Synergistic Strategies for Meta-heuristic Optimization Learning Algorithms

Jiann-Horng Lin

Abstract— **The theory of optimization in machine learning has benefited from various fields of research. Among these, nature-inspired meta-heuristic algorithms are becoming very powerful and increasingly popular in solving optimization problems and applications. Some algorithms have strong similarities, while others may be directly based on or inspired by some core algorithms. For most algorithms, we know their fundamental components, but how exactly they interact to achieve efficiency still remains partly unknown, which inspires more active studies. It will also be valuable for providing new directions for further important modifications on these algorithms or even pointing out innovative ways of developing new algorithms. In this paper, we propose some synergistic approaches to meta-heuristic search algorithms for optimization learning. Among most machine learning algorithms for optimization problem including meta-heuristic search algorithms, the solution is drawn like a moth to a flame and cannot keep away. That is, one of the major concerns in using learning algorithms to search complex state spaces is the problem of premature convergence. It means loss of diversity in the process of evolution. Maintaining a certain degree of diversity is widely believed to help avoid entrapment in non-optimal solutions. The fine balance between intensification (exploitation) and diversification (exploration) is very important to the overall efficiency and performance of a meta-heuristic search algorithm. Too little exploration and too much exploitation could cause the system to be trapped in local optima, which makes it very difficult or even impossible to find the global optimum. All meta-heuristic algorithms use certain tradeoff between local search and global exploration. The diversification via randomization provides a good way to move away from local search to the search on the global scale and avoids the solutions being trapped at local optima, while increases the diversity of the solutions. The good combination of these two major components will usually ensure that the global optimality is achievable. It is worth pointing that the use of a uniform distribution is not the only way to achieve randomization. In fact, random walks such as Levy flights on a global scale are more efficient. On the other hand, the track of chaotic variable can travel ergodically over the whole search space. In general, the chaotic variable has special characters, i.e., ergodicity, pseudo-randomness and irregularity. To enrich the searching behavior and to avoid being trapped into local optimum, chaotic sequence and a chaotic Levy flight are incorporated in the meta-heuristic search for efficiently generating new solutions. In addition to chaotic dynamics, we also propose some ideas inspired by psychology model of emotion for use in meta-heuristic algorithms which promises greater efficiency and perhaps solvability of problems to an optimization approach. We presented synergistic strategies for meta-heuristic optimization learning, with an emphasis on the balance between intensification and diversification. It provides us with efficient tools to get better insight into learning mechanisms.**

Jiann-Horng Lin, Dept. of Information Management, I-Shou University, Taiwan.

Index Terms— **Computational Intelligence, Learning algorithm, Machine Learning, Optimization Theory**

I. INTRODUCTION

This paper explores optimization theory and algorithms based on computational intelligence for complex systems. Nature-inspired meta-heuristic algorithms are becoming increasingly popular in optimization and applications. Meta-heuristics is about emergent behavior and patterns, self-organization, and simple processes leading to complex results. The meta-heuristics has ever since turned out to be a competitor in the field of numerical optimization. There are many reasons for this popularity and success, and one of the main reasons is that these algorithms have been developed by mimicking the most successful processes in nature, including biological systems, and physical and chemical processes. From a practical viewpoint, evolutionary algorithms are population-based meta-heuristics that provide the human engineer with a set of tools to address particular optimization problems. The core principles are built upon two complementary mechanisms, inspired from Darwin's original principles: blind variations (favoring the introduction of new candidates) and survival of the fittest (favoring pressure towards the best individuals). For most algorithms, we know their fundamental components, how exactly they interact to achieve efficiency still remains partly a mystery, which inspires more active studies. Several optimization techniques can be used, but we proposed here a simple yet effective meta-heuristic algorithm. Cuckoo search is an optimization algorithm developed by Xin-she Yang and Suash Deb in 2009 [8]. It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species). Cuckoo search idealized such breeding behavior, and can be applied for various optimization problems. Bat-inspired algorithm is an another metaheuristic optimization algorithm developed by Xin-She Yang in 2010 [12]. This bat algorithm is based on the echolocation behavior of microbats with varying pulse rates of emission and loudness. Furthermore, the firefly algorithm [13] is also a meta-heuristic algorithm, inspired by the flashing behavior of fireflies. The primary purpose for a firefly's flash is to act as a signal system to attract other fireflies. In firefly algorithm, the flashing light can be formulated in such a way that it is associated with the objective function to be optimized, which makes it possible to formulate the firefly algorithm. Among most machine learning algorithms for optimization problem including meta-heuristic search algorithms, the solution is drawn like a moth to a flame and cannot keep away. To enrich the searching behavior and to avoid being trapped into local optimum, some meta-heuristic search algorithms intended to introduce chaotic dynamics, Levy flights and psychology model of emotion into the algorithm are presented in this paper. We propose some synergistic approaches to meta-heuristic search optimization algorithms. Some new approaches to cuckoo search algorithm, bat algorithm and firefly algorithm as the synergistic meta-heuristics are developed. It provides us with efficient tools to get better insight into learning mechanisms. In simulation, we illustrate the application of the synergistic search algorithm in meta-heuristics for the reconstruction of chaotic dynamics. The proposed method includes both effects of chaotic dynamics and psychology model of emotion based search algorithm. Simulation results are provided to illustrate the effectiveness and feasibility of the proposed algorithm.

II. Meta-heuristic Optimization Search Algorithms

In this section, we will first discuss all the latest nature-inspired meta-heuristic algorithms including cuckoo search, bat-inspired algorithm and firefly algorithm.

A. *Meta-heuristic Cuckoo Search Algorithm*

Cuckoo Search is a meta-heuristic search algorithm which has been proposed recently by Yang and Deb [9]. The algorithm is inspired by the reproduction strategy of cuckoos. At the most basic level, cuckoos lay their eggs in the nests of other host birds, which may be of different species. The host bird may discover that the eggs are not its own and either destroy the egg or abandon the nest all together. This has resulted in the evolution of cuckoo eggs which mimic the eggs of local host birds [10]. To apply this as an optimization tool, Yang and Deb [9] used three idealized rules: (1) Each cuckoo lays one egg, which represents a set of solution co-ordinates, at a time and dumps it in a random nest; (2) A fraction of the nests containing the best eggs, or solutions, will carry over to the next generation; (3) The number of nests is fixed and there is a probability that a host can discover an alien egg. If this happens, the host can either discard the egg or the nest and this results in building a new nest in a new location. The steps involved in the Cuckoo Search are then derived from these rules and are shown in Fig. 1 [10]. An important component of a Cuckoo Search is the use of Levy flights for both local and global searching. The Levy flight process, which has previously been used in search algorithms [6], is a random walk that is characterized by a series of instantaneous jumps chosen from a probability density function which has a power law tail. This process represents the optimum random search pattern and is frequently found in nature [7]. When generating a new egg in Fig. 1, a Levy flight is performed starting at the position of a randomly selected egg, if the objective function value at these new coordinates is better than another randomly selected egg then that egg is moved to this new position. The scale of this random search is controlled by multiplying the generated Levy flight by a step size. For example setting could be beneficial in problems of small domains, in the examples presented here is used in line with the work by Yang and Deb [10]. Yang and Deb [9] do not discuss boundary handling in their formulation. When a Levy flight results in an egg location outside the bounds of the objective function, the fitness and position of the original egg are not changed. One of the advantages of CS over PSO is that only one parameter, the fraction of nests to abandon, needs to be adjusted. Yang and Deb [10] found that the convergence rate was not effected strongly by the value and they suggested setting . The use of Levy flights as the search method means that the Cuckoo Search can simultaneously find all optima in a design space and the method has been shown to perform well in comparison with PSO [10].

```
Cuckoo Search Algorithm
--------------------------------------------------------------------
begin
Objective function $f(x)$, $x = (x_1, \cdots, x_d)^T$;
Initial a population of n host nests $x_i$ $(i = 1, \cdots, n)$;
while ($t$ < Maximum Generation) or (stop criterion);
Get a cuckoo (say $i$) randomly
and generate a new solution by Levy flights;
Evaluate its quality/fitness; $F_i$
Choose a nest among n (say $j$) randomly;
if ($F_i > F_j$),
Replace $j$ by the new solution;
end
Abandon a fraction ($P_a$) of worse nests
[and build new ones at new locations via Levy flights];
Keep the best solutions (or nests with quality solutions);
Rank the solutions and find the current best;
end while
Post process results and visualization;
end
--------------------------------------------------------------------
```

Fig. 1. Pseudo code of the Cuckoo Search method

B. *Meta-heuristic Bat Search Algorithm*

Bat-inspired algorithm is a meta-heuristic search optimization developed by Xin-She Yang in 2010 [12]. This bat algorithm is based on the echolocation behavior of micro-bats with varying pulse rates of emission and loudness. The idealization of the echolocation of micro-bats can be summarized as follows: Each virtual bat flies randomly with a velocity $\vec{v}_i$ at position (solution) $\vec{x}_i$ with a varying frequency or wavelength and loudness A_i. As it searches and finds its prey, it changes frequency, loudness and pulse emission rate. Search is intensified by a local random walk. Selection of the best continues until certain stop criteria are met. A detailed introduction of meta-heuristic algorithms including the bat algorithm is given by Yang [14]. The basic idea behind the Bat Algorithm is that a population of n bats (possible solutions) of dimension d use echolocation to sense distance and fly randomly through a search space updating their positions $\vec{x}_i$ and velocities $\vec{v}_i$. Each solution

$\vec{x}_i = (x_1, \cdots, x_d)^T$ is evaluated by a fitness function $f(\vec{x}_i)$, $i = 1, \cdots, n$. The bats' flight aims at finding food/prey (best solutions). Two other parameters are: the loudness decay factor (α) that acts in a similar role as the cooling schedule in the traditional simulated annealing optimization method, and the pulse increase factor (γ) that regulates the pulse frequency. The properly update for the pulse rate (r_i) and the loudness (A_i) balances the exploitation and exploration behavior of each bat, respectively. As the loudness usually decrease once a bat has found its prey/solution (in order to do not loss the prey), the rate of pulse emission increases in order to raise the attack accuracy. The pseudo-code of the Bat Algorithm is shown in Fig. 2. In simulations, positions $\vec{x}_i$ and velocities $\vec{v}_i$ in a d-dimensional search space are updated. The new solutions $\vec{x}_i^{\,t}$ and velocities $\vec{v}_i^{\,t}$ at time step t are given by

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \cdot \beta \qquad (1)$$

$$\vec{v}_i^{\,t} = \vec{v}_i^{\,t-1} + (\vec{x}_i^{\,t} - \vec{x}_*) \cdot f_i \qquad (2)$$

$$\vec{x}_i^{\,t} = \vec{x}_i^{\,t-1} + \vec{v}_i^{\,t} \qquad (3)$$

where $\beta \in [0,1]$ is a random vector drawn from a uniform distribution. Here $\vec{x}_*$ is the current global best location (solution) which is located after comparing all the solutions among all the n bats. For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk

$$\vec{x}_{new} = \vec{x}_{old} + \varepsilon A^t \qquad (4)$$

where $\varepsilon \in [-1,1]$ is a random number, while A^t is the average loudness of all the bats at this time step. Furthermore, the loudness A_i and the rate r_i of pulse emission have to be updated accordingly as the iterations proceed. As the loudness usually decreases once a bat has found its prey, while the rate of pulse emission increases.

$$A_i^{t+1} = \alpha A_i^t \qquad (5)$$

$$r_i^{t+1} = r_i^0 \cdot (1 - e^{-\gamma t}) \qquad (6)$$

where α and γ are constants. In fact, α is similar to the cooling factor of a cooling schedule in the simulated annealing. For any $0 < \alpha < 1$ and $\gamma > 0$, we have $A_i^t \to 0$, $r_i^t \to r_i^0$, as $t \to \infty$. Their loudness and emission rates will be updated only if the new solutions are improved, which means that these bats are moving towards the optimal. solution.

Bat Algorithm
```
------------------------------------------------------------------------
Objective function $f(\vec{x})$, $\vec{x} = (x_1, \cdots, x_d)^T$;
Initialize the bat population $\vec{x}_i$ ($i = 1, \cdots, n$) and $\vec{v}_i$
Define pulse frequency $f_i$ at $\vec{x}_i$
Initialize pulse rates $r_i$ and the loudness $A_i$
Generate new solutions by adjusting frequency,
and updating velocities and locations/solutions [equations (1) to (3)]
  if (rand > $r_i$)
    Select a solution among the best solutions
    Generate a local solution around the selected best solution
  end if
  Generate a new solution by flying randomly
  if (rand < $A_i$ & $f(\vec{x}_i)$ < $f(\vec{x}_*)$)
    Accept the new solutions
    Increase $r_i$ and reduce $A_i$
  end if
Rank the bats and find the current best $\vec{x}_*$
end while
Post process results and visualization
------------------------------------------------------------------------
```
Fig. 2. Pseudo code of the bat algorithm

The update of the velocities and positions of bats have some similarity to the procedure in the standard particle swarm optimization [2] as f_i essentially controls the pace and range of the movement of the swarming particles. To a degree, Bat Algorithm can be considered as a balanced combination of the standard particle swarm optimization and the intensive local search controlled by the loudness and pulse rate.

C. Meta-heuristic Firefly Search Algorithm

The Firefly Algorithm (FA) is a meta-heuristic, nature-inspired, optimization algorithm which is based on the social (flashing) behavior of fireflies, or lighting bugs, in the summer sky in the tropical temperature regions [13][14]. It was developed by Dr. Xin-She Yang at Cambridge University in 2007, and it is based on the swarm behavior such as fish, insects, or bird schooling in nature. In particular, although the firefly algorithm has many similarities with other algorithms which are based on the so-called swarm intelligence, such as the famous Particle Swarm Optimization (PSO), Artificial Bee Colony optimization (ABC), and Bacterial Foraging (BFA) algorithms, it is indeed much simpler both in concept and implementation [13][14]. Furthermore, according to recent bibliography, the algorithm is very efficient and can outperform other conventional algorithms, such as genetic algorithms, for solving many optimization problems; a fact that has been justified in a recent research, where the statistical performance of the firefly algorithm was measured against other well-known optimization algorithms using various standard stochastic test functions [13][14]. Its main advantage is the fact that it uses mainly real random numbers, and it is based on the global communication among the swarming particles (i.e., the fireflies). The firefly algorithm has three particular idealized rules which are based on some of the major flashing characteristics of real fireflies [13][14]. These are the following: (1) all fireflies are unisex, and they will move towards more attractive and brighter ones regardless their sex. (2) The degree of attractiveness of a firefly is proportional to its brightness which decreases as the distance from the other firefly increases due to the fact that the air absorbs light. If there is not a brighter or more attractive firefly than a particular one, it will then move randomly. (3) The brightness or light intensity of a firefly is determined by the value of the objective function of a given problem. For maximization problems, the light intensity is proportional to the value of the objective function. In the firefly algorithm, the form of attractiveness function of a firefly is the following monotonically decreasing function [13][14]:

$$\beta(r) = \beta_0 \exp(-\gamma r^m), \text{ with } m \geq 1 \qquad (7)$$

where, r is the distance between any two fireflies, β_0 is the initial attractiveness at $r = 0$, and γ is an absorption coefficient which controls the decrease of the light intensity. The distance between any two fireflies i and j, at positions x_i and x_j, respectively, can be defined as a Cartesian or Euclidean distance as follows [13][14]:

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{8}$$

where $x_{i,k}$ is the *k*th component of the spatial coordinate x_i of the *i*th firefly and *d* is the number of dimensions we have. However, the calculation of distance *r* can also be defined using other distance metrics, based on the nature of the problem, such as Manhattan distance or Mahalanobis distance. The movement of a firefly *i* which is attracted by a more attractive (i.e., brighter) firefly *j* is given by the following equation [13][14]:

$$x_i = x_i + \beta_0 \exp(-\gamma r_{ij}^2) \cdot (x_j - x_i) + \alpha \cdot \varepsilon_i \tag{9}$$

where the first term is the current position of a firefly, the second term is used for considering a firefly's attractiveness to light intensity seen by adjacent fireflies, and the third term is used for the random movement of a firefly in case there are not any brighter ones. The coefficient α is a randomization parameter determined by the problem of interest, while ε_i is a vector of random numbers drawn from a Gaussian distribution or uniform distribution. As we will see in this implementation of the algorithm, we will use $\beta_0 = 1.0$, $\alpha \in [0,1]$, and the attractiveness or absorption coefficient $\gamma = 1.0$, which guarantees a quick convergence of the algorithm to the optimal solution. The convergence of the algorithm is achieved for any large number of fireflies (*n*) if *n* >> *m*, where *m* is the number of local optima of an optimization problem [11]. In this case, the initial location of *n* fireflies is distributed uniformly in the entire search space. The convergence of the algorithm into all the local and global optima is achieved, as the iterations of the algorithm continue, by comparing the best solutions of each iteration with these optima. However, it is under research a formal proof of the convergence of the algorithm and particularly that the algorithm will approach global optima when $n \to \infty$ and $t >> 1$ [11]. In practice, the algorithm converges very quickly in less than 80 iterations and less than 50 fireflies, as it is demonstrated in several research papers using some standard test functions [11]. Indeed, the appropriate choice of the number of iterations together with the γ, β, α and n parameters highly depends on the nature of the given optimization problem as this affects the convergence of the algorithm and the efficient find of both local and global optima. Note that the firefly algorithm has computational complexity of $O(n^2)$, where *n* is the population of fireflies. The larger population size becomes the greater the computational time is [11]. There are two important special cases of the firefly algorithm based on the absorption coefficient γ; that is, when $\gamma \to 0$ and $\gamma \to \infty$ [1]. When $\gamma \to 0$, the attractiveness coefficient is constant β_0, and the light intensity does not decrease as the distance *r* between two fireflies increases. Therefore, as the light of a firefly can be seen anywhere, a single local or global optimum can be easily reached. This limiting case corresponds to the standard Particle Swarm Optimization (PSO) algorithm. On the other hand, when $\gamma \to \infty$, the attractiveness coefficient is the Dirac delta function $\beta(r) \to \delta(r)$. In this limiting case, the attractiveness to light intensity is almost zero, and as a result, the fireflies cannot see each other, and they move completely randomly in a foggy place. Therefore, this method corresponds to a random search method. In a recent bibliography, a new meta-heuristic algorithm has been developed and formulated based on the concept of hybridizing the firefly algorithm. In particular, the new Levy flight firefly algorithm was developed by Dr. Xin-She Yang at Cambridge University in 2010 and it combines the firefly algorithm with the Levy flights as an efficient search strategy [15]. It combines the three idealized rules of the firefly algorithm together with the characteristics of Levy flights which simulate the flight behavior of many animals and insects. In this algorithm, the form of the attractiveness function and the calculation of distance between two fireflies are the same as in firefly algorithm, but in the movement function, the random step length is a combination of the randomization parameter together with a Levy flight. In particular, the movement of a firefly is a random walk, where the step length is drawn by the Levy distribution [15].

III. Synergistic Optimization Learning for Intensification and Diversification Balance

In this section, we propose some synergistic strategies for meta-heuristic optimization learning algorithms. The proposed methods include effects of chaotic dynamics, Levy flights and a psychology model of emotion. Two important characteristics of meta-heuristics are: intensification and diversification. Intensification (exploitation) intends to search locally and more intensively, while diversification (exploration) makes sure the algorithm explores the search space globally (hopefully also efficiently). Too little exploration and too much exploitation could cause the system to be trapped in local optima, which makes it very difficult or even impossible to find the global optimum. On the other hand, if too much exploration but too little exploitation, it may be difficult for the system to converge and thus slows down the overall search performance. One of the main tasks of designing new algorithms is to find a certain balance concerning this optimality and/or tradeoff.

A. Chaotic Levy Flight Bat Search Algorithm

One of the major concerns in using evolutionary algorithms to search complex state spaces is the problem of premature convergence. While premature convergence may be defined as the phenomenon of convergence to non-optimal solutions, move-generation (new solutions) means loss of diversity in the process of evolution. Maintaining a certain degree of diversity is widely believed to help avoid entrapment in non-optimal solutions.

Chaotic Levy Flights for Move-Generation

The notion of ergodicity asserts that a system having a number of possible states will, over a finite time, visit each one with equal frequency [1]. The track of chaotic variable can travel ergodically over the whole search space. In general, the chaotic variable has special characters, i.e., ergodicity, pseudo-randomness and irregularity. Furthermore, the Levy flight process, which has previously been used in search algorithms [6], is a random walk that is characterized by a series of instantaneous jumps chosen from a probability density function which has a power law tail. This process represents the optimum random search pattern and is frequently found in nature [7]. Broadly speaking, Levy flight is a random walk whose step length is drawn from the Levy distribution. Here we propose the chaotic Levy flight for the improved bat algorithm. In this paper, the well-known logistic map which exhibits the sensitive dependence on initial conditions is employed to generate the chaotic sequence c_s for the parameter in Levy flight:

$$c_s(t+1) = 4.0 \times c_s(t) \times (1 - c_s(t)), 0 \le c_s(0) \le 1 \quad (10)$$

On the other hand, due to infinite variance of Levy distribution, it permits occasionally long steps (the so called Levy flights) far from the neighborhood of the previous sample. Setting a smaller neighborhood range, and making small jumps, is helpful for finding optimum solutions in the region. However, large jumps are needed to avoid locals. Usually, there is no perspective to specify the above-mentioned regions. There is no specific definition of the mean and the variance of chaotic sequences. Thus, a combination of a chaotic sequence and Levy random process may result in better answers. In the proposed synergistic search algorithms, a new method to search a neighborhood is introduced. The new neighbor generation method is shown in the following equation:

$$x_{new} = x_{old} + c_s \otimes Levy(\lambda) \quad (11)$$

Levy distribution and a chaotic sequence are used to generate $Levy(\lambda)$ and c_s respectively. The product $\otimes$ means entry-wise multiplications. Levy flights essentially provide a random walk, while their random steps are drawn from a Levy distribution for large steps

$$Levy \sim u = t^{-\lambda}, (1 \le \lambda \le 3) \quad (12)$$

which has an infinite variance with an infinite mean. Because the chaotic sequence can generate several neighborhoods of suboptimal solutions to maintain the variability in the solutions, it can prevent the search process from becoming premature. Due to its ergodicity, chaotic sequence can generate several neighborhoods of near-optimal solutions. The algorithm probably converges to a space in the search space where good solutions are denser. The new solutions $\vec{x}_i^{\,t}$ and velocities $\vec{v}_i^{\,t}$ at time step t are then given by

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \cdot c_s \quad (13)$$

$$\vec{v}_i^{\,t} = \vec{v}_i^{\,t-1} + (\vec{x}_i^{\,t} - \vec{x}_g^{\,t}) \cdot f_i + (\vec{x}_i^{\,t} - \vec{x}_l^{\,t}) \cdot f_i \quad (14)$$

$$\vec{x}_i^{\,t} = \vec{x}_i^{\,t-1} + \vec{v}_i^{\,t} \quad (15)$$

where $\vec{x}_g^{\,t}$ is the global best of all the bats and $\vec{x}_l^{\,t}$ is the local best of each bat. Any single bat is following the best hunting position found by not only taking all bats into consideration, but also its own preference when searching for food. The reason for this added parameter in the velocity equation is because by choosing its own local hunting area the exploitation of the algorithm will be increased. The pseudo-code of the proposed Chaotic Levy Flight Bat Algorithm is shown in Fig. 3.

```
Chaotic Levy Flight Bat Algorithm
--------------------------------------------------------------------
Objective function $f(\vec{x})$, $\vec{x} = (x_1, \cdots, x_d)^T$;
Initialize the bat population $\vec{x}_i$ $(i = 1, \cdots, n)$ and $\vec{v}_i$
Define pulse frequency $f_i$ at $\vec{x}_i$
Initialize pulse rates $r_i$ and the loudness $A_i$
Generate the chaotic sequence $c_s$
Compute $f(\vec{x}_i)$
Find the current best $\vec{x}_*$
Generate new solutions by adjusting frequency,
and updating velocities and locations/solutions [equations (13) to (15)]
  if ($c_s > r_i$)
    Select a solution among the best solutions
    Generate a local solution around the selected best solution
    by flying randomly via chaotic Levy flights using equation (11)
  end if
  Generate a new solution by flying randomly via chaotic
  Levy flights using equation (11)
  if ($c_s < A_i$ & $f(\vec{x}_i) < f(\vec{x}_*)$)
    Accept the new solutions
    Increase $r_i$ and reduce $A_i$
  end if
Rank the bats and find the current best $\vec{x}_*$
end while
Post process results and visualization
--------------------------------------------------------------------
```

Fig. 3. Pseudo code of the proposed chaotic Levy flight bat algorithm

B. *Chaotic Levy Flight Firefly Search Algorithm*

Generally, the parameter in equation (9) is the key factor to affect the convergence of firefly algorithm. In fact, however, it cannot ensure the optimization's ergodicity entirely in phase space because they are absolutely random in the firefly algorithm. Here we propose the chaotic Levy flight for the improved firefly algorithm.

Chaotic Levy Flights for Move-Generation of Firefly Algorithm

In the proposed algorithm, a new method to search a neighborhood is introduced. The new neighbor generation method is shown in the following equation:

$$x_i = x_i + \beta_0 \exp(-c_s r_{ij}^2) \cdot (x_j - x_i) + c_s \otimes Levy(\lambda) \quad (16)$$

Levy distribution and a chaotic sequence are used to generate $Levy(\lambda)$ and c_s respectively. The product $\otimes$ means entry-wise multiplications. Levy flights essentially provide a random walk, while their random steps are drawn from a Levy distribution for large steps

$$Levy \sim u = t^{-\lambda}, (1 \le \lambda \le 3) \quad (17)$$

which has an infinite variance with an infinite mean. Because the chaotic sequence can generate several neighborhoods of suboptimal solutions to maintain the variability in the solutions, it can prevent the search process from becoming premature [3]. Due to its ergodicity, chaotic sequence can generate several neighborhoods of near-optimal solutions. The algorithm probably converges to a space in the search space where good solutions are denser. The proposed chaotic Levy flight firefly algorithm is shown in Fig. 4.

```
Chaotic Levy Flight Firefly Algorithm
--------------------------------------------------------------------
begin
Objective function $f(x)$, $x=(x_1,\cdots,x_d)^T$;
Generate initial population of fireflies $x_i$ $(i=1,\cdots,n)$
Light intensity $I_i$ at $x_i$ is determined by $f(x_i)$
Defined light absorption coefficient $\gamma$
while ($t$ < Maximum Generation)
  for $i=1:n$ all $n$ fireflies
    for $j=1:i$ all $n$ fireflies
      if ($I_j > I_i$),
        Move firefly $i$ towards $j$ in $d$-dimension via
        chaotic Levy flights using equation (5)
      end if
      Attractiveness varies with distance r via $\exp(-\gamma r^2)$
      Evaluate new solutions and update light intensity
    end for $j$
  end for $i$
  Rank the fireflies and find the current best
end while
Post process results and visualization
end
--------------------------------------------------------------------
```

Fig. 4. Pseudo code of the proposed chaotic Levy flight firefly algorithm

C. Emotional Chaotic Cuckoo Search Algorithm

Similarly, the parameter α in Levy flight is the key factor to affect the convergence of cuckoo search algorithm. It cannot ensure the optimization's ergodicity entirely in phase space because they are absolutely random in the cuckoo search algorithm. Here we propose the chaotic Levy flight for the move-generation of the cuckoo search algorithm. In addition, psychology model of emotion and chaotic sequence are also incorporated in the move-acceptance of the cuckoo search algorithm.

Psychology Model of Emotion and Chaotic Sequence for Move-Acceptance of Cuckoo Search Algorithm

Among most machine learning algorithms for optimization problem including meta-heuristic search algorithms, the solution is drawn like a moth to a flame and cannot keep away. To enrich the searching behavior and to avoid being trapped into local optimum, psychology model of emotion and chaotic sequence is proposed for move acceptance decision in cuckoo search algorithm. In psychology, emotion is considered a response to stimulus that involves characteristic physiological changes—such as increase in pulse rate, rise in body temperature, etc. Weber, the first psychologist who quantitatively studies the human response to a physical stimulus, found that the response was proportional to a relative increase in the weight. By Weber-Fechner Law, the relationship between stimulus and perception is logarithmic. This logarithmic relationship means that if the perception is altered in an arithmetic progression the corresponding stimulus varies as a geometric progression. That is to say, if the weight is 1 kg, an increase of a few grams will not be noticed. Rather, when the mass in increased by a certain factor, an increase in weight is perceived. If the mass is doubled, the threshold is also doubled. This kind of relationship can be described by a differential equation as:

$$dp = k\frac{dS}{S} \tag{18}$$

where dp is the differential change in perception, dS is the differential increase in the stimulus and S is the stimulus at the instant. A constant factor k is to be determined experimentally. Integrating the above equation.

$$p = k\ln S + C \tag{19}$$

with C is the constant of integration, ln is the natural logarithm. To determine C, put $p=0$, i.e. no perception; then

$$C = -k\ln S_0 \tag{20}$$

where S_0 is that threshold of stimulus below which it is not perceived at all, and can be called Absolute Stimulus Threshold (AST). Therefore, the equation becomes:

$$p = -k\ln\frac{S}{S_0} \tag{21}$$

For the proposed emotional chaotic cuckoo search algorithm, we define only two emotions cuckoos could have, positive and negative, and correspond to two reactions to perception respectively as follow:

IF ($c_s < e_s$) THEN positive
ELSE negative

where c_s is the chaotic sequence number. The emotion of cuckoos can determine by $e_s = p$. Here are two only two emotions cuckoo could have, positive and negative, and correspond to two reactions to perception respectively. The perception of cuckoo can be described by following:

$$e_s = -k\ln\left|\frac{S(F(x_i)-F(x_j))}{S_0}\right| \tag{22}$$

Here S_0 means stimulus threshold, S means stimulus function. $F(\bullet)$ is a fitness function. A candidate move is generated by chaotic Levy flight, and the system must decide whether to "accept" that move, based upon the chaotic sequence number and emotion factor. This process of move generation (by chaotic Levy flight) and move acceptance is repeated. This mechanism enables a system to transcend fitness barriers, thereby moving from one valley in the fitness landscape to another. The decision to accept new solutions is based on the acceptance criterion. We apply the psychology factor of emotion, which is given by (8):

$$P\{accept\} = \min(1, e_s) \tag{23}$$

This criterion produce real numbers in $[0,1)$ interval as the acceptance probability, which are used in the algorithm in the decision making process. Random numbers are replaced by chaotic sequence. The proposed algorithm compares the value of $P\{accept\}$ with a value from a chaotic sequence. The proposed emotional chaotic cuckoo search algorithm is shown in Fig. 5. The chaotic sequence used in this part produces not just a gradual divergence of the sequences of values, but also an exponential divergence, bringing the complexity and unpredictability features of chaotic theory into the proposed algorithm. Hence, the probability of evading local minima increases dramatically. The distinctive characteristics of the emotional chaotic cuckoo search algorithm are listed below to recapitulate the main proposal:

- Improved quality of the neighborhood search, and neighbor selection, using a chaotic sequence and a Levy random number generator.
- Increased probability of escaping from local minima by using the new acceptance criterion and the new method of space search.

Emotional Chaotic Cuckoo Search Algorithm

```
-------------------------------------------------------------------
begin
Objective function $f(x)$, $x=(x_1,\cdots,x_d)^T$;
Initial a population of n host nests $x_i$ $(i=1,\cdots,n)$;
while ($t$ < Maximum Generation) or (stop criterion);
Get a cuckoo (say $i$) randomly
and generate a new solution by Chaotic Levy flights;
Evaluate its quality/fitness; $F_i$
Choose a nest among n (say $j$) randomly;
if ($F_i > F_j$),
Replace $j$ by the new solution;
else if ($c_s < e_s$),
Replace $j$ by the new solution;
end
Abandon a fraction ($P_a$) of worse nests
[and build new ones at new locations via Chaotic Levy flights];
Keep the best solutions (or nests with quality solutions);
Rank the solutions and find the current best;
end while
Post process results and visualization;
end
-------------------------------------------------------------------
```

Fig. 5. Pseudo code of the proposed Emotional Chaotic Cuckoo Search Algorithm

IV. Simulation and Applications

For applications, we [4] applied the synergistic meta-heuristic optimization algorithms for the problem of parameter estimation (model calibration) in nonlinear dynamic models of biological systems. We applied the proposed optimization search algorithm and described a general methodology to adaptively select the values of the model parameters for the reconstruction of biological system dynamics. We illustrate the application of the method by jointly estimating the parameter vector of the dynamics of endocytosis. We also applied these optimization algorithms for the unsupervised robotic learning such as the maze exploration problem [5]. The proposed algorithms with quite general objective function are used to study the ability to develop unsupervised robotic learning. In this section, for simulation, we illustrate the application of the method for the reconstruction of chaotic dynamics. Simulation results are provided to illustrate the effectiveness and feasibility of the proposed algorithm by jointly estimating the complete parameter vector of a Lorenz system.

The rich nonlinear dynamics of chaos allows to model a broad variety of relevant systems in different fields of science and engineering, including complex biological structures and processes. The system of interest is usually observed through some time series and the modeling problem consists of adjusting the parameters of a model chaotic system until its dynamics is matched to the reference time series. It is often convenient to assume that the observed time series are originated by a primary chaotic system, with unknown parameters, that drives a coupled secondary chaotic system. If the coupling is suitable to lead the secondary system into the same behavior as the primary one when they share the same parameter values, the modeling problem reduces to the optimization of the secondary parameters in order to attain synchronization. Moreover, we can re-state the problem of parameter optimization as one of parameter identification, since the secondary parameters become estimates of the primary ones. In this section, we propose a new method for parameter identification in coupled chaotic systems that can be applied to a wide range of problems. Let us refer to the device generating the observed time series as the primary system, while the secondary system is the model with parameters to be adjusted. Both systems are coupled in a way that ensures synchronization when they are identical, i.e., when the parameters in the model exactly coincide with the primary system parameters. However, since the latter are unknown, the time evolution of the secondary system state variables is, in principle, different from that of the primary system. Our aim is to derive an algorithm that adaptively adjusts the model parameters until its response is matched to the observed time series. When this is achieved (according to an adequate criterion), the secondary system is synchronized and its parameters are estimates of the true parameters in the primary system. To be specific, let

$$\dot{x} = f(x, p) \qquad (24)$$

represent the primary system with state variables $x \in R^n$, and unknown parameters $p \in R^m$. Notice that $\dot{x}$ represents the temporal derivative of x. For conciseness, we usually leave the time dependence of dynamic variables implicit. If the functional form of equation (24) is known, we can build the secondary system as

$$\dot{y} = f(y, q) \qquad (25)$$

where $y \in R^n$ are the state variables and $q \in R^m$ are the parameters. We assume that there exists a unidirectional coupling scheme using any signal output of the primary system (equation (24)) which enables asymptotic synchronization of the secondary system (equation (25)), i.e., $y \to x$ as $t \to \infty$, as long as $q = p$. Notice that techniques are now available which enable the design of an unidirectional scheme which guarantees synchronization. The system in equation (25) is fully observed and we assume the ability periodically change the value of the parameter vector Finally, let $h(x): R^n \to R^k$ be the time series we obser from the primary system, which consists of a subset of t dynamic variables in equation (24). It is assumed that $h($ contains, at least, the signals needed for adequate coupling the systems. Our aim is to devise an algorithm to adaptive adjust the parameters in the secondary system, q, until t system variables, y, and the parameters themselves conver to their counterparts in the primary system, i.e., both $y \to$ and $q \to p$. In this way, synchronization between bc systems is achieved and the parameters of the primary syste are identified. Let $x = [x_1, \cdots, x_N]^T \in R^N$ be the state vector of the chaotic system, $\dot{x}$ is the derivative of the state vector x. Based on the measurable state vector $x = [x_1, \cdots, x_N]^T$, for individual i, we define the following fitness function

$$F_i = \sum_{t=0}^{k} ((x_1(t) - x_{i1}(t))^2 + \cdots + (x_N(t) - x_{iN}(t))^2) \qquad (2$$

where $t = 0, \cdots, k$. Therefore, the problem of paramet identification is transformed to that of using the propos emotional chaotic cuckoo search algorithm to search for t suitable value of the parameter q such that the fitne function is globally minimized.

In order to evaluate the proposed parameter identificati strategy, the Lorenz system are employed in the simulati study. Let us consider a primary system that obeys the Lore equations

$$\begin{aligned} \dot{x}_1 &= -\sigma_1(x_1 - x_2) \\ \dot{x}_2 &= r_1 x_1 - x_2 - x_1 x_3 \\ \dot{x}_3 &= -b_1 x_3 + x_1 x_2 \end{aligned} \qquad (27)$$

where $x = [x_1, x_2, x_3]^T \in R^3$ is the dynamic system state and $p = [\sigma_1, r_1, b_1]^T \in R^3$ contains the parameters. The system in the chaotic state when $\sigma_1 = 10$, $r_1 = 28$, $b_1 = 8/3$. In orc to estimate, let the parameters of the Lorenz system $\sigma_1 = 10$, $r_1 = 28$, $b_1 = 8/3$. The secondary systems is

$$\begin{aligned} \dot{y}_1 &= -\sigma_2(y_1 - y_2) \\ y_2 &= r_2 y_1 - y_2 - y_1 y_3 \\ \dot{y}_3 &= -b_2 y_3 + y_1 y_2 \end{aligned} \qquad (2$$

where $y = [y_1, y_2, y_3]^T \in R^3$ are the state variables a $q = [\sigma_2, r_2, b_2]^T \in R^3$ are the parameters that can be adjust in order to estimate p and attain synchronization. In order observe nonlinear dynamical searching process of tue emotional chaotic cuckoo search algorithm as a whole, we plot search values of all the individuals for parameters σ_1, r_1, b_1 in Fig. 6-8. From Fig. 6-8, we can see that the trajectories of the identification of the parameters converge at the real values of the parameters, indicating that the model of our proposed emotional chaotic cuckoo search algorithm can be used as an effective optimization model. Fig. 9 shows the identification of the parameters σ_1, r_1, b_1 in Lorenz system for the best fitness evolution. Fig. 10 shows the evolution of fitness function. Fig. 11 shows the reconstruction of Lorenz system. Simulation results show that the proposed method can provide greater efficiency and satisfactory accuracy.

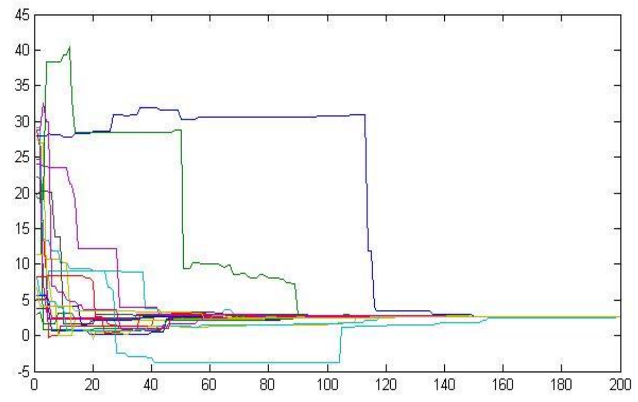
Fig. 6. Identification of the parameter σ_1 in Lorenz system

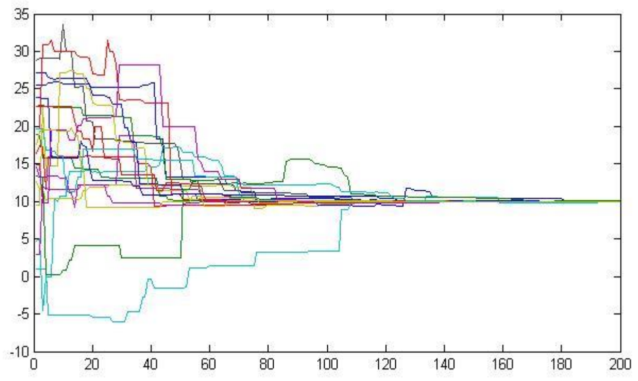
Fig. 7. Identification of the parameter r_1 in Lorenz system

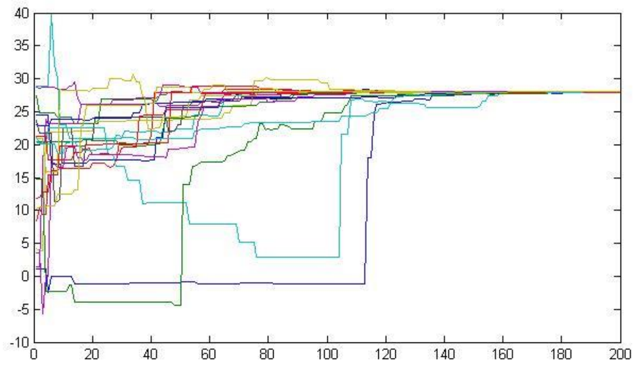
Fig. 8. Identification of the parameter b_1 in Lorenz system

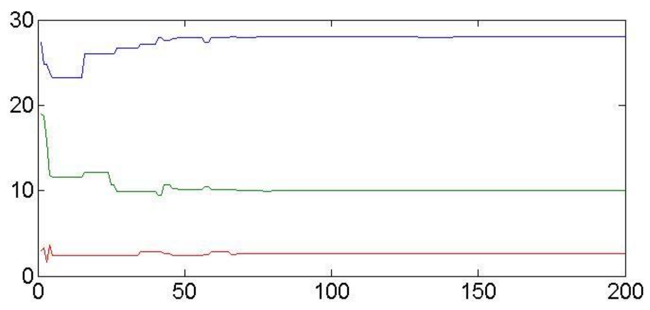
Fig. 9. Identification of the parameters σ_1, r_1, b_1 in Lorenz system for the best fitness evolution

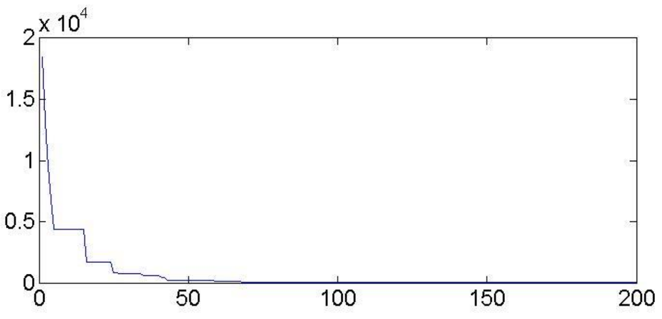
Fig. 10. Fitness function evolution

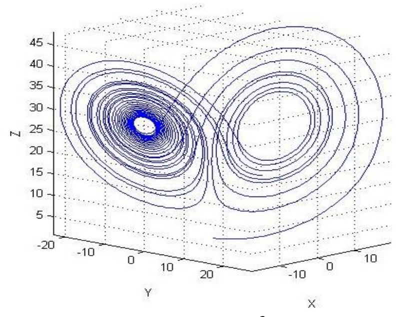
Fig. 11. Reconstruction of Lorenz System

V. Conclusion

In this paper we have proposed some synergistic meta-heuristic optimization algorithms. The proposed methods include effects of chaotic dynamics, Levy flights and a psychology model of emotion. Due to the easy implementation and special ability to avoid being trapped in local optima, chaos has been a good optimization technique and chaos-based searching algorithms have aroused intense interests. Simulation results of Lorenz system are provided to illustrate the effectiveness and feasibility of the proposed algorithm. Furthermore, one may hope that the further elaboration of optimization theory and algorithms based on computational intelligence for complex systems by using simple rules for complex intelligent behaviors will contribute to an interdisciplinary field of research.

References

[1] Buchanan, Mark, "Capturing Chaos", *Nature Publishing Group*, vol.435, 2005.

[2] Kennedy, J., Eberhart, R., "Particle swarm optimization", In: *Proc. IEEE Int. Conf. Neural Networks*, Perth, Australia, 1995, pp.1942–1945.

[3] Liao, Gwo-Ching and Tsao, Ta-Peng, "Application of a Fuzzy Neural Network Combined With a Chaos Genetic Algorithm and Simulated Annealing to Short-Term Load Forecasting", *IEEE Transaction On Evolutionary Computation*, vol. 10, no. 3, June 2006.

[4] Lin, Jiann-Horng, Chou, Chao-Wei, Yang, Chorng-Horng, Tsai, Hsien-Leing and Lee, I-Ho, "A Bio-inspired Optimization Algorithm for Modeling the Dynamics of Biological Systems", in *Proceedings of the 3rd International Conference on Innovations in Bio-Inspired Computing and Applications*, 2012.

[5] Lin, Jiann-Horng and Li, Yu-Lin, "A Metaheuristic Optimization Algorithm for Unsupervised Robotic Learning", *IEEE International Conference on Computational Intelligence and Cybernetics*, Bali, INDONESIA, 2012.

[6] Pavlyukevich, I., "Lévy Flights, Non-local Search and Simulated Annealing. *Journal of Computational Physics*,226,2007, pp.1830–1844.

[7] Viswanathan, GM., "Lévy flights and superdiffusion in the context of biological encounters and random searches", *Physics of Life Reviews*, 5, 2008, pp.133–150.

[8] Yang, X.-S. and Deb, S.,"Cuckoo search via Lévy flights". *World Congress on Nature & Biologically Inspired Computing (NaBIC 2009)*. IEEE Publications, 2009, pp. 210–214.

[9] Yang X.-S. and Deb, S., "Cuckoo search via Lévy flights", *Proceedings of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009, India)*, IEEE Publications, USA, 2009, pp. 210–214.

[10] Yang X.-S. and Deb, S., "Engineering optimisation by cuckoo search", *International Journal of Mathematical Modelling and Numerical Optimisation, 2010*;1:330–43.

[11] Yang, X.-S., "Firefly algorithms formultimodal optimization," in *Proceedings of the Stochastic Algorithms: Foundations and Applications (SAGA '09)*, vol. 5792 of Lecture Notes in Computing Sciences, Springer, Sapporo, Japan, 2009, pp. 178–178.

[12] Yang, X.-S., "A New Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization", (NISCO 2010) (Eds. J. R. Gonzalez et al.), *Studies in Computational Intelligence*, Springer Berlin, 284, Springer, 2010, pp.65-74.

[13] Yang, X.-S., "Firefly algorithm, stochastic test functions and design optimisation". *Int. J. Bio-inspired Computation* **2** (2), 2010, pp.78–84.

[14] Yang, X.-S., *Nature-Inspired Metaheuristic Algoirthms*, 2nd Edition, Luniver Press, 2010.

[15] Yang, X.-S. , "Firefly algorithm, Levy flights and global optimization," in *Research and Development in Intelligent Systems XXVI*, pp. 209–218, Springer, London, UK, 2010.

Author

Jiann-Horng Lin received his B. S. and M. S. both in Computer Science and Information Engineering from Feng-Chia University, Taiwan in 1987 and 1989, respectively. He then received his Ph.D. in Electrical Engineering and Computer Science from Syracuse University, New York in 1999. He is currently an assistant professor at the Department of Information Management at I-Shou University, Taiwan. He is also the department chair from 2004 to 2007. His research interests include artificial intelligence, data mining, chaos and information theory. He is also interested in the area of evolutionary computation and bioinformatics. Dr. Lin is a member of the IEEE, the Association for Computing Machinery and the Institute of Information and Computing Machinery.