IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 3, 2019, accepted November 26, 2019, date of current version December 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2956839

Enhanced Moth Search Algorithm for the Set-Union Knapsack Problems

YANHONG FENG[1,2], JIAO-HONG YI[3], AND GAI-GE WANG[4]
[1]School of Information Engineering, Hebei GEO University, Shijiazhuang 050031, China
[2]Hebei Center for Ecological and Environmental Geology Research, Hebei GEO University, Shijiazhuang 050031, China
[3]School of Information and Control Engineering, Qingdao University of Technology, Qingdao 266520, China
[4]College of Information Science and Engineering, Ocean University of China, Qingdao 266100, China

Corresponding author: Jiao-Hong Yi (yijiaohong@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61806069, Grant 41576011, Grant U1706218, Grant 41706010, and Grant 61503165.

ABSTRACT As an important and novel model with multitudinous practical applications, the set-union knapsack problem (SUKP) is a challenging issue in combinatorial optimization. In this paper, we present an enhanced moth search algorithm (EMS) for solving SUKP, which introduces an enhanced interaction operator (EIO) by integrating differential mutation into the global harmony search and then Lévy flight is replaced by EIO. Comparative experimental results, which were conducted on three types of 30 popular SUKP benchmark instances, demonstrate that EMS algorithm is superior to or competitive with the other state-of-the-art metaheuristic algorithm. In particular, EMS reaches the best-known solutions for the great majority of test instances and improves the best-known solutions for six instances. Two critical ingredients of EIO is investigated to confirm their impact on the performance of EMS. The results show that both components have an important role in improving the performance of EMS.

INDEX TERMS Differential mutation, global harmony search, moth search algorithm, set-union knapsack problem.

I. INTRODUCTION

The classical knapsack problem (KP) [1] is still one of the most challenging problems in combinatorial optimization. Since KP is an NP-hard problem and has many practical applications in reality, new varieties are emerging in recent years.

In this paper we consider an extension of KP, namely, the set-union knapsack problem (SUKP) [2], [3], which is a popular binary optimization problem with constraints. Although SUKP was proposed long ago, it has recently attracted more and more researchers to study this issue deeply, because it has been proved that there are many important applications in specific fields, such as public key prototype [4], data stream compression [5], and financial decision making [3]. In addition, SUKP is more complicated and challenging than the classical 0-1 KP. The classical 0-1 KP is characterized by one item with a profit and a weight. Nevertheless, there are a set of items and a set of elements in SUKP, in which each item has a profit and each element has a weight. Particularly, a set of items is required to pack into the knapsack in SUKP.

The associate editor coordinating the review of this manuscript and approving it for publication was Kai Li.

In view of its important application in practice and its theoretical research value, SUKP has attracted much attention in the community. According to the existing literature, the method of solving SUKP problem can be categorized into three groups based on their natures: (1) exact algorithm (2) approximate algorithm, and (3) heuristic approach. Here, we are mainly concerned with the most representative research work. The representative exact approach is dynamic programming (DP) algorithm. SUKP has been first introduced in the literature by Goldschmidt *et al.* with DP [2]. However, the high time complexity makes it difficult to apply in the real-world applications. Later, an approximation algorithm A-SUKP for the SUKP was presented by Arulselvan and some important proofs were provided [3]. Afterwards, Taylor designed several approximation strategies for SUKP and related problems [6]. Nevertheless, a satisfactory approximate solution cannot be obtained by this kind of method when facing large-scale SUKP instances.

In order to escape from the trouble when facing high-dimensional SUKP instances with exact algorithms and approximate algorithms, various heuristic methods have been proposed to solve SUKP. Recently a binary artificial bee colony algorithm (BABC) for SUKP was given by

He *et al.* [7]. Meanwhile, SUKP has also been addressed by Baykasoglu *et al.* [8], Ozsoydan and Baykasoglu [9]. The authors presented an effective binary swarm intelligence technique which is based on genetic algorithm (GA) [10] and particle swarm optimization (PSO) [11]. Lately, Baykasoglu *et al.* addressed SUKP by using binary weighted superposition attraction algorithm (WSA) [8]. Indeed, in the light of NP-hard characteristic, it is significant to investigate SUKP intensively, especially applying more novel metaheuristic algorithm.

Three metaheuristic algorithms based on the behavior of moths in nature have been proposed. The main inspiration of moth-flame optimization (MFO) [12] is transverse orientation navigation method of moth. Inspired by the orientation of moths towards moonlight, moth swarm algorithm (MSA) [13] is proposed. In MSA, moth swarm consists of three groups of moths according to their mission during flight, namely, pathfinders, prospectors, and onlookers. As recently introduced by Wang [14], moth search (MS) takes inspiration from the phototaxis and Lévy flights of the moths in nature. Similar to MSA, moth swarm in MS is divided into two subpopulations based on the flight mode to light source. Owing to its relative novelty, the related literature includes studies of MS are few. Feng *et al.* employed a binary MS algorithm (BMS) to solve discounted {0-1} knapsack problem (DKP) [15]. Although BMS can effectively solve DKP, whether MS can perform well in other complicated combinatorial optimization problems like SUKP, still needs to be studied.

This is the motivation for this work, in which an enhanced interaction operator (EIO) is specially designed to replace the Lévy flight operator in original MS and then an enhanced moth search algorithm (EMS) is proposed. The basic frame of EIO is embedding the mutation operator (MO) of differential evolution (DE) [16], [17] in global harmony search algorithm (GHS) [18] to make full use of their advantages. Concretely speaking, the effective combination of GHS and MO can enhance the ability of information interaction among individuals.

The main contributions of this work can be summarized as follows.

- For the first time, we investigate an enhanced MS algorithm for solving the SUKP. We replace Lévy flight operator with the combination of GHS and differential mutation operator that is capable of ensuring an effective diversification and intensification within the search space.
- We provide experimental results on 30 commonly used SUKP instances and compare the results of EMS with those of state-of-the-art SUKP algorithms in the literature. It should be noted especially that we update the best-known results for 6 SUKP instances.

The remainder of this paper is organized as follows. Section II formally defines SUKP problem. Section III gives a brief overview of the original MS algorithm. Section IV provides the detailed introduction of EMS to solve SUKP. Section V makes the comparison and analysis of the experimental results. Section VI draws conclusion of the present work and gives perspectives for future studies.

II. SET-UNION KNAPSACK PROBLEM

SUKP is essentially an extension of the 0-1 knapsack problem (0-1 KP) [19] by assigning some elements to each item. Formally, this problem can be defined as follows.

Given a SUKP instance with a set $U = \{1, 2, 3, \ldots, n\}$ of n elements and a set $S = \{1, 2, 3, \ldots, m\}$ of m items. Moreover, each item $i \in S$ $(i = 1, 2, \ldots, m)$ corresponds to a subset U_i of elements, and $U_i \neq \emptyset \wedge U_i \subset U \wedge \bigcup_{i=1}^{m} U_i = U$. Each item has non-negative profit p_i $(i = 1, 2, \ldots, m)$ and each of the elements has non-negative weight w_j $(j = 1, 2, \ldots, n)$. For an arbitrary subset $A \subseteq S$, total weight and total profit of subset A is defined as $W(A) = \sum_{j \in \bigcup_{i \in A} U_i} w_j$ and $P(A) = \sum_{i \in A} p_i$, respectively. Then the SUKP is to select a subset of items S^* such that $W(S^*) \leq C$ where C is the capacity limit of knapsack, while maximizing the total profit $P(S^*)$.

Then the mathematical model of SUKP can be formulated as follows:

$$Max\ P(A) = \sum_{i \in A} p_i \tag{1}$$

$$s.t.\ W(A) = \sum_{j \in \bigcup_{i \in A} U_i} w_j \leq C, \quad A \subseteq S \tag{2}$$

In order to solve SUKP easily by using metaheuristic algorithm, an integer programming model is proposed by He *et al.* [7]. The new mathematical model can be defined as follows:

$$Max\ f(Y) = \sum_{i=1}^{m} y_i p_i \tag{3}$$

$$s.t.\ W(A_Y) = \sum_{j \in \bigcup_{i \in A_Y} U_i} w_j \leq C \tag{4}$$

Here, any candidate solution $\mathbf{Y}$ can be represented by an m-dimensional binary vector $\mathbf{Y} = (y_1, y_2, y_3, \ldots, y_m)$, $A_y = \{i \,|y_i \in \mathbf{Y}, y_i = 1, 1 \leq i \leq m\} \subseteq S$ such that $y_i = 1$ if and only if $i \in A_Y$. Particularly, feasible solution $\mathbf{Y}$ satisfies Eq. (4), and infeasible solution otherwise.

III. MOTH SEARCH ALGORITHM

The MS was originally developed to solve continuous numerical optimization problem [14]. MS is a swarm-based nature-inspired metaheuristic algorithm. However, MS differs from other state-of-the-art methods including genetic algorithm (GA) [10], differential evolution algorithm (DE) [16], [17], particle swarm optimization (PSO) [11], and harmony search (HS) algorithm [20], [21]. In MS, there are two subpopulations, namely, subpopulation1 and subpopulation2. Therefore, MS searches the problem space by moving the moth individuals via Lévy flights in subpopulation1 and fly straightly in subpopulation2, respectively. The procedure of MS is illustrated in Figure 1.

Moth search algorithm

Begin

Step 1: Initialization.

Randomly initialize a population of *NP moth* individuals randomly.

Divide the whole population into two subpopulations with equal size according to the fitness, namely, subpopulation1 and subpopulation2

Step 2: While (termination condition is not satisfied) do

Update subpopulation1 by Eqs. (5) - (7) //Lévy flight phase

Update subpopulation2 by Eqs. (8) - (9) //Fly straightly phase

Recombine the newly-generated subpopulations.

Sort the population by fitness.

Divide the whole population into subpopulation1 and subpopulation2.

end while

Step 3: Return the best individual found so far.

End.

FIGURE 1. Moth search algorithm.

According to Figure 1, the main formulas of Lévy flights (Eqs. (5) - (7)) and fly straightly (Eqs. (8) - (9)) are described as follows.

$$x_i^{t+1} = x_i^t + \alpha L(s) \tag{5}$$

$$\alpha = S_{max}/t^2 \tag{6}$$

$$L(s) = \frac{(\beta-1)\Gamma(\beta-1)\sin(\frac{\pi(\beta-1)}{2})}{\pi s^{\beta}} \tag{7}$$

where x_i^{t+1} and x_i^t are respectively the position of moth i at generation $t+1$ and t. α refers to the scale factor based on the relevant problem. S_{max} is the max walk step and its value takes 1.0 in this paper. $L(s)$ represents the step drawn from Lévy flights and $\Gamma(x)$ is the gamma function. Parameter β is set to 1.5 for our experiments.

$$x_i^{t+1} = \lambda \times (x_i^t + \varphi \times (x_{best}^t - x_i^t)) \tag{8}$$

$$x_i^{t+1} = \lambda \times (x_i^t + 1/\varphi \times (x_{best}^t - x_i^t)) \tag{9}$$

where scale factor λ is set to a random number drawn by the standard uniform distribution and φ is an acceleration factor that its value equals golden ration. x_{best}^t is the best individual at generation t. Note that moth individual i updates the position with Eq. (8) or Eq. (9) with equal probability. These two update processes are shown in Figure 2. In Figure 2, old, best, new1, and new2 indicate the original, best, updated position by Eq. (8), and updated position by Eq. (9) for individual i, respectively. As can be seen in Figure 2, the control coefficient φ in Eq. (8) and $1/\varphi$ in Eq. (9) can well balance the individual to search in the two relative directions of the global best position.

IV. ENHANCED MS ALGORITHM FOR SUKP

To describe the enhanced moth search algorithm for the SUKP, we start with the solution representation. Then the constraint handling method is explained. Followed by a detailed introduction to EMS algorithm. Finally, we outline the framework of EMS for solving SUKP.

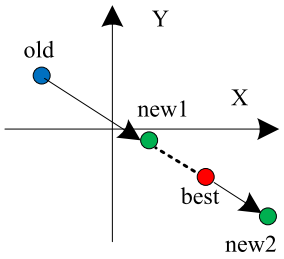

FIGURE 2. The two new individual generations of rectilinear flight.

A. SOLUTION REPRESENTATION

As mentioned earlier, given a SUKP instance with a set $S = \{1, 2, 3, \ldots, m\}$ of m items, any candidate solution $\mathbf{Y}$ can be expressed as an m-dimensional binary vector $\mathbf{Y} = (y_1, y_2, y_3, \ldots, y_m)$ such that $y_j = 1$ if the item j is selected, and $y_j = 0$ otherwise.

Since MS algorithm was originally proposed to solve numerical optimization problems, two operators perform the optimization process in a continuous search space. Nevertheless, SUKP belongs to constrained discrete optimization problem. In this work, we specifically employ two vectors $\mathbf{X}$ and $\mathbf{Y}$ to represent each moth individual, namely, $moth = <\mathbf{X}, \mathbf{Y}>$.

Let Ω^C be the set of all m-dimensional real-valued vectors, i.e.,

$$\Omega^C = \{\mathbf{X} \,|\mathbf{X} \in [-a, a]^m\} \tag{10}$$

Let Ω^D be the set of all m-dimensional binary vectors, i.e.,

$$\Omega^D = \{\mathbf{Y} \,|\mathbf{Y} \in \{0, 1\}^m\} \tag{11}$$

where $a = 5$ in this work. The evolution of MS in continuous space is still dependent on real-valued vector X. Meanwhile, the mapping of $\mathbf{X}$ to $\mathbf{Y}$ is implemented by using transfer function [22], [23]. In the present work, a simple and effective transfer function [24] $g : R^m \to \{0, 1\}^m$ is defined as follows:

$$y_i = \begin{cases} 1, & if\ x_i \geq 0 \\ 0, & else \end{cases} \tag{12}$$

Finally, the objective function f(Y) is defined as follows to evaluate the quality of any candidate solution.

$$f(Y) = \sum_{i=1}^{m} y_i p_i \tag{13}$$

B. CONSTRAINT HANDLING

Obviously, Ω^D consists of two parts: Feasible solutions and infeasible solutions. Therefore, the strategy of dealing with infeasible solutions effectively is one of the most important issues in solving SUKP. A repairing and optimization algorithm (named S-GROA) is specially proposed by He *et al.* [7]

S-GROA algorithm for SUKP

Begin

Step 1: Input: the candidate solution $\mathbf{Y} = \{y_1, y_2, ..., y_m\} \in \{0,1\}^m$.

Step 2: Preprocessing. The one-dimensional array H[1…*m*].

Step 3: Initialization. The *m*-dimensional binary vector $\mathbf{Z} = [0, 0, \ldots, 0]$.

Step 4: Greedy repair.

For $i = 1$ **to** m **do**

If $(y_{H[i]} = 1 \quad and \quad W(A_Z \cup \{H[i]\}) \le C)$

$Z_{H[i]} = 1 \quad and \quad A_Z = A_Z \cup \{H[i]\}$.

End if

End for

$\mathbf{Y} \leftarrow \mathbf{Z}$.

Step 5: Optimization.

For $i = 1$ **to** m **do**

If $(y_{H[i]} = 0 \quad and \quad W(A_Z \cup \{H[i]\}) \le C)$

$y_{H[i]} = 1 \quad and \quad A_Y = A_Y \cup \{H[i]\}$.

End if

End for

Step 6: Output: $\mathbf{Y} = \{y_1, y_2, ..., y_m\} \quad and \quad f(\mathbf{Y})$

End.

FIGURE 3. S-GROA algorithm for SUKP.

for this purpose and is employed in this work. The preprocessing phase of S-GROA can be summarized as follows:

1) Compute the frequency d_j of the element $j (j = 1, 2, 3, \ldots, n)$ in the subsets $U1, U2, U3, \ldots, U_m$.
2) Calculate the unit weight R_i of the item i $(i = 1, 2, 3, \ldots, m)$.

$$R_i = \sum\nolimits_{j \in U_i} (w_j / d_j) \tag{14}$$

3) Record the profit density of each item in S according to PD_i.

$$PD_i = p_i / R_i \quad (i = 1, 2, 3, \ldots, m) \tag{15}$$

4) Sort all the items in a non-ascending order based on PD_i $(i = 1, 2, 3, \ldots, m)$ and then the index value recorded in an array H[1…*m*].
5) Define a term $A_{\mathbf{Y}} = \{U_i \,|\, y_i \in \mathbf{Y} \wedge y_i = 1, 1 \le i \le m\}$ for any binary vector $\mathbf{Y} = [y_1, y_2, \ldots, y_m] \in \{0, 1\}^m$.

The pseudocode of S-GROA is outlined in Figure 3.

From Figure 3, one observes that S-GROA algorithm is composed of two phases. The first phase repairs only infeasible solutions by eliminating some of the violating items. After all the solutions have become feasible, the second phase optimize the remaining items by packing suitable items into knapsack with the aim of further utilizing the remaining capacity.

C. ENHANCED MS ALGORITHM

As one of the main operators of MS, Lévy flight [25], [26] should play a vital role in the optimization ability of the algorithm. However, previous works indicate that Lévy flights operator has relatively weak influence compared to fly straightly operator [15].

As a swarm intelligence algorithm [27], [28], the performance of the MS depends heavily on the interaction or information interchange among these individuals. However, the moths in subpopulation1 only fly around the global best individual in the form of Lévy flights. Clearly, there is a lack of sufficient information inheritance and interchange among the moths.

In this paper, an enhanced interaction operator (EIO) based on GHS [18] and mutation operator of DE [16], [17] was specially designed. The main consideration in this new operator is making full use of information sharing among individuals so that the exploration capability of the EMS can be improved. Additionally, Lévy flights have the characteristics of random flights, which do not fully reflect the mode of social cooperation. However, for differential mutation (DE/best/1bin), the best individual and any two individuals are selected to generate the mutation individual based on the social cooperation strategy. Meanwhile, GHS is a simple and effective heuristic global search algorithm than the original HS. Embedding the mutation operator of DE into GHS not only enhances the convergence of EMS, but also prevents the algorithm from falling into local optimum.

Consequently, Lévy flight operator was replaced by an enhanced interaction operator, and an enhanced MS algorithm (EMS) was proposed.

1) THE GLOBAL-BEST HARMONY SEARCH

In brief, harmony search (HS) [21], [29] is an efficient optimization metaheuristic inspired by the music improvisation process. In the last years, HS has attracted many researchers because of its excellent performance in solving various problems [30], [31].

Here, we adopt an efficient global-best harmony search (GHS) [18], where memory consideration, pitch adjustment, and random selection are calculated as follows:

$$x_i^k = x_j^k \; (j \sim U(1, \ldots, HMS)) \; if \quad rand \le HMCR \tag{16}$$

$$x_i^k = x_{best}^j \; if \; rand \le HMCR \wedge rand \le PAR \wedge j \sim U(1, N) \tag{17}$$

$$x_i^k = LB^k + rand \times (UB^k - LB^k) \; if \; rand > HMCR \tag{18}$$

where *HMCR*, *PAR*, *HMS*, and N are harmony memory considering rate, pitch adjusting rate, harmony memory size, and problem dimension, respectively. x_i^k, x_j^k are respectively the k th element of individual i, individual j. The jth of global best individual represents by x_{best}^j. *Rand* is a function generating a random number uniformly distributed in (0, 1). LB^k and UB^k are the lower and upper limits for the kth.

2) THE DIFFERENTIAL EVOLUTION

Differential evolution (DE) [17], [32] is undoubtedly one of the most promising stochastic real-parameter optimization algorithms. DE searches for a global optimum solution through three main stages: mutation with difference vectors, cross, and selection. Thereinto, mutation operator is the main

Enhanced interaction operator

Begin
  for i = 1 to $NP1$ **do**
    for j = 1 to m **do**
      if $rand(0, 1) < HMCR$ **then**
        $x_{i,j} = x_{k,j}$, where $k \sim U[1, 2,..., NP]$
        if $rand(0, 1) < PAR$ **then**
        $x_{i,j} = x_{best,j}$, where *best* is the index of global best individual
        else
          $x_i = x_{best} + \lambda(x_{r1} - x_{r2}) + F(x_{r3} - x_{r4})$
        end if
      else
        $x_{i,j} = rand(0,1) \times (x_{max} - x_{min}) + x_{min}$
      end if
    end for
  end for
End.

FIGURE 4. Enhanced interaction operator.

The framework of EMS for SUKP

Begin
Step 1: Sorting.
  Sort all items in descending order on the basis of PD_i ($0 \le i \le m$), and item numbers are stored in an array H[1…m].
Step 2: initialization.
  Randomly initialize a population P of NP moths.
  Divide the whole population into subpopulation1 and subpopulation2.
  Perform repair and optimization with S-GROA.
Step 3: while (termination condition is not satisfied) **do**
  Use enhanced interaction operator to update subpopulation1.
  Use fly straightly operator to update subpopulation2.
  Use S-GROA to repair infeasible solutions and optimize feasible solutions.
  Evaluate the fitness of the population.
  Recombine the two newly-generated subpopulations.
  Sort the population by fitness.
  Divide the whole population into two subpopulations.
end while
Step 4: Return the best individual found so far.
End.

FIGURE 5. The framework of EMS for SUKP.

component of DE. In this paper, DE/best/2/bin model is used as follows:

$$x_i = x_{best} + \lambda(x_{r1} - x_{r2}) + F(x_{r3} - x_{r4}) \tag{19}$$

where x_{r1}, x_{r2}, x_{r3}, and x_{r4} are mutually exclusive individual randomly chosen from subpopulation1 and they are also different from the base vector x_i. The vector x_{best} is the best individual of the entire population.

3) THE ENHANCED INTERACTION OPERATOR

Compared with original HS, GHS adds a social dimension to the HS which stems from PSO. While differential mutation can be regarded as self-cognition. Consequently, the effective combination of GHS and differential mutation can achieve a much better balance of exploration and exploitation than MS. Intuitively, this modification can enable EMS to solve continuous optimization problems and discrete optimization problems effectively. Then the primary steps of enhanced interaction operator are illustrated in Figure 4.

D. EMS FRAMEWORK FOR SUKP

After the special design of each component, the framework of EMS for solving SUKP is illustrated in Figure 5. It can be seen that the evolutionary process consists of three main stages if initialization stage is excluded. First stage, generating new individuals among subpopulation1 by employing enhanced interaction operator. Second stage, updating individuals of subpopulation2 by performing flight straightly operator. Third stage, using S-GROA to repair infeasible solutions and then optimize all feasible solutions.

E. COMPUTATIONAL COMPLEXITY OF THE EMS ALGORITHM

Computational complexity is an important factor in evaluating the running time of algorithms. Usually, it can be estimated in the light of the structure and implementation of the algorithm. As can be seen from Figures 3-5, the time cost of each iteration is mainly due to population size, the dimension of SUKP instances. Specifically, computational complexity depends mainly on Steps 1-3 of Figure 5. Note that Quicksort algorithm is selected in Step 1. The worst and the average time costs are $O(m^2)$ and $O(m\log m)$, respectively. In Step 2, the initialization of N moth individuals with m decision variable has time complexity $O(N \times m) = O(m^2)$. In Step 3, enhanced interaction operator and straight flight operator cost the same time $O(N/2 \times m) = O(m^2)$. The process of S-GROA and evaluation of the fitness of the population cost time $O(m \times n) = O(m^2)$ and $O(N)$, respectively. Similarly, the sorting process of the population via Quicksort has the worst time complexity and average time complexity of $O(N^2)$ and $O(N\log N)$, respectively. Therefore, the total time complexity can be calculated as $O(m\log m) + O(m^2) + O(m^2) + O(m^2) + O(m^2) + O(N) + O(N\log N) = O(m^2)$.

V. COMPUTATIONAL EXPERIMENTS

In this section, three types of 30 SUKP instances commonly used in literature are first provided. Then parameter settings and experimental environment are outlined. Followed by a lot of computational experiments to compare the proposed EMS with several state-of-the-art algorithms. Finally, the impact of two key components of EIO on the performance of EMS is investigated.

A. SUKP INSTANCES

These instances were first generated by He *et al.* in [7]. All instances are represented as $m\_n\_\alpha\_\beta$, where m and n represent the number of items and number of elements, respectively. The parameters α and β are called the density of elements and the ratio of knapsack capacity to the total weight of all elements, respectively. According to the size of

TABLE 1. The parameters for 30 SUKP instances.

No.	Instance	m	n	α	β	Capacity	Best*
1	F01	100	85	0.1	0.75	12015	13283
2	F02	100	85	0.15	0.85	12405	12274
3	F03	200	185	0.1	0.75	22809	13405
4	F04	200	185	0.15	0.85	25828	14044
5	F05	300	285	0.1	0.75	36126	11335
6	F06	300	285	0.15	0.85	40801	12245
7	F07	400	385	0.1	0.75	50856	11484
8	F08	400	385	0.15	0.85	56538	10710
9	F09	500	485	0.1	0.75	60351	11722
10	F10	500	485	0.15	0.85	67506	10022
11	S01	100	100	0.1	0.75	11223	14044
12	S02	100	100	0.15	0.85	15194	13508
13	S03	200	200	0.1	0.75	25630	12522
14	S04	200	200	0.15	0.85	29583	12317
15	S05	300	300	0.1	0.75	38289	12736
16	S06	300	300	0.15	0.85	45914	11425
17	S07	400	400	0.1	0.75	49822	11531
18	S08	400	400	0.15	0.85	57856	10927
19	S09	500	500	0.1	0.75	63902	10888
20	S10	500	500	0.15	0.85	73927	10194
21	T01	85	100	0.1	0.75	12180	12045
22	T02	85	100	0.15	0.85	14982	12369
23	T03	185	200	0.1	0.75	25405	13696
24	T04	185	200	0.15	0.85	28159	11298
25	T05	285	300	0.1	0.75	38922	11568
26	T06	285	300	0.15	0.85	44806	11517
27	T07	385	400	0.1	0.75	49815	10483
28	T08	385	400	0.15	0.85	57687	10338
29	T09	485	500	0.1	0.75	62516	11094
30	T10	485	500	0.15	0.85	71655	10104

m and n, three types of SUKP instances are provided. The first group contains 10 SUKP instances with $m > n$, named as F01-F10, respectively. The second group contains 10 SUKP instances with $m = n$, named as S01-S10, respectively. The third group contains 10 SUKP instances with $m < n$, named as T01-T10, respectively. The parameters and the best-known solution (Best*) are presented in Table 1.

B. PARAMETER SETTINGS AND EXPERIMENTAL ENVIRONMENT

To evaluate the comprehensive performance of the proposed EMS algorithm, seven state-of-the-art metaheuristic algorithms in the literatures are used as the basic comparison algorithms, including the binary artificial bee colony algorithm (BABC) [7], binary weighted superposition attraction algorithm (bWSA) [8], the hybrid of genetic algorithm and particle swarm optimization (gPSO* and gPSO) [9], firefly algorithm (FA) [33], [34], monarch butterfly optimization (MBO) [35], [36], and original MS [14]. These reference algorithms are among the best performing metaheuristic algorithms currently obtained through literature. The experimental results used in this paper for BABC, bWSA, gPSO*, and gPSO are adopted from the relevant literature.

The parameters of FA, MBO, MS, and the proposed EMS are empirically set (see Table 2). We define the maximum

TABLE 2. The parameter settings of four algorithms on SUKP.

Algorithm	Parameter	Value
FA	*Alpha*	0.2
	Beta	1.0
	Gamma	1.0
MBO	*Migration ratio*	3/12
	Migration period	1.4
	Butterfly adjusting rate	1/12
	Max step	1.0
MS	*Max step* S_{max}	1.0
	Acceleration factor φ	0.618
	β	1.5
EMS	*Acceleration factor* φ	0.618
	HMCR	0.9
	PAR	0.9
	Amplification factor λ	0.7
	Amplification factor F	0.7

number of iterations as the stopping condition according to the original paper [7], whose value is equal to $max\{m, n\}$. The population size of these four algorithms is set to be $N = 20$. All experimental results of FA, MBO, MS, and EMS are evaluated over 100 independent runs.

To make a fair comparison, all the proposed algorithms (FA, MBO, MS, and EMS) are programmed in C and compiled using the GNU GCC compiler. All the experiments are performed on a computer with Intel(R) Core (TM) i7-7500 CPU (2.90 GHz and 8.00 GB RAM).

C. COMPUTATIONAL RESULTS AND COMPARISONS

The computational results on 30 SUKP instances based on the above experimental design are summarized in Tables 3-7. In Tables 3-5, the first column shows the name of SUKP instance and the current best-known solution is recorded in parentheses under the corresponding instance. For each algorithm, three basic evaluation criteria, including the best objective value (Best), the average objective value (Mean), and the worst objective value (Worst) over 100 independent runs, are selected to assess the overall performance of all the reference algorithms. The best results of the eight algorithms are shown in bold if they are equal to or greater than the best-known solution reported in the literature.

It is important to note that the computational time is not considered as the comparison criteria in the present study. The prime reason is that, the running time of different comparison algorithms depends on the programming language, computing platform, and even the compiler. Therefore, it is difficult to make a fair comparison of the computational time. In addition, the worst value of BABC cannot be obtained from the literature [7] and "−" is used to express it.

The computational results on the first group SUKP instances with $m > n$ are recorded in Table 3. Experimental results (the last three rows in Table 3) demonstrate that the proposed algorithm EMS reaches the best solutions for five instances. In addition, EMS obtains the best mean values for four instances and the worst values for five instances. The number of the best values, the mean values, and the worst

TABLE 3. The experimental results on F01-F10 SUKP instances.

		BABC	bWSA	gPSO*	gPSO	FA	MBO	MS	EMS
F01	Best	13251	13044	13167	**13283**	**13283**	**13283**	**13283**	**13283**
(13283)	Mean	13028	12915	12937	13050	13041	12941	13062	**13152**
	Worst	-	12244	12217	**13044**	12778	12622	12692	12786
F02	Best	12238	12238	12210	12274	12348	12274	**12479**	12274
(12274)	Mean	12155	11527	11777	12084	12129	12103	**12167**	12149
	Worst	-	10408	11052	11631	11860	11907	**12062**	11537
F03	Best	13241	13250	13302	13405	13282	13381	**13521**	**13521**
(13405)	Mean	13064	12657	12766	**13286**	12544	12886	13193	13278
	Worst	-	11951	11974	12862	11959	12328	12694	**12865**
F04	Best	13829	13858	13993	**14044**	13801	13880	13880	**14044**
(14044)	Mean	13359	12585	12949	13492	12602	13326	13403	**13692**
	Worst	-	11836	11940	**12785**	11921	12305	12428	12536
F05	Best	10428	10991	10600	11335	10191	10786	11127	**11388**
(11335)	Mean	9994	10366	10090	10669	9092	10210	10302	**10881**
	Worst	-	9802	9629	9900	9495	9346	9526	**10265**
F06	Best	12012	12093	11935	12245	11468	11695	**12273**	12245
(12245)	Mean	10902	10901	10750	11607	10417	10366	**11964**	11958
	Worst	-	9912	9517	10536	9852	9178	10228	**11229**
F07	Best	10766	11321	10698	**11484**	9740	11142	11435	**11484**
(11484)	Mean	10065	10785	9946	**10915**	9226	10463	10411	10430
	Worst	-	9798	9209	**10158**	8879	9771	9499	10045
F08	Best	9649	10435	10168	**10710**	9305	9686	10099	10091
(10710)	Mean	9135	9587	9417	**9864**	8663	9156	8917	9117
	Worst	-	8695	8395	**9173**	8271	8408	8102	8778
F09	Best	10784	11540	11258	**11722**	11099	11546	11031	11716
(11722)	Mean	10452	10921	10565	11184	10473	10736	10716	**11552**
	Worst	-	10293	9817	10614	10081	10033	9536	**11385**
F10	Best	9090	9681	9756	**10022**	8710	9243	9037	9511
(10022)	Mean	8857	9013	8779	**9299**	8326	8667	8685	9256
	Worst	-	8479	8251	8556	8114	8160	8126	**8744**
Total(Best)		0	0	0	6	1	1	4	5
Total(Mean)		0	0	0	4	0	0	2	4
Total(Worst)		-	0	0	4	0	0	1	5

TABLE 4. The experimental results on S01-S10 SUKP instances.

		BABC	bWSA	gPSO*	gPSO	FA	MBO	MS	EMS
S01	Best	13860	**14044**	13963	**14044**	13814	**14044**	**14044**	**14044**
(14044)	Mean	13734	13492	13739	**13854**	13472	13612	13649	13735
	Worst	-	12625	13266	**13664**	13054	13023	13194	13492
S02	Best	**13508**	13407	13498	**13508**	13498	**13508**	13498	**13508**
(13508)	Mean	13352	12487	12937	13347	13408	13269	13374	**13427**
	Worst	-	10987	11818	12613	12752	12393	13074	**13407**
S03	Best	11846	12271	11972	**12522**	11406	11955	12350	**12522**
(12522)	Mean	11194	11430	11232	**11898**	10833	11056	11508	11876
	Worst	-	10622	10541	11048	10421	10357	10703	**11135**
S04	Best	11521	11804	12167	**12317**	11207	11569	11926	11846
(12317)	Mean	10945	11062	11026	**11584**	10354	10850	11064	11365
	Worst	-	10042	10109	**10510**	9741	9737	10191	10206
S05	Best	12186	12644	12736	**12736**	11398	12369	12598	12626
(12736)	Mean	11945	12227	11934	11934	10993	11604	11541	**12540**
	Worst	-	11365	11175	11175	10665	10616	10802	**11820**
S06	Best	10382	11113	10724	**11425**	9874	10152	10557	10770
(11425)	Mean	9859	10216	9906	**10568**	9339	9304	9821	10264
	Worst	-	9520	8740	9648	8855	8751	9145	**9953**
S07	Best	10626	11199	11048	**11531**	10241	10906	10727	11171
(11531)	Mean	10101	10624	10399	**10958**	9827	10237	10343	10803
	Worst	-	9818	9752	10205	9505	9676	9678	**10514**
S08	Best	9541	10915	10264	**10927**	9357	9370	10915	9793
(10927)	Mean	9032	9580	9195	9845	8562	8677	**9860**	9692
	Worst	-	8717	8600	9033	8248	8185	8644	**9268**
S09	Best	10755	10827	10647	**10888**	10057	10633	10355	10628
(10888)	Mean	10328	10482	10205	**10681**	9766	10139	9919	10466
	Worst	-	10147	9744	**10222**	9560	9486	9492	10139
S10	Best	9318	10082	9839	**10194**	9140	9926	9444	9765
(10194)	Mean	9180	9478	9106	**9703**	8630	9391	8937	9399
	Worst	-	8705	8512	8892	8348	8879	8061	**8918**
Total(Best)		1	1	0	10	0	2	1	3
Total(Mean)		0	0	0	7	0	0	1	2
Total(Worst)		-	0	0	3	0	0	0	7

values among the first group SUKP instances obtained by the reference algorithm gPSO is 6, 4, and 4, respectively. It can be concluded that the performance of EMS is competitive with gPSO and superior to the other six algorithms.

The computational results on the second group SUKP instances with $m = n$ are summarized in Table 4. Unfortunately, BABC, bWSA, gPSO*, FA, MBO, MS, and EMS algorithms get the best known solutions only for 1, 1, 0, 0, 2, 1, and 3 instances, respectively. However, gPSO maintains the best known solution for all 10 instances (S01-S10). It should be stated that EMS attains the best results for 7 instances in terms of the worst value.

The computational results on the third group SUKP instances with $m < n$ are reported in Table 5. Table 5 shows that the proposed algorithm EMS is efficient and EMS has the ability to compete with gPSO. Specifically, the EMS algorithm matches the best known solutions for 6 out of 10 instances, while the reference algorithms BABC, bWSA, gPSO*, MBO, and MS achieve the best known solutions only for 1 instance. Nevertheless, gPSO still retains the best-known solutions for 8 instances.

To evaluate the proximity between the best solution obtained by the algorithm and the best-known solution, the relative percentage deviation (*RPD*) is defined as follows.

$$RPD = (\text{Best}^* - \text{Best})/\text{Best}^* \times 100 \tag{20}$$

where Best* represents the best known solution that can be available through current literature [9]. Since SUKP is a maximum optimization problem, if Best $>$ Best*, $RPD <$ 0, which indicates the algorithm improves the best known solution. The results are reported in Table 6.

From Table 6, it can be observed that EMS matches the best known solutions for 14 instances ($RPD = 0.0$, out of 30 instances). Especially, our proposed EMS algorithm updates

TABLE 5. The experimental results on T01-T10 SUKP instances.

		BABC	bWSA	gPSO*	gPSO	FA	MBO	MS	EMS
T01	Best	11664	11947	11710	**12045**	11755	11748	11735	**12045**
(12045)	Mean	11182	11233	11237	**11486**	11226	11207	11287	11416
	Worst	-	10627	10657	**11202**	10797	10581	10892	11090
T02	Best	**12369**	**12369**	**12369**	**12369**	**12369**	**12369**	**12369**	**12369**
(12369)	Mean	12081	11342	11684	11994	12361	12229	12350	**12368**
	Worst	-	9774	11121	11274	12085	11858	12054	**12299**
T03	Best	13047	13505	13298	**13696**	11487	13008	13647	**13696**
(13696)	Mean	12522	12689	12514	13204	10880	12189	13000	**13381**
	Worst	-	11820	11919	12339	10393	11644	12159	**12969**
T04	Best	10602	10831	10856	11298	**11365**	10745	11298	11298
(11298)	Mean	10150	10228	10208	10801	10596	10095	10525	**10652**
	Worst	-	9467	9727	**10195**	10003	9297	9594	9907
T05	Best	11158	11538	11310	**11568**	11557	11090	11391	**11568**
(11568)	Mean	10775	**11105**	10761	10761	10983	10686	10816	11020
	Worst	-	10600	10278	10278	**10641**	9851	10363	10484
T06	Best	10528	11377	11226	**11517**	11160	10783	11353	**11517**
(11517)	Mean	9897	10452	10309	**10899**	10261	9977	10155	10775
	Worst	-	9519	9626	**10281**	9570	9269	8873	9861
T07	Best	10085	10414	9871	**10483**	9392	9770	9739	10325
(10483)	Mean	9537	9778	9552	**10013**	8895	9322	9240	10001
	Worst	-	9378	8786	**9519**	8633	8891	8514	9164
T08	Best	9456	10077	9389	10338	8980	9175	10058	**10506**
(10338)	Mean	9090	9203	8881	9524	8529	8604	9006	**9669**
	Worst	-	8600	8316	8816	8183	8052	8096	**9072**
T09	Best	10823	10835	10595	**11094**	10207	10661	10539	10855
(11094)	Mean	10483	10607	10145	**10687**	9783	10249	10190	10662
	Worst	-	10024	9583	10201	9482	9759	9492	**10394**
T10	Best	9333	9603	9807	**10104**	9141	9163	9284	9589
(10104)	Mean	9085	9141	8917	**9383**	8513	8758	8840	9078
	Worst	-	8562	8435	**8834**	8248	8289	8298	8821
Total(Best)		1	1	1	8	2	1	1	6
Total(Mean)		0	1	0	5	0	0	0	4
Total(Worst)		-	0	0	5	1	0	0	4

TABLE 6. Comparisons of the *RPD* values for 30 SUKP instances.

NO	BABC	bWSA	gPSO*	gPSO	FA	MBO	MS	EMS
F01	0.24	1.80	0.87	0.00	0.00	0.00	0.00	0.00
F02	0.29	0.29	0.52	0.00	**-0.60**	0.00	**-1.67**	0.00
F03	1.22	1.16	0.77	0.00	0.92	0.18	**-0.87**	**-0.87**
F04	1.53	1.32	0.36	0.00	1.73	1.17	1.17	0.00
F05	8.00	3.03	6.48	0.00	10.09	4.84	1.84	**-0.47**
F06	1.90	1.24	2.53	0.00	6.35	4.49	**-0.23**	0.00
F07	6.25	1.42	6.84	0.00	15.19	2.98	0.43	0.00
F08	9.91	2.57	5.06	0.00	13.12	9.56	5.70	5.78
F09	8.00	1.55	3.96	0.00	5.31	1.50	5.89	0.05
F10	9.30	3.40	2.65	0.00	13.09	7.77	9.83	5.10
S01	1.31	0.00	0.58	0.00	1.64	0.00	0.00	0.00
S02	0.00	0.75	0.00	0.00	0.07	0.00	0.07	0.00
S03	5.40	2.00	0.00	0.00	8.91	4.53	1.37	0.00
S04	6.46	4.16	1.22	0.00	9.01	6.07	3.17	3.82
S05	4.32	0.72	0.00	0.00	10.51	2.88	1.08	0.86
S06	9.13	2.73	6.14	0.00	13.58	11.14	7.60	5.73
S07	7.85	2.88	4.19	0.00	11.19	5.42	6.97	3.12
S08	12.68	0.11	6.07	0.00	14.37	14.25	0.11	10.38
S09	1.22	0.56	2.21	0.00	7.63	2.34	4.90	2.39
S10	8.59	1.10	3.48	0.00	10.34	2.63	7.36	4.21
T01	3.16	0.81	2.78	0.00	2.41	2.47	2.57	0.00
T02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
T03	4.74	1.39	2.91	0.00	16.13	5.02	0.36	0.00
T04	6.16	4.13	3.91	0.00	**-0.59**	4.89	0.00	0.00
T05	3.54	0.26	2.23	0.00	0.10	4.13	1.53	0.00
T06	8.59	1.22	2.53	0.00	3.10	6.37	1.42	0.00
T07	3.80	0.66	5.84	0.00	10.41	6.80	7.10	1.51
T08	8.53	2.52	9.18	0.00	13.14	11.25	2.71	**-1.63**
T09	2.44	2.33	4.50	0.00	8.00	3.90	5.00	2.15
T10	7.63	4.96	2.94	0.00	9.53	9.31	8.12	5.10
Mean	5.07	1.70	3.03	0	7.15	4.53	2.78	1.57
Greater	0	0	0	0	2	0	3	3
Equal	2	2	4	30	2	5	4	14
Less	28	28	26	0	26	25	23	13

the best known solutions for 3 instances ($RPD < 0$, new upper bounds). However, there are still 13 instances that EMS fails to find the best known result, which shows the performance of MS can be further improved. It can also be observed that FA and MS improve the best known results for 2 and 3 instances, respectively, which demonstrates these two algorithms are acceptable compared with BABC, bWSA, gPSO*, and MBO.

The ranking information of eight algorithms based on the best values is summarized in Table 7 with the aim of evaluating the optimization performance. As can be seen from Table 7, the average ranking value of gPSO is 1.84 and still maintains its excellent performance. In addition, EMS and bWSA are the second and the third best algorithm, respectively.

In conclusion, the above experimental results reveal that the proposed EMS algorithm is highly competitive compared to the state-of-the-art SUKP algorithms in the literature. For all 30 SUKP instances, EMS updates the best known solutions for 3 instances (10%) and reaches the best known solutions for 14 instances (46.67%).

In order to illustrate the overall performance of the proposed EMS and estimate the differences between EMS and MS, FA, and MBO, the Wilcoxon's rank sum tests with the 5% significance level are conducted. The p-value and h-value of for 30 SUKP instances is recorded in Table 8. Note that the results of significant difference between EMS algorithm and comparison algorithm are shown in bold.

As can be seen from Table 8, the p-value obtained by pairwise comparison between EMS and MS, FA, and MBO is less than 5% for 27, 26, and 30 instances, respectively. The mean p-value is less than 5% for EMS-FA and EMS-MBO or practically 5% for EMS-MS. It indicates that EMS is significantly different from the other three algorithms.

With the aim of analyzing the stability of the compared approaches, the box plots of six of the most representative instances, i.e., F09, F10, S09, S10, T09, and T10 are presented in Figures 6-8. The stability of each algorithm can be reflected by the span of the box. As can be seen from Figure 6,

TABLE 7. Ranks of eight algorithms based on the best values.

NO	BABC	bWSA	gPSO*	gPSO	FA	MBO	MS	EMS
F01	6	8	7	3	3	3	3	3
F02	6.5	6.5	8	4	2	4	1	4
F03	8	7	5	3	6	4	1.5	1.5
F04	7	6	3	1.5	8	4.5	4.5	1.5
F05	7	4	6	2	8	5	3	1
F06	5	4	6	2.5	8	7	1	2.5
F07	6	4	7	1.5	8	5	3	1.5
F08	7	2	3	1	8	6	4	5
F09	8	4	5	1	6	3	7	2
F10	6	3	2	1	8	5	7	4
S01	7	3	6	3	8	3	3	3
S02	3	8	3	3	6.5	3	6.5	3
S03	7	5	2	2	8	6	4	2
S04	7	5	2	1	8	6	3	4
S05	7	3	1.5	1.5	8	6	5	4
S06	6	2	4	1	8	7	5	3
S07	7	2	4	1	8	5	6	3
S08	6	2.5	4	1	8	7	2.5	5
S09	3	2	4	1	8	5	7	6
S10	7	2	4	1	8	3	6	5
T01	8	3	7	1.5	4	5	6	1.5
T02	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5
T03	6	4	5	1.5	8	7	3	1.5
T04	8	6	5	3	1	7	3	3
T05	7	4	6	1.5	3	8	5	1.5
T06	8	3	5	1.5	6	7	4	1.5
T07	4	2	5	1	8	6	7	3
T08	5	3	6	2	8	7	4	1
T09	4	3	6	1	8	5	7	2
T10	5	3	2	1	8	7	6	4
Mean	6.24	3.98	4.69	1.84	6.69	5.31	4.36	2.88
Rank	7	3	5	1	8	6	4	2

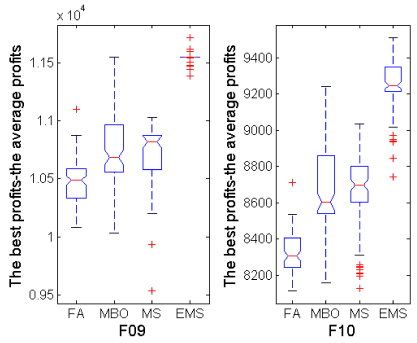

FIGURE 6. Boxplot of the best values for F09 and F10 in 100 runs.

for F09, EMS has decided advantage over its opponents, since the box span for EMS is much smaller than that of other three algorithms. EMS also achieves much smaller spans on other five instances.

To further demonstrate the effectiveness of the proposed EMS algorithm and especially to investigate the convergence speed, the evolutionary process of the iteration number and average best objective function value as a function are plotted in Figures 9-11.

It can be observed from Figures 9-11, the initial value of EMS is greater than that of other three algorithms and then

TABLE 8. Rank sum tests for EMS with FA, MBO, and MS on 30 SUKP instances.

NO	EMS-MS		EMS-FA		EMS-MBO	
	p-value	*h*-value	*p*-value	*h*-value	*p*-value	*h*-value
F01	1.31E-04	1	6.18E-08	1	5.33E-15	1
F02	**3.11E-01**	**0**	**1.93E-01**	**0**	9.17E-05	1
F03	8.68E-04	1	0.00E-00	1	0.00E-00	1
F04	2.51E-07	1	0.00E-00	1	4.64E-13	1
F05	0.00E-00	1	0.00E-00	1	0.00E-00	1
F06	**6.86E-01**	**0**	0.00E-00	1	0.00E-00	1
F07	**4.84E-01**	**0**	0.00E-00	1	4.60E-02	1
F08	2.66E-08	1	0.00E-00	1	7.20E-03	1
F09	0.00E-00	1	0.00E-00	1	0.00E-00	1
F10	0.00E-00	1	0.00E-00	1	0.00E-00	1
S01	8.06E-06	1	0.00E-00	1	9.62E-06	1
S02	1.80E-08	1	3.80E-03	1	1.78E-15	1
S03	9.36E-10	1	0.00E-00	1	0.00E-00	1
S04	4.18E-08	1	0.00E-00	1	0.00E-00	1
S05	0.00E-00	1	0.00E-00	1	0.00E-00	1
S06	0.00E-00	1	0.00E-00	1	0.00E-00	1
S07	0.00E-00	1	0.00E-00	1	0.00E-00	1
S08	1.34E-02	1	0.00E-00	1	0.00E-00	1
S09	0.00E-00	1	0.00E-00	1	0.00E-00	1
S10	0.00E-00	1	0.00E-00	1	5.13E-01	1
T01	2.26E-10	1	1.31E-14	1	1.15E-14	1
T02	1.50E-03	1	**5.36E-02**	**0**	0.00E-00	1
T03	2.51E-14	1	0.00E-00	1	0.00E-00	1
T04	1.47E-02	1	**1.30E-01**	**0**	0.00E-00	1
T05	1.08E-10	1	**7.49E-02**	**0**	0.00E-00	1
T06	2.22E-16	1	0.00E-00	1	0.00E-00	1
T07	0.00E-00	1	0.00E-00	1	0.00E-00	1
T08	0.00E-00	1	0.00E-00	1	0.00E-00	1
T09	0.00E-00	1	0.00E-00	1	0.00E-00	1
T10	0.00E-00	1	0.00E-00	1	0.00E-00	1
Mean	5.04E-02		1.52E-02		1.89E-02	
Difference		27		26		30
Similarity		3		4		0

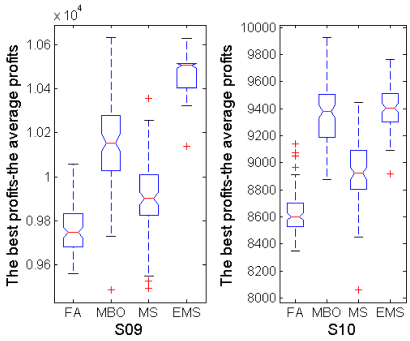

FIGURE 7. Boxplot of the best values for S09 and S10 in 100 runs.

EMS reaches the global optimum with a fast convergence speed. Although MBO outperforms EMS slightly in performance in the early period of evolutionary on S10, EMS and MBO have identical final solutions. Overall, Figures 9-11 clearly indicates that EMS has distinct advantage over FA, MBO, and MS on six high-dimensional instances.

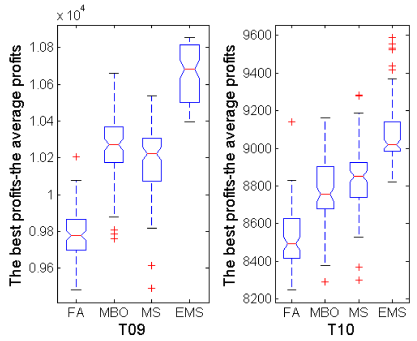

FIGURE 8. Boxplot of the best values for T09 and T10 in 100 runs.

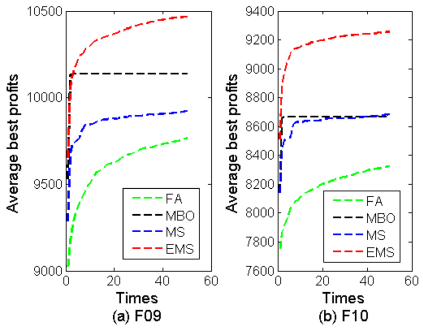

FIGURE 9. The convergence graph for F09 and F10.

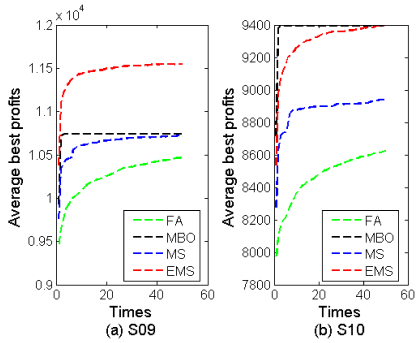

FIGURE 10. The convergence graph for S09 and S10.

Considering the experimental results shown in Figures 6-11, it can be concluded that EMS is an effective algorithm for solving SUKP problem, regardless of the quality of solution, the stability and the convergence rate.

D. THE EFFECT OF GLOBAL HARMONY SEARCH AND DIFFERENTIAL MUTATION ON THE PERFORMANCE OF EMS

In this section, the impact of two important components of EIO on the performance of EMS algorithm is analyzed. Therefore, EMS that Eq. (18) removes from enhanced interaction operator is renamed H-MS. Meanwhile, EMS that Eqs. (15) - (16) is excluded from enhanced interaction operator is called D-MS. Note that the difference between odd number instances and even numbering instances in each group is the value of α and β, and there is no essential difference. Therefore, fifteen odd numbering instances with

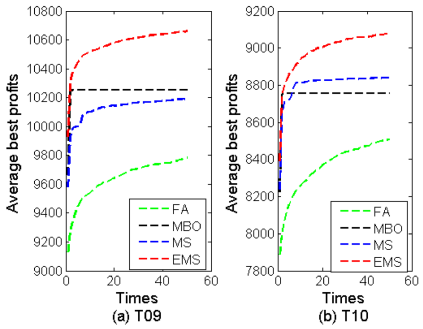

FIGURE 11. The convergence graph for T09 and T10.

TABLE 9. The effect of GHS and DM on the performance of EMS.

NO	Best*	MS		H-MS		D-MS		EMS	
		Best	Mean	Best	Mean	Best	Mean	Best	Mean
F01	13283	**13283**	13062	**13283**	13161	**13283**	**13246**	**13283**	13152
F03	13405	**13521**	13193	**13521**	13085	13322	12738	**13521**	**13278**
F05	11335	11127	10302	**11529**	**10978**	10702	9858	11388	10881
F07	11484	11435	10411	11042	10344	11109	**10516**	**11484**	10430
F09	11722	11031	10716	11410	11088	10817	10546	**11716**	**11552**
S01	14044	**14044**	13649	**14044**	**13784**	**14044**	13586	**14044**	13735
S03	12522	12350	11508	**12522**	**11877**	11951	10815	**12522**	11876
S05	12736	12598	11541	**12644**	12343	12075	11627	12626	**12540**
S07	11531	10727	10343	11087	10735	10814	10362	**11171**	**10803**
S09	10888	10355	9919	10589	10343	10273	10081	**10628**	**10466**
T01	12045	11735	11287	**12045**	**11436**	11755	11367	**12045**	11416
T03	13696	13647	13000	13647	**13389**	13355	12965	**13696**	13381
T05	11568	11391	10816	**11568**	**11058**	10955	10413	**11568**	11020
T07	10483	9739	9240	**10374**	9998	9701	9131	10325	**10001**
T09	11094	10539	10190	10817	10546	10633	10250	**10855**	**10662**
Total(Best)		3		9		2		12	
Total(Mean)			0		6		2		7

$\alpha = 0.1$ and $\beta = 0.75$ are selected to conduct comparative experiment. The results are summarized in Table 9, where column 1 and column 2 represent the instance name and the best-known solution, respectively. The last two rows respectively give the number of the best result in terms of the best solution value and the average solution value over 100 independent runs. The best values obtained by MS, H-MS, D-MS, and EMS are indicated in bold.

From Table 9, it can be observed that the number of the best values obtained by MS, H-MS, D-MS, and EMS is 3, 9, 2, and 12, respectively. Additionally, the number of the mean values obtained by MS, H-MS, D-MS, and EMS is 0, 6, 2, and 7, respectively. Thus, it can be seen that EMS performs the best among these four algorithms, which demonstrates enhanced interaction operator improves the performance of EMS by increasing information sharing between different solutions. Especially, H-MS outperforms D-MS, which reveals that the performance of EMS is more affected by global harmony search than differential mutation. Through careful analysis, it is not difficult to find that individual j in Eq. (15) selected randomly from the whole population. As a result, individuals in subpopulation1 inherit preeminent information on a greater

search space. From Eq. (16), the jth element of individual i inherits from the global optimal solution with a certain probability, which makes it more likely to transfer excellent gene to the next generation. In addition, D-MS is slightly inferior to MS, which shows the performance of EMS is not significantly improved with differential mutation alone.

VI. CONCLUSION

In this paper, we introduced an enhanced moth search algorithm (EMS) for solving the set-union knapsack problem. In EMS, an enhanced interaction operator is specifically designed by integrating differential mutation into global harmony search method with the aim of increasing information sharing and population diversity.

The experimental results on three types of 30 SUKP instances commonly used in the literature indicate the proposed algorithm EMS is superior to or at least quite competitive with state-of-the-art algorithms in the literature. Most importantly, the related algorithm in this paper update the best known solution for six instances and then give new best solutions for these instances, including F02 (MS, 12479), F03 (MS and EMS, 13521), F05 (H-MS, 11529), F06 (MS, 12273), T04 (FA, 11365), and T08 (EMS, 10506).

The impact of global harmony search and differential mutation on the performance of EMS is investigated. The comparative results show that both parts play an important role on the performance of the algorithm. However, global harmony search is more effective than differential mutation.

Additionally, although EMS is an effective alternative method for solving SUKP problems, the accuracy of some instances need to be further improved and the performance of EMS does not show obvious advantages, especially compared with gPSO algorithm. Therefore, it is necessary to make a profound study on the evolution mechanism of MS algorithm and propose more efficient improvement strategies.

As future work, several potential research topics are outlined as follows. First, to further improve MS, it is necessary to an in-depth study of alternative strategies for maintaining population diversity and sharing excellent information such as information feedback mechanism [37]. Second, it would be interesting to investigate other swarm intelligence, such as earthworm optimization algorithm (EWA) [38], fruit fly optimization algorithm (FOA) [39], invasive weed optimization algorithm (IWO) [40], cuckoo search (CS) [41], krill herd (KH) [42], for solving SUKP. Finally, it is certainly worth extending MS to other more complex combinatorial optimization problems including the knapsack problem with setup (KPS) [43], the 0-1 multidimensional knapsack problem (MKP) [44], unbounded knapsack problem (UKP) [45], constrained knapsack problems in dynamic environments (DKPs) [46].

REFERENCES

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.
[2] O. Goldschmidt, D. Nehme, and G. Yu, "Note: On the set-union knapsack problem," *Naval Res. Logistics*, vol. 41, no. 6, pp. 833–842, 2015.
[3] A. Arulselvan, "A note on the set union knapsack problem," *Discrete Appl. Math.*, vol. 169, pp. 214–218, May 2014.
[4] B. Schneier, *Appiled Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. Hoboken, NJ, USA: Wiley, 1996.
[5] X. Yang, A. Vernitski, and L. Carrea, "An approximate dynamic programming approach for improving accuracy of lossy data compression by Bloom filters," *Eur. J. Oper. Res.*, vol. 252, pp. 985–994, Aug. 2016.
[6] R. Taylor, "Approximations of the densest k-subhypergraph and set union knapsack problems," Oct. 2016, *arXiv:1610.04935*. [Online]. Available: https://arxiv.org/abs/1610.04935
[7] Y. C. He, H. Xie, T.-L. Wong, and X. Wang, "A novel binary artificial bee colony algorithm for the set-union knapsack problem," *Future Generat. Comput. Syst.*, vol. 78, pp. 77–86, Jan. 2018.
[8] A. Baykasoglu, F. B. Ozsoydan, and M. E. Senol, "Weighted superposition attraction algorithm for binary optimization problems," in *Operational Research*, 2018, doi: 10.1007/s12351-018-0427-9.
[9] F. B. Ozsoydan and A. Baykasoglu, "A swarm intelligence-based algorithm for the set-union knapsack problem," *Future Gener. Comput. Syst.*, vol. 93, pp. 560–569, Apr. 2019.
[10] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Mach. Learn.*, vol. 3, no. 2, pp. 9–95, Oct. 1988.
[11] J. Liu, Y. Mei, and X. Li, "An analysis of the inertia weight parameter for binary particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 666–681, Oct. 2015.
[12] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[13] A.-A. A. Mohamed, Y. S. Mohamed, A. A. M. El-Gaafary, and A. M. Hemeida, "Optimal power flow using moth swarm algorithm," *Electr. Power Syst. Res.*, vol. 142, pp. 190–206, Jan. 2017.
[14] G.-G. Wang, "Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Comput.*, vol. 10, pp. 151–164, Jun. 2018.
[15] Y.-H. Feng and G.-G. Wang, "Binary moth search algorithm for discounted {0-1} knapsack problem," *IEEE Access*, vol. 6, pp. 10708–10719, 2018.
[16] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.
[17] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.
[18] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Appl. Math. Comput.*, vol. 198, pp. 643–656, May 2008.
[19] H. Kellerer, U. Pferschy, and D. Pisinger, "Multidimensional knapsack problems," in *Knapsack Problems*. Berlin, Germany: Springer, 2004, pp. 235–283.
[20] Z. W. Geem, *Music-Inspired Harmony Search Algorithm Theory and Applications*. Berlin, Germany: Springer, 2009.
[21] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Comput. Methods Appl. Mech. Eng.*, vol. 194, pp. 3902–3933, Sep. 2005.
[22] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary particle swarm optimization," *Swarm Evol. Comput.*, vol. 9, pp. 1–14, Apr. 2013.
[23] S. Saremi, S. Mirjalili, and A. Lewis, "How important is a transfer function in discrete heuristic algorithms," *Neural Comput. Appl.*, vol. 26, pp. 625–640, Apr. 2014.
[24] H. Zhu, Y. He, X. Wang, and E. C. C. Tsang, "Discrete differential evolutions for the discounted {0-1} knapsack problem," *Int. J. Bio-Inspired Comput.*, vol. 10, no. 4, pp. 219–238, 2017.
[25] X. S. Yang, "Firefly algorithm, Lévy flights and global optimization," in *Research and Development in Intelligent Systems XXVI*. London, U.K.: Springer, 2010, pp. 209–218.
[26] H. Sharma, J. C. Bansal, and K. V. Arya, "Opposition based Lévy flight artificial bee colony," *Memetic Comput.*, vol. 5, pp. 213–227, Sep. 2013.
[27] A. P. Engelbrecht, *Computational Intelligence: An Introduction*. Hoboken, NJ, USA: Wiley, 2007.
[28] E. Bonabeau, M. Dorigo, and G. Theraulaz, "Swarm intelligence: From natural to artificial systems," in *Santa Fe Institute Studies on the Sciences of Complexity*. Oxford, U.K.: Oxford Univ. Press, 1999.
[29] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, pp. 8–60, Feb. 2001.
[30] D. Manjarres, I. Landa-Torres, S. Gil-Lopez, J. D. Ser, M. N. Bilbao, S. Salcedo-Sanz, and Z. W. Geem, "A survey on applications of the harmony search algorithm," *Eng. Appl. Artif. Intell.*, vol. 26, pp. 1818–1831, Sep. 2013.

[31] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Appl. Math. Comput.*, vol. 216, pp. 830–848, Apr. 2010.
[32] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, pp. 341–359, Dec. 1997.
[33] Y. Feng, G.-G. Wang, and L. Wang, "Solving randomized time-varying knapsack problems by a novel global firefly algorithm," *Eng. Comput.*, vol. 34, pp. 621–635, Jul. 2018.
[34] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," Mar. 2010, *arXiv:1003.1409*. [Online]. Available: https://arxiv.org/abs/1003.1409
[35] Y. Feng, G.-G. Wang, W. Li, and N. Li, "Multi-strategy monarch butterfly optimization algorithm for discounted {0-1} knapsack problem," *Neural Comput. Appl.*, vol. 30, pp. 3019–3036, Nov. 2018.
[36] G. G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Comput. Appl.*, vol. 31, no. 7, pp. 1995–2014, 2019.
[37] G.-G. Wang and Y. Tan, "Improving metaheuristic algorithms with information feedback models," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 542–555, Feb. 2019.
[38] G. G. Wang, S. Deb, and L. dos Santos Coelho, "Earthworm optimization algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Int. J. Bio-Inspired Comput.*, vol. 7, pp. 1–23, Oct. 2018.
[39] L. Wang, X. L. Zheng, and S. Y. Wang, "A novel binary fruit fly optimization algorithm for solving the multidimensional knapsack problem," *Knowl.-Based Syst.*, vol. 48, no. 2, pp. 17–23, 2013.
[40] H.-Y. Sang, Q.-K. Pan, J.-Q. Li, P. Wang, Y.-Y. Han, K.-Z. Gao, and P. Duan, "Effective invasive weed optimization algorithms for distributed assembly permutation flowshop problem with total flowtime criterion," *Swarm Evol. Comput.*, vol. 44, pp. 64–73, Feb. 2019.
[41] G. G. Wang, A. H. Gandomi, X. J. Zhao, and H. C. E. Chu, "Hybridizing harmony search algorithm with cuckoo search for global numerical optimization," *Soft Comput.*, vol. 20, no. 1, pp. 273–285, Jan. 2016.
[42] G.-G. Wang, S. Deb, A. H. Gandomi, and A. H. Alavi, "Opposition-based krill herd algorithm with Cauchy mutation and position clamping," *Neurocomputing*, vol. 177, pp. 147–157, Feb. 2016.
[43] K. Chebil and M. Khemakhem, "A dynamic programming algorithm for the knapsack problem with setup," *Comput. Oper. Res.*, vol. 64, pp. 40–50, Dec. 2015.
[44] L. Wang, R. Yang, H. Ni, W. Ye, M. Fei, P. M. Pardalos, "A human learning optimization algorithm and its application to multi-dimensional knapsack problems," *Appl. Soft Comput.*, vol. 34, pp. 736–743, Sep. 2015.
[45] H. Becker and L. S. Buriol, "An empirical analysis of exact algorithms for the unbounded knapsack problem," *Eur. J. Oper. Res.*, vol. 277, pp. 84–99, Aug. 2019.
[46] S. Qian, Y. Liu, Y. Ye, and G. Xu, "An enhanced genetic algorithm for constrained knapsack problems in dynamic environments," *Natural Comput.*, vol. 18, pp. 913–932, Dec. 2019.

YANHONG FENG graduated from the School of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang, Hebei, in 2001, received the bachelor's degree, and M.E. degree from the Department of Computer, North China Electronic Power University, Baoding, Hebei, in 2006. She is currently an Associate Professor with the School of Information Engineering, Hebei GEO University. Her major research interests including swarm intelligence, evolutionary computation, and approximation algorithm.

JIAO-HONG YI was a Researcher with the School of Information and Control Engineering, Qingdao University of Technology. Her main research interests are intelligent algorithm, communications, and geodetic engineering.

GAI-GE WANG is currently an Associate Professor with the Ocean University of China, China. His entire publications have been cited more than 5000 times (Google Scholar). Thirteen and 39 article are selected as Highly Cited Paper by Web of Science, and Scopus (until November 2019), respectively. One paper is selected as Top Articles from Outstanding S&T Journals of China-F5000 Frontrunner. The latest Google H-index and i10-index are 40 and 78, respectively. His research interests are swarm intelligence, evolutionary computation, and big data optimization. He is a Senior Member of SAISE, SCIEI, IEEE CIS and ISMOST. He served as the Editors-in-Chief of *OAJRC Computer and Communications*, an Editorial Advisory Board Member of *Communications in Computational and Applied Mathematics* (CCAM), an Associate Editor of IJCISIM, an Editorial Board Member of IEEE Access, *Mathematics*, IJBIC, *Karbala International Journal of Modern Science*, and the *Journal of Artificial Intelligence and Systems*. He served as a Guest Editor for many journals including *Mathematics*, IJBIC, FGCS, *Memetic Computing*, and *Operational Research*.