IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 21, 2019, accepted September 6, 2019, date of publication September 10, 2019,
date of current version September 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2940538

Estimation of Distribution Algorithm Based on Lévy Flight for Solving the Set-Union Knapsack Problem

XUE-JING LIU AND YI-CHAO HE
College of Information and Engineering, Hebei GEO University, Shijiazhuang 050031, China

Corresponding author: Xue-Jing Liu (lxjpass@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61806069, in part by the Scientific Research Project Program of Colleges and Universities in Hebei Province under Grant QN2019075, in part by the Scientific Research Project Program of Colleges and Universities in Hebei Province under Grant ZD2016005, and in part by the Natural Science Foundation of Hebei Province under Grant F2016403055.

ABSTRACT This article investigates how to use the estimation of distribution algorithm based on Lévy flight to solve the set-union knapsack problem (SUKP). First, the mathematical model of the SUKP is introduced. Then, a quadratic greedy repair and optimization algorithm (Q-GROA), which deals with infeasible solutions, is proposed. Thereby, a new approach, the estimation of distribution algorithm based on Lévy flight (LFEDA) combined with the Q-GROA is also proposed to solve the SUKP. A number of experiments are performed on the SUKP datasets to evaluate the performance of our proposed model. The results verify that the proposed method is significantly better than other algorithms with respect to the solution's performance.

INDEX TERMS Set-union knapsack problem, estimation of distribution algorithm, Lévy flight, repair and optimization.

I. INTRODUCTION

The 0-1 knapsack problem (0-1 KP) [1] is a classic NP-hard problem; it has important applications in project selection, cargo loading, investment decision-making and so on. The 0-1 KP is as follows: given n items, where the item i ($1 \leq i \leq n$) has weight w_i and profit p_i, and a knapsack with capability C, the 0-1 KP describes packing a number of items into the knapsack without exceeding the capability C, so that the items in it have the maximal value among all possible combinations.

The set-union knapsack problem (SUKP) [2]–[5], which is a variation of the 0-1 KP [1], is an NP-complete problem, and it has important applications in many fields such as machine loading in flexible manufacturing systems [2]–[4], the allocation of storage to fields in a database [4], data stream compression and smart cities. First, Goldschmidt *et al.* [2] studied the SUKP and proposed a dynamic program running in exponential time to solve it exactly, but the dynamic program is not applicable to this problem. Arulselvan and Ashwin [4] presented an approximation algorithm named the A-SUKP for solving the SUKP with the additional restriction that the A-SUKP approximates a special case of the SUKP within a constant factor, and the A-SUKP greatly improved the speed for solving the SUKP. Then, He *et al.* [5] proposed a novel binary artificial bee colony algorithm (BABC) with the greedy repair and optimization algorithm (S-GROA) for solving the SUKP, and the BABC provides a new idea for solving the SUKP using evolutionary algorithms (EAs). Baykasoğlu *et al.* [6] used the binary weighted superposition attraction algorithm (bWSA) to solve the SUKP. At present, the SUKP is a hotspot of KP problems, and there are fewer EAs to solve it. Therefore, estimation of distribution algorithms (EDAs) [7]–[9] are a trending research topic in the field of evolutionary computations and will be explored to solve the SUKP.

EDAs are new stochastic optimization algorithms based on statistical principles, which explore the space of potential solutions by establishing a probabilistic model and sampling the candidate solutions based on the model. The explicit application of probabilistic models in optimization provides more important advantages than other types of

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

metaheuristics. At present, the related theoretical research of EDAs has achieved many results, and EDAs have good optimization performance in many fields, including combinatorial problems, pattern matching, secondary distributions, mechanical structure design, numerical optimization problems and so on. Population-based incremental learning (PBIL) is the earliest EDA model [10]. EDAs were first proposed in 1996 [7] and had been rapidly developed over the following years. Hauschild and Pelikan [11] discussed the advantages of EDAs using probability models over other types of metaheuristics and outlined many different types of EDAs. Pelikan *et al.* [12] proposed a Bayesian optimization algorithm based on EDAs. Aickelin *et al.* [13] used EDAs to implement the explicit learning of rule-based nurse scheduling, and an ant-miner method was used to improve each solution that was generated in each generation. Zhang *et al.* [14] proposed multi-objective EDAs based on the rule model for the variable links of continuous multi-objective optimization problems. Chang *et al.* [15] used three mechanisms to effectively combine particle swarm optimization with EDAs, and proposed a new particle swarm EDA framework that retained the original unique functions of the two algorithms. A new type of EDA was proposed to solve a special kind of nonlinear bilevel programming problem [16]. Ceberio *et al.* [17] proposed a hybrid method consisting of an EDA and a variable neighborhood search to solve the problem of flow shop scheduling. Inspired by the organization and division of the bee colony, Novais *et al.*[18] used a clustering method in the target space and combined the EDAs to propose a hybrid multi-objective estimation distribution algorithm based on an artificial bee colony and clustering to solve continuous variables. Luo *et al.* [19] divided the particle swarm into several subgroups and built a probability model for each subgroup based on the EDAs to solve the complex multi-objective optimization problem of reservoir flood control operations. Alden and Miikkulainen [20] introduced a Markovian learning EDA by employing a Markov random field model, and applied it in computational biology and autonomous agent design. Pérez-Rodríguez and Hernández-Aguirre [21] combined EDAs and the Mallows distribution in order to build better sequences for flexible job shop scheduling problems with process plan flexibility. Min *et al.*[22] used the Monte Carlo method and transfer learning technique to propose a domain EDA based on domain adaptation and nonparametric estimation to solve the dynamic multi-objective optimization problem. Based on Pareto EDAs, Shao *et al.* [23] constructed three probability models to solve the multi-objective distributed no-wait flow shop scheduling problem with the sequence correlation setup time and achieved better results. While EDA applications have been very extensive and achieved certain results, they are less frequently applied to the knapsack problem; therefore, this paper will study EDAs to solve the SUKP problem.

The remainder of this paper is organized as follows. In section II, the definition and the mathematical model of the SUKP are introduced. Section III introduces EDAs. In the following section, Lévy flight is reviewed, and an efficient quadratic greedy repairing and optimization algorithm (Q-GROA) that can deal with the infeasible solution of the SUKP is presented. Subsequently, we applied Lévy flight and the Q-GROA to EDAs (named LFEDAs) to form a new approach for solving the SUKP. In section V, a large-scale comparison between LFEDAs and other methods including the A-SUKP, binary differential evolution (binDE) [24], BABC [5], binary weighted superposition attraction (bWSA) [6] and EDAs for solving three real-world problems is conducted, and the experimental results show that LFEDAs are superior to other algorithms with respect to efficiency and performance. We summarize our results and suggest future research directions in section VI.

II. DEFINITION AND MATHEMATICAL MODELS

The SUKP [2],[3],[5] contains a set of items $S = \{1,2,\ldots,m\}$ and a set of elements $U = \{1,2,\ldots,n\}$. Each item $i(i = 1,2,\ldots,m)$ has a nonnegative value p_i corresponding to a subset of elements, and each element j $(j = 1,2,\ldots,n)$ has a nonnegative weight w_j. For a nonempty set $A \subseteq S$, the weight of set A is $W(A) = \sum_{j \in \bigcup_{i \in A} U_i} w_j$ and the profit of set A is $P(A) = \sum_{i \in A} p_i$. Finding a subset $S^* \subseteq S$ is the purpose of the SUKP, which will maximize $P(S^*)$ and meet the conditions $W(S^*) \leq C$, where C is a nonnegative value and is the given capacity of the knapsack. Without loss of generality, the SUKP can be expressed as follows:

$$\text{maximize} \quad P(A) = \sum\nolimits_{i \in A} p_i \tag{1}$$

$$\text{subject to} \quad W(A) = \sum\nolimits_{j \in \bigcup_{i \in A} U_i} w_j \leq C, \quad A \subseteq S \tag{2}$$

The above mathematical model cannot be solved using EAs. Therefore, an integer model based on the above model is introduced, which is easier for EAs.

Let $A_{\boldsymbol{X}} = \{i | x_i \in \boldsymbol{X}, x_i = 1, i = 1,2,\ldots,m\} \subseteq S$, where $\boldsymbol{X} = [x_1, x_2,\ldots,x_m] \in \{0,1\}^m$ is an m-dimension 0–1 vector. Obviously, the relationship between $\boldsymbol{X}$ and $A_{\boldsymbol{X}} \subseteq S$ is a one-to-one mapping. Through this one-to-one mapping relationship, the SUKP can be modeled as follows:

$$\text{maximize} \quad f(\boldsymbol{X}) = \sum\nolimits_{i=1}^{m} x_i p_i \tag{3}$$

$$\text{subject to} \quad W(A_{\boldsymbol{X}}) = \sum\nolimits_{j \in \bigcup\limits_{i \in A_{\boldsymbol{X}}} U_i} w_j \leq C \tag{4}$$

$$X = [x_1, x_2, \ldots, x_m] \in \{0, 1\}^m \tag{5}$$

According to the model above, all 0-1 vectors of $\boldsymbol{X}$ are the only potential solutions of the SUKP, and only the solutions that satisfy the limit condition (4) and (5) are the feasible solutions.

III. ESTIMATION OF DISTRIBUTION ALGORITHMS

EDAs are EAs that are stochastic optimization algorithms. They explore the space of potential solutions by using the constructed explicit probabilistic model to find the most promising solution. In EDAs, there is no traditional crossover or mutation. Usually, EDAs start with the initial population that is generated based on a uniform distribution in the

solution space. Then, the population is evaluated using the fitness function, and the population is sorted according to the fitness function. For example, from the sorted population, when we use 60% as the selection operator, the subset of the top 60% best solutions is selected. The probabilistic model is constructed using the selected solution method in order to estimate the probability distribution of the solution. Once a probabilistic model has been established, some new solutions can be sampled based on this probabilistic model and a new population can be generated. Until the termination conditions are met, we repeat this process, and each iteration is usually called a generation of the EDAs. The basic procedure of EDAs is shown in Algorithm 1. Here, NP is the population size, $\boldsymbol{X}$ is the n-dimensional vector, $Miter$ is the maximum number of iterations, P_m is the selection factor, $\boldsymbol{B}$ is the optimal solution, and the function f is the fitness value.

Algorithm 1 EDAs Pseudocode

Input: parameters NP, $Miter$, P_m, t
Output: $\boldsymbol{B}$ and $f(\boldsymbol{B})$

1) generate initial population $P(0)=\{X_k(0)|1\leq k \leq NP\}$ randomly;
2) compute fitness $f(P(0))$;
3) for $t \leftarrow 1$ to $Miter$
4) sort $P(t)$ by $f(P(t))$;
5) generate the selection population $S(t)$ from $P(t)$ using the selection operator P_m;
6) build probabilistic model $M(t)$ from $S(t)$;
7) generate new feasible solutions $P'(t)$ by sampling $M(t)$;
8) generate new $P(t)$ by $P'(t)$ and $P(t)$;
9) calculate $\boldsymbol{B}$ and $f(\boldsymbol{B})$;
10) end for
11) return $\boldsymbol{B}$ and $f(\boldsymbol{B})$.

An important step in distinguishing EDAs from many other metaheuristic methods is the construction of the model and the attempt to capture the probability distribution of a promising solution. This task is not trivial because the goal is not a perfect representation of a promising solution for the population, but rather a more general distribution that captures the functionality of the chosen solution and makes it better than any other candidate. In addition, we must ensure that models are efficiently built and sampled.

IV. EDAS BASED ON LÉVY FLIGHT

A. LÉVY FLIGHT

The Lévy distribution is a probability distribution [25] that was proposed by the French mathematician Lévy in the 1930s. As a random movement, Lévy flight is an alternate way of walking with a short distance search combined with occasional longer distance walking [26], which is a random search path that obeys a Lévy distribution. So far, scholars have proven that the foraging trajectory of many animals and insects in nature, such as albatrosses, bees and fruit flies, conform to a Lévy distribution (Lévy flights) pattern. Lévy flight can increase the diversity of the population and broaden the scope of the search, which makes it is easier to jump out of local optimal points in the intelligent optimization algorithm.

The Lévy flight location update formula is as follows [27]:

$$\boldsymbol{X}_i^{t+1} = \boldsymbol{X}_i^t + \alpha \oplus Levy(\lambda), \quad i = 1, 2, \ldots, n \tag{6}$$

where $\boldsymbol{X}_i$ is the position of the i-th individual, $\boldsymbol{X}_i^t$ represents $\boldsymbol{X}_i$'s t generation position, $\oplus$ is point-to-point multiplication, α represents the step control amount, and Lévy(λ) is a random search path. The latter satisfies the following:

$$\text{Lévy} \sim u = t^{-\lambda}, \quad 1 < \lambda \leq 3 \tag{7}$$

At present, Lévy flight has been applied to firefly algorithms (FAs) [27], [28], moth search algorithms (MSAs) [29], cuckoo search (CS) [30] and many state-of-the-art algorithms because Lévy flight significantly improves the search ability of metaheuristic algorithms.

B. QUADRATIC GREEDY REPAIRING AND OPTIMIZATION

The SUKP is a constrained optimization problem that will definitely generate infeasible solutions when metaheuristic algorithms are used to solve it. Because infeasible solutions reduce the effectiveness of the algorithm, how to deal with infeasible solutions is the key problem that needs to be solved. The most common methods for dealing with infeasible solutions are the repair approach, the penalty function approach, the purist approach, the separation method and the hybrid approach [31], [32]. They have their own advantages and disadvantages for the general constrained optimization problem, but for the 0-1 KP, Michalewicz [33] showed that the repair approach is more suitable than any other approach to address nonfeasible solutions. He *et al.* [5], [34] proposed a repair and optimization algorithm for solving the infeasible solutions of randomized time varying knapsack problems, the discount {0-1} knapsack problem (D{0-1}KP) and the SUKP. On the basis of the S-GROA [5], Q-GROA is proposed to handle the infeasible solutions of the SUKP.

According to the SUKP, suppose that c_i ($0<= c_i <= m$) (i =1,2,...,n) counts the number of occurrences of the element $i(i \in U)$ in the subsets U_1, U_2,...,U_m. *Let* $W\_N_j = \sum_{i\in U_j}(w_i/c_i)(j = 1, 2, \ldots, m)$. Let $H[1\ldots m]$ be the descending subscript order in which the Quicksort algorithm is employed to sort all items in S in descending order according to p_j/W_N_j (j =1,2,...,m). For an arbitrary m-dimensional 0–1 vector $\boldsymbol{Y}=[y_1,y_2,\ldots,y_m]\in\{0,1\}^m$, we simplify the notation as $A_{\boldsymbol{Y}} =\{i|y_i \in \boldsymbol{Y} \,\&\&\, y_i = 1, 1\leq i \leq m\}$. The detailed steps of the Q-GROA are shown in Algorithm 2.

In the Q-GROA, the Quicksort algorithm's time complexity is $O(m\log m)$, where step 1 calculates the parameters c_i (i =1,2,...,n), $W\_N_j(j$ =1,2,...,m) and $H[1\ldots m]$, and the time complexity is $O(m\log m) + O(mn)$. Step 2 assigns 0 to the potential solution Y, and the time complexity is $O(m)$. A repair stage (steps 3 to 7) is used to obtain a preliminary feasible solution, and the time complexity is $O(mn)$. Without considering the elements that have been

Algorithm 2 Q-GROA

Input: $\boldsymbol{X}=[x_1,x_2,\ldots,x_m]\in\{0,1\}^m$
Output: $\boldsymbol{Y}=[y_1,y_2,\ldots,y_m]\in\{0,1\}^m, f(\boldsymbol{Y})$
1) calculate c_i ($i=1,2,\ldots,n$), W_N_j($j=1,2,\ldots,m$) and $H[1\ldots m]$
2) for $i=1$ to m do $y_i=0$ endfor
3) for $i=1$ to m do
4) if($x_{H[i]}=1\&\&W(A_Y\cup\{H[i]\})\leq C$) then
5) $y_{H[i]}=1, A_Y=A_Y\cup\{H[i]\}$
6) endif
7) endfor
8) do not consider the elements that have been added to the knapsack, recalculate $c_i(i=1,2,\ldots,n)$,$W\_N_j(j=1,2,\ldots,m)$ and $H[1\ldots m]$
9) for $i=1$ to m do
10) if($x_{H[i]}=0\&\&W(A_Y\cup\{H[i]\}\leq C$) then
11) $y_{H[i]}=1, A_Y=A_Y\cup\{H[i]\}$
12) endif
13) endfor
14) return($\boldsymbol{Y}$,$f(\boldsymbol{Y})$)

added to the knapsack (step 8), the frequencies of the items that are not included in the knapsack and the weight of each item after the correction are recalculated, and quicksort is used to reassign $H[1\ldots m]$ with a time complexity of $O(m\log m)+O(mn)$. In the optimization phase (steps 9 to 13), the feasible solution $\boldsymbol{Y}$ is further optimized and the feasible solution is obtained with a time complexity of $O(mn)$. Therefore, the total time complexity is $O(m\log m)+O(mn)+O(m)+O(mn)+O(m\log m)+O(mn)+O(mn)=O(mn)$. Obviously, the Q-GROA is an efficient method for handling the infeasible solutions of the SUKP.

C. APPLICATION OF EDAS BASED ON LÉVY FLIGHT TO SUKP

In this section, the Q-GROA and Lévy flights are applied to EDAs for the SUKP in order to deal with infeasible solutions and increase the diversity of solutions. Therefore, Algorithm 1 needs to be further modified in the following three aspects.

(1) Before generating the initial population, according to p_j/W_N_j ($j=1,2,\ldots,m$), all items in S are sorted in descending order, and the index of each ordering item is stored in array $H[1\ldots m]$.

(2) To increase the diversity of the population, the Lévy flight strategy is introduced with a flight probability of α when generating new populations.

(3) The Q-GROA is used to repair and optimize the potential solutions that are generated in every iteration, and the output of the objective function of the feasible solutions is considered the fitness.

The detailed steps of the EDAs based on Lévy flight (LFEDAs) for the SUKP are shown in Algorithm 3.

To show the LFEDAs more clearly, a flowchart is shown in Fig. 1.

Algorithm 3 LFEDAs Pseudo-Code

input: knapsack parameter P, W, C, m, n
output: approximate solution $\boldsymbol{Y}$ and the objective function $f(\boldsymbol{Y})$
1) initialize parameter: maxiteration $MaxIter$, iteration t, the population size NP, flight probability α, selection operator p_m
2) $H[1\ldots m]\leftarrow$ Quicksort(p_j/W_N_j) ($j=1,2,\ldots,m$)
3) generate initial population $P(0)$ of NP individuals randomly, and apply the Q-GROA to $P(0)$
4) evaluate $f(P(0))$
5) while $t<MaxIter$ do
6) sort $P(t)$ by $f(P(t))$
7) generate selection population $S(t)$ from $P(t)$ by p_m
8) build probabilistic model $M(t)$ from $S(t)$
9) for $i=1$ to NP do
10) if rand()$>\alpha$ then
11) generate new individual i by sampling $M(t)$
12) else
13) generate new individual i by Lévy flights
14) end if
15) apply the Q-GROA on individual i
16) recalculate $f(P(t))$ and probabilistic model $M(t)$
17) end for
18) $t=t+1$
19) find the best individual Y and $f(Y)$
20) end while
21) return Y and $f(Y)$

Let $MaxIter$ and NP be the constant times of $\max\{m, n\}$, where $N=\max\{m, n\}$. In step 2, the time complexity of the quicksort algorithm is $O(m\log m)=O(N\log N)$. The time complexity of the Q-GROA is $O(mn)=O(N^2)$. In step 3, the time complexity is $O(mn)+O(NP^*m)=O(N^2)+O(N^2)=O(N^2)$. In steps 5-20, the time complexity is $O(MaxIter^*NP^*m^*n)=O(N^4)$. Therefore, the total time complexity of the LFEDAs is $O(N\log N)+O(N^2)+O(N^2)+O(N^4)=O(N^4)$. In other words, it is a random approximation algorithm with polynomial time complexity for solving the SUKP.

V. EXPERIMENTAL RESULTS

We use three types of large-scale SUKP instances to verify the performance of the EDAs and LFEDAs, and the SUKP instances can be downloaded from http://sncet.com/ThreekindsofSUKPinstances(EAs).rar. An $m\times n$ 0–1 matrix $M=(r_{ij})$ ($i=1,2,\ldots,m$;$j=1,2,\ldots,n$) denotes the subsets $U=\{U_1, U_2,\ldots, U_m\}$, and when $r_{ij}=1$, it denotes $j\in U_i$. The name of the three types of SUKP instances is unified as sukp $m\_n\_\alpha\_\beta$, where m is the number of items, n is the number of elements, $\alpha=(\sum_{i=1}^{m}\sum_{j=1}^{n}r_{ij})/(mn)$, α represents the density of element 1 in matrix M, $\beta=C/\sum_{j=1}^{n}w_j$, and β is the ratio of C to the sum of all elements. As shown in Table 1, when

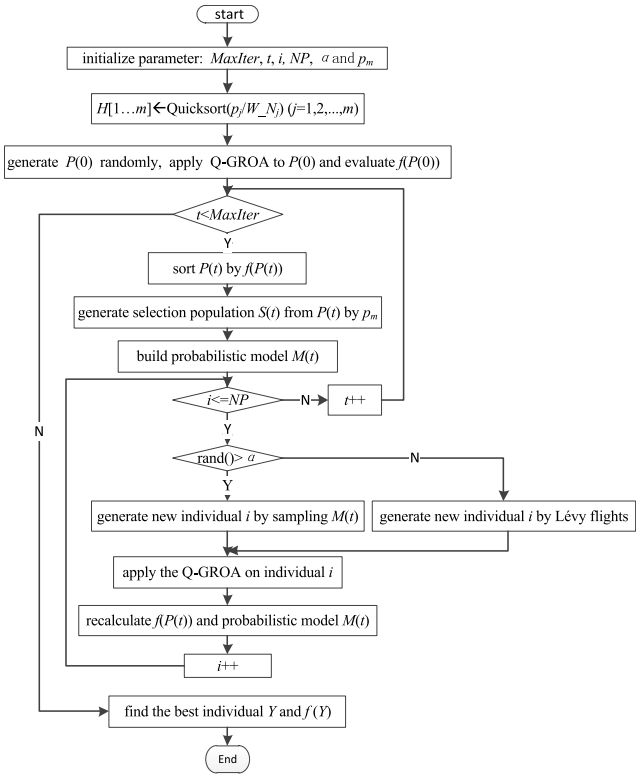


FIGURE 1. The flowchart of LFEDAs.

$m > n$, Fir1-Fir10 are the first kind of SUKP instances; when $m = n$, Sec1∼Sec10 are the second kind of SUKP instances; and when $m < n$, Thi1∼Thi10 are the third kind of SUKP instances. The experimental platform is a 1.7 GHz Intel (R) Core (TM) i3-4005U CPU with 4 GB of RAM (3.75 GB available). The operating system is Microsoft Windows 7. All algorithms are written with C language in the Code-Blocks environment, and the line charts are implemented by MATLAB7.14.0.739 (R2012A).

A. PARAMETER SETTINGS

The value of the parameter flight probability α for the LFEDAs is determined experimentally. Assume that α is 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. Due to space limitations, we conducted experiments by LFEDAs on only 6 instances, Fir1, Fir5, Sec1, Sec5, Thi1 and Thi5, after being independently run 100 times each. In Table 2 and Figs. 2-7, the results of the Kruskal–Wallis test of different α value are given. In Table 2, the p value of the Kruskal–Wallis test is given; if the p value is near zero, this casts doubt on the null hypothesis and suggests that at least one sample median is significantly different from the others. Fig. 2 to Fig. 7 show box plots of the 100 best results.

It can be seen from Table 2 and Fig. 2 - Fig. 7 that when $\alpha = 0.5$, the best solutions that are obtained by the six cases are the largest, and the median is the largest or the second largest; therefore, $\alpha = 0.5$ is a suitable choice.

The parameter settings of the LFEDAs are as follows: a population size of NP=100, a selection operator of P_m =0.6,

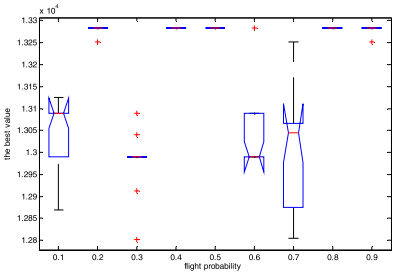


FIGURE 2. Experimental results for Fir1.

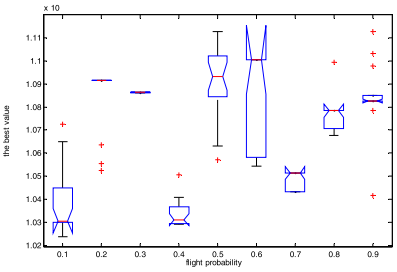


FIGURE 3. Experimental results for Fir5.

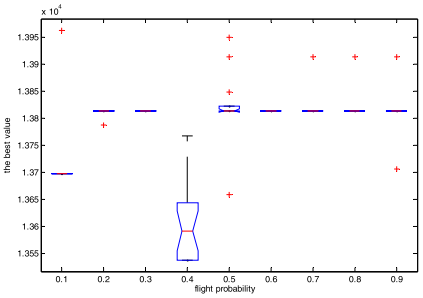


FIGURE 4. Experimental results for Sec1.

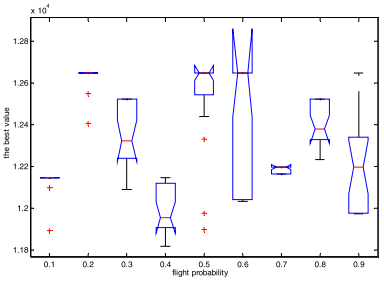


FIGURE 5. Experimental results for Sec5.

a flight probability of α =0.5, and the maxiteration is $MaxIter$=Max{m, n} for all SUKP instances, where m is the number of items and n is the number of elements in the instances.

B. COMPARISON OF EXPERIMENTAL RESULTS

To study the performance of the LFEDAs for solving the SUKP, the results are compared with those of the A-SUKP, BABC [5], binDE [24], bWSA [6] and EDAs for the three kinds of SUKP instances.

In Table 3–5, *Cbest* is the best result that is currently known for the instance. *Best*, *Mean*, and *Std* are the best value, the mean value and the standard deviation that are achieved by

TABLE 1. The id of three kinds of SUKP instances.

Id	The First Sets($m>n$)	Id	The Second Sets($m=n$)	Id	The Third Sets($m<n$)
Fir1	sukp 100_85_0.1_0.75	Sec1	sukp 100_100_0.1_0.75	Tir1	sukp 85_100_0.1_0.75
Fir2	sukp 100_85_0.15_0.85	Sec2	sukp 100_100_0.15_0.85	Tir2	sukp 85_100_0.15_0.85
Fir3	sukp 200_185_0.1_0.75	Sec3	sukp 200_200_0.1_0.75	Tir3	sukp 185_200_0.1_0.75
Fir4	sukp 200_185_0.15_0.85	Sec4	sukp 200_200_0.15_0.85	Tir4	sukp 185_200_0.15_0.85
Fir5	sukp 300_285_0.1_0.75	Sec5	sukp 300_300_0.1_0.75	Tir5	sukp 285_300_0.1_0.75
Fir6	sukp 300_285_0.15_0.85	Sec6	sukp 300_300_0.15_0.85	Tir6	sukp 285_300_0.15_0.85
Fir7	sukp 400_385_0.1_0.75	Sec7	sukp 400_400_0.1_0.75	Tir7	sukp 385_400_0.1_0.75
Fir8	sukp 400_385_0.15_0.85	Sec8	sukp 400_400_0.15_0.85	Tir8	sukp 385_400_0.15_0.85
Fir9	sukp 500_485_0.1_0.75	Sec9	sukp 500_500_0.1_0.75	Tir9	sukp 485_500_0.1_0.75
Fir10	sukp 500_485_0.15_0.85	Sec10	sukp 500_500_0.15_0.85	Tir10	sukp 485_500_0.15_0.85

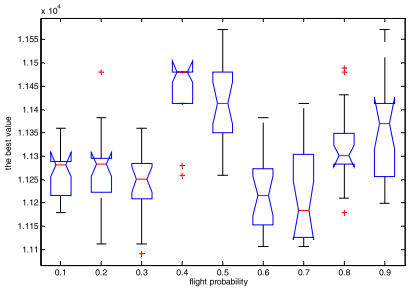


FIGURE 6. Experimental results for Thi1.

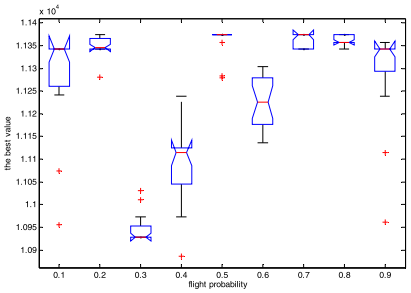


FIGURE 7. Experimental results for Thi5.

TABLE 2. The results of Kruskal-Wallis test.

instance	p	instance	p
Fir1	6.5890e-30	Fir5	1.8328e-23
Sec1	1.1144e-21	Sec5	2.6198e-19
Thi1	1.0689e-14	Thi5	6.5136e-27

all algorithms over 100 times independently. In the A-SUKP approximation algorithm, *Best* is the same as *Mean*, and they are approximate results.

It can be seen from Fig. 8(a), Fig. 9(a), and Fig. 10(a) that in terms of the best solution, the LFEDAs are inferior to the EDAs only in Thi1, and the LFEDAs are better or equal to the EDAs in other cases. It can be seen from Fig. 8(b), Fig. 9(b), and Fig. 10(b) that the LFEDAs are superior to the EDAs in terms of the mean performance for the three kinds of SUKP instances. Therefore, the Lévy flight strategy is feasible for solving the SUKP because Lévy flight increases the diversity of the solutions and it helps us find a better solution. Therefore, we only compare the LFEDAs with other algorithms in the following text.

Since the EA is a stochastic approximation algorithm, in order to evaluate its performance, it is also necessary to consider the average statistical performance. The gap fitting curve can be used to compare the average performances of all algorithms, where the Gap measure is the relative difference between the optimal value Opt and the mean value Mean, and the formula is given in (8). The closer the gap curve is to the transverse axis, the smaller its value is, and the better the average performance of the algorithm.

$$Gap = \frac{|Opt - Mean|}{Opt} * 100(\%) \tag{8}$$

The Opt of the SUKP instances is unknown, so Opt is replaced by *Cbest* when calculating the Gap among the A-SUKP, BABC, binDE and LFEDAs. The Gap curve of each algorithm is given in Fig. 11- Fig. 13.

It can be seen from Fig. 11 to Fig. 13 that the Gap value of the LFEDAs is closest to the X axis; therefore, the average performance of the LFEDAs is the best, the A-SUKP is the worst, the bWSA ranks second in most instances, and the BABC is superior to binDE in most cases. Obviously, when solving the first type of SUKP instance, the Gap value of the LFEDAs is less than 5% except for Fir10. When solving the second and third kinds of instances, the Gap values of the LFEDAs are no more than 10%, while the Gap values of the other algorithms are nearly 25%.

Tables 6-8 show the comparison of LDEDAs with other algorithms in the optimal solution of three types of instances. "+" represents LFEDAs are poor, "-" denotes LFEDAs are better, and "=" represents the same optimal solution for both algorithms. In addition, the total number of "+", "-", "=" is listed in the last three lines of the table. Obviously, it can be seen that LFEDAs are superior to other algorithms in terms of optimal solutions.

From Fig. 14, we observe that the Stds of the BABC and binDE are less than 300 in most cases, and the Stds of the LFEDAs are smaller than those of the bWSA in most cases; therefore, we conclude that the BABC and binDE are more robust algorithms than the LFEDAs and bWSA for the first

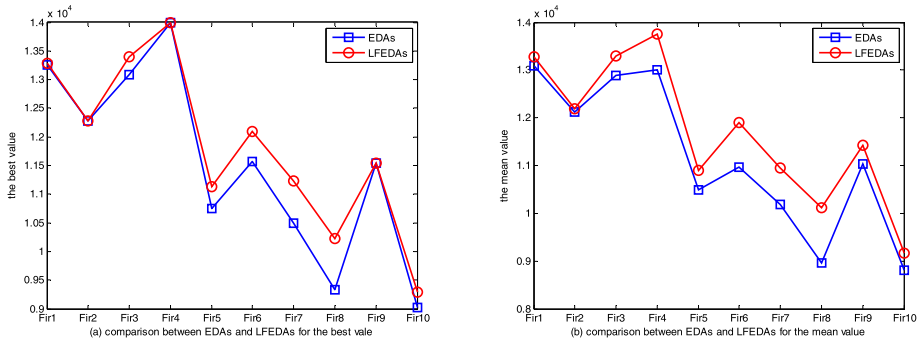


FIGURE 8. Comparison between EDAs and LFEDAs for Fir1-Fir10.

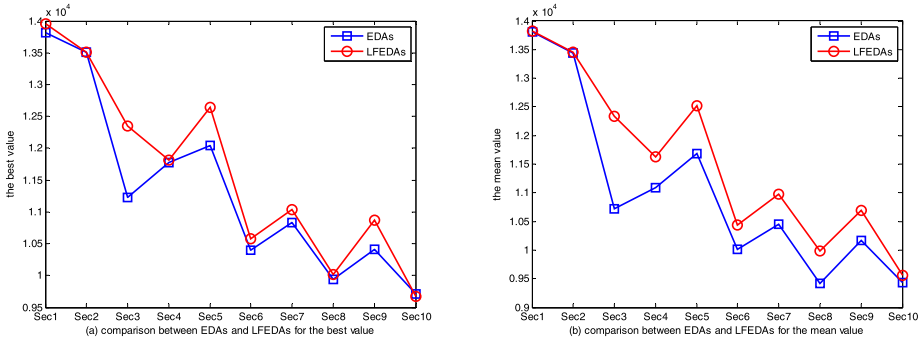


FIGURE 9. Comparison between EDAs and LFEDAs for Sec1-Sec10.

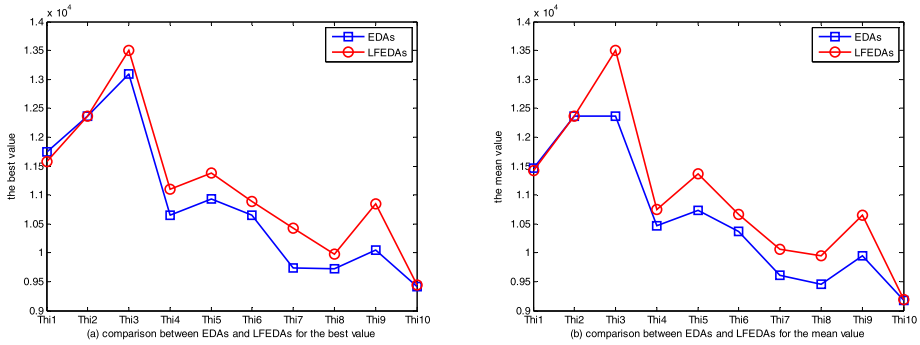


FIGURE 10. Comparison between EDAs and LFEDAs for Thi1-Thi10.

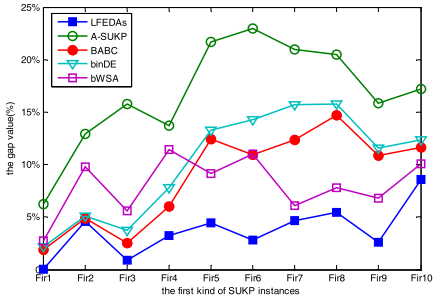


FIGURE 11. The Gap fitting curves for Fir1-Fir10.

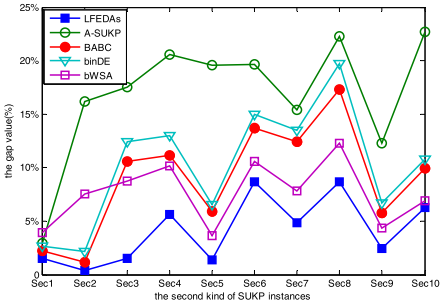


FIGURE 12. The Gap fitting curves for Sec1-Sec10.

kind of SUKP instances. From Fig. 15, we find that the Stds of the LFEDAs are less than 100 in 7 cases, the Stds of the LFEDAs are larger than those of the binDE and BABC only for the Sec5 instance, and the Stds of the LFEDAs are smaller than those of the other three algorithms on all other examples. Therefore, the LFEDAs are the most robust algorithm for

TABLE 3. The results of the first kind of SUKP instances.

algorithm	results	Fir1	Fir2	Fir3	Fir4	Fir5	Fir6	Fir7	Fir8	Fir9	Fir10
	Cbest	13283	12774	13405	14215	11413	12245	11484	10710	11722	10022
	Best	13251	**12274**	13093	**13993**	10747	11570	10488	9329	11546	9019
EDAs	*Mean*	13084.3	12120	12881.5	13008.4	10488.6	10970.5	10191.9	8964.1	11032.9	8807.9
	Std	103.83	77.03	126.63	459.9	114.09	346.98	105.48	192.99	241.55	129.67
	Best	**13283**	**12274**	**13405**	**13993**	**11127**	12088	11228	10220	**11548**	9287
LFEDAs	*Mean*	**13281.4**	**12185.9**	**13288.2**	**13753.1**	**10901.7**	**11896.1**	**10951.2**	**10123.3**	**11418.6**	**9159.9**
	Std	96.26	52.31	150.41	301.01	233.49	429.86	358.15	394.21	403.69	140.69
	Best	12459	11119	11292	12262	8941	9432	9076	8514	9864	8299
A-SUKP	*Mean*	12459	11119	11292	12262	8941	9432	9076	8514	9864	8299
	Std	0	0	0	0	0	0	0	0	0	0
	Best	13251	12238	13241	13829	10428	12012	10766	9946	10784	9090
BABC	*Mean*	13028.5	12155	13064.4	13359.2	9994.8	10902.9	10065.2	9136	10452.2	8857.9
	Std	92.63	53.29	99.57	234.99	154.03	449.45	241.45	151.9	114.35	94.55
	Best	13044	**12274**	13241	13671	10420	11661	10576	9649	10586	9191
binDE	*Mean*	12991	12123.9	12904.7	13110	9899.2	10499.4	9681.5	9020.9	10363.8	8784
	Std	75.95	67.61	205.7	269.69	153.18	403.95	275.05	150.99	93.39	131.05
	Best	13044	12238	13250	13858	10991	**12093**	**11321**	**10435**	11540	**9681**
bWSA	*Mean*	12915.67	11527.41	12657.65	12585.35	10366.21	10901.59	10785.74	9875.72	10921.58	9013.09
	Std	185.45	332.27	319.58	302.66	257.1	508.79	361.45	360.29	351.69	204.85

TABLE 4. The results of the second kind of SUKP instances.

algorithm	results	Sec1	Sec2	Sec3	Sec4	Sec5	Sec6	Sec7	Sec8	Sec9	Sec10
	Cbest	14044	13508	12522	12317	12695	11425	11531	10927	10960	10194
	Best	13814.0	**13508.0**	11228.0	11773.0	12038.0	10402.0	10832.0	9938.0	10414.0	9716.0
EDAs	*Mean*	13808.3	13444.9	10724.5	11090.5	11679.2	10011.8	10449.3	9411.1	10167.8	9431.6
	Std	25.10	46.50	164.60	272.10	145.40	173.50	118.40	185.10	172.90	95.20
	Best	13950.0	**13508.0**	**12350.0**	**11812.0**	**12646.0**	10582.0	11041.0	10021.0	**10872.0**	9674.0
LFEDAs	*Mean*	**13825.6**	**13453.1**	**12333.5**	**11625.6**	**12515.7**	**10435.0**	**10970.8**	**9980.4**	**10691.2**	**9554.1**
	Std	46.80	76.60	9.50	221.30	250.60	147.20	24.90	10.20	75.30	92.10
	Best	13634.0	11325.0	10328.0	9784.0	10208.0	9183.0	9751.0	8497.0	9615.0	7883.0
A-SUKP	*Mean*	13634.0	11325.0	10328.0	9784.0	10208.0	9183.0	9751.0	8497.0	9615.0	7883.0
	Std	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Best	13860.0	**13508.0**	11846.0	11521.0	12186.0	10382.0	10626.0	9541.0	10755.0	9318.0
BABC	*Mean*	13734.9	13352.4	11194.3	10945.0	11945.8	9859.7	10101.1	9032.9	10328.5	9180.7
	Std	70.76	155.14	249.58	255.14	127.8	177.02	196.99	194.18	91.62	84.91
	Best	13814.0	13407.0	11535.0	11469.0	12304.0	10382.0	10462.0	9388.0	10546.0	9312.0
binDE	*Mean*	13675.9	13212.8	10969.4	10717.1	11864.4	9710.4	9975.8	8768.4	10227.7	9096.1
	Std	119.53	287.45	302.52	341.08	160.42	208.48	185.57	212.24	103.32	145.45
	Best	**14044.0**	13407.0	12271.0	11804.0	12644.0	**11113.0**	**11199.0**	**10915.0**	10827.0	**10082.0**
bWSA	*Mean*	13492.71	12487.88	11430.23	11062.06	12227.56	10216.71	10624.79	9580.64	10482.80	9487.71
	Std	325.34	718.23	403.33	423.90	308.11	351.12	266.46	411.83	165.62	262.44

the second kind of SUKP instances. From Fig. 16, the Stds of the LFEDAs are less than 200 in all cases, the Stds of the LFEDAs are slightly larger than the BABC only for the Thi10 instance and slightly larger than the BABC and binDE for the Thi4 instance, and the Stds of the LFEDAs are smaller than those of the other three algorithms for all other examples. There is no doubt that the LFEDAs are the most robust of all algorithms in the third categories of SUKP cases.

Based on the above analysis, we draw the following two conclusions.

TABLE 5. The results of the third kind of SUKP instances.

algorithm	results	Thi1	Thi2	Thi3	Thi4	Thi5	Thi6	Thi7	Thi8	Thi9	Thi10
	Cbest	12045	12369	13696	11298	11568	11763	10483	10338	11094	10104
	Best	11752.0	**12369.0**	13095.0	10646.0	10936.0	10655.0	9739.0	9725.0	10048.0	9419.0
EDAs	*Mean*	11459.2	**12369.0**	12368.6	10470.8	10731.7	10373.1	9614.3	9463.0	9942.0	9172.4
	Std	123.81	0.00	319.21	157.11	123.04	207.26	101.52	182.97	58.56	115.00
	Best	11572.0	**12369.0**	**13505.0**	**11106.0**	**11374.0**	10883.0	**10420.0**	9978.0	**10845.0**	9438.0
LFEDAs	Mean	**11421.5**	**12369.0**	**13500.7**	**10751.5**	**11363.1**	**10661.8**	**10055.7**	**9942.2**	**10646.7**	**9185.9**
	Std	88.97	0.00	8.75	177.30	27.66	77.86	103.52	107.40	74.48	121.19
	Best	10231.0	10483.0	11508.0	8621.0	9961.0	9618.0	8672.0	8064.0	9559.0	8157.0
A-SUKP	*Mean*	10231.0	10483.0	11508.0	8621.0	9961.0	9618.0	8672.0	8064.0	9559.0	8157.0
	Std	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Best	11664.0	**12369.0**	13047.0	10602.0	11158.0	10528.0	10085.0	9456.0	10823.0	9333.0
BABC	*Mean*	11182.7	12081.6	12522.8	10150.6	10775.9	9897.9	9537.5	9090.0	10483.4	9085.6
	Std	183.57	193.79	201.35	152.91	116.8	186.53	184.62	156.69	228.34	115.62
	Best	11352.0	**12369.0**	13024.0	10547.0	11152.0	10528.0	9883.0	9352.0	10728.0	9218.0
binDE	*Mean*	11075.0	11875.9	12277.5	10085.4	10661.3	9832.3	9314.6	8847.0	10159.4	8919.6
	Std	119.42	336.94	234.24	160.60	149.84	232.72	191.59	210.91	198.49	168.90
	Best	**11947.0**	**12369.0**	**13505.0**	10831.0	11538.0	**11377.0**	10414.0	**10077.0**	10835.0	9603.0
bWSA	*Mean*	11233.16	11342.70	12689.09	10228.07	11105.09	10452.03	9778.03	9203.52	10607.21	9141.94
	Std	216.67	474.76	336.51	286.92	197.78	416.76	221.49	303.12	191.86	180.42

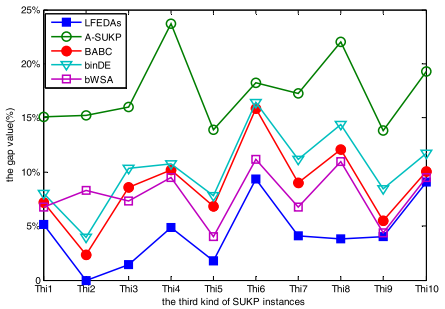


FIGURE 13. The Gap fitting curves for Thi1-Thi10.

TABLE 6. Comparison of LFEDAs with other algorithms on Fir1-Fir10.

algorithm	EDAs	A-SUKP	BABC	binDE	bWSA
Fir1	-	-	-	-	-
Fir2	=	-	-	=	-
Fir3	-	-	-	-	-
Fir4	=	-	-	-	-
Fir5	-	-	-	-	-
Fir6	-	-	-	-	+
Fir7	-	-	-	-	+
Fir8	-	-	-	-	+
Fir9	-	-	-	-	-
Fir10	-	-	-	-	+
+	0	0	0	0	4
=	2	0	0	1	0
-	8	10	10	9	6

(1) Among all the algorithms, LFEDAs provide better solutions than EDAs, which obviously shows that the Lévy flight strategy for EDAs is an effective way to solve the SUKP.

TABLE 7. Comparison of LFEDAs with other algorithms on Sec1-Sec10.

algorithm	EDAs	A-SUKP	BABC	binDE	bWSA
Sec1	-	-	-	-	+
Sec2	-	-	=	-	-
Sec3	-	-	-	-	-
Sec4	-	-	-	-	-
Sec5	-	-	-	-	-
Sec6	-	-	-	-	+
Sec7	-	-	-	-	+
Sec8	-	-	-	-	+
Sec9	-	-	-	-	-
Sec10	-	-	-	-	+
+	0	0	0	0	5
=	0	0	0	0	0
-	10	10	10	10	5

TABLE 8. Comparison of LFEDAs with other algorithms on Thi1-Thi10.

algorithm	EDAs	A-SUKP	BABC	binDE	bWSA
Thi1	-	-	-	-	+
Thi2	=	-	=	=	=
Thi3	-	-	-	-	=
Thi4	-	-	-	-	-
Thi5	-	-	-	-	-
Thi6	-	-	-	-	+
Thi7	-	-	-	-	-
Thi8	-	-	-	-	+
Thi9	-	-	-	-	-
Thi10	-	-	-	-	-
+	0	0	0	0	3
=	1	0	1	1	2
-	9	10	9	9	5

(2) When using LFEDAs to solve the SUKP, the calculation results are better than those of other algorithms. The results show that the algorithm based on the probability model is more suitable for solving SUKP than other algorithms.

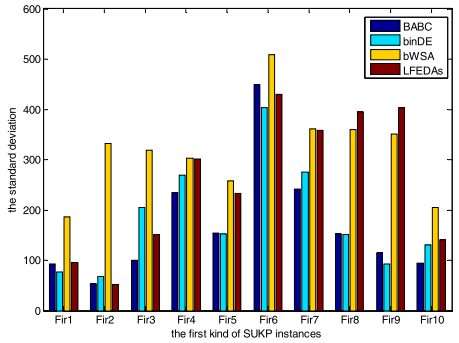


FIGURE 14. The Stds for Fir1-Fir10.

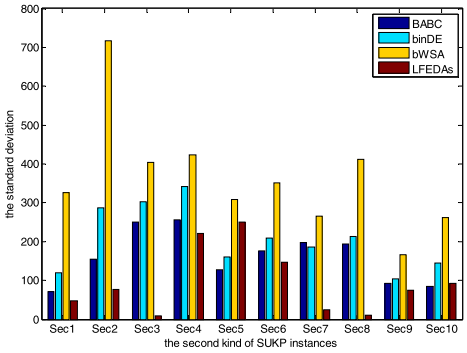


FIGURE 15. The Stds for Sec1-Sec10.

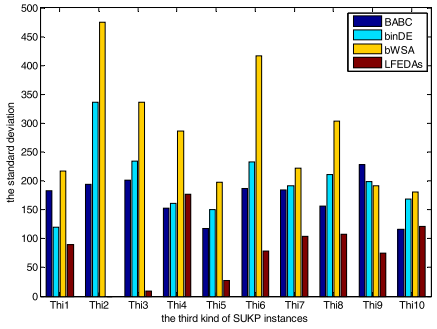


FIGURE 16. The Stds for Thi1-Thi10.

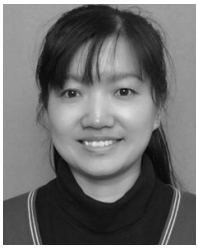
VI. CONCLUSION

In this paper, EDAs based on Lévy flight are proposed to solve the SUKP problem. To apply the LFEDAs to solve the SUKP, we propose the Q-GROA to solve the infeasible solutions. Compared with the BABC and binDE using the S-GROA, LFEDAs are more effective than the BABC and binDE, and the average solution performance is better. In fact, the Q-GROA is a general-purpose algorithm that can be applied to other EAs (e.g., the artificial algae algorithm (AAA) [35], the fireworks algorithm (FWA) [36], brainstorming optimization (BSO) [37], particle swarm optimization (PSO) [38] and the moth flame optimization algorithm (MFO) [39] to solve the infeasible solutions of the SUKP. Since the SUKP is an NP-complete problem and there is no pseudopolynomial to solve it, it is necessary to find an efficient and fast approximation algorithm. The research results in this paper show that using LFEDAs to design approximation algorithms is a feasible research direction. In our future research, we will continue to solve the SUKP by developing EDAs and EAs such as the AAA, FWA, BSO, PSO and MFO and will find the best performing EAs. The other two research directions are (i) using more accurate discriminators to consolidate recent machine learning methods [40], and (ii) using adaptive frameworks to improve the robustness and accuracy.

REFERENCES

[1] A. Singh, *Elements of Computation Theory*. London, U.K.: Springer, 2009.
[2] O. Goldschmidt, D. Nehme, and G. Yu, "Note: On the set-union knapsack problem," *Nav. Res. Logistics*, vol. 14, no. 6, pp. 833–842, Oct. 1994.
[3] D. A. Nehme-Haily, "The set-union knapsack problem," Ph.D. dissertation, Dept. Math., Univ. Texas Austin, Austin, TX, USA, 1996.
[4] A. Arulselvan, "A note on the set union knapsack problem," *Discrete Appl. Math.*, vol. 169, pp. 214–218, May 2014.
[5] 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.
[6] A. Baykasoğlu, F. B. Ozsoydan, and M. E. Senol, "Weighted superposition attraction algorithm for binary optimization problems," *Oper. Res.*, to be published. doi: 10.1007/s12351-018-0427-9.
[7] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I. Binary parameters," in *Proc. 4th Int. Conf. Parallel Problem Solving Nature*, London, U.K., 1996, pp. 178–187.
[8] M. Pelikan, D. E. Goldberg, and F. G. Lobo, "A survey of optimization by building and using probabilistic models," *Comput. Optim. Appl.*, vol. 21, no. 1, pp. 5–20, 2002.
[9] C. González, J. A. Lozano, and P. Larrañaga, "Mathematical modeling of discrete estimation of distribution algorithms," in *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, 6th ed. Norwell, MA, USA: Kluwer, 2001, pp. 147–163.
[10] B. Shummet, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," Dept. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-94-163, 1994.
[11] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 111–128, 2011.
[12] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "Linkage problem, distribution estimation, and Bayesian networks," *Evol. Comput.*, vol. 8, no. 3, pp. 311–340, 2000.
[13] U. Aickelin, E. K. Burke, and J. Li, "An estimation of distribution algorithm with intelligent local search for rule-based nurse rostering," *J. Oper. Res. Soc.*, vol. 58, no. 12, pp. 1574–1585, Dec. 2007.
[14] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 41–63, Feb. 2008.
[15] C. W. Ahn, J. An, and J.-C. Yoo, "Estimation of particle swarm distribution algorithms: Combining the benefits of PSO and EDAs," *Inf. Sci.*, vol. 192, pp. 109–119, Jun. 2012.
[16] Z. Wan, L. Mao, and G. Wang, "Estimation of distribution algorithm for a class of nonlinear bilevel programming problems," *Inf. Sci.*, vol. 256, pp. 184–196, Jan. 2014.
[17] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A distance-based ranking model estimation of distribution algorithm for the flow-shop scheduling problem," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 286–300, Apr. 2014.
[18] F. T. Novais, L. S. Batista, A. J. Rocha, and F. G. Guimarães, "A multiobjective estimation of distribution algorithm based on artificial bee colony," in *Proc. BRICS Congr. Comput. Intell. 11th Brazilian Congr. Comput. Intell.*, Ipojuca, Brazil, Sep. 2013, pp. 415–421.
[19] J. Luo, Y. Qi, J. Xie, and X. Zhang, "A hybrid multi-objective PSO–EDA algorithm for reservoir flood control operation," *Appl. Soft Comput.*, vol. 34, pp. 526–538, Sep. 2015.
[20] M. Alden and R. Miikkulainen, "MARLEDA: Effective distribution estimation through Markov random fields," *Theor. Comput. Sci.*, vol. 633, pp. 4–18, Jun. 2016.

[21] R. Pérez-Rodríguez and A. Hernández-Aguirre, "A hybrid estimation of distribution algorithm for flexible job-shop scheduling problems with process plan flexibility," *Appl. Intell.*, vol. 48, no. 10, pp. 3707–3734, Oct. 2018.
[22] J. Min, L. Qiu, Z. Huang, and G. G. Yen, "Dynamic multi-objective estimation of distribution algorithm based on domain adaptation and nonparametric estimation," *Inf. Sci.*, vol. 435, pp. 203–223, Apr. 2018.
[23] W. Shao, D. Pi, and Z. Shao, "A Pareto-based estimation of distribution algorithm for solving multiobjective distributed no-wait flow-shop scheduling problem with sequence-dependent setup time," *IEEE Trans. Automat. Sci. Eng.*, vol. 16, no. 3, pp. 1344–1360, Jul. 2019. doi: 10.1109/TASE.2018.2886303.
[24] A. P. Engelbrecht and G. Pampara, "Binary differential evolution strategies," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, Sep. 2007, pp. 1942–1947.
[25] C. Tsallis, S. V. F. Levy, A. M. C. Souza, and R. Maynard, "Statistical-mechanical foundation of the ubiquity of Lévy distributions in nature," *Phys. Rev. Lett.*, vol. 75, no. 20, pp. 3589–3593, Nov. 1995.
[26] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, E. J. Murphy, P. A. Prince, and H. E. Stanley, "Lévy flight search patterns of wandering albatrosses," *Nature*, vol. 381, pp. 413–415, May 1996.
[27] X.-S. Yang, "Firefly algorithm, Lévy flights and global optimization," in *Research and Development in Intelligent Systems*, vol. 26. London, U.K.: Springer-Verlag, 2010, pp. 209–218.
[28] X.-S. Yang, "Cuckoo search and firefly algorithm: Overview and analysis," in *Cuckoo Search and Firefly Algorithm*. Cham, Switzerland: Springer, 2014.
[29] G. G. Wang, "Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Comput.*, vol. 10, no. 2, pp. 151–164, Jun. 2016.
[30] X. Li and M. Yin, "Modified cuckoo search algorithm with self adaptive parameter method," *Inf. Sci.*, vol. 298, pp. 80–97, Mar. 2015.
[31] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.
[32] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Comput. Methods Appl. Mech. Eng.*, vol. 191, nos. 11–12, pp. 1245–1287, Jan. 2002.
[33] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin, Germany: Springer-Verlag, 1996.
[34] Y. He, X. Zhang, W. Li, X. Li, W. Wu, and S. Gao, "Algorithms for randomized time-varying knapsack problems," *J. Combinat. Optim.*, vol. 31, no. 1, pp. 95–117, Jan. 2016.
[35] S. A. Uymaz, G. Tezel, and E. Yel, "Artificial algae algorithm (AAA) for nonlinear global optimization," *Appl. Soft Comput.*, vol. 31, pp. 153–171, Jun. 2015.
[36] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Proc. ICSI*. Berlin, Germany: Springer-Verlag, 2010, pp. 355–364.
[37] S. Cheng, Q. Qin, J. Chen, and Y. Shi, "Brain storm optimization algorithm: A review," *Artif. Intell. Rev.*, vol. 46, no. 4, pp. 445–458, 2016.
[38] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, Australia, Nov./Dec. 1995, pp. 1942–1948.
[39] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[40] S. Bang, J. Kang, M. Jhun, and E. Kim, "Hierarchically penalized support vector machine with grouped variables," *Int. J. Mach. Learn. Cybern.*, vol. 8, no. 4, pp. 1211–1221, Aug. 2017.



XUE-JING LIU was born in 1980. She received the bachelor's degree from the School of Computer Science and Technology, Hebei University of Technology, Tianjin, in 2002, and the master's degree in computer application technology from the Beijing University of Technology, Beijing, in 2010. She is currently a Lecturer with the School of Information Engineering, Hebei GEO University. Her major research interests include swarm intelligence, evolutionary computation, and approximation algorithms.

YI-CHAO HE was born in 1969. He received the master's degree. He is currently a Professor. His main research interests include the theory and applications of evolutionary algorithms, the design and analysis of algorithms, computational complexity theory, and group testing theory.