2.48E + 03	2.50E + 03	2.53E + 03	
	F25	2.57E + 03	2.58E + 03	2.59E + 03	
	F26	2.63E + 03	2.63E + 03	2.63E + 03	
	F27	2.79E + 03	3.07E + 03	3.51E + 03	
	F28	3.12E + 03	3.15E + 03	3.95E + 03	
	F29	3.00E + 03	3.05E + 03	4.20E + 03	
	F30	3.36E + 03	1.71E + 04	6.20E + 05	

Table 14
HFA Complexity CEC 2014 benchmark functions.

	T_0	T_1	$\widehat{T_2}$	$(\widehat{T_2} - T_1)/T_0$
D = 10	0.1040	0.2433	0.3637	1.1585
D = 30		0.3346	0.6763	3.2861
D = 50		0.4062	0.7467	3.4664
D = 100		0.8688	1.4658	5.7408

Table 15
CEC-C06 2019 benchmark functions.

No	Functions	Dimension	Range	f_{min}
F1	Storn's Chebyshev Polynomial Fitting Problem	9	[-8192, 8192]	1
F2	Inverse Hilbert Matrix Problem	16	[-16384, 16384]	1
F3	Lennard-Jones Minimum Energy Cluster	18	[-4, 4]	1
F4	Rastrigin's Function	10	[-100, 100]	1
F5	Grienwank's Function	10	[-100, 100]	1
F6	Weiersrass Function	10	[-100, 100]	1
F7	Modified Schwefel's Function	10	[-100, 100]	1
F8	Expanded Schaffer's F6 Function	10	[-100, 100]	1
F9	Happy Cat Function	10	[-100, 100]	1
F10	Ackley Function	10	[-100, 100]	1

performance between the two optimization algorithms was considered to be significantly different if the corresponding average ranks differed by at least the CD. In this test information for IEEE ECE 2019, benchmark test function is shown in Table 18. According to the information in this table, there is a significant difference between the HFA and other algorithms. There are also significant differences between algorithms COA and PSO.

Carefully in Eqs. (2)–(4), it will be understood that the HFA algorithm has good convergence because everyone is moving towards the elites. To avoid getting stuck in local minima, in Eqs. 3 and 4, the coefficient w and c must be changed within the appropriate range. If the distance between the numerical value of w and the number one is large, convergence does not develop well and people do not get very close to the elites. In addition, if w is too close to one, people will quickly approach the elites and the problem space will not be well searched. Therefore, selecting the appropriate numerical range for changes to this parameter is very important. In Formula 3, in order for Disciples to approach the elite, parameter c is selected close to one. In Formula 4, in order for General People in society to be influenced by the elite but not get too close to them, the value of c is chosen much less than one. After performing many simulations, the appropriate values of w, c in Eqs. 3 and 4 are suggested in section 4.2. The important point is that the values of w, c are chosen completely randomly at the specified numerical range and are not a fixed number during the simulation time. Figs. 9 and 10 are presented to investigate the convergence of the HFA optimization algorithm. In these forms, the average felicity of members of society is plotted before each revolution. Because the maximum value of each CEC-C06 2019 benchmark test function is different, drawing benchmark test functions in one Figure eliminates precise observation of the values of the functions. Therefore, to draw Fig. 7 and. 8 in an innovative operation by multiplying a constant coefficient in the optimization functions, the values of the functions are changed between 1 and 1000.

Table 18
Statistical analysis using Non-parametric Nemenyi test for IEEE ECE 2019 benchmark test function, where '1' indicates significant difference and '0' otherwise.

	PSO	CCS	ASMO	EHO	VNBA	RFO	HFA	COA
PSO	NAN	0	0	0	0	0	1	1
CCS	0	NAN	0	0	0	0	1	0
ASMO	0	0	NAN	0	0	0	1	0
EHO	0	0	0	NAN	0	0	1	0
VNBA	0	0	0	0	NAN	0	1	0
RFO	0	0	0	0	0	NAN	1	0
HFA	1	1	1	1	1	1	NAN	1
COA	1	0	0	0	0	0	1	NAN

Table 16
IEEE ECE 2019 benchmark test function results.

FN	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	2.62E + 09	4.19E + 10	2.02E + 09	1.43E + 11	8.30E + 06	5.13E + 08	4.03E + 07	2.17E + 09	5.32E + 09	1.95E + 09	1.32E + 09	2.15E + 10	2.62E + 03	4.39E + 08	2.61E + 03	2.36E + 08
F2	2.90E + 04	2.86E + 04	2.08E + 04	1.01E + 05	2.52E + 01	1.35E-02	1.70E + 01	7.11E + 02	7.65E + 01	4.05E-02	7.10E + 03	1.50E + 04	9.79E + 00	2.15E-03	2.18E + 02	1.01E + 05
F3	1.52E + 01	7.50E-03	1.74E + 01	2.44E-13	1.88E + 01	8.10E-03	1.34E + 01	5.67E-03	1.34E + 01	0.00E + 00	1.37E + 01	1.00E-03	9.24E + 00	2.23E-10	2.74E + 01	2.44E-11
F4	1.53E + 02	3.33E + 03	5.53E + 01	2.21E + 02	5.72E + 01	6.00E + 02	3.87E + 02	3.35E + 03	3.37E + 02	2.24E + 03	7.25E + 01	5.39E + 02	6.86E + 00	1.25E + 01	3.93E + 01	5.31E + 02
F5	2.24E + 00	3.90E + 00	1.56E + 00	2.41E + 00	3.03E + 00	2.87E + 00	2.68E + 00	2.63E + 00	2.51E + 00	2.63E + 00	5.01E + 00	1.08E + 00	1.16E + 00	1.05E + 00	1.56E + 00	3.81E + 00
F6	2.74E + 00	3.61E + 01	3.65E + 00	4.56E + 01	2.38E + 00	4.02E + 01	3.00E + 00	1.33E + 01	2.77E + 00	9.29E + 00	2.57E + 00	1.20E + 01	2.67E + 00	7.69E + 00	2.75E + 00	2.36E + 01
F7	4.36E + 02	5.43E + 02	2.20E + 02	2.81E + 02	5.63E + 02	7.85E + 02	4.81E + 02	2.67E + 02	5.67E + 02	1.75E + 02	2.37E + 02	1.32E + 02	9.30E + 01	2.59E + 02	8.71E + 02	9.21E + 02
F8	7.00E + 00	1.41E + 00	7.17E + 00	2.12E + 00	8.74E + 00	1.58E + 00	6.77E + 00	4.06E-01	6.74E + 00	3.84E-01	5.91E + 00	6.39E-01	3.85E + 00	1.94E + 00	3.84E + 00	2.92E-01
F9	3.67E + 00	2.34E + 01	2.60E + 00	4.98E-01	4.03E + 00	6.38E + 00	4.65E + 00	2.33E + 01	4.74E + 00	1.49E + 01	2.48E + 00	3.53E + 00	1.37E + 00	3.12E + 00	1.36E + 00	1.82E + 00
F10	2.37E + 01	2.84E + 00	2.78E + 01	3.47E + 00	2.89E + 01	2.11E + 00	7.45E + 00	1.39E + 00	7.44E + 00	1.00E + 00	2.34E + 00	7.67E-01	8.86E + 00	3.15E-02	4.94E + 00	6.37E-01
	1	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**1**	**2**	**1**	**1**	**5**	**5**	**1**	**1**
	7	**6**	**7**	**7**	**6**	**6**	**5**	**6**	**7**	**5**	**8**	**7**	**0**	**0**	**7**	**6**

Table 17
Friedman test based on the results of table 16.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
Friedman test based on mean values algorithms	5.91	4.82	4.30	4.13	4.42	4.63	2.29	3.94
Final rank algorithms based on mean values	8	7	4	3	5	6	1	2
Friedman test based on SD values algorithms	3.98	4.91	4.45	4.32	3.96	4.88	3.49	3.86
Final rank algorithms based on SD values	4	8	6	5	3	7	1	2

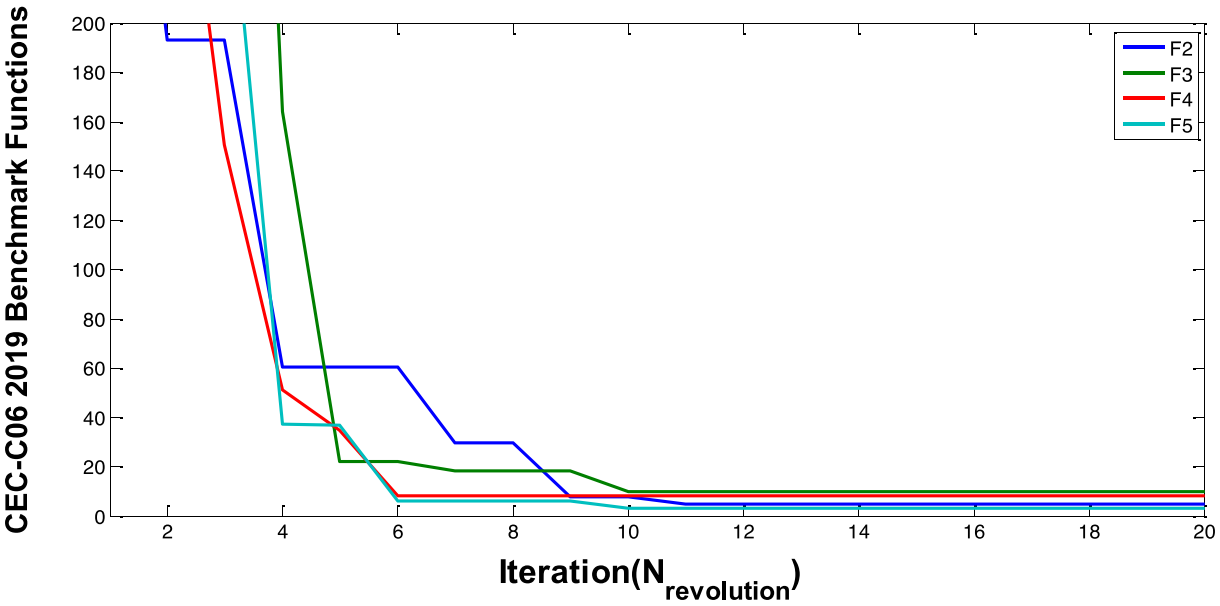

Fig. 9. CEC-C06 2019 Benchmark Functions F2-F5 for the HFA algorithm convergence curve.

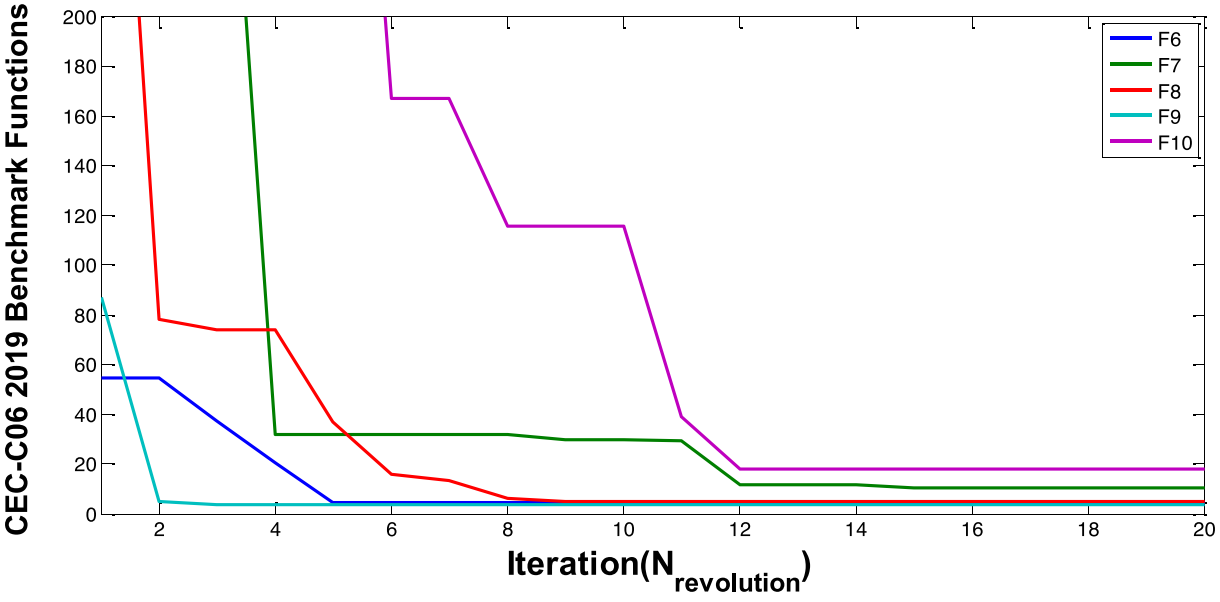

Fig. 10. CEC-C06 2019 Benchmark Functions F6-F10 for the HFA algorithm convergence curve.

With this method, it is possible to better compare the performance of the HFA in optimizing the CEC-C06 2019 benchmark test function.

5.3. CEC 2020 benchmark test functions

In this section, the proposed algorithm was compared with other optimization algorithms on CEC 2020 benchmark functions for single objective bound-constrained optimization. More details of CEC 2020 benchmark functions are demonstrated in Table 18. These benchmark functions set consists of 10 optimization real problems mentioned as f1-f10. In summary, function 1 is unimodal, functions 2–4 are basic, functions 5–7 are hybrid functions and 8–10 are composition functions. Therefore, finding the best response to these problems is a challenging task due to the hybrid, combination, and multi-mode features. To compare the optimization methods, the use CEC 2020 benchmark functions with dimensions D = 5; 10; 15 and 20 is recommended [47, 48]. Due to the comparisons made in the previous sections and the space constraints, this paper is simulated only the CEC 2020 benchmark functions with dimension 10 to compare the optimization methods.

All optimization methods for each optimization function are run 30 times. The average and standard deviation of the optimal solution are calculated in the last iteration. The mean and SD values of each optimization method for CEC 2020 benchmark functions with 30D are shown in Table 19. The number of functions with the best mean or standard deviation for each optimization algorithm is shown in the penultimate row. It also can be seen from the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 1,1, 1, 0, 1, 1,4, and 2 times optimal mean results from 10 benchmark functions, respectively. In addition, it can be seen in the penultimate row that PSO, CCS, ASMO, EHO, VNBA, RFO, HFA, and COA can find 0, 1, 2, 1, 0, 1,4, and 2 times optimal SD results from 10 benchmark functions, respectively. The last row shows a comparison between the current strategy

Table 19
CEC 2020 benchmark functions.

	No	Functions	Range	f_{min}
Unimodal Functions	F1	Shifted and Rotated Bent Cigar Function (CEC 2017 F1)	[-100, 100]	100
Basic Functions	F2	Shifted and Rotated Schwefel's Function (CEC 2014 F11)	[-100, 100]	1100
	F3	Shifted and Rotated Lunacek bi-Rastrigin Function (CEC 2017 F7)	[-100, 100]	700
	F4	Expanded Rosenbrock's plus Griewangk's Function (CEC2017 f19)	[-100, 100]	1900
Hybrid Functions	F5	Hybrid Function 1 (N = 3) (CEC 2014 F17)	[-100, 100]	1700
	F6	Hybrid Function 2 (N = 4) (CEC 2017 F16)	[-100, 100]	1600
	F7	Hybrid Function 3 (N = 5) (CEC 2014 F21)	[-100, 100]	2100
Composition Functions	F8	Composition Function 1 (N = 3) (CEC 2017 F22)	[-100, 100]	2200
	F9	Composition Function 2 (N = 4) (CEC 2017 F24)	[-100, 100]	2400
	F10	Composition Function 3 (N = 5) (CEC 2017 F25)	[-100, 100]	2500

and the HFA on benchmark functions. According to the last line, the mean of HFA for 7, 6, 8, 6, 7, 6, and 5benchmark functions was less than the mean of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively. In addition, the SD of HFA for 5, 5, 4, 6, 6, 6, and 6 benchmark functions was less than the SD of PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms, respectively.

Table 20 shows the ranking of the algorithms in table 14 based on the Friedman test. According to mean, HFA obtains the best rank, COA ranks 2, and the following are VNBA, RFO, CCS, ASMO, EHO, and PSO. In addition, According to SD, HFA obtains the best rank, EHO ranks 2, and the following are VNBA, ASMO, PSO, CCS, RFO, and COA. In the Friedman test, the p-value was less than 0.003 (Table 21.).

In this section, because the Friedman test is significant, the post hoc Nemenyi test is used to better analyze the Friedman test. The non-parametric Nemenyi test for the IEEE ECE 2020 benchmark function is presented in Table 22. In this section, the number of test functions is 10 and the number of algorithms that are compared with each other is 8, so by placing in Eq. (5), the value of CD = 1.32 is calculated. Carefully in this table, it can be observed that there is a significant difference between the HFA and other algorithms except for COA. With the exception of COA and PSO algorithms, which have significant differences, there is no significant difference between the other algorithms.

As seen in sections 5.1, 5.2, 5.3, in complicated optimization problems, HFA has an optimal capability in achieving total optimization and escaping from local minimums in complex problems with high dimensions. Also, HFA has a more favorable solution than many

Table 22
Statistical analysis using Non-parametric Nemenyi test for IEEE ECE 2020 benchmark test function, where '1' indicates significant difference and '0' otherwise.

	PSO	CCS	ASMO	EHO	VNBA	RFO	HFA	COA
PSO	NAN	0	0	0	0	0	1	1
CCS	0	NAN	0	0	0	0	1	0
ASMO	0	0	NAN	0	0	0	1	0
EHO	0	0	0	NAN	0	0	1	0
VNBA	0	0	0	0	NAN	0	1	0
RFO	0	0	0	0	0	NAN	1	0
HFA	1	1	1	1	1	1	NAN	0
COA	1	0	0	0	0	0	0	NAN

Table 20
IEEE ECE 2020 benchmark test function results.

FN	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	1.10E + 03	7.14E + 03	9.75E + 02	1.38E + 03	1.09E + 03	1.00E + 03	1.09E + 03	1.19E + 03	1.09E + 03	9.54E + 02	1.09E + 03	1.25E + 03	1.09E + 03	9.52E + 02	1.08E + 03	1.27E + 03
F2	4.96E + 03	1.19E + 03	4.42E + 03	2.63E + 02	4.90E + 03	2.08E + 02	3.98E + 03	1.75E + 02	2.48E + 03	1.08E + 03	3.45E + 03	1.93E + 02	1.16E + 03	1.52E + 00	1.15E + 03	1.21E + 00
F3	3.08E + 03	1.78E + 03	8.47E + 02	3.91E + 02	8.52E + 02	2.01E + 02	1.00E + 03	2.96E + 02	7.73E + 02	3.89E + 02	8.10E + 02	4.91E + 02	8.80E + 02	2.37E + 02	8.21E + 02	1.98E + 02
F4	1.32E + 04	2.67E + 04	3.92E + 03	7.90E + 03	2.94E + 03	9.95E + 02	4.03E + 03	4.45E + 03	3.12E + 03	1.64E + 03	3.52E + 03	2.96E + 02	3.79E + 03	3.56E + 03	2.94E + 03	7.25E + 02
F5	1.71E + 04	4.73E + 05	1.71E + 04	5.41E + 06	1.71E + 04	1.18E + 05	1.71E + 04	1.28E + 05	1.71E + 04	1.88E + 05	1.71E + 04	9.66E + 04	1.71E + 04	7.84E + 02	1.71E + 04	8.23E + 04
F6	9.93E + 03	9.20E + 01	2.97E + 03	1.29E + 00	3.01E + 03	8.69E + 01	1.99E + 03	2.94E + 01	2.77E + 03	1.14E + 03	2.87E + 03	4.31E + 03	2.92E + 03	5.86E + 02	3.01E + 03	7.45E + 01
F7	2.11E + 04	3.40E + 04	2.11E + 04	9.48E + 05	2.11E + 04	2.11E + 04	2.11E + 04	1.24E + 04	2.11E + 04	1.81E + 04	2.11E + 04	8.04E + 03	2.21E + 03	8.54E + 01	2.11E + 04	2.11E + 04
F8	9.97E + 03	3.21E + 03	2.45E + 03	5.80E + 02	2.85E + 03	8.94E + 02	3.53E + 03	4.91E + 02	2.52E + 03	1.37E + 03	2.49E + 03	3.11E + 03	2.32E + 03	2.00E + 03	2.29E + 03	1.95E + 03
F9	1.18E + 04	5.29E + 02	3.05E + 03	1.20E + 02	3.02E + 03	4.81E + 01	4.05E + 03	5.61E + 01	2.89E + 03	1.18E + 03	2.97E + 03	3.71E + 03	2.58E + 03	1.88E + 03	2.48E + 03	1.96E + 02
F10	1.51E + 04	3.93E + 03	3.79E + 03	8.52E + 02	4.12E + 03	5.23E + 02	5.27E + 03	4.58E + 02	3.76E + 03	1.72E + 03	3.78E + 03	4.72E + 03	2.63E + 03	2.62E + 03	2.97E + 03	2.62E + 03
	1	**0**	**1**	**1**	**1**	**2**	**0**	**1**	**1**	**0**	**1**	**1**	**4**	**4**	**2**	**2**
	7	**5**	**6**	**5**	**8**	**4**	**6**	**6**	**7**	**6**	**6**	**6**	**0**	**0**	**5**	**6**

Table 21
Friedman test based on the results of table 20.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
Friedman test based on mean values algorithms	5.02	4.20	4.75	4.96	3.89	3.92	2.10	3.41
Final rank algorithms based on mean values	8	5	6	7	3	4	1	2
Friedman test based on SD values algorithms	4.12	4.74	3.79	3.31	4.96	4.91	3.05	3.75
Final rank algorithms based on SD values	5	6	4	2	3	7	1	8

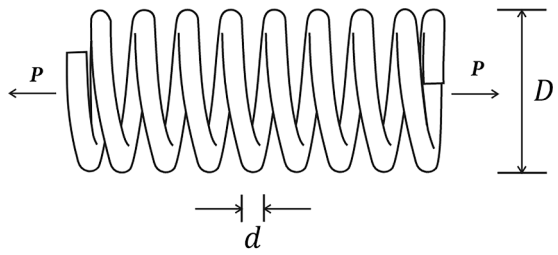

Fig. 11. Tension/compression spring design.

optimization algorithms such as PSO, CCS, ASMO, EHO, VNBA, RFO, and COA. As shown in problems having many dimensions, local optimizations, HFA with a lower frequency and lower number of elements can to achieve the total optimization. It should also be noted that the configurable parameters of the proposed algorithm are more than many other algorithms. Therefore, using this algorithm is difficult for beginner and non-professional researchers.

5.4. Application of HFA classic engineering optimization problems

Engineering problems are some real world problems that involve designing and building of systems and/or products. It is a decision making process that contains complex objective functions and a large number of decision variables. Mainly complex engineering problems involve large number of design variables. Therefore In this section, a comparison of HFA with some other meta-heuristics and golden standard results from analytical methods for some classic engineering optimization problems is presented.

5.5. 1. Compression spring problem

The goal of this design problem is to minimize the weight of the spring based on the constraints of minimum deflection, oscillation frequency, and shear stress. As shown in Fig. 11, the variables in this problem are $x = [x_1, x_2, x_3] =$ [dDN]. The variable ranges are:$x_1 \in [0.05, 2], x_2 \in [0.25, 1.3]$*and*$x_1 \in [2, 15]$. The objective function and constraints are given in Eqs. (6)–(7).

$$f(x) = (2+x_3)x_2x_1^2 \tag{6}$$

$$\begin{cases} g_1(x) = 1 - \dfrac{x_3x_2^3}{71785x_1^4}\hat{a}\copyright½0 \\ g_2(x) = \dfrac{4x_2^2 - x_2x_3}{12566(x_2x_1^3 - x_1^4)} + \dfrac{1}{5108x_1^2} - 1\hat{a}\copyright½0 \\ g_3(x) = 1 - \dfrac{140.45x_1}{x_2^2+x_1}\hat{a}\copyright½0 \\ g_4(x) = \dfrac{x_2+x_1}{1.5} - 1\hat{a}\copyright½0 \end{cases} \tag{7}$$

The optimal construction solution for the compression spring problem is stated in Table 23 for different optimization methods and golden standard results from analytical methods. Based on the performed simulations, the best solutions were for COA, RFO, HFA, EHO, ASMO, VNBA, PSO, CCS methods, respectively.

5.6. 2. Welded beam design problem

In this optimization, a construction composed of two components - beam and weld is considered. The welded beam design problem is mainly a minimization problem consisting of four variables that are presented in Fig. 12. The variables in this problem are physical quantities of the weld like height, length, etc. The variable ranges are $x_1, x_4 \in [0.1, 2]$*and*$x_2, x_3 \in [0.1, 10]$. The objective function and constraints is given in Eqs. (8)–(10).

$$f(x) = 1.1047x_1^2x_2 + 0.0481x_3x_4(14 + x_2) \tag{8}$$

$$\begin{cases} g_1(x) = \tau(x) - 13600\hat{a}\copyright½0 \\ g_2(x) = \sigma(x)\tau - 30000\hat{a}\copyright½0 \\ g_3(x) = x_1 - x_4\hat{a}\copyright½0 \\ g_4(x) = P - PC\hat{a}\copyright½0 \\ g_5(x) = 0.1047x_1^2 + 0.04811x_3x_4(14 + x_2) - 5\hat{a}\copyright½0 \\ g_6(x) = \delta(x) - 0.25\hat{a}\copyright½0 \\ g_7(x) = 0.125 - x_1\hat{a}\copyright½0 \end{cases} \tag{9}$$

where τ , σ , δ and PC are shear stress, the bending stress in the beam, deflection , the bending load on the rod, respectively.

$$\begin{aligned} \tau(x) &= \sqrt{(\tau')^2 + (\tau'')^2 + \frac{2\tau'\tau''x_2}{2R}} \\ \tau' &= \frac{P}{\sqrt{2}x_1x_2} \\ \tau'' &= \frac{MP}{J}, M = P\left(L + \frac{x_2}{2}\right) \\ R &= \sqrt{\left(\frac{x_2}{2}\right)^2 - \left(\frac{x_1+x_3}{2}\right)^2} \\ \sigma(x) &= \frac{6PL}{x_4x_3^2}, \delta(x) = \frac{6PL^3}{Ex_4x_3^2} \\ Pc(x) &= \frac{4.013E\sqrt{\frac{x_4^6x_3^2}{36}}}{L^2}\left(1 - \frac{x_3\sqrt{\frac{E}{4G}}}{2L}\right) \end{aligned} \tag{10}$$

where P = 6000 lb , L = 14in, E = $30*10^6$ psi, G = $12*10^6$ psi.

The obtained optimal variables for the welded beam design problem are expressed in Table 24 for different optimization methods and golden

Table 23
Optimal construction solution for the compression spring problem.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA	Gold Standard Results	
									Method	
									Constraint correction	Mathematical optimization
x_1	5.21E-02	5.21E-02	5.19E-02	5.14E-02	5.21E-02	5.22E-02	5.12E-02	5.22E-02	5.07E-02	5.07E-02
x_2	3.68E-01	3.71E-01	3.61E-01	3.54E-01	3.59E-01	3.70E-01	3.59E-01	3.65E-01	3.59E-01	3.59E-01
x_3	1.17E + 01	1.18E + 01	1.16E + 01	1.19E + 01	1.16E + 01	1.08E + 01	1.17E + 01	1.17E + 01	1.16E + 01	1.16E + 01
$f(x)$	1.37E-02	1.39E-02	1.32E-02	1.30E-02	1.33E-02	1.25E-02	1.26E-02	1.25E-02	1.24E-02	1.24E-02

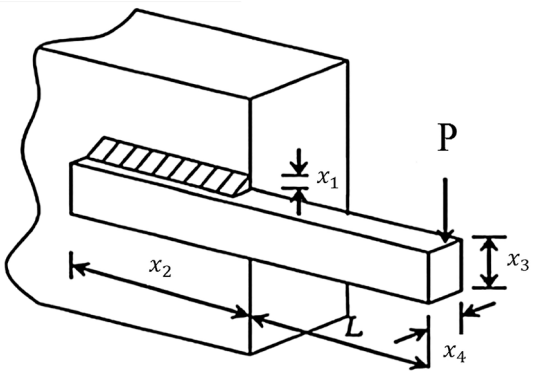

Fig. 12. Welded beam design.

Table 24
The obtained optimal variables for the welded beam design problem.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA	Gold Standard Results	
									Method	
									Siddall	Ragsdell
x_1	2.52E-01	2.87e-01	2.60E-01	2.64E-01	2.68E-01	2.69E-01	2.44E-01	2.65E-01	2.44E-01	2.45E-01
x_2	6.51E + 00	6.72e + 00	6.49E + 00	6.38E + 00	6.39E + 00	6.51E + 00	6.22E + 00	6.48E + 00	6.22E + 00	6.02E + 00
x_3	8.63E + 00	8.37e + 00	8.98E + 00	8.81E + 00	8.88E + 00	8.87E + 00	8.29E + 00	8.59E + 00	8.29E + 00	8.27E + 00
x_4	2.45E-01	2.21e-01	2.20E-01	2.19E-01	2.20E-01	2.25E-01	2.33E-01	2.29E-01	2.44E-01	2.45E-01
$f(x)$	2.5426	2.4609	2.4317	2.3869	2.4230	2.4893	2.3308	2.4405	2.3815	2.3859

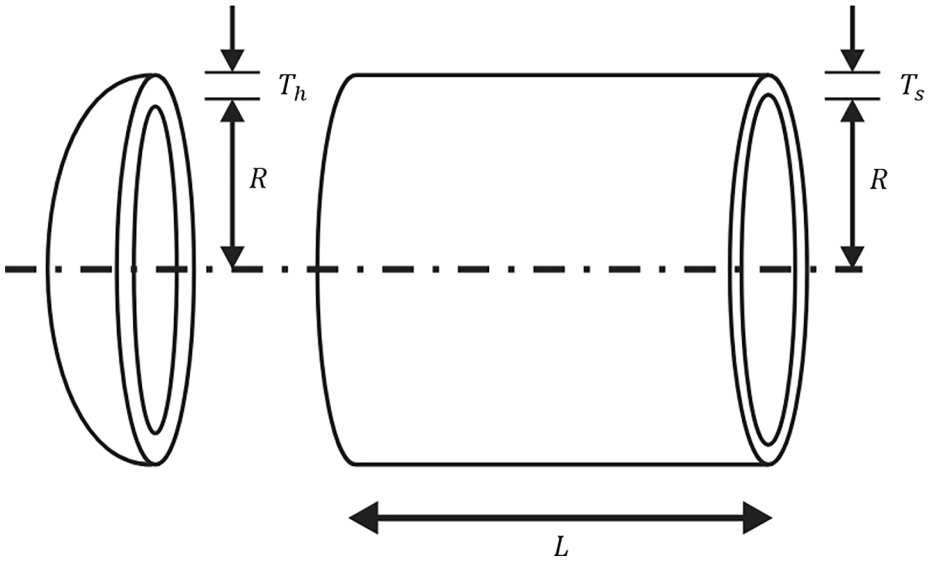

Fig. 13. Pressure vessel design.

Table 25
Optimal construction variables for the pressure vessel design problem.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA	Gold Standard Results	
									Method	
									Lagrange multiplier	Branch-bound
x_1	8.25E-01	8.27E-01	8.25E-01	8.18E-01	8.18E-01	8.25E-01	8.17E-01	8.27E-01	1.25E-00	1.25E-01
x_2	5.12E-01	4.50E-01	4.43E-01	4.52E-01	4.52E-01	4.49E-01	4.46E-01	4.53E-01	6.25E-01	6.25E-01
x_3	4.23E + 01	4.26E + 01	4.26E + 01	4.22E + 01	4.22E + 01	4.23E + 01	4.23E + 01	4.22E + 01	5.82E + 01	4.77E + 01
x_4	1.77E + 02	1.77E + 02	1.76E + 02	1.77E + 02	1.77E + 02	1.77E + 02	1.77E + 02	1.77E + 02	4.36E-01	1.17E + 03
$f(x)$	6.42E + 03	6.28E + 03	6.23E + 03	6.20E + 03	6.18E + 03	6.22E + 03	6.15E + 03	6.17E + 03	7.19E + 03	8.12E + 03

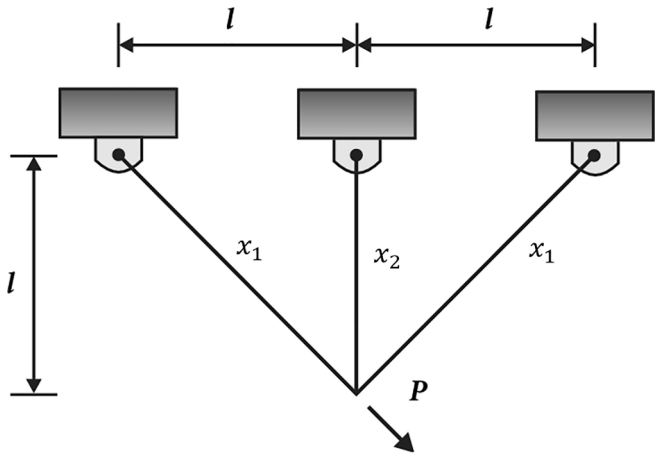

Fig. 14. Three-bar truss design.

standard results from analytical methods. According to this table, the best answers were for HFA, EHO, ASMO, VNBA, COA, CCS, RFO, PSO methods, respectively.

5.7. 3. Pressure design vessel problem

The main objective of this problem is to minimize the manufacturing, welding and material cost of the pressure vessel. Four variables are $x = [x_1, x_2, x_3, x_4] = [T_s T_h \mathrm{RL}]$ as shown in Fig. 13.

Objective function and condition for dimensions of the cylinder are described in Eqs. (11)–(12), respectively.

The variable ranges are $x_1, x_2 \in [0, 100] and x_2, x_3 \in [10, 200]$.

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_3^2x_2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{11}$$

$$\begin{cases} g_1(x) = -x_1 + 0.0193x_3 \leqslant 0 \\ g_2(x) = 0.00954x_2 - x_3 \leqslant 0 \\ g_3(x) = 129600 - \frac{4}{3}\pi x_3^3 - \pi x_3^2 x_4 \leqslant 0 \\ g_4(x) = x_4 - 240 \leqslant 0 \end{cases} \tag{12}$$

Optimal construction variables for the pressure vessel design problem are expressed in Table 25 for different optimization methods and golden standard results from analytical methods. Based on the information in this table, the best answers were for HFA, COA, VNBA, EHO, RFO, ASMO, CCS, PSO methods, respectively.

5.7.1. Three-bar truss design

The three-bar truss design problem is intended to minimize the volume of the three-bar truss with each truss member

constrained by stress r. Three variables are $x = [x_1, x_2, x_3]$ as shown in Fig. 14.

Objective function and condition for the three-bar truss design problem are described in Eqs. (13)–(14), respectively.

The variable ranges are $x_1, x_2 \in [0, 1] and x_1 = x_3$.

$$f(x) = l\left(2\sqrt{2}x_1 + x_2\right) \tag{13}$$

$$\begin{cases} g_1(x) = \dfrac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leqslant 0 \\ g_1(x) = \dfrac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leqslant 0 \\ g_3(x) = \dfrac{1}{\sqrt{2}x_1 + 2x_1} P - \sigma \leqslant 0 \end{cases} \tag{14}$$

where L = 100 cm , σ = P = 2 kN/cm^2.

Optimal construction variables for the three-bar truss optimization problem are expressed in Table 26 for different optimization methods. By looking at this table, it can be seen that the best designs were for VNBA, HFA, COA, RFO, ASMO, EHO, CCS, PSO methods, respectively.

5.8. Application of HFA to Optimize Tuning Hyper-Parameters:

The objective of the support vector machine (SVM) algorithm is to find a hyperplane in an N-dimensional space (N-the number of features) that distinctly classifies the data points. SVM has many applications in machine learning science, such as image analysis tasks, risk analysis system, automated car steering system, quality analysis, product design analysis, voice recognition, inventory cost consulting systems, financial analysis (Cervantes, Garcia-Lamont, Rodríguez-Mazahua, & Lopez, 2020; Chandra & Bedi, 2018).

In practice, the performance of SVMs usually depends on its hyper-parameters. There are two major types of algorithms in SVMs: classification and regression. In this experiment, we apply SVMs to classify some real-world practical datasets. Suppose dataset are m data points $\{x_i, y_i\}_{i=1}^m$ with n-dimensional features $x_i \in R^n$ and respective class labels. The mathematical expressions of SVM is shown as in Eq. (15).

$$\max \sum_j \alpha_j - \frac{1}{2}\sum_{j,k} \alpha_j \alpha_k y_j y_k k\left(x_j, x_k\right) \tag{15}$$

$$s.t : 0 \leq \alpha_j \leq c \quad and \quad \sum_j \alpha_j y_j = 0$$

where $\alpha = [\alpha_1, \ldots, \alpha_m]$ is Lagrange multipliers and C is the tunable penalty factor and K is the kernel function. Due to the outstanding performance of radial based function (RBF) kernel function, it is used in this test as stated in Eq.16.

$$k\left(x_j, x_k\right) = \exp\left(-\frac{\|x_j - x_k\|^2}{2\sigma^2}\right) \tag{16}$$

where σ is another tunable parameter. Using this kernel in the SVM classifier, we can get the decision function as shown in Eq.17.

$$f(x) = sign\left[\sum_i \alpha_j y_j \exp\left(-\frac{\|x - x_i\|^2}{2\sigma^2}\right) + b\right] \tag{17}$$

In this test, we have to optimize two hyper-parameters: the penalty factor C and σ for classification problems (Wang & Pardalos, 2014).

Table 26
Optimal construction variables for the three-bar truss optimization problem.

	PSO	CCS	ASMO	EHO	VNBA	RFO	Proposed Algorithm HFA	COA
x_1	8.32E-01	8.10E-01	7.66E-01	7.85E-01	7.803E − 01	7.39E-01	7.34E-01	7.65E-01
x_2	6.32E-01	6.34E-01	5.82E-01	6.64E-01	4.330E − 01	7.84E-01	6.22E-01	5.89E-01
$f(x)$	2.99E + 02	2.93E + 02	2.72E + 02	2.88E + 02	2.64E + 02	2.87E + 02	2.70E + 02	2.75E + 02
g_1	−2.19E-01	−1.80E-01	−6.77E-02	−1.46E-01	−4.24E-04	−1.06E-01	−1.68E-02	−6.77E-02
g_2	−1.38E + 00	−1.35E + 00	−1.32E + 00	−1.31E + 00	−1.44E + 00	−1.19E + 00	−1.26E + 00	−1.32E + 00
g_3	−8.41E-01	−8.29E-01	−7.48E-01	−8.39E-01	−5.64E-01	−9.18E-01	−7.60E-01	−7.48E-01

Table 27
The performance of algorithms in tuning hyper-parameters of SVM optimization.

Dataset	PSO		CCS		ASMO		EHO		VNBA		RFO		Proposed Algorithm HFA		COA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Australian Credit Approval	0.7871	0.0522	0.7696	0.0231	0.7802	0.0159	0.7654	0.0089	0.8165	0.7365	0.7365	0.0031	0.7261	0.0072	0.7665	0.0090
HCC Survival	0.7990	0.0456	0.6892	0.0625	0.6792	0.0519	0.6921	0.0429	0.7365	0.7132	0.7164	0.0536	0.7069	0.0421	0.6272	0.0415
Iris	0.9710	0.0008	0.9752	0.0031	0.9773	0.0041	0.9773	0.0011	0.9772	0.9773	0.9773	0.0017	0.9574	0.0002	0.9774	0.0002
Somerville Happiness Survey	0.5494	0.0180	0.5587	0.0128	0.5691	0.0029	0.5291	0.0097	0.5354	0.5398	0.5564	0.0095	0.5490	0.0098	0.5565	0.0197
Wine	0.9723	0.0075	0.9982	0.0083	0.9501	0.0019	0.9591	0.0009	0.9604	0.0007	0.9415	0.0027	0.9614	0.0013	0.9644	0.0011

The 5 most popular databases used are as follows:

- Australian Credit Approval: A well-known dataset that concerns credit card applications approval in Australia (Dua & Graff, 2017);
- HCC Survival: HCC dataset was obtained at a University Hospital in Portugal and contains several demographic, risk factors, laboratory and overall survival features of 165 real patients diagnosed with HCC (Santos, Abreu, García-Laencina, Simão, & Carvalho, 2015);
- Iris: This is perhaps the best known dataset to be found in machine learning. It is to classify type of iris plant (Dua & Graff, 2017);
- Somerville Happiness Survey: A dataset about life survey (Koczkodaj et al., 2017).
- Wine: This dataset is the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars (Dua & Graff, 2017);

All datasets as listed above are publicly available at (Dua & Graff, 2017). As for the searching space, we set C, $\sigma \in [10^{-5}, 10^{+5}]$. The accuracy of 10-f old cross validation is computed as the fitness in the evaluation. The maximum iteration is set as 200.

The detailed results are illustrated into Table 27 where the mean fitness represents the average classification accuracy of 30 runs. HFA outperforms over all other algorithms on 3 out of 5 datasets. HFA has performed best in the Australian Credit Approval, Iris and Wine databases. COA and VNBA performed best in the HCC Survival and Somerville Happiness Survey databases, respectively. The HFA algorithm had the lowest SD values in the Australian Credit Approval and Iris databases, and the COA, RFO and VNBA algorithms had the best SD values in the HCC Survival, Somerville Happiness Survey and Wine databases, respectively. Comparison of the algorithm in five famous databases shows that the proposed HFA has a good performance of tuning hyper-parameters and can be used as a powerful tool by researchers.

5.9. Constraints and the tuning of the HFA parameters

In most problems, improper adjustment of the parameters optimization algorithms makes that the time to reach the appropriate answer become long or impossible. The precise tuning of the optimization algorithms, as well as the proposed HFA, depends on various indicators such as search space, the dimensions, and the complexity of the problem. It is difficult and impossible to fine-tune and optimize the parameters without examining the problem. Usually, the adjustment parameters at all stages are assumed to be fixed. The adjustment parameters should have the necessary changes during the optimization steps according to the different conditions of the problem. There are also methods and experiments to properly adjust the parameters of optimization algorithms .According to the above, it is not possible to set the optimal values of HFA parameters without considering the optimization problem. In Table 28, the appropriate and suggested values for the HFA parameters is presented, which lead to the appropriate answer in solving most of the problems. Also, the constraints and practical tips for setting HFA parameters are stated in the Table 28.

6. Conclusion

Many innovative optimization algorithms have been inspired by nature and collective intelligence. In this paper, a new optimization method, named HFA, is presented. This algorithm was inspired by the movement of human beings towards happiness and felicity.

The main step in achieving a sense of felicity is due to the thoughts of people in society to the universe. Therefore, the goal of the HFA in optimizing issues is to change the thoughts of the world and achieve a maximum sense of felicity. In fact, in HFA, a change of felicity is occurred by changing the thoughts of people in society. Dimensions the problem of optimization is the dimensions of thought to the world, optimization function is felicity, and the best thought is the thought that

Table 28
Constraints and the Tuning Of the HFA Parameters.

Parameter	Description	Recommended Amount	Constraints	
			Select a larger parameter	**Select a smaller parameter**
N_{pop}	Initial population	It depends entirely on the size of the problem and its complexity	- Problem solving time getting longer	- Not reaching the optimal answer - Problem solving time getting longer
N_{elite}	Number of elites	2-4 (larger initial solution steps can be selected and reduced over time)	Lack of proper search of the whole space (searches only around the elite)	Stuck in the local minimum
$N_{disciple}$	the number of disciples	Consider 2 points: 1- About 20% to 30% of the*Npop*. 2- 1 to 3 times that of the elite. The minimum number obtained from the above two constraints should be allocated as the number of the disciple.	Lack of proper search of the whole space (searches only around the elite)	Convergence occurs slowly
$N_{popular}$	Number of ordinary people	$Npop - Nelite - Ndisciple$		
$s(n)$	Step length in local search	Depends on the search space (it is recommended to choose a shorter step length for the elite and a larger step length for ordinary people)	- Convergence occurs slowly - Not reaching the optimal answer	It looks like there was no local search
$N_{personal}$	Number of local searches	2-4 (It is suggested to choose a larger number for elites and disciples(	Problem solving time getting longer	Not reaching the optimal answer
$N_{unidirection}$	Number of moves in the right direction	2-4 (It is suggested to choose a larger number for elites and disciples)	Problem solving time getting longer	Not reaching the optimal answer
$N_{alteration}$	Number of moves to the elite	10-30	Problem solving time getting longer	- Problem solving time getting longer - Convergence occurs slowly
$N_{revolution}$	Number of revolutions	It depends entirely on the size of the problem and its complexity	Problem solving time getting longer	Not reaching the optimal answer
N_{rev}	The number of people who participate in each revolution	(60% - 90%)*$Npop$ (Revolution does not happen for the elite)		- Not reaching the optimal answer - Stuck in the local minimum

results in the greatest felicity. To achieve the steps of HFA, ways have been studied to change the mindset of human beings to achieve maximum felicity.1) In the first step, people in society are influenced by elites and prophets. Therefore, in this step people are divided into three categories: A) the elites in a society who enjoy higher felicity and are considered as intellectual leaders of part of the society, B) the disciples who are trained by those elites and try to observe the world from their perspectives and C) ordinary people who are influenced by several elites. These three categories try to achieve felicity based on their personal experiences, being influenced by one another, and sudden changes in their thoughts. In this algorithm, to determine the elite, only the latest situation of the people in the society is not examined and the situation of the people in the society is deliberated in a period (the situation of the person in the last few stages). This is done to identify false and untrue prophets. 2) In the second step, each person has his own experiences that can change his thoughts. 3) In the third step, for various reasons, such as famine, war, and the death of loved ones, a deep spiritual revolution may occur in some people in society and their worldview may change fundamentally.

To test the capability of the proposed algorithm in solving various real-world complex engineering problems, three types of test functions have been used: 1) CEC 2014 benchmark test functions 30D, 50D and 100D 2) CEC-C06 2019 benchmark test functions 3) CEC 2020 benchmark test functions. Besides, for a more accurate comparison, the proposed algorithm is compared with PSO, CCS, ASMO, EHO, VNBA, RFO, and COA algorithms in all stages. In addition, the Friedman test is used to rank the algorithms in each of the modes, and the P-value is calculated. In CEC 2014 benchmark test functions, for unimodal functions, simple multimodal functions, hybrid functions, and composition functions HFA can find optimal solution 88%, 82%, 87%, and 60% respectively. The mean and standard deviation of the proposed algorithm in CEC-C06 2019 benchmark test functions had the best response among all algorithms 80% and 60%, respectively. In CEC 2020 benchmark test functions, for unimodal functions, basic functions, hybrid functions, and Composition Functions, HFA can find optimal solution 100%, 66%, 100%, and 33% respectively. Also, to evaluate the ability of HFA to solve classic engineering optimization problems, 4 standard optimization problems are examined and compared. HFA found the best answer in two of these engineering problems, the second best answer in one of the engineering problems, and the third best answer in one of the engineering problems among the simulated algorithms. To test the ability of the proposed algorithm in hyper-parameters tuning, the five-database classification was simulated using the SVM algorithm. HFA outperforms over all other algorithms on 3 out of 5 datasets. Therefore, it can be concluded that the HFA has a high ability to find the best answer for different objective functions and has good performance in optimizing real complex problems with many dimensions. It can also be claimed that the performance of the HFA is better than many optimization algorithms introduced in recent years.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abdullah, J. M., & Ahmed, T. (2019). Fitness dependent optimizer: Inspired by the bee swarming reproductive process. *IEEE Access, 7*, 43473–43486.

Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A., & Gandomi, A. H. (2021). Aquila Optimizer: A novel meta-heuristic optimization Algorithm. *Computers & Industrial Engineering, 157*, Article 107250.

Askari, Q., Younas, I., & Saeed, M. (2020). Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowledge-Based Systems, 195*, Article 105709.

Bäck, T., & Schwefel, H. P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation, 1*(1), 1–23.

Bangyal, W. H., Ahmad, J., Rauf, H. T., & Pervaiz, S. (2018). An overview of mutation strategies in bat algorithm. *International Journal of Advanced Computer Science and Applications (IJACSA), 9*(8), 523–534.

Bonabeau, E., Theraulaz, G., & Dorigo, M. (1999). Swarm intelligence (pp. 32-77). Oxford.

Bonyadi, M. R., & Michalewicz, Z. (2017). Particle swarm optimization for single objective continuous space problems: A review. *Evolutionary Computation, 25*(1), 1–54.

Carrasco, J., García, S., Rueda, M. M., Das, S., & Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation, 54*, Article 100665.

Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing, 408*, 189–215.

Chandra, M. A., & Bedi, S. S. (2018). Survey on SVM and their application in image classification. *International Journal of Information Technology*, 1–11.

Cheng, R., He, C., Jin, Y., & Yao, X. (2018). Model-based evolutionary algorithms: A short survey. *Complex & Intelligent Systems, 4*(4), 283–292.

Cheng, R., & Jin, Y. (2014). A competitive swarm optimizer for large scale optimization. *IEEE Transactions on Cybernetics, 45*(2), 191–204.

Covic, N., & Lacevic, B. (2020). Wingsuit flying search—A novel global optimization algorithm. *IEEE Access, 8*, 53883–53900.

Das, B., Mukherjee, V., & Das, D. (2020). Student psychology based optimization algorithm: A new population based optimization algorithm for solving optimization problems. *Advances in Engineering Software, 146*, Article 102804.

De Vasconcelos Segundo, E. H., Mariani, V. C., & dos Santos Coelho, L. (2019a). Design of heat exchangers using Falcon Optimization Algorithm. *Applied Thermal Engineering, 156*, 119–144.

De Vasconcelos Segundo, E. H., Mariani, V. C., & dos Santos Coelho, L. (2019b). Metaheuristic inspired on owls behavior applied to heat exchangers design. *Thermal Science and Engineering Progress, 14*, Article 100431.

Derrac, J., García, S., Hui, S., Suganthan, P. N., & Herrera, F. (2014). Analyzing convergence performance of evolutionary algorithms: A statistical approach. *Information Sciences, 289*, 41–58.

Dhal, K. G., Ray, S., Das, A., & Das, S. (2019). A survey on nature-inspired optimization algorithms and their application in image enhancement domain. *Archives of Computational Methods in Engineering, 26*(5), 1607–1638.

Dhiman, G., & Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems, 165*, 169–196.

Dua, D., & Graff, C. (2017). UCI machine learning repository.

Eberhart, R., & Kennedy, J. (1995, October). A new optimizer using particle swarm theory. In MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science (pp. 39-43). IEEE.

Eftimov, T., & Korošec, P. (2019). A novel statistical approach for comparing meta-heuristic stochastic optimization algorithms according to the distribution of solutions in the search space. *Information Sciences, 489*, 255–273.

Elhosseini, M. A., El Sehiemy, R. A., Rashwan, Y. I., & Gao, X. Z. (2019). On the performance improvement of elephant herding optimization algorithm. *Knowledge-Based Systems, 166*, 58–70.

Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems, 191*, Article 105190.

Fogel, D. B. (2006). *Evolutionary computation: toward a new philosophy of machine intelligence*. John Wiley & Sons.

Gabis, A. B., Meraihi, Y., Mirjalili, S., & Ramdane-Cherif, A. (2021). A comprehensive survey of sine cosine algorithm: Variants and applications. *Artificial Intelligence Review*, 1–72.

Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and machine learning.

Hashim, F. A., Houssein, E. H., Hussain, K., Mabrouk, M. S., & Al-Atabany, W. (2022). Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Mathematics and Computers in Simulation, 192*, 84–110.

Houssein, E. H., Saad, M. R., Hashim, F. A., Shaban, H., & Hassaballah, M. (2020). Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems. *Engineering Applications of Artificial Intelligence, 94*, Article 103731.

Isiet, M., & Gadala, M. (2020). Sensitivity analysis of control parameters in particle swarm optimization. *Journal of Computational Science, 41*, Article 101086.

Jain, M., Singh, V., & Rani, A. (2019). A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm and Evolutionary Computation, 44*, 148–175.

Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation, 214*(1), 108–132.

Kashan, A. H., Tavakkoli-Moghaddam, R., & Gen, M. (2019). Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm: An effective algorithm with new evolutionary operators for global optimization. *Computers & Industrial Engineering, 128*, 192–218.

Koczkodaj, W. W., Kakiashvili, T., Szymanska, A., Montero-Marin, J., Araya, R., Garcia-Campayo, J., … Strzalka, D. (2017). How to reduce the number of rating scale items without predictability loss? *Scientometrics, 111*(2), 581.

Koza, J. R. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing, 4*(2), 87–112.

Koza, J. R., & Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*. MIT press.

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems, 111*, 300–323.

Li, J., Guo, L., Li, Y., & Liu, C. (2019). Enhancing elephant herding optimization with novel individual updating strategies for large-scale optimization problems. *Mathematics, 7*(5), 395.

Li, W., Wang, G. G., & Alavi, A. H. (2020). Learning-based elephant herding optimization algorithm for solving numerical optimization problems. *Knowledge-Based Systems, 195*, Article 105675.

Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 635, 490.

Liu, L., Liu, X., Wang, N., & Zou, P. (2018). Modified cuckoo search algorithm with variational parameters and logistic map. *Algorithms, 11*(3), 30.

Mashayekhi, M., Harati, M., & Estekanchi, H. E. (2019). Development of an alternative PSO-based algorithm for simulation of endurance time excitation functions. *Engineering Reports, 1*(3), Article e12048.

Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120–133.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163–191.

Mohapatra, P., Das, K. N., & Roy, S. (2017). A modified competitive swarm optimizer for large scale optimization problems. *Applied Soft Computing, 59*, 340–362.

Moosavi, S. H. S., & Bardsiri, V. K. (2019). Poor and rich optimization algorithm: A new human-based and multi populations algorithm. *Engineering Applications of Artificial Intelligence, 86*, 165–181.

Muthiah-Nakarajan, V., & Noel, M. M. (2016). Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion. *Applied Soft Computing, 38*, 771–787.

Pierezan, J., & Coelho, L. D. S. (2018). Coyote optimization algorithm: a new metaheuristic for global optimization problems. In *2018 IEEE congress on evolutionary computation (CEC)* (pp. 1–8). IEEE.

Połap, D. (2017). Polar bear optimization algorithm: Meta-heuristic with fast population movement and dynamic birth and death mechanism. *Symmetry, 9*(10), 203.

Połap, D., & Woźniak, M. (2021). Red fox optimization algorithm. *Expert Systems with Applications, 166*, Article 114107.

Price, K. V., Awad, N. H., Ali, M. Z., & Suganthan, P. N. (2018). The 100-digit challenge: Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. Nanyang Technological University.

Rahman, C. M., & Rashid, T. A. (2021). A new evolutionary algorithm: Learner performance based behavior algorithm. *Egyptian Informatics Journal, 22*(2), 213–223.

Santos, M. S., Abreu, P. H., García-Laencina, P. J., Simão, A., & Carvalho, A. (2015). A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of Biomedical Informatics, 58*, 49–59.

Shabani, A., Asgarian, B., Salido, M., & Gharebaghi, S. A. (2020). Search and rescue optimization algorithm: A new optimization method for solving constrained engineering optimization problems. *Expert Systems with Applications, 161*, Article 113698.

Sharma, A., Sharma, A., Panigrahi, B. K., Kiran, D., & Kumar, R. (2016). Ageist spider monkey optimization algorithm. *Swarm and Evolutionary Computation, 28*, 58–77.

Shayanfar, H., & Gharehchopogh, F. S. (2018). Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems. *Applied Soft Computing, 71*, 728–746.

Singh, P. R., Abd Elaziz, M., & Xiong, S. (2019). Ludo game-based metaheuristics for global and engineering optimization. *Applied Soft Computing, 84*, Article 105723.

Slowik, A., & Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 1–17.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4), 341–359.

Sulaiman, M. H., Mustaffa, Z., Saari, M. M., & Daniyal, H. (2020). Barnacles mating optimizer: A new bio-inspired algorithm for solving engineering optimization problems. *Engineering Applications of Artificial Intelligence, 87*, Article 103330.

Tong, H., Zhu, Y., Pierezan, J., Xu, Y., & dos Santos Coelho, L. (2021). Chaotic coyote optimization algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 1–21.

Wang, G. G., Deb, S., Gandomi, A. H., Zhang, Z., & Alavi, A. H. (2014). A novel cuckoo search with chaos theory and elitism scheme. In *2014 International Conference on Soft Computing and Machine Intelligence* (pp. 64–69). IEEE.

Wang, G. G., Deb, S., Gandomi, A. H., Zhang, Z., & Alavi, A. H. (2016). Chaotic cuckoo search. *Soft Computing, 20*(9), 3349–3362.

Wang, H., Geng, Q., & Qiao, Z. (2014). Parameter tuning of particle swarm optimization by using Taguchi method and its application to motor design. In *2014 4th IEEE international conference on information science and technology* (pp. 722–726). IEEE.
Wang, G. G., Lu, M., & Zhao, X. J. (2016). An improved bat algorithm with variable neighborhood search for global optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1773–1778). IEEE.
Wang, X., & Pardalos, P. M. (2014). A survey of support vector machines with uncertainties. *Annals of Data Science, 1*(3–4), 293–309.
Yadav, A. (2019). AEFA: Artificial electric field algorithm for global optimization. *Swarm and Evolutionary Computation, 48*, 93–108.
Yapici, H., & Cetinkaya, N. (2019). A new meta-heuristic optimizer: Pathfinder algorithm. *Applied Soft Computing, 78*, 545–568.
Yuan, Z., Wang, W., Wang, H., & Yildizbasi, A. (2020). Developed coyote optimization algorithm and its application to optimal parameters estimation of PEMFC model. *Energy Reports, 6*, 1106–1117.
Yue, C. T., Price, K. V., Suganthan, P. N., Liang, J. J., Ali, M. Z., Qu, B. Y., … Biswas, P. (2019). *Problem Definitions and Evaluation Criteria for the CEC 2020 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization*. Technial Report. Singapore: Nanyang Technological University.
Zhang, M., Xu, Z., Lu, X., Liu, Y., Xiao, Q., & Taheri, B. (2021). An optimal model identification for solid oxide fuel cell based on extreme learning machines optimized by improved Red Fox Optimization algorithm. *International Journal of Hydrogen Energy, 46*(55), 28270–28281.
Zhao, W., Wang, L., & Zhang, Z. (2019). Supply-demand-based optimization: A novel economics-inspired algorithm for global optimization. *IEEE Access, 7*, 73182–73206.