Cogn Comput (2013) 5:188–199
DOI 10.1007/s12559-012-9189-5

Mussels Wandering Optimization: An Ecologically Inspired Algorithm for Global Optimization

Jing An · Qi Kang · Lei Wang · Qidi Wu

Received: 28 April 2012 / Accepted: 2 September 2012 / Published online: 26 September 2012

Abstract Over the last decade, we have encountered various complex optimization problems in the engineering and research domains. Some of them are so hard that we had to turn to heuristic algorithms to obtain approximate optimal solutions. In this paper, we present a novel metaheuristic algorithm called mussels wandering optimization (MWO). MWO is inspired by mussels' leisurely locomotion behavior when they form bed patterns in their habitat. It is an ecologically inspired optimization algorithm that mathematically formulates a landscape-level evolutionary mechanism of the distribution pattern of mussels through a stochastic decision and Lévy walk. We obtain the optimal shape parameter μ of the movement strategy and demonstrate its convergence performance via eight benchmark functions. The MWO algorithm has competitive performance compared with four existing metaheuristics, providing a new approach for solving complex optimization problems.

Keywords Optimization · Ecologically inspired algorithm · Mussel wandering · Lévy walk

Introduction

One of the most fundamental problems worldwide is the search for an optimal state. Optimization is about finding the best possible solutions for given problems [1]. Formally, the goal of optimization is to find the best possible elements x^* from a feasible set X according to a set of criteria $F = \{f_1, f_2, \ldots, f_n\}$, the so-called objective functions. For a single-objective minimization problem, the goal is to find the minimum solution $x^* \in X$ where $f(x^*) \leq f(x)$ for all x.

Over the past few decades, many optimization algorithms have been proposed to solve complex optimization problems. These algorithms are usually divided into deterministic algorithms, probabilistic algorithms, and metaheuristics [2]. Deterministic algorithms are mainly used if a clear relation between the characteristics of the possible solutions and their utility for a given problem exists, i.e., branch and bound, and mathematical programming methods. However, it is always hard to solve a problem analytically in most cases, because the relation between a solution candidate and its "fitness" is always not obvious or too complicated, or the dimensionality of the search space is very high for many real-world problems. Probabilistic algorithms model a problem or search a problem space using a probabilistic model of candidate solutions [3]. Monte Carlo-based approaches belong to the family of probabilistic algorithms, which trade off guaranteed correctness of the solution for shorter runtime. Metaheuristics are designed to tackle complex optimization problems where other optimization methods have failed to be either effective or efficient. These methods have come to be recognized as some of the most practical approaches for solving many complex problems, particularly for the many real-world problems that are combinatorial in nature.

In particular, nature-inspired metaheuristics have attracted increasing attention, becoming powerful methods to solve modern global optimization problems [4]. Among these, the most successful are evolutionary algorithms

J. An · Q. Kang (✉) · L. Wang · Q. Wu
Department of Control Science and Engineering, Tongji University, Shanghai 201804, China
e-mail: qkang@tongji.edu.cn

J. An
School of Electrical and Electronic Engineering, Shanghai Institute of Technology, Shanghai 201418, China

(EAs), which draw their inspiration from evolution by natural selection [5]. There are several different types of EAs, including genetic algorithms [6], genetic programming [7], evolutionary programming [8], the estimation of distribution algorithm [9], and differential evolution [10]. Later, swarm intelligence offered more effective approaches for complex optimization problems, being inspired from collective animal behavior [11], i.e., particle swarm optimization [12], ant colony optimization [13], artificial bee colony [14], group search optimizer [15], and bacterial foraging optimization [16]. Such successful algorithms also include simulated annealing [17], artificial neural network [18], artificial immune algorithm [19], chemical reaction optimization [20], memetic algorithm [21], and harmony algorithm [22]. More recently, a number of ecological optimization algorithms have been proposed and successfully applied, i.e., biogeography-based optimization [23], cuckoo search [24], shuffled frog-leaping algorithm [25], and the self-organizing migrating algorithm [26].

All these metaheuristics have unique computational models, mimicking specific behaviors for the performance of diversity. However, most of them satisfy the unifying principle of being population based with randomization being involved in the solution process. Notably, these forms of population behavior patterns exhibit both deterministic and stochastic characteristics in order to guarantee some inheritance as well as to promote healthy evolution. This stochastic feature plays a significant role in such algorithms [27]. However, many existing algorithms just borrow some random parameters to maintain diversity and promote its evolution. More comprehensive stochastic behavior-oriented optimizers are seldom reported. It will be exciting and challenging to deeply understand the integral stochastic behaviors or mechanisms of an ecological system in order to develop new, efficient stochastic optimization models for complex problems.

Recently, Daniel [28] reported that mussels in intertidal beds use random movement behaviors to optimize their formation of patches to balance feeding and mortality. This species of mussel, despite exhibiting leisurely locomotion, exhibits economy of movement and savvy behavioral strategies that approach a theoretical ideal as well as, or better than, more visibly athletic species. Inspired by this important discovery, this paper formulates a metaheuristic named mussels wandering optimization (MWO) for the first time, using the mussels' stochastic walking strategy. In particular, mussels use a Lévy walk during the formation of a spatially patterned bed. Mussels move according to others' behaviors and environmental complexity. The interaction between individuals and the complexity of the habitat shapes the mussels' movement in ecological systems.

In the rest of the paper, we first introduce the bed formatting behavior of natural mussels in section "Bed Formatting Behavior of Mussels for Optimization." Then, the mussels wandering optimization (MWO) algorithm is formulated in section "Mussels Wandering Optimization." We demonstrate its performance via a set of standard benchmarks and compare it with some existing metaheuristic algorithms in section "Simulation and Results." The paper is concluded in section "Conclusions."

Bed Formatting Behavior of Mussels for Optimization

Nature is full of biological landscapes that seem static to the casual observer but that actually contain highly dynamic spatial features. These features are shaped dramatically, if slowly, by subtle interplays between large-scale population-level patterns and small-scale movements of individuals [28]. Mussel is a species of mollusk, unusually thriving on rocky-shore and soft-bottom habitats, and in the intake mouths of coastal power stations [29]. Mussels depend on not only physical processes, such as water flow rate, temperature, salinity, and desiccation, but also biological processes, such as amount of food resources for survival, growth, and reproduction [30].

Mussels form extensive beds of high or varying density at complex soft or hard surfaces so as to provide themselves with habitat structure. Mussel bed patterns of distribution and abundance result from a combination of constraints that occur from regional processes at the macrohabitat scale [24] to local interactions at the microhabitat scale [25] over periods of various decades, in the range of their lifespan [30, 31]. When mussels move, they usually use a random strategy with respect to local mussel density, predators, and habitat quality parameters such as the amount of biomass and the water flow rate [32].

In particular, pattern formation in mussel beds depends on two opposing biological mechanisms: cooperation and competition. Mussels can aggregate not only to provide physical protection to the underlying bed, thus enabling the bed to withstand erosion by high tidal currents and wave action, but also to resist predators and high wave stress. Within a cluster, mussels move less when surrounded by conspecifics, possibly to minimize predation, wave stress or dislodgement losses. By moving into cooperative aggregations, mussels increase their local density. However, mussel behavior prevents the formation of such large clusters, and possibly mussels decide to move when the cluster size becomes too large, because of competition for algae. This interaction between local cooperation and long-range competition results in simultaneous risk reduction and minimization of competition for algae [32].

Experiments show that the probability of a mussel moving decreases with the short-range density and increases with long-range density [28]. When a mussel

decides to move, it adopts a Lévy walk during the formation of spatially patterned beds, and models reveal that this Lévy movement accelerates bed pattern formation. The emergent patterning in mussel beds, in turn, improves individual fitness. Moreover, much research has demonstrated that, in fish, insect, or bird foraging, and human travel, the step length is also a stochastic process which approaches a Lévy walk [33–35].

The step lengths for mussel are not constant but rather are chosen from a probability distribution with a power-law tail [34], which is characterized by rare but extremely long step lengths, and the same sites are revisited much less frequently than in a normal diffusion process, represented as [32],

$$f(l) = C_\mu l^{-\mu}, \tag{1}$$

where C_μ is a normalization constant. The shape parameter $1 < \mu < 3$ is known as the Lévy exponent or scaling exponent and determines the movement strategy. When μ is close to 1, the resulting movement strategy resembles ballistic, straight-line motion, as the probability to move a very large distance is equal to the chance of making a small displacement.

Behaviorally, power-law distributions require cognitive memory of the duration of the current move. It seems likely that organisms that perform Lévy walks have specific adaptations for doing so. Ecologically, Lévy walks are important because they yield a "scale-free" blend of long and short movements that can be more efficient in locating and exploiting resources that are patchy at multiple scales. Notably, in mussels, this efficiency extends beyond exploitation to the fundamental mechanisms underlying the formation of favorably patchy mussel beds.

Notably, this bed pattern formatting behavior with a random walk is just an optimization process to search for the most suitable place for each mussel in the habitat. Therefore, it is promising to design an optimization algorithm by modeling the ecological population behavior of mussels for complex global optimization problems.

Mussels Wandering Optimization

A population of mussels includes N individuals in a certain spatial region of marine "bed" called the habitat. The habitat is mapped to a d-dimensional space S^d of the problem to be optimized. The objective function value $f(s)$ at each point $s \in S^d$ represents the nutrition provided by the habitat. Each mussel has a position $x_i := (x_{i1}, \ldots, x_{id}), i \in \mathbf{N}_N = \{1, 2, \ldots, N\}$ in S^d, forming a specified spatial bed pattern, accordingly.

For simplicity in describing MWO, the following idealized rules are given: (1) Mussel size is negligible; (2) When mussels are wandering for food and avoiding negative circumstance factors, there is no breeding and mortality; and (3) Mussels can walk across any other's body.

The spatial distance D_{ij} between m_i and m_j in S^d can be calculated by

$$D_{ij} := \|x_i - x_j\| = \left[\sum_{k=1}^{d}(x_{ik} - x_{jk})^2\right]^{1/2}, \quad i, j \in \mathbf{N}_N. \tag{2}$$

In MWO, all mussels are uniformly located in the habitat at the beginning. At each iteration, a mussel moves or remains still with a certain probability in the search space. The moving probability of a mussel depends on its short-range density and long-range density [28].

For mussel m_i, given a short-range reference r_s (the radius of the inner circle in 2-D space shown in Fig. 2), the short-range density ξ_{si} is defined as the ratio of the number of mussels that are located away from m_i but within distance r_s to the population size N in unit distance, i.e.,

$$\xi_{si} := \sharp(D_i < r_s)/(r_s N), \tag{3}$$

where $\sharp(A < b)$ is used to compute the count in set A satisfying $a < b, a \in A$; D_i is the distance matrix from m_i to other mussels in the population.

Given the long-range reference r_l (the radius of the outer circle in Fig. 2), the long-range density ξ_{li} of mussel m_i is defined as

$$\xi_{li} := \sharp(D_i < r_l)/(r_l N). \tag{4}$$

The short-range reference r_s and long-range reference r_l change through the iterations. In iteration t, given the distance data, the dynamic short-range reference $r_s(t)$ and long-range reference $r_l(t)$ are defined and calculated by

$$\begin{cases} r_s(t) := \alpha \cdot \max_{i,j \in \mathbf{N}_N}\{D_{ij}(t)\}/\delta \\ r_l(t) := \beta \cdot \max_{i,j \in \mathbf{N}_N}\{D_{ij}(t)\}/\delta \end{cases} \tag{5}$$

where α and β are positive constant coefficients with $\alpha < \beta$; $\max_{i,j \in \mathbf{N}_N}\{D_{ij}(t)\}$ is the maximum distance among all mussels at iteration t, and δ is a scale factor of space which depends on the problem to be solved.

In the MWO algorithm, mussels are likely to move when surrounded by species at high long-range density, but likely to stay at their home position for high short-range density. Therefore, for mussel m_i, given short-range density ξ_{si} and long-range density ξ_{li}, its moving probability is calculated by

$$P_i := \begin{cases} 1, & \text{if } a - b\xi_{si} + c\xi_{li} > z \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where a, b, and c are positive constant coefficients and z is a value randomly sampled from the uniform distribution [0, 1]. If $P_i = 0$, mussel m_i stays still; and if $P_i = 1$, mussel m_i moves.

Table 1 Benchmarks for simulations

Name	Mathematical representation	Multimodal?	Separable?	Regular?	Range	Minimum
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	No	Yes	Yes	±100	0
Rosenbrock	$f_2(x) = \sum_{i=1}^{n-1} [(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2]$	No	No	Yes	±100	0
Quartic	$f_3(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0,1)$	No	Yes	Yes	±1.28	0
Rastrigin	$f_4(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	Yes	Yes	Yes	±5.12	0
Griewank	$f_5(x) = \frac{1}{4{,}000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	Yes	No	Yes	±600	0
Ackley	$f_6(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	Yes	No	Yes	±30	0
Schwefel2.22	$f_7(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	Yes	No	No	±10	0
Penalty1	$f_8(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + \sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4), y_i = 1 + 1/4(x_i + 1),$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	Yes	No	Yes	±50	0

Table 2 Comparison among MWO with different μ ($d = 20$)

Function	Results	$\mu = 1.5$	$\mu = 1.8$	$\mu = 1.9$	$\mu = 2.0$	$\mu = 2.1$	$\mu = 2.2$	$\mu = 2.5$
f_1	Best	273.99 (7)	2.02e−8 (4)	4.74e−14 (3)	2.60e−18 (1)	9.86e−16 (2)	3.48e−4 (5)	20.88 (6)
	Mean	1.47e+4 (7)	3.51e−7 (4)	9.57e−13 (2)	4.83e−16 (1)	4.48e−8 (3)	0.38 (5)	575.93 (6)
f_2	Best	1.33e+8 (7)	0.35 (2)	0.25 (1)	1.44 (3)	5.27 (4)	93.98 (5)	3.20e+4 (6)
	Mean	5.47e+9 (7)	1.43e+7 (6)	63.74 (1)	107.98 (2)	359.90 (3)	3.73e+3 (4)	1.82e+6 (5)
f_3	Best	0.025 (7)	4.39e−19 (4)	8.69e−31 (3)	3.63e−36 (2)	3.44e−36 (1)	9.54e−19 (5)	3.87e−5 (6)
	Mean	16.31 (7)	4.12e−17 (4)	1.57e−26 (3)	5.08e−33 (1)	1.03e−28 (2)	2.85e−10 (5)	1.50e−3 (6)
f_4	Best	199.74 (7)	7.52e−5 (2)	2.05e−8 (1)	2.98 (3)	5.97 (4)	13.82 (5)	16.31 (6)
	Mean	234.44 (7)	0.75 (1)	3.58 (2)	12.09 (3)	19.86 (4)	30.83 (5)	42.66 (6)
f_5	Best	6.07 (7)	2.21e−7 (4)	3.37e−13 (3)	2.66e−15 (2)	2.22e−16 (1)	2.62e−2 (5)	1.31 (6)
	Mean	176.59 (7)	3.56e−2 (2)	3.57e−2 (3)	5.29e−3 (1)	7.66e−2 (4)	0.34 (5)	3.12 (6)
f_6	Best	8.37 (7)	3.21e−5 (4)	8.74e−8 (3)	9.90e−10 (1)	1.66e−9 (2)	0.078 (5)	4.43 (6)
	Mean	18.30 (7)	2.20e−4 (3)	5.74e−7 (2)	4.79e−8 (1)	0.68 (4)	2.77 (5)	5.75 (6)
f_7	Best	70.43 (7)	3.34e−6 (3)	6.18e−9 (2)	3.63e−10 (1)	6.91e−6 (4)	0.038 (5)	2.94 (6)
	Mean	9.06e+4 (7)	6.78 (6)	6.60e−8 (1)	6.35e−5 (2)	0.064 (3)	1.06 (4)	5.64 (5)
f_8	Best	5.58 (7)	1.80e−10 (4)	8.15e−17 (2)	5.53e−19 (1)	2.99e−13 (3)	5.67e−5 (5)	1.99 (6)
	Mean	1.58e+7 (7)	2.39e−8 (1)	1.60e−2 (3)	1.01e−2 (2)	0.10 (4)	0.71 (5)	7.10 (6)

Once mussel m_i decides to move, the step length is subject to a Lévy distribution. Its new position x'_i is calculated by

$$x'_i := \begin{cases} x_i + \ell_i \Delta_g, & \text{if} \quad P_i = 1 \\ x_i, & \text{if} \quad P_i = 0 \end{cases} \tag{7}$$

where the step length $\ell_i := \gamma \quad [1 - \text{rand}()]^{-1/(\mu-1)}$, $1.0 < \mu < 3.0$, and γ denotes the walk scale factor, which is a positive real number; $\Delta_g := x_g - x_i$ is defined as the distance from m_i to the best position x_g found by all mussels with the optimum nutrition value.

The main steps of the MWO algorithm are presented as follows:

Step 1: Initialize a population of mussels and the algorithm parameters

At the beginning, N mussels are generated and uniformly placed in space S^d. Set the maximum generation G, coefficients of range references α and β, space scale factor δ, moving coefficients a, b, and c, and walk scale factor γ.

Evaluate the initial fitness of mussel m_i by computing the objective function $f(x_i)$. Find the best mussel and record its position as x_g.

Step 2: Calculate the short-range density ξ_s and long-range density ξ_l for each mussel

Using all mussels' coordinate positions, compute the distances $D_{ij}, i, j \in \mathbf{N}_N$ between any two mussels by using Eq. (2), and then compute the short-range reference r_s and long-range reference r_l by (5). For all mussels, calculate their ξ_{si} and $\xi_{li}, i \in \mathbf{N}_N$ by using (3) and (4), respectively.

Table 3 Comparison among MWO with different μ ($d = 30$)

Function	Results	$\mu = 1.5$	$\mu = 1.8$	$\mu = 1.9$	$\mu = 2.0$	$\mu = 2.1$	$\mu = 2.2$	$\mu = 2.5$
f_1	Best	4.55e+3 (7)	1.40e−3 (4)	3.28e−7 (3)	1.29e−9 (1)	3.04e−7 (2)	2.30 (5)	157.99 (6)
	Mean	5.35e+4 (7)	1.33e−2 (4)	3.59e−6 (1)	3.87e−6 (2)	6.10e−3 (3)	31.41 (5)	724.69 (6)
f_2	Best	3.15e+8 (7)	75.65 (3)	9.63 (1)	12.20 (2)	90.20 (4)	6.291e+3 (5)	1.82e+6 (6)
	Mean	2.58e+10 (7)	1.33e+5 (4)	309.77 (2)	142.68 (1)	3.05e+3 (3)	4.35e+5 (5)	1.25e+7 (6)
f_3	Best	3.63 (7)	1.01e−10 (4)	1.32e−16 (3)	1.33e−22 (1)	4.07e−20 (2)	1.97e−10 (5)	3.10e−3 (6)
	Mean	88.81 (7)	8.89e−9 (4)	1.27e−14 (2)	1.02e−17 (1)	1.43e−11 (3)	8.75e−6 (5)	3.81e−2 (6)
f_4	Best	367.06 (7)	5.47 (3)	1.04 (1)	4.97 (2)	29.85 (5)	29.42 (4)	63.66 (6)
	Mean	401.67 (7)	75.43 (5)	44.32 (3)	17.09 (1)	44.11 (2)	58.26 (4)	85.09 (6)
f_5	Best	73.13 (7)	8.00e−3 (3)	6.06e−6 (2)	2.99e−9 (1)	3.40e−2 (4)	1.00 (5)	3.01 (6)
	Mean	503.11 (7)	5.12e−2 (3)	2.87e−2 (1)	2.96e−2 (2)	0.18 (4)	1.82 (5)	7.02 (6)
f_6	Best	19.93 (7)	1.09e−2 (3)	2.37e−4 (2)	1.24e−5 (1)	3.10e−2 (4)	3.45 (5)	5.04 (6)
	Mean	20.25 (7)	1.86 (3)	6.97e−2 (2)	6.61e−2 (1)	1.98 (4)	4.64 (5)	7.78 (6)
f_7	Best	8.09e+6 (7)	4.40e−3 (3)	7.71e−5 (2)	3.19e−6 (1)	9.50e−2 (4)	0.87 (5)	10.94 (6)
	Mean	5.87e+9 (7)	12.51 (5)	1.62e−4 (1)	2.32e−2 (2)	1.45 (3)	4.37 (4)	13.72 (6)
f_8	Best	1.39e+6 (7)	1.08e−5 (3)	1.34e−8 (2)	3.92e−11 (1)	1.61e−2 (4)	1.76 (5)	5.93 (6)
	Mean	4.57e+8 (7)	1.06e−2 (2)	1.04e−2 (1)	1.95e−2 (3)	0.64 (4)	3.65 (5)	16.68 (6)

Step 3: Determine the movement strategy for each mussel

Calculate the moving probability P_i of mussel m_i according to the short-range density ξ_{si} and long-range density ξ_{li} by Eq. (6); If $P_i = 1$, calculate its step length by $\ell_i = \gamma[1 - \text{rand}()]^{-1/(\mu-1)}$.

Step 4: Update the position for all mussels

Compute the new position coordinate x'_i of mussel m_i in S^d by using Eq. (7).

Step 5: Evaluate the fitness of mussel m_i after position updating. Calculate the objective function $f(x')$ for the new positions. Rank the solutions so as to find the global best mussel m_g, and update the best record [best position x_g and optimal fitness $f^*(x_g)$].

Step 6: Examine if the termination criteria is satisfied? If it is true, stop the algorithm and output the optimized results; otherwise, go to step 2 to start the next iteration.

The pseudocode for the MWO algorithm is listed as follows.

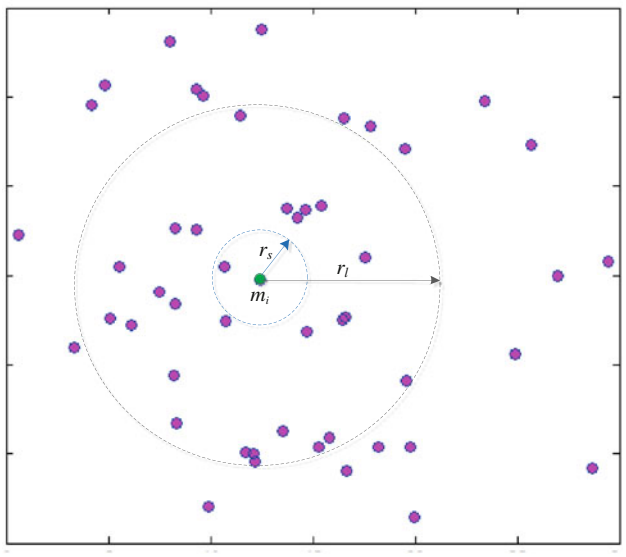

Fig. 1 Sketch of short range and long range for a mussel in 2-D space

Algorithm 1 Pseudocode for the MWO algorithm

Initialization:
Set $t := 0$;
FOR (mussel m_i , $i = 1$ to N)
Uniformally randomize initial positions $x_i(0)$ for all mussels m_i;
Calculate the fitness values of initial mussels: $f(x_i(0))$;
END FOR
Find the global best mussel and record its position as x_g.
Iteration:
WHILE ($t \leq G$, or $f(x^*) > \epsilon$)
FOR (mussel m_i , $i = 1$ to N)
Calculate distances from m_i to other mussels by equation (2);
Calculate the short-range reference $r_s(t)$ and long-range reference $r_l(t)$ by equation (5);
Calculate its short-range density ξ_{si} and long-range density ξ_{li} by using equations (3)-(4);
Compute its moving probability $P_i(t)$ according to equation (6);
If: $P_i = 1$
Generate its step length $\ell_i(t)$ by a Lévy distribution;
Else If: $P_i = 0$
$\ell_i(t) = 0$.
End If
Compute its position coordinate $x'_i(t)$ by equation (7);
Evaluate the fitness value of the new position $f(x'_i(t))$;
END FOR
Rank all mussels according to the fitness from the best to worst, find the global best mussel and update the best position x_g;
Set $t := t + 1$;
END WHILE
Output the optimized results and end the algorithm.

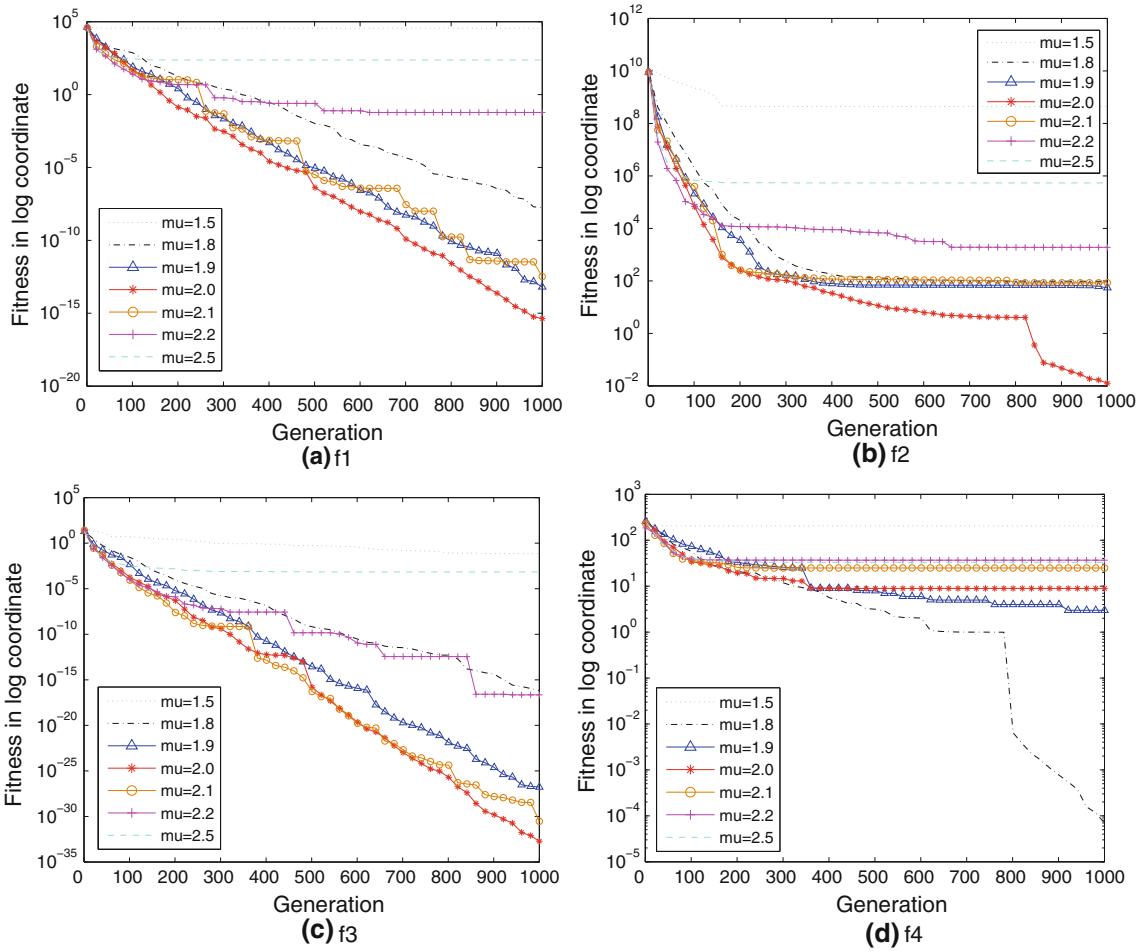

Fig. 2 The optimal dynamics of MWO with different μ values for $f_1 - f_4$

Simulation and Results

In this section, we test the convergence performance of MWO and compare it with some existing population-based metaheuristic algorithms.

Benchmarks

To evaluate the performance of the MWO algorithm for some chosen problems, we employ a set of eight standard benchmark functions, which are given in Table 1.

These benchmarks have been widely used in the literature for comparison of optimization methods. Some are multimodal, which means that they have multiple local minima. Some are nonseparable, which means that they cannot be written as a sum of functions of individual variables. Some are regular, which means they are analytical (differentiable) at each point of their domain [23, 36]. Each of the functions has a global minimum $f(x^*) = 0$.

Experimental Setting

In this work, we investigate the effect of the Lévy walk in the MWO algorithm. Different values of the shape parameter μ of the Lévy distribution are tested, i.e., $\mu = 1.5$, $\mu = 1.8$, $\mu = 1.9$, $\mu = 2.0$, $\mu = 2.1$, $\mu = 2.2$, and $\mu = 2.5$.

To explore the benefits of MWO, we compare its performance on various benchmark functions with four well-known metaheuristic algorithm: (1) genetic algorithm (GA), (2) biogeography-based optimization (BBO), (3) particle swarm optimization (PSO), and (4) group search optimizer (GSO). We executed GA, BBO, PSO, and GSO algorithms with their standard versions by adopting the empirical parameters setting.

The parameter setting of the MWO algorithm is summarized as follows. The population size $N = 50$ is used in the paper. The initial population is generated uniformly at random in the search space S^d. The short-range reference coefficient $\alpha = 1.1$, the long-range reference coefficient

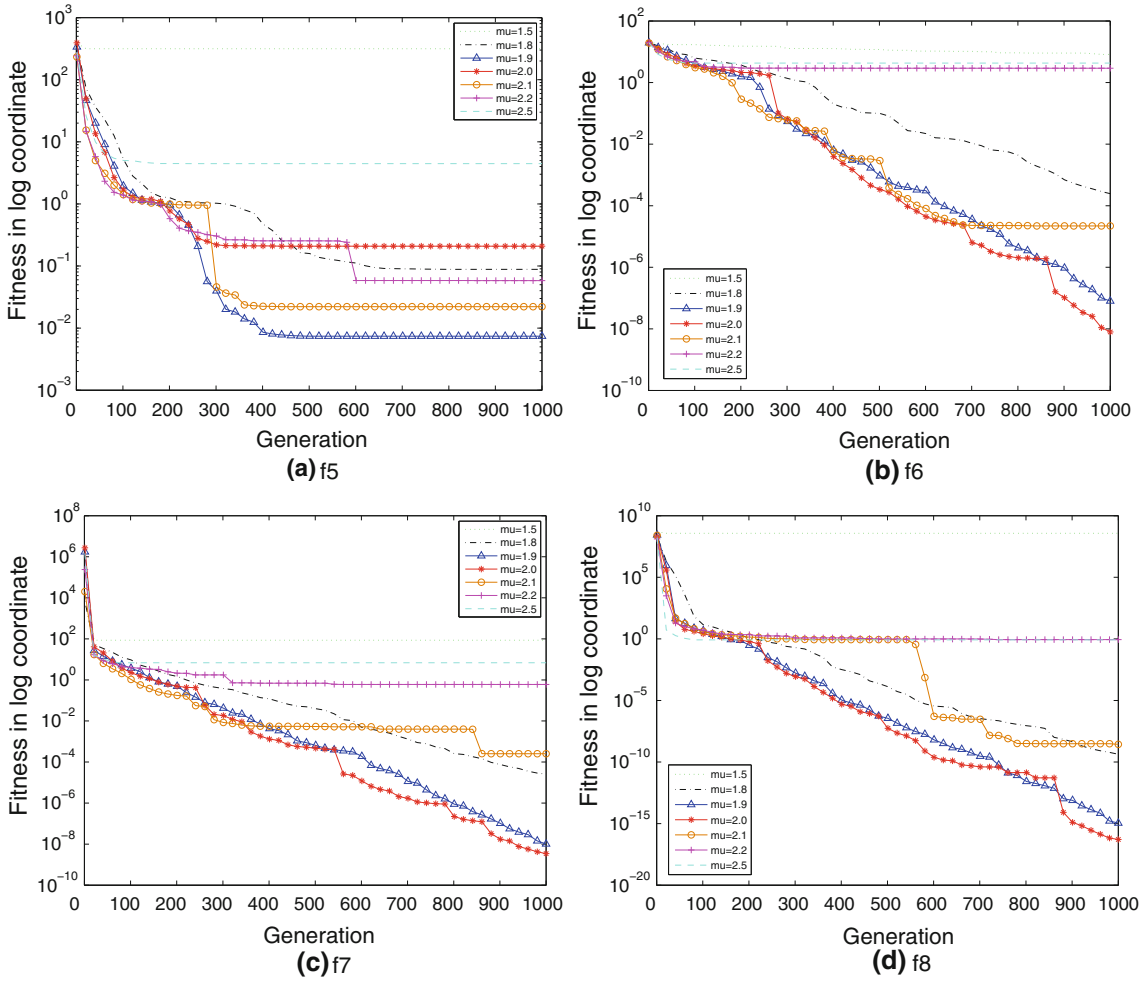

Fig. 3 The optimal dynamics of MWO with different μ values for $f_5 - f_8$

$\beta = 7.5$, the moving coefficients $a = 0.63$, $b = 1.26$, and $c = 1.05$, the walk scale factor $\gamma = 0.1$, and the scale factor of space δ is set as $\delta = 25$ for f_1 and f_2, $\delta = 0.3$ for f_3, $\delta = 1.2$ for f_4, $\delta = 150$ for f_5, $\delta = 10$ for f_6, $\delta = 2.5$ for f_7, and $\delta = 15$ for f_8.

In this work, we ran 50 Monte Carlo simulations for each case with different μ values on each benchmark to obtain representative performance. The simulations were executed on 20-D and 30-D functions, respectively. We carried out the simulations with two different termination criteria: (1) the maximum number of iterations $G = 1{,}000$ is reached, and (2) the algorithm converges to a predefined precision called the error goal ϵ, that is $f(x^*) \leq \epsilon$, e.g., $\epsilon = 10^{-10}$ for f_1, $\epsilon = 15$ for f_2, $\epsilon = 10^{-6}$ for f_3, $\epsilon = 10$ for f_4, $\epsilon = 10^{-3}$ for f_5–f_7, and $\epsilon = 0.1$ for f_8 defined in this work.

The experiments were carried out on a PC with a 2.40-GHz Intel processor and 4.0 GB RAM. All programs were written and executed in MATLAB 7.1. The operating system was Microsoft Windows 7.

Results and Discussion

The experiments included an average test on all cases for each benchmark function. Tables 2 and 3 list the best and mean values of the 20-D and 30-D functions obtained by MWO in 1,000 iterations, respectively. Numbers in parenthesis indicate the ranking in seven simulations of MWO with different settings of μ.

Table 2 shows that the best and mean function values are all not good when μ is set as a larger ($\mu = 2.5$) or smaller value ($\mu = 1.5$) in its valid range. When $\mu = 2.0$, MWO performs well for all eight functions, e.g., the best function results rank first or second for f_1, f_3, and f_5–f_8, and third for f_2 and f_4; the mean function results rank first or second for all benchmarks except f_4. For f_2 and f_4, the MWO with $\mu = 1.9$ performs the best, but its minimum function result ranks second after $\mu = 1.8$ for f_4. For f_3, the best result obtained with $\mu = 2.1$ ranks first and its mean function result is inferior to that of $\mu = 2.0$. Generally,

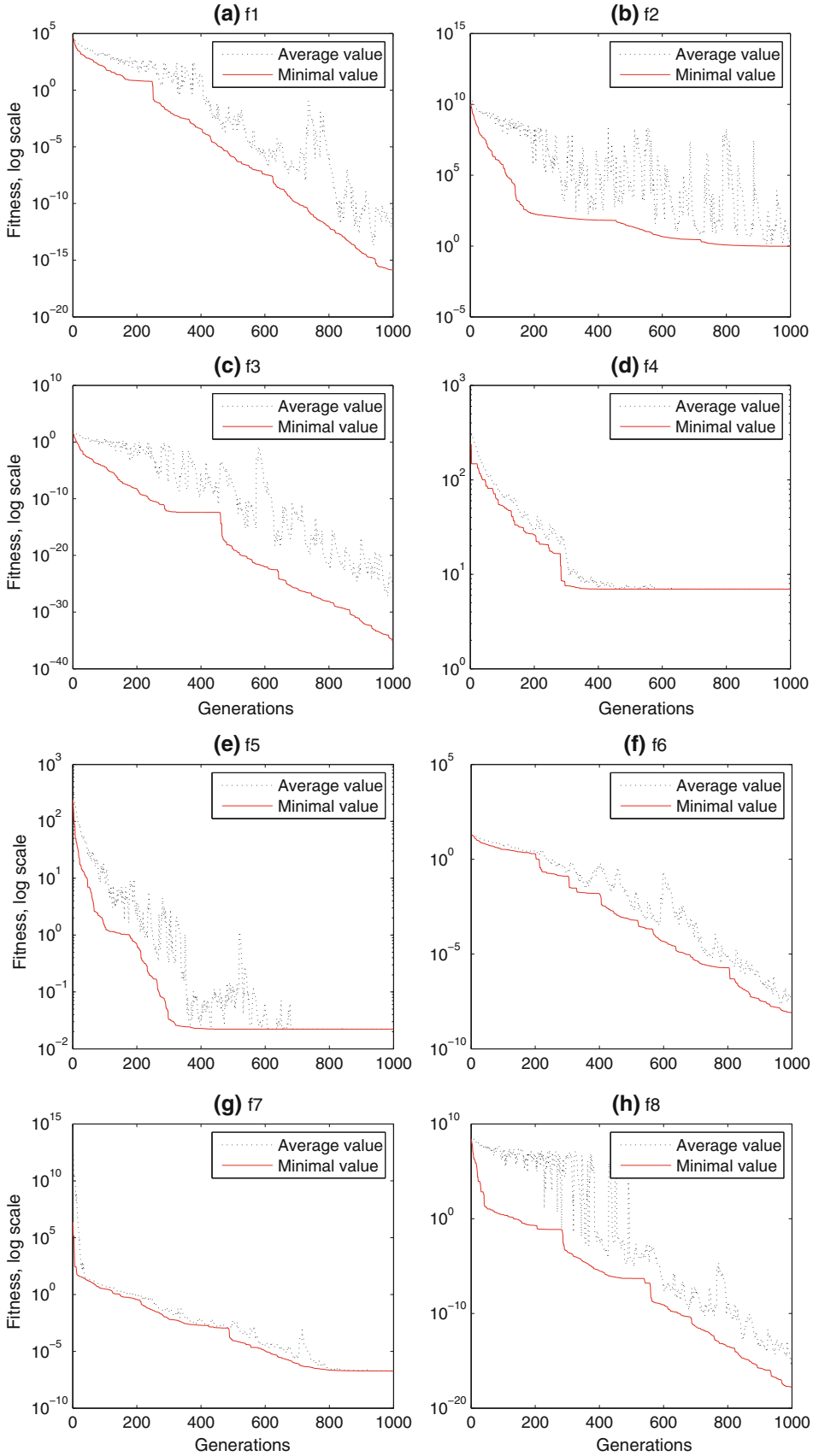

Fig. 4 Typical dynamics of fitness on log scale for 20-D functions ($\mu = 2.0$)

Table 4 Optimal results of MWO ($\mu = 2.0$), GA, BBO, PSO and GSO on 20-D benchmark functions

Function	Results	GA	BBO	PSO	GSO	MWO ($\mu = 2.0$)
f_1	Best	1.59 (5)	0.44 (4)	5.83e−28 (1)	4.55e−12 (3)	2.60e−18 (2)
	Mean	10.93 (5)	1.72 (4)	4.85e−27 (1)	2.37e−8 (3)	4.83e−16 (2)
f_2	Best	380.20 (3)	297.70 (2)	441.74 (5)	420.24 (4)	1.44 (1)
	Mean	1.58e+3(3)	915.28 (2)	1.88e+5(5)	7.89e+3 (4)	107.98 (1)
f_3	Best	3.28e−7 (4)	4.17e−8 (3)	1.27e−13 (2)	8.03e−5(5)	3.63e−36 (1)
	Mean	1.37e−6 (3)	2.06e−7 (2)	1.40e−5 (4)	6.24e−3 (5)	5.08e−33 (1)
f_4	Best	7.09 (4)	9.92e−2 (1)	6.99 (3)	15.77 (5)	2.98 (2)
	Mean	13.50 (3)	0.83 (1)	28.96 (4)	59.35 (5)	12.09 (2)
f_5	Best	1.15 (5)	0.67 (4)	9.21e−15 (3)	4.29e−3 (2)	2.66e−16 (1)
	Mean	1.30 (5)	0.96 (4)	0.17 (3)	1.52e−2 (2)	5.29e−3 (1)
f_6	Best	3.27 (5)	0.48 (4)	8.17e−7 (2)	6.81e−5 (3)	9.90e−10 (1)
	Mean	6.88 (5)	0.79 (4)	0.39 (3)	3.33e−2 (2)	4.79e−8 (1)
f_7	Best	1.66 (5)	0.24 (3)	4.61e−15 (1)	0.28 (4)	3.63e−10 (2)
	Mean	2.75 (3)	0.35 (2)	11.37 (4)	23.56 (5)	6.35e−5 (1)
f_8	Best	5.20e−3 (3)	2.30e−3 (2)	0.82 (5)	4.83e−2 (4)	5.53e−19 (1)
	Mean	4.23e−2 (3)	1.65e−2 (2)	5.73 (4)	29.04 (5)	1.01e−2 (1)

Table 5 Optimal results of MWO ($\mu = 2.0$), GA, BBO, PSO and GSO on 30-D benchmark functions

Function	Results	GA	BBO	PSO	GSO	MWO ($\mu = 2.0$)
f_1	Best	45.19 (5)	3.48 (4)	7.37e−12 (1)	8.35e−5 (3)	1.29e−9 (2)
	Mean	138.81 (4)	7.34 (3)	0.59 (2)	157.11(5)	3.87e−6 (1)
f_2	Best	7.96e+3 (4)	1.25e+3 (2)	5.81e+3 (5)	2.57e+3 (3)	12.20 (1)
	Mean	4.19e+4 (4)	3.04e+3 (2)	4.54e+5 (5)	3.59e+4 (3)	142.68 (1)
f_3	Best	2.07e−5 (4)	5.68e−7 (2)	1.25e−5 (3)	7.89e−2 (5)	1.33e−22 (1)
	Mean	1.26e−4 (3)	5.65e−6 (2)	5.20e−3 (4)	9.87e−2 (5)	1.02e−17 (1)
f_4	Best	39.99 (5)	1.21 (1)	11.08 (3)	27.43 (4)	4.97 (2)
	Mean	64.18 (4)	2.39 (1)	43.62 (3)	91.23 (5)	17.09 (2)
f_5	Best	1.86 (5)	1.00 (4)	5.31e−13 (1)	0.26 (3)	2.99e−9 (2)
	Mean	2.73 (5)	1.06 (3)	0.18 (2)	1.17 (4)	2.96e−2 (1)
f_6	Best	6.83 (5)	0.74 (4)	6.22e−15 (1)	2.41e−5 (3)	1.24e−5 (2)
	Mean	11.70 (5)	1.21 (4)	6.48 (3)	3.34e−3 (2)	6.61e−2 (1)
f_7	Best	7.95 (5)	0.50 (4)	1.06e−14 (2)	2.74e−3 (3)	3.19e−6 (1)
	Mean	8.62 (3)	0.78 (2)	63.71 (4)	17.50 (5)	2.32e−2 (1)
f_8	Best	0.11 (4)	1.02e−2 (3)	8.37 (5)	6.58e−4 (2)	3.92e−11 (1)
	Mean	0.57 (3)	4.04e−2 (2)	24.18 (5)	1.90 (4)	1.95e−2 (1)

$\mu = 2.0$ is an optimal setting for MWO on average, and [1.9, 2.1] presents a good range of μ of MWO for solving 20-D functions. From Table 3, we can obtain a similar conclusion for 30-D functions. $\mu = 2.0$ is the optimal setting on average, and a recommended range of μ is [1.9, 2.0].

Figures 2 and 3 show typical optimal dynamics of MWO with different μ values for f_1–f_8. It is obvious that MWO with $\mu = 2.0$ converges fastest and achieves the best solution for most functions, except f_4 and f_5. MWO with $\mu = 1.8$ converges fast for f_4. For f_5, MWO with $\mu = 1.9$ performs well, but all algorithms encounter a saturation state.

Furthermore, we present the evolving fitness dynamics of the 20-D benchmarks for MWO with the optimal setting of $\mu = 2.0$ in Fig. 4. In Fig. 4, each subfigure represents a typical evolving dynamics of the global minimal value (red line) and the average value of all mussels (black dotted line), in logarithmic coordinates. These curves of minimum values are in accordance with the results in Table 2. From these figures, we find that MWO keeps converging to its optimum solution for the benchmarks except f_4 and f_5. We

Table 6 Average CPU time in seconds of MWO ($\mu = 2.0$), GA, BBO, PSO, and GSO

Function	Dimension	GA	BBO	PSO	GSO	MWO ($\mu = 2.0$)
f_1	20	11.18	12.16	11.99	15.97	12.15
	30	11.94	12.86	12.47	16.84	13.46
f_2	20	11.01	8.64	11.64	13.22	12.89
	30	11.97	9.61	12.38	16.08	13.11
f_3	20	10.70	8.05	11.68	13.05	12.00
	30	11.23	8.89	12.64	16.49	13.15
f_4	20	11.20	8.49	11.79	13.32	13.07
	30	11.92	9.40	12.32	16.51	12.85
f_5	20	10.83	9.90	11.60	13.05	12.93
	30	11.39	10.47	11.81	16.09	12.43
f_6	20	11.04	8.09	11.45	14.29	13.02
	30	11.43	8.69	11.69	16.01	12.34
f_7	20	10.96	8.16	11.54	13.22	12.03
	30	11.10	8.62	11.62	15.28	12.61
f_8	20	11.96	10.53	12.34	15.57	13.83
	30	12.13	10.99	12.62	16.72	13.28

Table 7 Comparisons between BBO, PSO, and MWO under $f(x^*) \leq \epsilon$ ($d = 20$)

Function	Error goal	BBO		PSO		MWO	
		Ave. its. (CPU time)	Success rate (%)	Ave. its. (CPU time)	Success rate (%)	Ave. its. (CPU time)	Success rate (%)
f_1	10^{-10}	/	0	720 (8.39 s)	100	722 (9.36 s)	100
f_2	15	/	0	/	0	1,641 (21.65 s)	100
f_3	10^{-6}	893 (10.25 s)	100	995 (11.63 s)	100	170 (1.16 s)	100
f_4	10	193 (1.64 s)	100	1,470 (17.64 s)	10	532 (6.69 s)	98
f_5	10^{-3}	/	0	1,363 (15.74 s)	68	1,768 (24.12 s)	92
f_6	10^{-3}	/	0	2,294 (26.96 s)	80	540 (7.09 s)	100
f_7	10^{-3}	5,139 (37.87 s)	98	2,974 (21.87 s)	100	450 (5.46 s)	100
f_8	10^{-1}	519 (5.84 s)	100	/	0	252 (3.23 s)	100

its. iterations

also find that, when the best global value decreases in the iterative process, the average fitness appears to oscillate for all benchmarks, especially f_2, f_3, f_5, and f_8. These results prove that the mussels preferred Lévy walk can maintain the diversity of the population, because the Lévy walk adopts long steps with low probabilities and revisits the same sites far less often, which helps MWO to escape from local optima and find optimal results more efficiently.

We compared the optimal results of MWO with $\mu = 2.0$ with GA, BBO, PSO, and GSO for the 20-D and 30-D functions. The best and mean results are presented in Tables 4 and 5. Table 4 shows that MWO generates significantly better results than GA and GSO on all the functions with $d = 20$. From the comparisons between MWO and BBO, we can see that MWO has obviously better performance than BBO on f_1, f_3, and f_5–f_8. However, MWO yields statistically inferior results on f_4 compared with BBO. From the comparisons between MWO and PSO, we can see that MWO has better average performance than the PSO algorithm on all the functions except f_1. For f_7, PSO presents better optimal results than MWO does.

From Table 5, we see that MWO performs best on average for the majority of the 30-D benchmarks. BBO exceeds MWO on f_4. PSO is the second most effective, followed by BBO. PSO runs in first place in terms of the best value for functions f_1, f_5, and f_6, but its mean function values are no match for MWO.

Table 6 presents the average CPU time in seconds for MWO ($\mu = 2.0$) and the other four algorithms. From this table, we see that the average CPU time required by BBO is less than those of the other algorithms for all eight functions except f_1. Among the other four algorithms, the

average CPU time of GSO is the longest; GA, MWO, and PSO have similar CPU time cost, or rather MWO requires slightly more time than GA and PSO.

In this section, we also describe the test of the convergence performance of MWO based on a predefined precision called the error goal ϵ, compared with BBO and PSO, which are the best two among the four metaheuristics. The error goal is the criterion selected to stop the algorithm, which means that the termination condition is $f(x^*) \leq \epsilon$. Table 7 presents the simulation results of BBO, PSO, and MWO for 20-D functions, in which a successful run is defined as the algorithm converging to the error goal ϵ in 10,000 iterations. The average convergence time only takes successful runs into account.

From the results in Table 7, we see that MWO needs fewer runs in terms of number of iterations and CPU time in seconds on average than PSO to reach the precision ϵ defined in the table, for most functions except f_1. It performs better than BBO except for f_4. For 50 Monte Carlo simulations, the average success rate of MWO (100 % for almost all the functions) is significantly higher than that of PSO for f_2, f_4, and f_5–f_8. It also exceeds BBO for f_1, f_2, and f_5–f_7. Therefore, MWO has competitive performance on the whole to other metaheuristics in terms of accuracy and convergence speed.

Conclusions

This paper presents a novel metaheuristic algorithm—mussels wandering optimization (MWO)—for the first time, through mathematically modeling mussels' leisurely locomotion behavior when they format their bed pattern in a habitat. MWO emphasizes competition and cooperation among mussels via stochastic decisions based on the mussel density in the habitat, and random walks. In the computational model, besides the Lévy distribution of step length adopted by each mussel, its moving decision behavior is also a stochastic variable.

MWO is conceptually simple and easy to implement. The performance of MWO is demonstrated via a set of eight benchmarks and compared with four well-known metaheuristics, i.e., GA, BBO, PSO, and GSO. The primary conclusion is that MWO beats the four algorithms on average. MWO provides a new approach for solving a variety of large-scale optimization problems, which makes it particularly attractive for real-world applications.

One of the most significant merits of MWO is that it provides an open framework to tackle hard optimization problems, by utilizing research in spatial bed formatting, landscape-level pattern evolution, and the leisurely locomotion behavior of mussels in their habitat. Our future planned work includes: (1) presenting the mathematical models of distribution patterns of mussels based on some typical random walks, (2) proving the convergence of the proposed models for complex optimization problems with and without noise, and (3) exploring its promising real-world applications.

Acknowledgments The authors thank Prof. M. Zhou and Prof. R. Kozma for helpful discussions and constructive comments. The authors also thank the reviewers for their instrumental comments in improving this paper from its original version. This work was supported in part by the National Science Foundation of China (grants no. 61005090, 61034004, 61272271, and 91024023), the Natural Science Foundation Program of Shanghai (grant no. 12ZR1434000), the Program for New Century Excellent Talents in University of MOE of China (grant no. NECT-10-0633), and the Ph.D. Programs Foundation of MOE of China (grant no. 20100072110038).

References

1. Weise T. Global optimization algorithms theory and application, Germany. 2009. it-weise.de.
2. Michalewicz Z, Fogel DB. How to solve it: modern heuristics, 2nd ed. Berlin: Springer; 2004.
3. Nobakhti A. On natural based optimization. Cognit Comput. 2010;2:97–119.
4. Shadbolt Nigel Nature-inspired computing. IEEE Intell Syst. 2004;1/2:1–3.
5. Zhang J, Zhan Z, et al. Enhancing evolutionary computation algorithms via machine learning techniques: a survey. IEEE Comput Intell Mag. 2011;68–75.
6. Zhang J, Chung H, Lo W Clustering-based adaptive crossover and mutation probabilities for genetic algorithms. IEEE Trans Evol Comput. 2007;11(3):326–35.
7. Chen Shu-Heng, et al. Genetic programming: an emerging engineering tool. Int J Knowl Based Intell Eng Syst. 2008;12(1):1–2.
8. Fogel LJ. Intelligence through simulated evolution : forty years of evolutionary programming. New York: Wiley; 1999.
9. Price K, Storn R, Lampinen J. Differential evolution: a practical approach to global optimization. Berlin: Springer; 2005.
10. Gao Y, Culberson J. Space complexity of estimation of distribution algorithms. Evol Comput. 2005;13(1):125–43.
11. Kennedy J, Eberhart RC. Swarm intelligence. San Francisco: Morgan Kaufmann; 2001.
12. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of the international conference on neural networks. Australia: Perth; 1995. pp. 1942–48.
13. Dorigo M, Gambardella L. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput. 1997;1(1):53–66.
14. Karaboga D, Akay B. A comparative Study of artificial bee colony algorithm. Appl Math Comput. 2009;214:108–32.
15. He S, Wu Q, Saunders J. Group search optimizer: an optimization algorithm inspired by animal searching behavior. IEEE Trans Evol Comput. 2009;13(5):973–90.
16. Passino K. Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag. 2002;22:52–67.
17. Kirkpatrick S, Gelatt C, Vecchi M. Optimization by simulated annealing. Science, 1983;220(4598):671–80.
18. Haykin S. Neural networks: a comprehensive foundation. Englewood: Prentice Hall; 1999.
19. Bagheria A, Zandiehb M, Mahdavia Iraj, Yazdani M. An artificialimmunealgorithm for the flexible job-shop scheduling problem. Futur Gener Comput Syst. 2010;26(4):533–41.

20. Lam A, Li V. Chemical-reaction-inspired metaheuristic for optimization. IEEE Trans Evol Comput. 2010;14(3):381–99.
21. Chen X, Ong Y, Lim M. Research frontier: memetic computation—past, present and future. IEEE Comput Intell Mag. 2010;5(2):24–36.
22. Geem Z, Kim J, Loganathan G. A new heuristic optimization algorithm: harmony search. Simulation, 2001;76(2):60–68.
23. Simon D. Biogeography-based optimization. IEEE Trans Evol Comput. 2008;12(6):702–13.
24. Yang X-S. Cuckoo search via Lévy flights. World Congr Nat Biol Inspired Comput, 2009.
25. Eusuffa M, Lanseyb K, Pashab F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Eng Optim. 2006;38(2):129–54.
26. Zelinka I. SOMA: self-organizing migrating algorithm. Berlin: Springer; 2004. pp. 167–217.
27. Kang Q, An J, Wang L, Wu Q. Unification and diversity of computation models for generalized swarm intelligence. Int J Artif Intell Tools. 2012;21(3):1240012.
28. Daniel G. Why did you Lévy?. Sci Technol Human Values 2011;332:1514.
29. Alfaro Andrea C. Population dynamics of the green-lipped mussel, Perna canaliculus, at various spatial and temporal scales in northern New Zealand. J Exp Mar Biol Ecol. 2006;334:294–315.
30. Haag WR, Warren ML. Role of ecological factors and reproductive strategies in structuring freshwater mussel communities. Can J Fish Aquat Sci. 1998;55:297–306.
31. Strayer D, Downing J, Haag W Changing perspectives on pearly mussels-North America's most imperiled animals. BioScience, 2004;54(5):429–39.
32. de Jager M. Lvy walks evolve through interaction between movement and environmental complexity. Science, 2011;332:1551–53.
33. Viswanathan G. Fish in Lévy-flight foraging. Nat Environ Pollut Technol. 2010;465:1018–19.
34. Viswanathan G. Lévy flight search patterns of wandering albatrosses, Nature. 1996;381:413–15.
35. Brockmann D, Hufnagel L, Geisel T. The scaling laws of human travel. Nature 2006;439:462–65.
36. Cai Z, Wang Y. A multiobjective optimization-based evolutionary algorithm for constrained optimization. IEEE Trans Evol Comput. 2006;10:658–75.