International Journal of Computational Intelligence and Applications

Vegetation Evolution: An Optimization Algorithm Inspired by the Life Cycle of Plants

Jun YU

Institute of Science and Technology, Niigata University, Niigata,Japan
URL: http://www.eng.niigata-u.ac.jp/~yujun

Received (received date)
Revised (revised date)

We have observed that different types of plants in nature can use their own survival mechanisms to adapt to various living environments. A new population-based vegetation evolution (VEGE) algorithm is proposed to solve optimization problems by interactively simulating the growth and maturity periods of plants. In the growth period, individuals explore their local areas and grow in potential directions, while individuals generate many seed individuals and spread them as widely as possible in the maturity period. The main contribution of our proposed VEGE is to balance exploitation and exploration from a novel perspective, which is to perform these two periods in alternation to switch between two different search capabilities. To evaluate the performance of the proposed VEGE, we compare it with three well-known algorithms in the evolutionary computation community: differential evolution, particle swarm optimization, and enhanced fireworks algorithm - and run them on 28 benchmark functions with 2-dimensions (2-D), 10-D, and 30-D with 30 trial runs. The experimental results show that VEGE is efficient and promising in terms of faster convergence speed and higher accuracy. In addition, we further analyze the effects of the composition of VEGE on performance, and some open topics are also given.

Keywords: Evolutionary Computation; Vegetation Evolution; Meta-heuristic Algorithm.

1. Introduction

Evolutionary computation (EC) is a kind of population-based optimization techniques, and has developed many novel and effective algorithms inspired by biological evolution theory, natural/social phenomena or behaviors, human culture, and others. EC algorithms have attracted much attention after decades of vigorous development thanks to their excellent features, such as robustness, simplicity, parallelism, high efficiency, and user-friendliness. Many novel EC algorithms are constantly being raised since they became popular in the 1960s. For example, Storn and Price proposed the well-known differential evolution (DE) [1], which uses differential information to perturb individuals such that they evolve toward an optimal area. Particle swarm optimization (PSO) [2] simulates the foraging behavior of birds to achieve individual cooperation and thus show a high degree of intelligence. Fireworks algorithm (FWA) [3] is inspired by the explosions of real fireworks in the sky and simulates the explosion process to iteratively find the optimal solution. Besides, there are some other EC algorithms that also have strong optimization capabilities, such as bacterial foraging optimization [4], artificial bee colony [5], bat algorithm [6], and krill herd [7].

More practitioners are committed to introduce new search strategies or mechanisms into existing EC algorithms to further improve their performance [8], [9], [10]. For example, JADE [11] features a new mutation strategy with optional external archive and parameters to improve the performance of classic DE. The enhanced fireworks algorithm (EFWA) [12] introduces five major improvements to FWA to further enhance its performance. Besides, some researchers use approximate models of the fitness landscape to aid evolution and reduce computational costs [13], [14]. For example, a convergence point is estimated mathematically using the gradient information between parent individuals and their offspring individuals [15]; then the estimated convergence point, as an elite individual, replaces the worst individual in the current population to accelerate convergence [16]. Compared with the previous two approaches, this direction has not received enough attention, but there is still huge potential and space to continue research.

Although there have been dozens of EC algorithms based on different meta heuristics, their optimization ideas are similar. Specifically, they pay attention to the ability of exploration in the early stage, and gradually emphasize the ability of exploitation with the convergence of the population. Here, we try to balance exploration and exploitation from a new perspective instead of the inverse relationship, and develop a new population-based vegetation evolution (VEGE) algorithm to find the global optimum, which mainly simulate the growth and maturity of plants repeatedly to balance exploration and exploitation well.

The main purpose of this paper is to simplify the growth and maturity mechanism observed from various real plants, and then simulate two different periods (grow and maturity) of plants to form a new population-based optimization framework, i.e. VEGE involves cyclically switching between two different optimization capabilities (exploitation and exploration) to solve various optimization problems. The secondary purpose is to analyze the performance of our proposed VEGE as well as its pros and cons. Finally, we give some topics which are for open discussion.

The remaining parts are organized as follows. We briefly review some typical survival patterns of plants that we can find everywhere in our life in Section 2. The proposed VEGE is fully described in Section 3. To evaluate the performance of the proposed VEGE, we compare it with three other famous EC algorithms, and run them on 28 benchmark functions from CEC 2013 of 3 different dimensions in Section 4. We analyze the impact of VEGE parameter settings on performance and discuss some open topics in Section 5, and finally conclude the current work and present some future opportunities in Section 6.

2. Vegetation Survival

A great number of researchers obtain inspiration from the natural selection and group behavior of organisms to develop multiple novel EC algorithms. However, a few people have paid attention to the evolution and growth of vegetation and found some inspiration, e.g. flower pollination [17] and dandelion algorithm [18]. Actually, of all living things on the earth, a great part is made up of not animals and humans, but of vegetation. Humans, together with all other animals, accounted for only a small portion of life. Vegetation has evolved various effective and intelligent survival mechanisms to adapt to different environments, which include the valuable innovation of perennial evolution. We thus try to summarize some of their mechanisms and develop a new EC algorithm based on our observation.

Just as human and some animals grow up from fertilized eggs, many plants usually begin their lives from seeds. Although different plants have different times to become maturity and different growth patterns, for example, some can grow into towering trees after decades, while some grow into grass after only a few months. They have a similar pattern, i.e., plants take time to grow from seed to maturity, during which time they use their unique survival mechanisms to ensure their growth. Once plants have absorbed enough nutrients to grow and mature, they generate their own seeds to ensure the continuation of their species. Generally, plants do not generate only one seed but hundreds of seeds. These generated seeds are dispersed everywhere and open a new round of their growth in new environments.

Although different plants have evolved different complex survival mechanisms, we can still extract some commonalities from these mechanisms and simplify their life cycle to develop a new EC algorithm. By carefully analyzing the growth and propagation patterns, we realize that the entire life cycle of plants can be roughly divided into two different periods. First, plants need to absorb nutrients to grow in places where they take roots. This period can be seen as an exploitation in local areas. Next, the mature plants produce lots of seeds and use various forces to disperse their seeds widely. This period can be viewed as an exploration over wide areas. Together these two periods can form an optimization framework for the new VEGE algorithm, and the details will be given in the following Section.

3. Vegetation Evolution

Each real plant is transformed into an individual (i.e., a candidate solution), and it will experience differently during its life cycle, the *growth* period and the *maturity* period, to undertake different search capabilities. In short, the optimization process of VEGE can be summarized as follows. The initial population, containing multiple individuals, is randomly generated, and all individuals grow independently to discover better local areas around them. Here, every exploration of an individual is regarded as a *growth* operation, and the individual only grows in a better direction, i.e., a better offspring individual replaces its parent individual, otherwise, it keeps unchanged. Once the predefined maximum number of times of growth is reached, all individuals transition from the growth state to the maturity state. At this time, each individual generates the same number of multiple seed individuals (i.e., candidate solutions) rather than just one seed individual, forming a temporary seed population. Finally, individuals who survive to the next generation are selected from a mixed population consisting of the current population and the temporary seed population. The above steps are repeated until a termination condition is satisfied, and finally VEGE outputs the global optimum found. Fig. 1 shows visually the general process of our proposed VEGE algorithm.

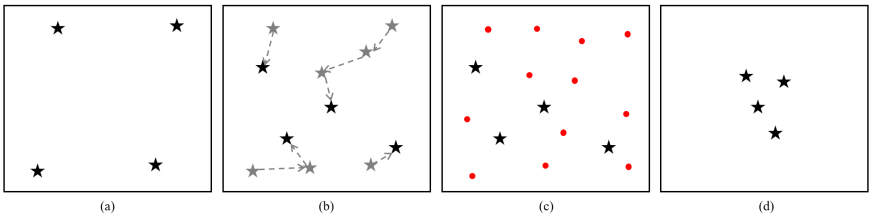

Fig. 1. The universal search process of the proposed VEGE algorithm. (a) The initial population is randomly generated, (b) All individuals experience the growth period until the predefined maximum number of growth is reached. Dotted arrows indicate the growth directions of individuals within local areas. (c) All individuals enter the maturity period, and each individual generates multiple seed individuals. Red circles indicate generated seed individuals and all of them form a temporary seed population. (d) Individuals in the next generation are selected from all individuals in step (c). All selected individuals experience new life cycles again, and steps (b) and (d) are iterated until a termination condition is satisfied.

Next, we focus on the two core operations of the new proposed VEGE algorithm, the growth period and the maturity period.

Growth period

When plants root in a new environment, they try to absorb nutrients to maximize their healthy growth by extending the roots and opening branches. Inspired by these observations, we model the process of plant growth as local search and evolve into potential directions. All individuals during this growth period are made responsible for exploitation by emphasizing competition among individuals.

The key problem is that how can we implement reasonable local search for individuals? Although different real-life plants have different mechanisms to control their growth, we roughly extract two factors from them to simulate local search: growth direction and growth length. Once these two factors are determined, the morphology of plant growth is also generally determined.

Corresponding to our proposed VEGE, we introduce a new parameter, *growth radius*, to determine the maximum radius of the local search. As with our first attempt, we use a stochastic strategy to randomly generate offspring individuals within the growth radius. Thus, the direction starting from an individual to its randomly generated offspring individual can imply a growth direction and the distance between them shows the growth length. When a generated offspring individual is better than its parent individual, we believe that growing along this direction is a potential choice. In such case, we classify it as a successful build local search, and we replace the parent with its offspring individual. Otherwise, the growth direction is poor, and the parent individual does not move to its offspring and is kept the same. This growth mode is also very common in nature. Some plants do not grow constantly, but instead sometimes choose to stop growing so that they can endure harsh environments - or, conversely, they grow more vigorously. In this paper, we create a simple model of real plant growth for the local search of our proposed VEGE algorithm. Since there are many more complex mechanisms involved in the control of growth in the real world, it will be a challenge in the future to further develop more efficient local searches by learning more about the novel mechanisms employed by vegetation.

The next important problem to be solved is how to control the number of growth operations. In general, not all plants can keep growing grow all the time without any restriction in the real world; some plants stop growing when they reach certain conditions. Similarly, the number of fitness evaluations has also an upper limit when we apply EC algorithm to solve real-world problems. Another new parameter, *growth cycle*, is thus introduced to control the maximum number of growth to avoid unlimited growth operations and start the subsequent maturity operations. It's worth noting that we count the number of growth operations not only when generated offspring is better than its parent but also when the opposite is true (see the 9th line in Algorithm 1). Finally, these two parameters can determine the whole growth period of individuals. Once the number of growth cycles reaches the predefined maximum number, individuals enter the maturity period. There is no doubt that these two parameters can be designed to be adaptive. But, we set these two parameters as constants in our following experiments. Algorithm 1 gives the general framework for an individual behavior in the growth period.

Maturity period

When plants become mature in nature, they usually generate dozens to thousands of seeds rather than just one seed to ensure that some of the seeds can survive in the new environment. Not all generated seeds will succeed in growing as new plants, and most are eliminated and then die due to not adapting to the current environment. With the help of this reproductive phenomenon, we use a one-to-many generation relationship to increase population diversity: each individual which has entered its maturity period generates multiple seeds regardless of its fitness to achieve exploration in a wide area.

There are two key problems in the maturity period, one is how to generate many seed individuals and the other is how to disperse them as widely as possible.

Algorithm 1 The general framework for an individual searching during the growth period. GC, GR and x_i mean the growth cycle, the growth radius and the i-th individual in the current population, respectively.

```
count1 = 1.
while count1 ≤ GC do
    An individual generates an offspring individual for exploitation, x_next, using
    x_next = x_i + GR * rand(−1, 1) in each dimension.
    if the x_next is better than the x_i then
        The generated x_next individual replaces its parent x_i.
    else
        The x_i is kept as a parent individual.
    end if
    count1 = count1 + 1.
end while
```

Although plants have a variety of different ways to disperse their generated seeds, we also use two paramaters to decide the spread of seeds roughly: *dispersed direction* and *dispersed distance*. Once these two factors are determined, we can easily decide the new rooting positions for the generated seeds.

In complex natural environments, many plants depend on the cooperation of individuals to generate seeds. Corresponding to our proposed VEGE, we use the differential information between individuals to simulate the cooperation between plant individuals. Precisely, we randomly select two different individuals and make a difference vector between them. The first reason we use differential vectors is that it is easily calculated, and the second one is that any differential vector between any two individuals forms a share of the population distribution and therefore differential vectors contain a part of this information. A differential vector thus can represent the propagation direction of a seed individual. In addition, we also introduce a new parameter, *moving scaling*, to scale dispersed distance. It is a random generated number because, following the analogy for the spread of seeds in nature which is often influenced by many factors, such as water flow velocity, wind strength, and others. The new parameter, *moving scaling*, simulates these uncertain factors to increase the random search of the proposed VEGE algorithm.

Another problem that needs to be addressed is how to determine the number of seeds generated by each maturity individual. Actually, different kinds of plants generate different numbers of seeds, and even the number of generated seeds by different individuals of the same species may be different. As a first attempt, we do not simulate the various complex realities determining seed numbers but introduce only a new parameter, *number of seeds*, to control the number of seed individuals generated by each maturity individual. We treat each maturity individual equally and let them generate exactly the same number of seed individuals. Thus, we can easily embed the breeding model of real plants into the new VEGE. Algorithm 2 summarizes the general framework for an individual behavior during its maturity period.

Algorithm 2 The general framework of an individual search during the maturity period. MS is the moving scale; x_i is the i-th individual in the current population; x_1 and x_2 are two randomly selected different individuals and are different from x_i.

1: $count2 = 1$.
2: **while** $count2$ is less than the maximum number of generated seed individuals of x_i **do**
3: The individual generates a new seed individual, x_{seed}, using $x_{seed} = x_i + MS * (x_1 - x_2)$. Note that MS is randomly generated on each dimension.
4: Record the generated seed individual into a seed population for selecting the next generation.
5: $count2 = count2 + 1$.
6: **end while**

The last important problem is how to reasonably select individuals to survive to the next generation during the maturity period. Since an individual generates multiple seed individuals during its maturity period, the total number of generated seed individuals must exceed the current population size. Thus, we should use a reasonable method for selecting some potential individuals for the next generation, and also keep the balance between elite individuals and diversity. There are many famous selection methods, such as truncation selection, tournament selection, ranking selection, and others. Here, we use a greedy strategy to select the most promising individuals from a mixed population consisting of the current population and the seed population. All individuals are sorted according to their fitness, and the top PS individuals are selected to the next generation. Although this selection method can always keep the most outstanding individuals, it also has some risks, such as loss of diversity, becoming trapped at a local optimum area, and others. Thus, one of potential challenges is to develop other efficient selection methods. A feasible approach for solving this problem would be to model the more sophisticated and complex selection mechanisms employed by real plants.

We explain our inspirations from real plants in detail, and describes how to map them to the proposed VEGE. Each individual iterating the growth period and the maturity period can be considered as a life cycle. The newly selected individuals in the next generation begin a new round of the cycle from the growth period. Thus, our proposed VEGE algorithm iteratively executes this cycle towards finding the optimum. Algorithm 3 outlines the general framework of the whole of the proposed VEGE. Additionally, Fig 2 abstractly illustrates the growth cycle of a tomato, which may help to understand the optimization principle of our proposed VEGE. To keep consistent with most existing EC algorithms, we define that a cycle

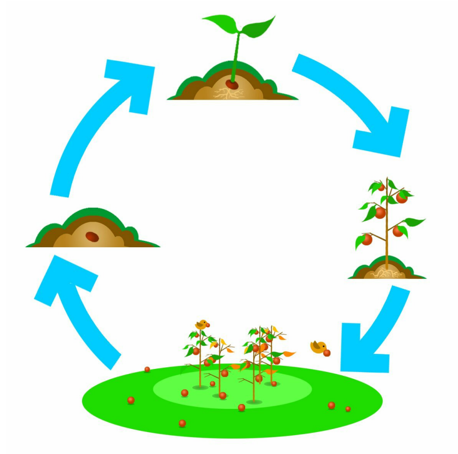

Fig. 2. The abstract growth state of plants. We divide it into two different periods subjectively, the growth period and the maturity period.

(including multiple growth operations and one maturity operation) does not equal one generation, but includes multiple generations. In other words, it is regarded as a generation update when all individuals perform growth or maturity operations. Thus, an individual does not perform both the growth period and the maturity period contiguously in one generation, but performs either a growth operation or a maturity operation.

Algorithm 3 The general framework of the proposed VEGE algorithm.

```
Initialize the population randomly.
Evaluate the population.
if an individual is in a growth period then
    for i = 1, . . . , population size do
        Perform Algorithm 1 for local search (exploitation).
    end for
else
    for i = 1, . . . , population size do
        Perform Algorithm 2 for wide search (exploration).
    end for
    Mix the current population and seed population, and apply a selection strat-
    egy to update the current population.
end if
Output the found optima.
```

4. Evaluation Experiments

Since the CEC2013 test suite [19] is devoted to the approaches, algorithms and techniques for solving real parameter single objective optimization, we select 28 benchmark functions with three dimensional settings, 2-dimensions (2-D), 10-D, and 30-D as a test bed. To analyze the performance of proposed VEGE, we run VEGE and three other algorithms (e.g., DE, PSO and EFWA) on these selected functions, and each function is run 30 times independently to educe contingency.

Table 1. Benchmark Functions: Uni=unimodal, Multi=multimodal, Comp.=Composition

No.	Types	Characteristics	Optimum fitness
F_1	Uni	Sphere function	−1400
F_2		Rotated high conditioned elliptic function	−1300
F_3		rotated Bent Cigar function	−1200
F_4		Rotated discus function	−1100
F_5		different powers function	−1000
F_6	Multi	Rotated Rosenbrock's function	−900
F_7		Rotated Schaffers function	−800
F_8		Rotated Ackley's function	−700
F_9		Rotated Weierstrass function	−600
F_{10}		Rotated Griewank's function	−500
F_{11}		Rastrigin's function	−400
F_{12}		Rotated Rastrigin's function	−300
F_{13}		Non-continuous rotated Rastrigin's function	−200
F_{14}		Schwefel's function	−100
F_{15}		Rotated Schwefel's function	100
F_{16}		Rotated Katsuura function	200
F_{17}		Lunacek BiRastrigin function	300
F_{18}		Rotated Lunacek BiRastrigin function	400
F_{19}		Expanded Griewank's plus Rosenbrock's function	500
F_{20}		Expanded Scaffer's F_6 function	600
F_{21}	Comp.	Composition Function 1 (n=5,Rotated)	700
F_{22}		Composition Function 2 (n=3,Unrotated)	800
F_{23}		Composition Function 3 (n=3,Rotated)	900
F_{24}		Composition Function 4 (n=3,Rotated)	1000
F_{25}		Composition Function 5 (n=3,Rotated)	1100
F_{26}		Composition Function 6 (n=5,Rotated)	1200
F_{27}		Composition Function 7 (n=5,Rotated)	1300
F_{28}		Composition Function 8 (n=5,Rotated)	1400

Table 1 summarizes their types, optimum fitness values, variable ranges, characteristics, and these landscapes include rotated, unimodal, shifted, and multi-modal characteristics. Table 2, Table 3, Table 4, and Table 5 show the parameter configuration of VEGE, DE, PSO, and EFWA, respectively. Since the ranges for all variables

Table 2. VEGE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	10
growth cycle GC	6
growth radius GR	a random number in [-1,1]
total seed individuals SI for 2-D, 10-D, and 30-D search	60
moving scaling MS	a random number in [-2,2]
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 3. DE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	60
scale factor F	0.8
crossover rate	0.9
DE operations	DE/rand/1/bin
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 4. PSO algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	60
inertia factor w	1
constant c_1 and c_2	1.49445, 1.49445
max. and min. speed V_{max} and V_{min}	2.0, -2.0
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

of all benchmark functions are in [-100, 100], we set the upper and lower limits of the growth radius, GR, to 1 and -1 for the growth operation. The maximum moving scaling, MS, is a random number generated in [-2, 2] for a wide search.

To ensure the fairness of the comparison as much as possible, we use the number of fitness calls rather than generations to terminate the evaluation experiments. Since we run each benchmark function with 30 trial runs in 3 different dimensional spaces, the Kruskal-Wallis test and Holm's multiple comparison test are applied to check whether there is a difference among all the algorithms at the stop condition, i.e. the maximum number of fitness calculations. Table 6 gives the result of these statistical tests.

Table 5. EFWA algorithm parameter settings.

# of fireworks for 2-D, 10-D and 30-D search	5
# of sparks m	50
# of Gauss mutation sparks,	5
constant parameters	$a = 0.04$ $b = 0.8$
maximum amplitude A_{max}	40
initial and final min. amplitude A_{init}, A_{final}	0.1, 0.05
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 6. Statistical test results of the Kruskal-Wallis test and Holm's multiple comparison for the average fitness values of 30 trial runs among 4 methods at the stop condition. ($A \gg B$) and ($A > B$) mean that A is significantly better than B with significance levels of 1% and 5%, respectively. ($A \approx B$) means that there is no significant difference among them.

	2-D	10-D	30-D
F_1	$VEGE \gg DE \approx PSO \gg EFWA$	$EFWA > VEGE \gg PSO \gg DE$	$EFWA \gg VEGE \gg PSO \gg DE$
F_2	$DE \approx VEGE \gg EFWA \approx PSO$	$PSO \approx VEGE \gg DE \approx EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_3	$VEGE > DE \gg PSO \approx EFWA$	$VEGE \gg PSO > DE \approx EFWA$	$VEGE \approx PSO \gg EFWA \gg DE$
F_4	$VEGE \approx DE > EFWA \approx PSO$	$PSO \approx DE \gg VEGE \gg EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_5	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg PSO \gg DE > EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_6	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE > PSO \approx EFWA \gg DE$
F_7	$VEGE \gg DE \approx PSO > EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$	$VEGE \approx DE \approx PSO \gg EFWA$
F_8	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx DE \approx EFWA$
F_9	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \approx PSO \gg EFWA > DE$	$PSO \approx VEGE \gg EFWA \gg DE$
F_{10}	$VEGE \gg DE \gg PSO \approx EFWA$	$VEGE \gg PSO > EFWA \gg DE$	$EFWA \gg PSO > VEGE \gg DE$
F_{11}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$
F_{12}	$VEGE \gg DE > PSO \approx EFWA$	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$
F_{13}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$VEGE \gg DE \gg PSO > EFWA$
F_{14}	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx EFWA \gg DE$	$EFWA \approx VEGE \approx PSO \gg DE$
F_{15}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \approx PSO \gg EFWA \gg DE$	$VEGE > EFWA \approx PSO \gg DE$
F_{16}	$EFWA \gg VEGE \approx PSO \approx DE$	$VEGE \approx PSO \gg EFWA \gg DE$	$EFWA \gg VEGE \gg DE \approx PSO$
F_{17}	$VEGE \approx DE \gg PSO \approx EFWA$	$VEGE \approx PSO \gg DE \gg EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$
F_{18}	$VEGE > DE \approx PSO \approx EFWA$	$VEGE \approx PSO \gg DE \approx EFWA$	$VEGE \approx EFWA \gg DE \gg PSO$
F_{19}	$VEGE > DE \gg PSO \approx EFWA$	$VEGE \gg PSO \approx EFWA \gg DE$	$PSO \approx VEGE > EFWA \gg DE$
F_{20}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE > DE \gg EFWA \gg PSO$
F_{21}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \approx EFWA \gg DE \approx PSO$	$EFWA \approx VEGE \gg DE \gg PSO$
F_{22}	$VEGE \gg PSO \approx DE > EFWA$	$VEGE > PSO \approx EFWA \gg DE$	$VEGE \approx EFWA \approx PSO \gg DE$
F_{23}	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg PSO > EFWA \approx DE$	$VEGE \gg PSO \approx EFWA \gg DE$
F_{24}	$VEGE \gg DE \approx PSO > EFWA$	$VEGE \gg DE > PSO \gg EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$
F_{25}	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$
F_{26}	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$DE > VEGE \gg PSO > EFWA$
F_{27}	$VEGE \approx PSO > DE > EFWA$	$VEGE \gg PSO > DE \approx EFWA$	$VEGE \gg PSO \gg DE \approx EFWA$
F_{28}	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \gg DE \gg PSO > EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$

5. Discussion

5.1. *Analysis of VEGE components*

We start our discussion with the new benefits of the proposed VEGE algorithm. The new VEGE is inspired by the life cycle of plants, and realizes different search capabilities by simulating two different life periods of plants. All individuals in the growth period are responsible for exploitation, where there is no communication between them and they compete independently to search for potential local areas. When the number of growth operations reaches the predefined maximum number of growths, all individuals enter the maturity period. Next, each individual generates many diverse seed individuals which are responsible for exploration through mutual cooperation between individuals. Then, a certain number of promising individuals are selected into the next generation, and they continue to reopen the growth trajectory like their parents.

The proposed VEGE emphasizes exploitation and exploration alternately to achieve a good balance, rather than focusing on one ability during a specific search period. It is the main difference from most existing EC algorithms that emphasize the exploration ability at the beginning and gradually transfer to the exploitation ability with convergence. Thanks to this new way of thinking, VEGE not only can guarantee that an individual has a strong local search ability, but also can use generated seed individuals to jump out from local areas to avoid premature convergence. Furthermore, the length of difference vectors changes adaptively according to the convergence of the population, and the difference vectors can keep variable distribution shape information of the population. Overall, VEGE can maintain the balance between exploitation and exploration well, and it changes adaptively with the convergence of population. Owing to the iteration of the proposed VEGE, that is, repeatedly simulating the growth and maturity of plants, the fitness of all individuals can be gradually improved. Finally, the population has a high probability of converging to the global optimum.

Secondly, we would like to point out that the performance of our proposal is mainly influenced by three operations: growth, maturity and selection. The growth operation controls the local search ability of individuals using two introduced parameters, GR and GC, to generate better offspring in local areas. We adopt a one-to-one growth relationship to achieve the growth operation in this paper, where one-to-one growth means that an individual grows in only one direction at a time. Actually, the growth of plants in nature is diverse, and, for example, it is common in nature to grow multiple branches on a tree trunk. Corresponding to the proposed VEGE, this would mean that multiple directions would be simultaneously explored in growth. Although this would increase the cost of fitness evaluation, it may deepen the exploration of the local fitness landscape and avoid falling into a local optima. Thus, developing an efficient local search mechanism is the main task of this operation. We will continue to observe the behaviour of plants in nature and implement more promising growth operations.

The maturity operation is designed to increase diversity and can ease the pressure of local search by using two introduced parameters, MS and SI. This operation generates diverse seed individuals and spreads them to explore as many unexplored areas as possible. In this paper, we have just roughly simulated the simplest implementation. Actually, plants in nature have multiple ways to produce seeds such as sexual reproduction and asexual reproduction. It is now possible to achieve hybridization between different species through artificial means. There are also many ways for plants to spread their seeds. Thus, how to generate diverse potential seeds and spread them is the main tasks of this operation. We are going to focus on developing new efficient strategies to generate seed individuals and spread seed individuals which are inspired by natural plants.

The last, but also important, operation is the selection operation. Here, we use fitness ranking to select individuals for the next generation. Although this operation can maintain faster convergence speed, it also has the risk of rapidly losing population diversity. To overcome this limitation, we intend to develop a new adaptive selection operation based on the current optimization situation in our forthcoming works. For example, diversity selection can be appropriately emphasized to avoid over-concentration of the population and stagnation. Or, different selection methods can be used according to the dynamics of population diversity. In short, how to maintain the balance between the three operations to achieve better performance is a valuable topic that needs continuous research. Too much emphasis on one of them may weaken the performance of our proposal, by causing the population to stagnate or fall into local areas and slow convergence.

Thirdly, we discuss the potential of our proposed VGEG algorithm. We only simulated the rough survival mechanisms of plants to propose an optimization framework. In fact, we have not yet thoroughly researched the many plants in nature, for which there must be many known and unknown survival mechanisms. It is a huge challenge to further enhance the performance of our proposed algorithm by introducing more novel and efficient mechanisms observed from natural plants. For example, we can develop some variants of the currently proposed version, such as an *adaptive version*, a *parallel version*, and a *hybrid version*.

Adaptive version: All individuals of the proposed VEGE algorithm perform two different periods alternately and generate lots of seed individuals. In nature, many plants have different growth conditions depending on their living environment, i.e., plants will grow better on fertile land. Different individuals may generate different numbers of seeds even though they are the same species. Inspired by this phenomenon, we can consider their fitness to be analogous to their growth environments. It is possible to develop an adaptive version of VEGE, where each individual has a different generative cycle to reach maturity and generates a different number of seed individuals according to its own fitness. Usually, an individual with higher fitness may be closer to the global optimum, such an individual should be given a longer growth period and generate more seed individuals.

Parallel version: Since individuals in the next generation are selected after seed individuals from all parent individuals are collected, parallel computing cannot be applied to this process. We can design an independent selection mechanism to maintain multiple different small groups, where each group consists of an individual and all seed individuals generated by it. Further, we can introduce an exclusion mechanism to ensure that different small groups can locate multiple different extremes for solving multimodal optimization problems. Although each seed individual is collaboratively generated by multiple individuals, it is possible to implement parallel computing thanks to the independent selection mechanism.

Hybrid version: The proposed algorithm is flexible and easy to combine with other EC algorithms. In other words, different EC search operations can replace a part of the VEGE strategies, and the replaced strategy(es) can be applied to the growth period and/or the maturity period to obtain the advantages of different EC algorithms. According to our design philosophy, all individuals focus on their exploitation ability in the growth period, while their exploration ability is emphasized during the maturity period. Thus, we can embed one or two different EC algorithms into our proposed optimization framework by setting different parameters to achieve two different search abilities. Although the overall framework of our proposal has not changed in this hybrid version, the specific implementation details may change. For example, when we integrate DE or PSO into our framework, we naturally introduce the parameters of the corresponding algorithm. So, these works need to be further refined deeply in the near future.

Finally, we apply the Kruskal-Wallis test and Holm's multiple comparison test at the stop condition to compare the performance of four EC algorithms. The results of statistical test shown in Table 6 confirm that our proposed VEGE algorithm is effective and has potential compared to the other three well-known algorithms. We noticed that our proposal performs the best in all algorithms except F_{16} in 2-D where it is second best. As the dimension increases, VEGE is not ranked as the best in F_1 and F_4 in 10-D but is also not ranked the worst. When the dimension is increased to 30, the VEGE performance is further reduced, but it is still the best overall. In addition, DE and PSO win or lose each other on different dimensions and types of problems, while EFWA basically has the worst performance, especially for low dimensional multimodal problems.

Generally speaking, the complexity of optimization problems becomes more complicated as the dimensionality increases. Suppose the population size of the proposed VEGE is set to PS, the total number of possible differential vectors is $(PS-1)*(PS-2)$. Specifically, the number of differential vectors that may be used in each generation is 72 because the PS is set to 10 in our experiments. Thus, such parameter settings may be sufficient for a low-dimensional space but insufficient for a high-dimensional space. To solve the above-mentioned problem of insufficient differential vectors in the high-dimensional space, one of the most intuitive methods is to increase the initial population, but this will increase the cost of fitness evaluation. Another feasible option is to use outstanding individuals from past generations to provide sufficient differential vectors, such as with the optional external archive [11] and an individual pool [20]. In addition, the use of greedy selection may further accelerate the loss of diversity in the population. This indicates that a small of population size is not recommended for high-dimensional problems. Thus, it is necessary to further investigate the relationship between spatial dimensions and population size.

5.2. *Analysis of VEGE parameter settings*

Like other EC algorithms, we also use a fixed population size, PS, to represent a set of all current individuals, where each individual is a candidate solution. Due to the use of differential vectors, the population size should not be less than 3 in our proposed optimization framework.

In addition, we have introduced four new special parameters. Two of the four parameters are random numbers, and only GC and MS need to be set carefully according to the characteristics of the optimization problems. In the following, we describe their goals and roles respectively.

Growth Cycle (GC): Controls the number of local searches. Individuals enter their maturity period after they perform local searches. The larger the GC is, the stronger the ability for an individual to explore locally. However, too large GC makes the population fall into local areas easily, whereas populations with too small GC may not be able to search deeply. Thus, appropriate parameter values help the VEGE algorithm to maintain good performance. In our future work, we will investigate the relationship between the settings and convergence periods, and summarize the relationship with optimization problem characteristics, i.e., how to set the size of the growth cycle for what kind of characteristics.

Growth Radius (GR): A randomly generated number which controls the radius of the local search of an individual. A random number for GR can increase the randomness of the VEGE algorithm to adapt to different optimization problems rather than a fixed number. If the GC is set to the boundary, it evolves into a random search of the whole search space. Setting GR too large or too small is not conducive to growth. Certainly, the maximum radius should be determined based on the optimization problem. In our evaluation experiments, we set GR in $[-1, 1]$, which means the radius for a local search of each individual does not exceed 1.

Number of Seeds (NS): This is the number of seed individuals generated by each individual during its maturity period. The larger the NS is, the more seed individuals are generated. Although many plants in nature may generate thousands of seeds to survive, it is unwise for the VEGE algorithm to generate too many seed individuals because it uses a lot of fitness evaluation. However, generating too few seed individuals will prevent an extensive search being achieved, and thus the search cannot jump out from a local area. Therefore, we must further study and observe the generation mechanism of vegetation and implement them into our proposal in a more rational way. We made all individuals equally generate the same number of seed individuals in this paper. In fact, one approach would be to make each individual generate a different number of seed individuals while keeping the same total number of generated seed individuals.

Moving Scaling (MS): A randomly generated number which controls the direction and scale length of the difference vector. In our evaluation experiments, the maximum size of the differential vector does not exceed two times itself. We use this parameter value because seeds are affected by many factors during the propagation process. It is expected to increase the diversity among seed individuals even when they are diffused in the same direction. Naturally, this parameter alone cannot simulate the complex propagation process of natural plants. Thus, we are considering how to improve it to achieve more effective propagation.

5.3. *Potential and future topics*

In our first attempt, we proposed a new optimization algorithm by extracting some common mechanisms from life cycle of plants in nature, then simulated them repeatedly to find the global optimum. There must be more effective and elaborate mechanisms waiting to be discovered and integrated into our framework to further improve its performance. Here, we list some future topics, although the possibilities are not limited by this list alone.

(1) Analysis of the effect of the two periods of an individual on the performance. Further, how to properly share the proportion of these two periods.
(2) Analysis of the relationship between parameter settings and optimization problem characteristics. Develop an adaptive version.
(3) Extending the algorithm to other application scenarios, such as multimodal optimization, multi-objective optimization, large scale problems, dynamic optimization and multi-objective and many-objective optimization.
(4) Development of hybrid algorithms to aggregate the advantages of different EC algorithms.
(5) Analysis of the convergence characteristics of our proposal and the characteristics of individual trajectories.
(6) Applications of the VEGE algorithm to real world tasks.

6. Conclusion

We proposed a novel population-based VEGE algorithm to balance exploitation and exploration by simulating the life cycle of plants. The controlled experiments confirmed that VEGE is effective and promising in comparison with other popular EC algorithms regardless of the speed of convergence or accuracy. Besides, it also shows that switching exploration and exploitation repeatedly is also a potential search idea.

In our future work, we will further analyze the effectiveness of VEGE theoretically, improve the current version to develop more powerful versions, and expand its usage scenarios.

References

1. R. Storn and K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization.* **11**(4) (1997) 341–359.
2. J. Kennedy and R. Eberhart, Particle swarm optimization, *IEEE International Conference on Neural Networks.* (1995) 1942–1948.
3. Y. Tan and YC. Zhu, Fireworks algorithm for optimization, *The First international conference on Advances in Swarm Intelligence.* (2010) 355–364.
4. PM. Passino, Biomimicry of Bacterial Foraging for Distributed Optimization and Control, *IEEE Control Systems.* **22**(3) (2002) 52–67.
5. K. Dervis and B. Bahriye, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm, *Journal of Global Optimization.* **39**(3) (2007) 459–471.
6. XS. Yang and AH. Gandomi, Bat algorithm: A novel approach for global engineering optimization, *Engineering Computations.* **29**(5) (2010) 464–483.

7. AH. Gandomi and AH. Alavi, Krill herd: A new bio-inspired optimization algorithm, *Communications in Nonlinear Science and Numerical Simulation.* **17**(12) (2012) 4831–4845.
8. RJ. Mullen, D. Monekosso, S. Barman and P. Remagnino, A review of ant algorithms, *Expert Systems with Applications.* **36**(6) (2009) 9608–9617.
9. J. Yu and H. Takagi, Acceleration for fireworks algorithm based on amplitude reduction strategy and local optima-based selection strategy, *eighth International Conference on Swarm Intelligence.* (2017) 477–484.
10. J. Yu, Y. Pei and H. Takagi, Competitive Strategies for Differential Evolution, *The 2018 IEEE International Conference on Systems, Man, and Cybernetics.* (2018) 268–273.
11. JQ. Zhang and AC. Sanderson, JADE: Adaptive differential evolution with optional external archive, *IEEE Transactions on Evolutionary Computation.* **13**(5) (2009) 945–958.
12. SQ. Zheng, A. Janecek and Y. Tan, Enhanced fireworks algorithm, *2013 IEEE Congress on Evolutionary Computation.* (2013) 2069–2077.
13. YC. Jin, M. Olhofer and B. Sendhoff, A Framework for Evolutionary Optimization with Approximate Fitness Functions, *IEEE Transactions on Evolutionary Computation.* **6**(5) (2002) 481–494.
14. YC. Jin, A comprehensive survey of fitness approximation in evolutionary computation, *Soft Computing.* **9**(1) (2005) 3–12.
15. N. Murata, R. Nishii, H. Takagi and Y. Pei, Analytical estimation of the convergence point of populations, *2015 IEEE Congress on Evolutionary Computation.* (2015) 2619–2624.
16. J. Yu, Y. Pei and H. Takagi, Accelerating Evolutionary Computation Using Estimated Convergence Point, *2016 IEEE Congress on Evolutionary Computation.* (2016) 1438–1444.
17. P. Kopciewicz and S. Łukasik, Exploiting flower constancy in flower pollination algorithm: improved biotic flower pollination algorithm and its experimental evaluation, *Neural Computing and Applications.* (2019).
18. X. Li, S. Han, L. Zhao, C. Gong and X. Liu, New Dandelion Algorithm Optimizes Extreme Learning Machine for Biomedical Classification Problems, *Computational Intelligence and Neuroscience.* **2017** (2017) 1–13.
19. JJ. Liang, BY. Qu, PN. Suganthan and GHD. Alfredo, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization (2013).
20. J. Yu and H. Takagi, Estimation of the Convergence Points of a Population Using an Individual Pool, *IEEE 10th International Workshop on Computational Intelligence and Applications.* (2017) 67–72.