Diversified sine cosine algorithm based on differential evolution for multidimensional knapsack problem

Shubham Gupta[a,*], Rong Su[a], Shitu Singh[b]

[a]*School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798*
[b]*Department of Mathematics, South Asian University, New Delhi 110021, India*

Abstract

The sine cosine algorithm (SCA) is one of the simplest and efficient stochastic search algorithms in the field of metaheuristics. It has shown its efficacy in solving several real-life applications. However, in some cases, it shows stagnation at local optima and premature convergence issues due to low exploitation ability and insufficient diversity skills. To overcome these issues from the SCA, its enhanced version named ISCA is developed in this paper. The proposed ISCA is designed based on modifying the original search mechanism of the SCA and hybridizing it with a differential evolution (DE) algorithm. The search procedure in the ISCA switches between the modified search mechanism of the SCA and DE based on the evolutionary states of candidate solutions and a parameter called the switch parameter. The modified SCA enhances the exploitation ability and convergence speed, while the DE maintains the diversity of the population to avoid local optimal solutions. The parameters of the ISCA are tuned in such as way that they could balance the exploration and exploitation features. Validation of the ISCA is conducted on a benchmark set of 23 continuous optimization problems through different performance measures, which reveals its effectiveness as a better optimizer for continuous optimization problems. Furthermore, the proposed ISCA is extended to develop its efficient binary version named BISCA for solving multidimensional knapsack problems. A benchmark collection of 49 instances is used for the performance evaluation of the BISCA. Comparison of results produced by the BISCA with other algorithms and previous studies indicates its better search efficiency and verifies it as an effective alternative for solving the MKP.

Keywords: Global optimization; Sine cosine algorithm; Differential evolution; Binary optimization; Multidimensional knapsack problem.

1. Introduction

Nowadays, metaheuristic algorithms have gained great attention for solving real global optimization problems due to their simplicity, exploration and exploitation skills, and local-optima avoidance ability [1, 2]. Generally, these metaheuristic algorithms are developed by modeling the behavior of natural phenomena and the food-foraging behavior of social creatures. Some of the well-known metaheuristic algorithms are particle swarm optimization (PSO) [3], firefly algorithm (FA) [4], ant colony optimization (ACO) [5], differential evolution (DE) [6], and so forth. Although these metaheuristic algorithms have shown their effectiveness in solving several real-life optimization problems, stagnation and premature convergence are the two serious issues that still exist while solving complex

*Corresponding author
Email addresses: shubham.gupta@ntu.edu.sg (Shubham Gupta), rsu@ntu.edu.sg (Rong Su), singh.shitu93@gmail.com (Shitu Singh)

multimodal problems. The No Free Lunch (NFL) theorem [7] verifies this fact and states that there does not exist a unique optimizer, which performs well on all the optimization problems. Due to this theorem, improving the search efficiency of existing metaheuristic algorithms is a trending and hot research topic.

Since the metaheuristic algorithms are originally designed for continuous optimization problems, their extension is required to deal with the diverse category of search space like binary and mixed-integer. For example, a common and simplest approach to transform the continuous variable to a discrete variable is the rounding procedure, where the continuous variables are rounded to their nearest integer value. This approach has been used by many researchers to perform their optimization tasks [8, 9, 10, 11]. Mirjalili and Lewis [12] have proposed S-shape and V-shaped transfer functions to map the continuous search space into binary search space. In addition to these approaches, some researchers have directly transformed the search equations of metaheuristics to deal with discrete search space [13, 14, 15, 16].

The sine cosine algorithm (SCA) is a recently designed metaheuristic algorithm by Mirjalili [17]. It has shown its efficiency in solving many real-life application problems such as parameter optimization of support vector regression [18], optimal power flow [19], feature selection [20, 21], image segmentation [22], structural damage detection [23], short-term hydrothermal scheduling [24], image copyright protection [25], etc. Although these applications show the excellent search behavior of the SCA due to its elite guidance-based search procedure and exploration capability, in some cases, the SCA suffers from local optima stagnation and premature convergence [26, 22]. One of the main reasons for these shortcomings is an inappropriate balance between exploration and exploitation with the algorithm. Therefore, in the literature, researchers have attempted to improve the search efficiency of the SCA through several modifications. Nayak et al. [27] have embedded the mutation scheme to propose a new version of the SCA to deal with the stagnation issues. Abd Elaziz et al. [26] have developed opposition-based learning (OBL) based SCA to speed up the convergence rate of the algorithm and to explore the unseen areas of the search space. Although this OBL-based strategy has increased the computational efforts of the algorithm, it shows promising performance on unimodal problems in improving the convergence behavior. However, on some complex multimodal problems, this OBL-based SCA is unable to provide near-optimal solutions and shows a stagnation issue. In [28], the inertia weight is introduced in the search equation of the SCA and a new modified SCA is proposed. In this algorithm, the transition control parameter is also modified and the levy-mutation is embedded. These two schemes are introduced in the SCA to avoid the situation of getting prone towards the local optima during the search procedure. Gupta and Deep [29] have used the OBL at a low jumping rate to enhance the diversity of the algorithm. In this algorithm, an issue of extra computational efforts due to OBL is avoided based on the jumping rate parameter. The personal best history of each candidate solution is also used to modify the search scheme in this algorithm. This strategy tries to involve the contribution of each individual candidate for the search process and to provide a jump from local optima by sharing the information of search space between candidate solutions. In [22], a crossover scheme and a new search equation to enhance the diversity skills of the algorithm are applied to develop an improved version of the SCA. This algorithm shows promising performance on all categories of benchmark test problems. Li et al. [30] have developed an improved version of the SCA by introducing a new approach called dimension by dimension dynamics. In this algorithm, the impact of the SCA search scheme is examined at each component of the candidate solutions. Although this algorithm shows promising performance, it requires evaluation of newly generated candidate solution corresponding to each dimension, which increases the computational cost of the algorithm. In [31], a hybrid version of the SCA and GWO is developed to utilize the balancing ability between exploitation and exploration of the GWO. Issa et al. [32] and Nenavath et al. [33] have combined the PSO with SCA to speed up the convergence rate and to strike a good balance between

exploitation and exploration. In [34], SCA and genetic algorithm (GA) are hybridized to provide a proper balance between exploration and exploitation. A brief literature review on the development of modified SCA variants can be obtained from [35, 36].

In [37], a hybrid version of SCA and DE called Hybrid SCA-DE is proposed. In Hybrid SCA-DE, after the SCA search scheme, a search procedure of the DE including mutation, crossover, and selection is applied to further exploit the new promising solutions based on the information available in the population. The algorithm has been validated on a set of 23 well-known benchmark optimization problems. Although the Hybrid SCA-DE performs better than the classical version of the SCA, it adds extra computational cost due to the additional search phase executed by the DE. Hence, the computational cost of the Hybrid SCA-DE is twice to the cost of the classical SCA. Later, Altay and Alatas [38] also proposed the hybrid version of the SCA and DE called HDESC by replacing the mutation operator of DE with the SCA search equation. Although this algorithm extracts elite guidance-based search procedure of the SCA, it discards the diversity skills of the DE mutation operator. This behavior degrades the search performance of the HDESC on multimodal problems, where the population diversity plays an important role to avoid local optimal solutions during the search procedure.

By inspiring the advantages of the SCA and DE and considering the drawbacks of the above described hybrid versions of the SCA and DE, we have proposed a new version of the SCA named ISCA. The ISCA utilizes the full advantages of the SCA and DE to provide a comparatively more efficient search mechanism without adding any extra computational efforts of evaluating candidate solutions. Both of these features such as complete utilization of SCA and DE search schemes and avoidance of extra computation cost are absent in the exiting hybrid versions of the SCA and DE as discussed above. The main goal of our proposal is to level up the exploitation skills of the SCA by modifying its search scheme and enhancing its diversity skills using the DE algorithm. The aim of improving the search performance of the SCA is to enhance the exploitation ability and to explore the neighborhood areas of the elite candidate solution obtained so far. The second extension of hybridizing improvised SCA with the DE is to maintain sufficient diversity within the algorithm. In the ISCA, the parameters are adjusted in such as way that they could manage an appropriate balance between exploration and exploitation. The ISCA is validated on a commonly used benchmark set consisting of 23 continuous optimization problems with varying difficulty levels. A comparison of performance on this test set verifies the search efficiency of the proposed ISCA. In the next step, as an application of the ISCA, its binary version named BISCA is developed based on the transformation function and repair operator and used to solve the multidimensional knapsack problem. A comparison of results shows the effectiveness of the BISCA in solving the multidimensional knapsack problem in a more efficient way as compared to other algorithms.

The rest of the paper is organized as follows: Section 2 summarizes the multidimensional knapsack problem, classical versions of the SCA and DE. The proposed enhanced version of the SCA named ISCA and its binary version named BISCA are discussed briefly in Section 3. The performance comparison and evaluation of the ISCA on a well-known benchmark set of continuous optimization problems is conducted in Section 4. In the same section, our developed BISCA is used for solving the multidimensional knapsack problem. Finally, Section 5 concludes with a summary of major results and future research suggestions.

2. Literature review

In this section, brief descriptions of the multidimensional knapsack problem, SCA and DE are provided. Moreover, the computational steps of the SCA and DE are also summarized in this section.

2.1. Multidimensional knapsack problem

The multidimensional knapsack problem (MKP) is a well-known optimization problem, which is NP-hard [39, 40]. The goal of this problem is to fill a multidimensional capacity-limited knapsack by a subset of items in an optimal way to maximize the associated profit. The selection of items depends on varying resource capacity corresponding to different knapsacks. Formally, the MKP is stated as follows:

Maximize:

$$F(X) = \sum_{j=1}^{n} c_j x_j \tag{1}$$

subject to:

$$\sum_{j=1}^{n} w_{kj} x_j \leq W_k, \quad k = 1, 2, \ldots, m \tag{2}$$

$$x_j \in \{0, 1\} \quad j = 1, 2, \ldots, n. \tag{3}$$

where n is the number of items and m is the number of knapsack constraints with capacities W_k ($k = 1, 2, \ldots, m$). $X = (x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$ is a n-dimensional vector with each coordinate from the set $\{0, 1\}$. $x_j = 1$ if and only if the j^{th} item is selected. w_{kj} is the required resource units for an item j to yield a profit c_j. The variables $c_j > 0, w_{kj} \geq 0$ and $W_j > 0$. For a well-stated MKP, it is assumed that $w_{kj} \leq W_j$ and $\sum_{j=1}^{n} w_{kj} \geq W_j$, because any violation in the first inequality will automatically imply $x_j = 0$ and the violation in the second inequality results in the elimination of k^{th} constraint from eq. (2).

Optimization methods for solving the MKP can be separated into two categories: deterministic and non-deterministic. The former category includes dynamic programming [41], branch and bound method [42], backtracking algorithm [43], implicit enumeration [44], etc. Since the MKP is NP-hard, there does not exist a deterministic algorithm that can solve this problem in a polynomial time. These methods are limited to solving small-scale problems with endurable computational time. When the number of items and the constraints increases in MKP, these deterministic methods show their computational inefficiency. On the other hand, the second category is of non-deterministic algorithms that have gained comparatively more attention for solving the MKP. These methods mainly include approximation algorithm [45], randomized algorithms [46], and metaheuristic algorithms. In the literature, various metaheuristic algorithms such as PSO [47], tabu search [48], fruit fly optimization algorithm [49], grey wolf optimizer [50], harmony search algorithm [51], etc. are applied to solve the MKP. By inspiring the performance of the metaheuristic algorithms in solving the MKP, this paper introduces a diversified sine cosine algorithm based on the differential evolution for solving the MKP more efficiently.

2.2. Sine Cosine Algorithm (SCA)

The SCA is one of simplest and newly developed algorithm in the field of population-based metaheuristics. It has been developed by Mirjalili [17] from the inspiration of mathematical features of sine and cosine trigonometric functions. The behavior of these functions are utilized in the SCA to explore the search space of the problem. In the SCA, the search agents of the population are referred to as candidate solutions and the best search agent of the population is called destination point. In the search procedure of the SCA, when the initial population is randomly generated, the search process is performed using eq. (4)

$$x_{i,j}^{t+1} = \begin{cases} \dot{x}_{i,j}^{t+1} & r_{i,j}^{t} \leq 0.5 \\ \hat{x}_{i,j}^{t+1} & r_{i,j}^{t} > 0.5 \end{cases} \tag{4}$$

where,

$$\dot{x}_{i,j}^{t+1} = x_{i,j}^{t} + r_{1,j}^{t} \times sin(r_{2_{i,j}}^{t}) \times |r_{3_{i,j}}^{t} \times x_{D,j}^{t} - x_{i,j}^{t}| \tag{5}$$

$$\hat{x}_{i,j}^{t+1} = x_{i,j}^{t} + r_{1,j}^{t} \times cos(r_{2_{i,j}}^{t}) \times |r_{3_{i,j}}^{t} \times x_{D,j}^{t} - x_{i,j}^{t}| \tag{6}$$

$$r_{1,j}^{t} = 2 - 2 \times \left(\frac{t}{t_{max}}\right) \tag{7}$$

where $r_{i,j}^{t}$ is a random number produced by using a uniform distribution with bounds 0 and 1, $x_{i,j}^{t+1}$ is the j^{th} component of the updated candidate solution X_i^{t+1}, $r_{2_{i,j}}^{t}$ and $r_{3_{i,j}}^{t}$ are random numbers from the interval $(0, 2\pi)$ and $(0, 1)$, respectively, and generated using a uniform distribution. The parameter $r_{1,j}^{t}$, given by eq.(7), is called the transition control parameter and is used to control exploration and exploitation features in the SCA. More precisely, when $r_{1,j}^{t} > 1$, this parameter focuses on exploring the search space, while the condition $r_{1,j}^{t} < 1$ indicates the exploitation ability of this parameter. Moreover, $r_{3_{i,j}}^{t}$ is a parameter that focuses on both exploration and exploitation but in a random manner. This parameter is introduced in the SCA to supports exploration ability, when the parameter $r_{1,j}^{t}$ fails to do so. The destination point is denoted by X_D. The symbol t indicates the iteration counter and the maximum iterations are represented by t_{max}. The computational steps of the SCA are explained in Algorithm 1.

Algorithm 1 Pseudo-code of the SCA

Initialize the population of the SCA with N candidate solutions randomly within the search space
Evaluate the fitness of each candidate solution using a given objective function
Select the destination point X_D from the population
while *($t < t_{max}$ and $fes < fes_{max}$)* **do**
  Obtain the value of the transition control parameter r_1^t using eq. (7)
  for *each candidate solutions* X_i^t **do**
    Obtain a new updated candidate solution X_i^{t+1} using eq. (4)
    Evaluate the fitness $f(X_i^{t+1})$ of updated candidate solution X_i^{t+1}
    $fes = fes + 1$
  end
  Memorize the destination point X_D
  $t = t + 1$
end

2.3. Differential evolution (DE)

The DE is one of the popular metaheuristic algorithms in the field of evolutionary algorithms. It has been developed by Storn and Price [6] using three elementary operators: mutation, crossover, and selection. In the DE, search agents are referred to as individuals or target vectors. On each individual of the DE, mutation, and crossover operators are applied to generate an offspring individual called a trial vector. The selection operator compares the fitness of the trial vector with the fitness of the target vector to pass the best-fitted individual in the next generation of the search

procedure. The initial population of the DE is generated the same as in other metaheuristics. After this initialization step, a mutant vector $V_i^t = (v_{i,1}^t, v_{i,2}^t, \ldots, v_{i,d}^t)$ is generated corresponding to the target vector X_i^t using one of the following mutation rules.

DE/rand/1:

$$V_i^t = X_{r_1}^t + F \times (X_{r_2}^t - X_{r_3}^t) \tag{8}$$

DE/rand/2:

$$V_i^t = X_{r_1}^t + F \times (X_{r_2}^t - X_{r_3}^t) + F \times (X_{r_4}^t - X_{r_5}^t) \tag{9}$$

DE/best/1:

$$V_i^t = X_{best}^t + F \times (X_{r_1}^t - X_{r_2}^t) \tag{10}$$

DE/best/2:

$$V_i^t = X_{best}^t + F \times (X_{r_1}^t - X_{r_2}^t) + F \times (X_{r_3}^t - X_{r_4}^t) \tag{11}$$

DE/current-to-best/1:

$$V_i^t = X_i^t + F \times (X_{best}^t - X_i^t) + F \times (X_{r_1}^t - X_{r_2}^t) \tag{12}$$

where $X_{r_1}, X_{r_2}, X_{r_3}, X_{r_4}$ and X_{r_5} are distinct individuals from the population, i.e., indices r_1, r_2, r_3, r_4, and r_5 are different from each other. Control parameter F is called scale factor, which controls the magnitude of difference vector(s) involved in the mutation rules. When the mutant vector is generated, DE executes the crossover operator to generate a trial vector. This crossover operator is implemented as follows:

$$u_{i,j}^t = \begin{cases} v_{i,j}^t & \text{if } rand_{i,j}^t < CR \text{ or } j = j_r \\ x_{i,j}^t & \text{otherwise} \end{cases} \tag{13}$$

where $U_i^t = (u_{i,1}^t, u_{i,2}^t, \ldots, u_{i,d}^t)$ is a crossover vector or trial vector and CR is a crossover rate. This CR decides how many components of the trial vector will be inherited from the target vector and mutant vector. j_r is a random index chosen from the set $\{1, 2, \ldots, d\}$ to ensure that the crossover vector will differ the target vector and the mutant vector by at least one component. When the trial vector U_i^t is generated, a selection operator is applied between the target vector and trial vector to pass the best-fitted individual in the next generation. This selection operator is executed as follows:

$$X_i^{t+1} = \begin{cases} U_i^t & \text{if } f(U_i^t) \leq f(X_i^t) \\ X_i^t & \text{otherwise} \end{cases} \tag{14}$$

where $f(X)$ indicates the fitness of the individual X. The framework of the classical DE is explained through Algorithm 2.

Due to the better exploration ability of the DE, it has been used to solve several challenging real-life application problems from different fields such as chemical engineering [52, 53], electrical engineering [54, 55], image processing [56, 57],and so forth. A detailed applications of the DE can be obtained from [58]. Although DE has adequate exploration ability, its exploitation ability is weak and converges is slow [59, 38]. In some cases, the DE also suffers from the premature convergence and stagnation issues due to improper balance between exploration and exploitation abilities [60, 38]. To reduce these shortcomings of the DE, several efforts have been done in the literature. A brief overview of them can be accessed from [58, 61].

Algorithm 2 Pseudo-code of the DE

Initialize the population of the DE with N individuals randomly within the search space
Evaluate the fitness value of each individual vector using a given objective function
Select the best individual X_{best} from the population
while *($t < t_{max}$ and $fes < fes_{max}$)* **do**
  for *Each individual* X_i^t **do**
    Apply one of the mutation operator given by eqs. (8) to (12) to generate a mutant vector V_i^t
    Apply the crossover operator using eq. (13) to generate a trial vector U_i^t
    Evaluate the fitness of trail vector
    $fes = fes + 1$
    if $f(U_i^t) < f(X_i^t)$ **then**
      $X_i^{t+1} = U_i^t, \quad f(X_i^{t+1}) = f(U_i^t)$
    else
      $X_i^{t+1} = X_i^t, \quad f(X_i^{t+1}) = f(X_i^t)$
    end
  end
  Memorize the best individual X_{best}^t
  $t = t + 1$
end

3. Proposed enhanced version of the SCA for solving the MKP

Although the classical SCA has been applied to solve several real-life application problems as discussed in Section 1, in some cases, it shows stagnation at local optima and premature convergence issues [26, 37]. These issues are caused by an inappropriate balance between diversity and convergence within the algorithm. To realize these issues in the SCA, we have drawn the diversity and convergence behavior of the algorithm in Figure 1 for two different multimodal problems namely, F9 and F21, which are stated in Table 1. The diversity represents the average distance between candidate solutions of the population [62] and the convergence behavior indicates the decay of objective function value over the optimization process. From Figures 1(a) and 1(b) of problem F5, it can be seen after certain iterations, the diversity of the population becomes zero and the algorithm is unable to provide a better solution as compared to the best solution obtained so far. This behavior shows the premature convergence issue of the SCA. On the other hand, figures 1(c) and 1(d) indicate that after certain iterations, the algorithm is unable to provide a better solution as compared to the solution recorded in previous iterations while the population has still the diversity skills. This behavior indicates the issue of stagnation at local optima. To deal with these issues of the SCA, we propose a hybridized version of the SCA with DE named ISCA. The ISCA firstly improves its exploitation and convergence abilities by modifying the search procedure of the SCA, and later, this modified SCA is hybridized with the DE search scheme to improve the diversity skill of the algorithm. These search strategies along with an appropriate parameter selection scheme establish a comparatively better balance between the exploitation and exploration within the search procedure of the ISCA.

In this section, we firstly provide a detailed description of the continuous version of the ISCA with all of its search strategies. Then we present its binary version named BISCA using transformation and repair operators to solve MKP. We also provide computational steps for the ISCA and the BISCA in this section to extract an idea about the flow of algorithms.

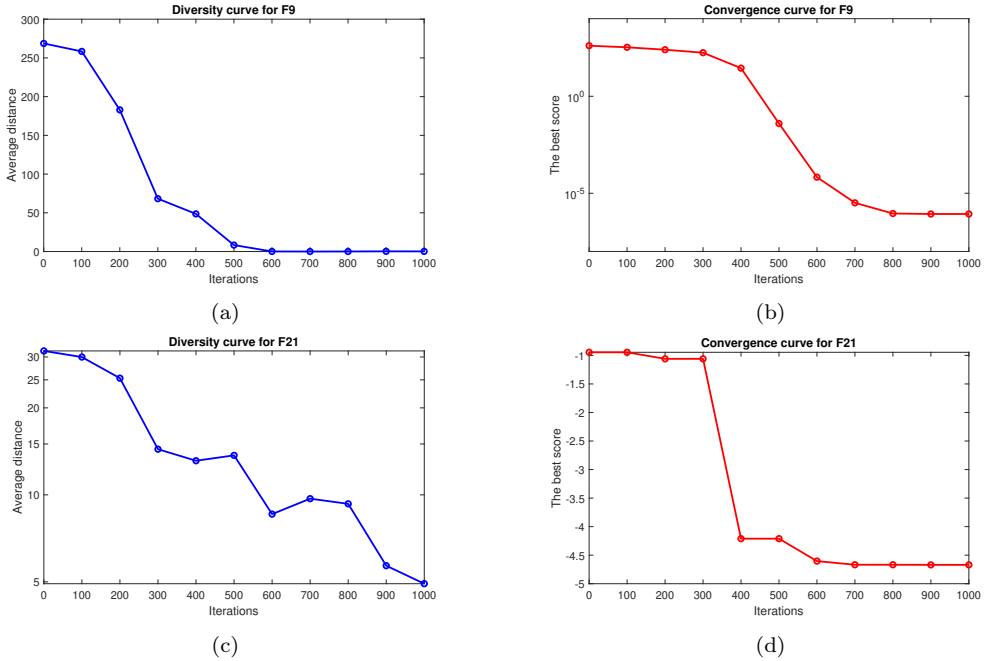

Figure 1: Premature convergence and stagnation issues in the classical SCA

3.1. Continuous ISCA

The classical version of the SCA executes the position update process of an individual X based on an absolute value of difference computed between the vectors $r_2 \times X_D$ and X. Due to the involvement of absolute value, high randomness of search is present in the original structure of the SCA and this misleads the search procedure as no proper search guidance is received at each time. These shortcomings degrade the performance of the SCA and result in weak exploitation ability and improper balance between exploitation and exploration. Based on this observation, the search scheme of the SCA can be modified to focus on exploring the obtained elite areas of the search space. The modified search equation of the SCA is stated as follows:

$$x_{i,j}^{t+1} = \begin{cases} \dot{x}_{i,j}^{t+1} & r_{i,j}^{t} > 0.5 \\ \hat{x}_{i,j}^{t+1} & r_{i,j}^{t} \leq 0.5 \end{cases} \tag{15}$$

where,

$$\dot{x}_{i,j}^{t+1} = x_{i,j}^{t} + |r_{1,j}^{t} \times sin(r_{2_{i,j}}^{t})| \times (r_{3_{i,j}}^{t} \times x_{D,j}^{t} - x_{i,j}^{t}) \tag{16}$$

$$\hat{x}_{i,j}^{t+1} = x_{i,j}^{t} + |r_{1,j}^{t} \times cos(r_{2_{i,j}}^{t})| \times (r_{3_{i,j}}^{t} \times x_{D,j}^{t} - x_{i,j}^{t}) \tag{17}$$

From the above equations, it is clear that we have just removed the absolute function applied on the vector $(r_{3_{i,j}}^{t} \times x_{D,j}^{t} - x_{i,j}^{t})$ and introduced an absolute function for the parameters $r_{1,j}^{t} \times sin(r_{2_{i,j}}^{t})$ and $r_{1,j}^{t} \times cos(r_{2_{i,j}}^{t})$. By doing such modification, now the search process performs the exploration and exploitation around the destination point X_D only with step sizes are controlled by the transition control parameter r_1 and sine-cosine trigonometric functions. These functions together with the control parameter maintain the randomness and transition from exploration to exploitation over the iterations.

Although the above modifications are sufficient to increase the exploitation ability of the algorithm, the diversity skills are compromised due to the biasedness of the search process around the

current elite candidate solution. This biasedness may create the problem of stagnation at local optima while solving complex multimodal problems. Therefore, in our proposed algorithm ISCA, we have hybridized our improvised SCA with a well-known algorithm called DE. In order to maintain the diversity within the algorithm, the DE/current-to-rand/1 mutation scheme is used, which is given by:

$$V_i^t = X_i^t + F_1 \times (X_{r_1}^t - X_i^t) + F_2 \times (X_{r_2}^t - X_{r_3}^t) \tag{18}$$

where, V_i^t is a mutant vector generated for a candidate solution X_i^t, F_1 and F_2 are the scale factors to control their associated difference vectors. In this paper, to reduce the burden of parameter-tuning and to provide an efficient movement of search from the current state of a candidate solution being updated, parameters F_1 and F_2 are defined based on the fitness and evolutionary state of candidate solutions involved in the mutation scheme as follows:

$$F_1 = \begin{cases} \frac{f(X_i^t) - f(X_{r_1}^t)}{f(X_w^t) - f(X_{r_1}^t)} & if\ f(X_w^t) \neq f(X_{r_1}^t) \\ 0 & otherwise \end{cases} \tag{19}$$

$$F_2 = \begin{cases} \frac{f(X_{r_3}^t) - f(X_{r_2}^t)}{f(X_w^t) - f(X_{r_2}^t)} & if\ f(X_w^t) \neq f(X_{r_2}^t) \\ 0 & otherwise \end{cases} \tag{20}$$

where $f(X)$ indicates the objective function value at candidate solution X, X_w is the worst candidate solution in the population. From the above equations, it can be noticed that in our definition of scale factors, the restriction of being a positive real number is ignored. The reason for this is to provide a search direction from less-fitted candidate solution to more-fitted candidate solution associated with the difference vectors. In our proposed ISCA, to generate a trail vector, the crossover rate CR is fixed to 1 to completely utilize the feature of the introduced mutation strategy in enhancing the diversity of the algorithm.

In the proposed ISCA, the hybridization of the improvised SCA and DE is done based on the evolutionary state of the candidate solution which is being updated. The evolutionary state ES_i for a candidate solution X_i is determined as follows:

$$ES_i^t = \frac{f(X_i^t) - f(X_D^t)}{f(X_w^t) - f(X_D^t)} \tag{21}$$

Based on the above-defined evolutionary state of the candidate solution X_i, the modified SCA or DE is applied to update the candidate solution X_i. To maintain the balance between exploration and exploitation, a switch parameter S_P is defined which manages the search between the modified SCA and DE. The S_P is chosen as a time-varying variable with iterations, which is given by eq. (22)

$$S_P^t = S_{P,start} - (S_{P,start} - S_{P,end}) \times \frac{t}{t_{max}} \tag{22}$$

In the ISCA, an evolutionary state for each candidate solution is determined using eq. (21) and if this is less than the parameter S_P^t, a modified SCA is applied, otherwise the DE is used to update the candidate solution. Since at the early stages of the search process, enough diversity is present in the population, the search process based on the modified SCA is focused more. On the other hand, with the increasing iterations, the DE is focused more than the improvised SCA to maintain the diversity of the population. This feature helps to minimize the chance of getting trapped at local optima and premature convergence within the algorithm. In this paper, the values of parameters $S_{P,start}$ and $S_{P,end}$ are set to 0.8 and 0.2, respectively.

3.1.1. Computational procedure and complexity of the ISCA

In this subsection, we describe the computational procedure and complexity of the ISCA. Computational procedure summarizes the steps followed by the ISCA and computational complexity provides an idea about the computational resources required to execute the ISCA.

Based on the modified search mechanism of the SCA, hybridizing it with the DE and other factors such as switch parameter, fitness-based scale factors, an improvised version of the SCA named ISCA is developed in this paper. The ISCA first initializes the population and other algorithm parameters and then it repeats the search procedure discussed in the previous section until the preset stopping criteria for the algorithm is not completed. The computational flow of the ISCA search procedure is explained in Algorithm 3.

According to the pseudo-code of the ISCA, the computational complexity can be calculated. The complexity of the initialization process in the ISCA is $O(N \times d)$, updating the parameters r_1 and S_P is $O(t_{max})$, F_1 and F_2 is $O(N \times t_{max})$. Here N is the population size, d is the dimension of the problem and t_{max} is the maximum number of iterations set as a stopping criterion for the ISCA. The total complexity for updating the positions of the population is $O(t_{max} \times N \times d)$. Hence, by summing up all the complexities, an overall order of the complexity level for the ISCA is $O(t_{max} \times N \times d)$, which is identical to the complexity of the classical SCA. Hence the proposed ISCA does not add any extra computationally expensive steps in its search process.

Algorithm 3 Pseudo-code of the ISCA

Initialize the population of the ISCA with N candidate solutions randomly within the search space
Evaluate the fitness of each candidate solution using a given objective function
Select the destination point X_D from the population
while *($t < t_{max}$ and $fes < fes_{max}$)* **do**
  for *Each individual* X_i^t **do**
    Determine the evolutionary state ES_i if the candidate solution using eq. (21)
    Evaluate the switch parameter S_P^t using eq. (22)
    if $ES_i^t < S_P^t$ **then**
      Apply the improvised SCA search procedure given by eq. (15) - (17) to obtain a new position X_i^{t+1}
    else
      Apply the DE search procedure with the proposed mutation strategy given by eq. (18) and scale factor given by eqs. (19) and (20) to obtain a new position X_i^{t+1}
    end
    Evaluate the fitness $f(X_i^{t+1})$ of newly obtained candidate solution X_i^{t+1}
    Increase the function evaluation count as $fes = fes + 1$
    Apply the greedy selection scheme as follows:
    if $f(X_i^t) < f(X_i^{t+1})$ **then**
      $X_i^{t+1} = X_i^t$ and $f(X_i^{t+1}) = f(X_i^t)$
    end
    Update the value of the transition control parameter r_1 using eq. (7)
    Memorize the destination point X_D
  end
  $t = t + 1$
end
Return the best solution or destination point X_D

3.2. Binary ISCA

In this section, the representation of candidate solutions, the initialization process of the algorithm population, and the repair operator to generate feasible solutions for solving the MKP will be described. Furthermore, the framework of the binary ISCA, denoted by BISCA, will be summarized with the utilized transfer function to transform the continuous search space into binary search space and repair operator to handle the constraints.

3.2.1. Representation of candidate solutions

The first step of developing a binary ISCA (BISCA) for solving the 0-1 MKP is the selection of an appropriate representation scheme of the population of the ISCA. Since the search space of the MKP is binary, candidate solutions within the population of the ISCA will contain only 0 and 1 as its components. More precisely, a candidate solution of the BISCA will be represented by $X = (x_1, x_2, \ldots, x_d)$, where $x_j \in \{0, 1\}$ to indicate whether an item j is selected or not.

3.2.2. Initialization of BISCA population

The population of the BISCA is consist of N candidate solutions, which will be represented as binary vector as explained above. However, random initialization procedure for each component of the candidate solution may generate an infeasible solution. To start the search procedure of the BISCA with feasible candidate solutions, a specific process is adopted by utilizing the knowledge of the MKP, especially the pseudo-utility of items. The pseudo-utility of items is computed through different methods [43]. Although in the classical 0-1 knapsack problems, it easy to determine the preference of items using the profit produced by units of resource consumption. This concept of giving preferences to items is extended through pseudo-utility [63]. In this paper, we have adopted the following definition of pseudo-utility, which is based on surrogate multipliers σ_j.

$$v_j = \frac{c_j}{\sum_{k=1}^{m} \sigma_j w_{kj}} \tag{23}$$

where v_j indicates the pseudo-utility for j^{th} item, w_{kj} is the amount of resource k consumed by item j to produce a profit c_j. With this pseudo-utility concept, a preference is given to the items having higher pseudo-utility to make more profit. In our algorithm BISCA, we have used the same procedure that has been used by Luo and Zhao [50] to initialize the population. In this procedure, the items are picked based on their preferences calculated by the pseudo-utility. The steps of generating the initial population of the BISCA is given in Algorithm 4.

3.2.3. Search process of the BISCA

The search procedure of the BISCA for solving the MKP is same as the ISCA but an additional phase is added to transform the produced continuous variables to binary variables. In the literature, several transformation function are available, which are divided into two categories namely, S-shaped and V-shaped [12]. Mirjalili and Lewis [12] have experimentally shown that the V-shaped functions have performed better than the S-shaped, when tested them in developing binary PSO. We have performed several experiments using different V-shaped functions in our proposed BISCA and found that the transformation function $T(y) = y/\sqrt{(1 + y^2)}$ has performed better than other V-shaped functions. Hence, from now onwards our proposed BISCA refers the binary version of the ISCA with $T(y) = y/\sqrt{(1 + y^2)}$ transformation function.

Hence, using the transformation function, the obtained position component $x_{i,j}$ of i^{th} candidate solution is transformed into binary variable $\hat{x}_{i,j}$ as follows:

$$\hat{x}_{i,j} = \begin{cases} 1 & rand_j < T(x_{i,j}) \\ 0 & otherwise \end{cases} \tag{24}$$

Algorithm 4 Initialization procedure in the BISCA

Initialize each candidate solution as a zero binary vector and their fitness is zero, i.e.,
$X_i = (0, 0, \ldots, 0)$ and $F(X_i) = 0, \quad \forall i = 1, 2, \ldots, N$
for *Each candidate solution* X_i **do**
  Reset the amount of consumed resources as $z_k = 0 \; \forall k \in \{1, 2, \ldots, m\}$
  Sort the items in descending order of their pseudo-utility,
  let the sorted items are $J = \{I_1, I_2, \ldots, I_n\}$
  for $j = I_1$ *to* I_n **do**
    if $rand() < 0.5$ *and* $z_k + w_{kj} \leq W_k \quad \forall k \in \{1, 2, \ldots, m\}$ **then**
      $x_{i,j} = 1, f(X_i) = f(X_i) + c_j$ and $z_k = z_k + w_{kj}$
    else
      $x_{i,j} = 0$
    end
  end
end

where T is a transformation function, $rand_j$ is a uniformly distributed random number from the interval (0,1).

With the above procedure of the ISCA, a binary vector is successfully generated. To satisfy the constraints of the MKP, a well-known approach known as repairing of solution vector is adopted to update the generated binary vector. The repair operator is introduced by Chu and Beasley [63] and utilized in many metaheuristic algorithms [51, 49, 64, 50]. The repair operator utilizes the knowledge of MKP about the pseudo-utility of items. In the repair operator, two phases are executed: first, the picked items are dropped in ascending order of pseudo-utility, while in the second phase, the dropped items are tried to be picked in the knapsack based on their descending order of pseudo-utility without violating any of the resource constraints. Luo and Zhao [50] have introduced a modified repair operator. The only difference is at the time of dropping items, where instead of checking the feasibility and the condition of whether an item is selected or not simultaneously, the feasibility condition is checked first to prevent an unnecessary computational burden. The procedure of the embedded repair operator is summarized in Algorithm 5.

3.2.4. Complete procedure of the BISCA

The binary version of our proposed ISCA is an extension with an initialization procedure based on pseudo-utility, transformation function, and repair operator to produce feasible solutions. Flow-chart of the proposed BISCA is presented in Figure 2. This flow-chart indicates that this could also be applied to other optimization problems based on the related initialization and repair operators.

4. Experimental results

In this section, firstly, the validation of the developed ISCA algorithm is carried out on a set of 23 benchmark optimization problems using several performance metrics. Later, its binary version named BISCA is used to solve 49 instances of the MKP.

4.1. Validation of the continuous ISCA

Before applying the binary version of ISCA (BISCA) for solving MKPs, its continuous version named ISCA has to be validated on continuous optimization problems. To do this, a well-known and commonly used benchmark set of 23 optimization problems [65], given in Table 1, is adopted

Algorithm 5 Repair operator based on the pseudo-utility of items

Calculate the consumed amount of resources as $z_k = \sum_{j=1}^{n} w_{kj} \times x_{i,j}, \quad \forall k \in \{1, 2, \ldots, m\}$
Sort the items in descending order order of their pseudo-utility,
let the sorted items are $J = \{I_1, I_2, \ldots, I_n\}$
for $j = I_1$ *to* I_n **do**
  if $z_k \leq W_k, \quad \forall k \in \{1, 2, \ldots, m\}$ **then**
    break
  end
  if $z_k > W_k$, *and* $x_{i,j} = 1$ **then**
    $x_{i,j} = 0$, $z_k = z_k - w_{kj} \quad \forall k \in \{1, 2, \ldots, m\}$
  end
end
for $j = I_n$ *to* I_1 **do**
  if $x_{i,j} = 0$ *and* $z_k + w_{kj} \leq W_k \quad \forall k \in \{1, 2, \ldots, m\}$ **then**
    $x_{i,j} = 1$ and $z_k = z_k + w_{kj} \quad \forall k \in \{1, 2, \ldots, m\}$
  end
end

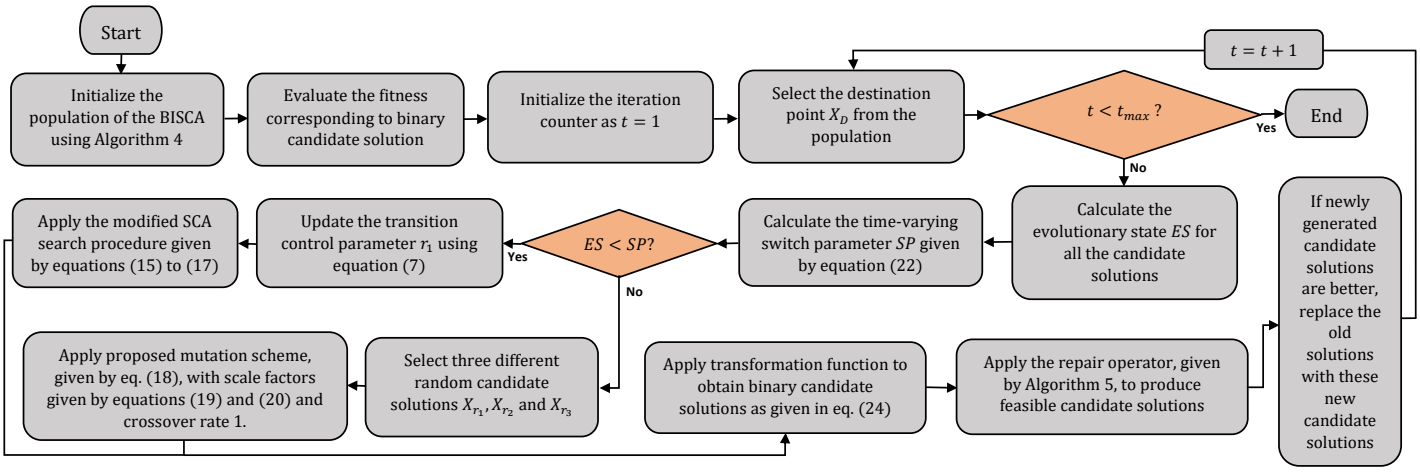

Figure 2: Flow-chart of the proposed binary ISCA (BISCA)

for the performance comparison of the ISCA. In the literature, this problem set is used to validate several algorithms [17, 66, 67, 68, 69]. On this problem set, the ISCA is compared with other metaheuristics such as particle swarm optimization (PSO) [70], salp-swarm algorithm (SSA) [71], differential evolution (DE) [6], transit search (TS) algorithm [72], classical SCA (SCA) [17], modified SCA (mSCA) [73], opposition-based learning SCA (OBSCA) [26], the dimension by dimension dynamic SCA (DDSCA) [30], enriched SCA based on the adaptive parameters and chaotic exploitative strategy (ASCA) [74], improved DE with neighborhood mutation operators and opposition-based learning (NBOLDE) [75], improved DE with replacement strategies (RSDE) [76], hybrid versions of SCA with DE such as HDESC [38], and HSCA [37]. Each algorithm is executed 30 times on each problem to analyze the robustness of the algorithm. The performance measures, which are used for comparing the performance of the ISCA with these algorithms are mean (Mean) and standard deviation (Std) of objective function values, rank comparison of each algorithm, and statistical tests using Wilcoxon signed-rank test [77].

To produce optimization results, the population size is fixed to 100 in all the algorithms, the maximum function evaluations are fixed to 100000 and the maximum number of iterations is set to 1000. All the performance measures based on the recorded results are shown in Table 2. In the last five rows of Table 2, overall performance on the benchmark set is provided to conclude the best-performing algorithm.

For unimodal problems (F1 to F7), the ISCA has outperformed on all the problems as compared to the classical SCA, on six problems than PSO, SSA, DE, OBSCA, DDSCA, HDESC, HSCA, on four problems than ASCA, on three problems than TS, RSDE, on two problems than mSCA and NBOLDE. This performance behavior shows the excellent exploitation ability of the ISCA than the PSO, SSA, DE, SCA, OBSCA, DDSCA, ASCA, HDESC and HSCA. The results of the TS, mSCA, NBOLDE, and RSDE indicate their better search performance in terms of exploitation of search space than the ISCA. For the second difficulty level, i.e. for multimodal problems (F8 to F13), the ISCA provides better results than the classical SCA on all the six problems, than the SSA, OBSCA, in four problems, than the PSO, mSCA, DDSCA in three problems, than the DE, ASCA, NBOLDE, RSDE, HDESC, HSCA in only two problems and than the TS in only one problem. The results of the ISCA on these problems verify the reasonable exploration strength of the ISCA. The results for low-dimensional multimodal problems (F14 to F23), reported in Table 2, indicate the outperform search efficiency of the ISCA than SSA, SCA, mSCA, OBSCA, DDSCA, ASCA, NBOLDE, RSDE, and HDESC and very competitive solution quality to the remaining algorithms. Based on these problems, the balancing ability of the ISCA between exploration and exploitation is verified, which is the impact of maintaining a proper synergy between the modified SCA and DE search schemes.

Although the ISCA has shown very competitive results on most of the unimodal and multimodal problems, in some cases (F6, F8 and F13), its performance is poor than most of the other algorithms. In problem F5, except for the RSDE algorithm, all the other algorithms are unable to provide near-to-optimal results. This competitive search performance of the ISCA on the unimodal problems verifies the impact of the modified SCA search scheme, which focuses on improving the exploitation ability of the algorithm and performing the search around the elite solution of the population. On the other hand, the competitive search performance of the ISCA on multimodal problems shows the impact of the DE search scheme in the ISCA. The results comparison on fixed-dimensional multimodal problems verifies that the applied strategies in the ISCA are able to balance the exploitation and exploration skills during the search procedure. Therefore, these problems illustrate that applied strategies such as modified SCA search scheme, hybridization with DE and parameters selected based on the fitness and evolutionary state of the candidate solutions have balanced the exploration and exploitation features within the ISCA. The overall ranking of all the algorithms, listed in the bottom part of the table, concludes the ISCA as a winner optimizer based on the comparison performed on the

benchmark set. Furthermore, statistical results are presented in Table 2 by comparing the obtained p values received from the Wilcoxon test and a significance level $\alpha = 0.05$. This test compares the optimization results obtained by all the algorithms statistically and gives an indication whether the algorithm significantly provides promising results. The statistical outcomes (Stat) are reported in table with the symbols "+/-/≈" to indicate the better, worst, or identical performance of the proposed ISCA than its competitive algorithm. An overall counting of these outcomes demonstrates the better search efficiency of the ISCA in producing better optimization results than other compared algorithms. To analyze the errors occurred by the algorithms, we have used a normalization equation given in [72] and is stated as follows:

$$error(A_i) = \Big| \frac{R_{A_i} - R_O}{max\{R_{A_1}, R_{A_2}, \dots R_{A_n}\}} \Big| \tag{25}$$

where R_{A_i} is the objective function value achieved by the algorithm A_i, R_O indicates the optimal value of the objective function, $error(A_i)$ indicates the error computed corresponding to the i^{th} algorithm A_i and n indicates the number of algorithms used for comparison. In our case, the value of n is 14. The error values for all the considered problems using the above equation are provided in Table 3. We have computed the rank based on these error values, which are shown in the last two rows of the table. The table ensures that the proposed ISCA has achieved the top rank in this comparison.

To analyze the convergence behavior of the ISCA and compare it with other algorithms, we have plotted the convergence curves for some selected benchmark problems. These curves are shown in Figure 3, where the competitive convergence behavior of the ISCA can be verified. It can be noticed from these figures that the ISCA has better convergence ability compared to the PSO, SSA, DE, TS, SCA, mSCA, OBSCA, HDESC, and HSCA on most of the problems. The flat convergence curves and poor solution quality of the classical SCA on most of the multimodal problems indicate its issues of stagnation at local optima and premature convergence. On the other hand, it can be verified that the ISCA has provided promising results with better convergence than the classical SCA. This fact concludes that the embedded strategies in the ISCA are successful to avoid the shortcomings of the classical SCA. Along with the convergence behavior comparison, we have also recorded the average computational time consumed by the proposed ISCA and other compared algorithms on a set of benchmark problems. These computational times are shown in Figure 4. The comparison of computation times indicates that the ISCA provides competitive results within reasonable computation efforts. The computational times of most of the algorithms are very close to each other except for the TS. The DDSCA has the lowest recorded computational time while the algorithm TS has the largest computational time among all the compared algorithms.

Hence, based on different performance measures, the effectiveness of the proposed ISCA is verified on continuous optimization problems. The algorithms mSCA and NBOLDE have outperformed on unimodal problems. However, their performances are not good on several low-dimensional multimodal problems, which indicates their low balancing ability between exploration and exploitation. The performances of the classical SCA, SSA, OBSCA and DDSCA have shown their low exploitation and weak diversity skills as compared to the ISCA as the results of these algorithms are worst than the ISCA on most of the problems. PSO shows weak exploitation and exploration than the ISCA as it does not perform better on most of the unimodal and multimodal problems. However, on low-dimensional problems, its performance is competitive to the ISCA which shows that the algorithm can deal with the problems having a low number of local optima only due to insufficient diversity skills. DE shows its weak exploitation and slow convergence as it does not perform well on unimodal problems as compared to the ISCA. On multimodal problems, its performance is promising due to reasonable exploration skills. The results of the ASCA, RSDE, HDESC and HSCA show its better

exploration ability but weak exploitation and imbalance between exploitation and exploration as compared to the ISCA. mSCA is superior in exploitation ability as it provides promising results on unimodal problems, but on multimodal problems, the results are not good as compared to the ISCA. This behavior shows the weak exploration in the mSCA as compared to the ISCA. By analyzing all these results, it can be concluded that on average, the ISCA outperforms all the other algorithms as it has comparatively better exploitation, exploration and balancing ability between exploitation and exploration.

Therefore, with this efficient search procedure of the ISCA, we can use its binary version named BISCA for solving the MKPs. The next section will accomplish this task based on some benchmark MKPs.

Table 1: Details of benchmark problems

Test Function	**Nature**	**Dimension**	**Search Range**	$f_{\min}$
$F1(x) = \sum_{i=1}^{d} x_i^2$	unimodal	30	[−100,100]	0
$F2(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	unimodal	30	[−10,10]	0
$F3(x) = \sum_{i=1}^{d} \left(\sum_{j-1}^{i} x_j \right)^2$	unimodal	30	[−100,100]	0
$F4\,(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le d \}$	unimodal	30	[−100,100]	0
$F5(x) = \sum_{i=1}^{d-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	unimodal	30	[−30,30]	0
$F6(x) = \sum_{i=1}^{d} \left([x_i + 0.5] \right)^2$	unimodal	30	[−100,100]	0
$F7(x) = \sum_{i=1}^{d} i x_i^4 + random[0,1)$	unimodal	30	[−1.28,1.28]	0
$F8(x) = \sum_{i=1}^{d} -x_i \sin \left(\sqrt{\lvert x_i \rvert} \right)$	multimodal	30	[−500,500]	$-418.9829 \times d$
$F9(x) = \sum_{i=1}^{d} \left[x_i^2 - 10 \cos (2\pi x_i) + 10 \right]$	multimodal	30	[−5.12,5.12]	0
$F10(x) = -20 \exp(-0.2 \sqrt{\frac{1}{d} \sum_{i=1}^{d} x_i^2}) - \exp \left(\frac{1}{d} \sum_{i=1}^{d} cos\,(2\pi x_i) \right) + 20 + e$	multimodal	30	[−32,32]	0
$F11(x) = \frac{1}{4000} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1$	multimodal	30	[−600,600]	0
$F12(x) = \frac{\pi}{d} \left\{ 10 \sin (\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 \left[1 + 10 \sin^2 (\pi y_{i+1}) \right] + (y_d - 1)^2 \right\}$ $+ \sum_{i=1}^{d} u(x_i, 10, 100, 4)\ y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	multimodal	30	[−50,50]	0
$F13(x) = 0.1 \left\{ \sin^2 (3\pi x_1) + \sum_{i=1}^{d} (x_i - 1)^2 \left[1 + \sin^2 (3\pi x_i + 1) \right] \right\}$ $+0.1 \left\{ (x_d - 1)^2 \left[1 + \sin^2 (2\pi x_d) \right] \right\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$	multimodal	30	[−50,50]	0
$F14(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	multimodal	2	$[-65, 65]$	0.998
$F15(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	multimodal	4	$[-5, 5]$	0.00030
$F16(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3} x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	multimodal	2	$[-5, 5]$	−1.0316
$F17\,(x) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$	multimodal	2	$[-5, 5]$	0.398
$F18(x) = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2 \right) \right]$ $\times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2 \right) \right]$	multimodal	2	$[-2, 2]$	3
$F19(x) = -\sum_{i=1}^{4} c_i \exp \left(-\sum_{j=1}^{3} a_{ij} \left(x_j - p_{ij} \right)^2 \right)$	multimodal	3	$[1, 3]$	−3.86
$F20(x) = -\sum_{i=1}^{4} c_i \exp \left(-\sum_{j=1}^{6} a_{ij} \left(x_j - p_{ij} \right)^2 \right)$	multimodal	6	$[0, 1]$	−3.32
$F21(x) = -\sum_{i=1}^{5} \left[(X - a_i) (X - a_i)^T + c_i \right]^{-1}$	multimodal	4	$[0, 10]$	−10.1532
$F22(x) = -\sum_{i=1}^{7} \left[(X - a_i) (X - a_i)^T + c_i \right]^{-1}$	multimodal	4	$[0, 10]$	−10.4028
$F23(x) = -\sum_{i=1}^{10} \left[(X - a_i) (X - a_i)^T + c_i \right]^{-1}$	multimodal	4	[0.10]	−10.5363

Table 2: Comparison of optimization results on benchmark test problems

Problem	Results	PSO	SSA	DE	TS	SCA	mSCA	OBSCA	DDSCA	ASCA	NBOLDE	RSDE	HDESC	HSCA	ISCA
F1	Mean	3.44E-14	7.00E-09	2.34E-11	1.05E-13	9.57E-05	0.00E+00	5.60E-13	1.71E-04	5.81E-02	7.90E-221	2.44E-23	2.50E-14	1.14E-07	2.15E-22
	Std	9.62E-14	1.07E-09	7.55E-12	1.42E-13	2.50E-04	0.00E+00	3.05E-12	1.06E-04	8.83E-02	0.00E+00	1.82E-23	9.86E-14	5.04E-08	7.84E-22
	Rank	6	10	9	7	12	1	8	13	14	2	3	5	11	4
	Stat	+	+	+	+	+	-	+	+	+	-	≈	+	+	NA
F2	Mean	8.86E-10	1.35E-01	2.09E-07	4.98E-08	5.31E-07	5.42E-167	8.11E-16	3.37E-03	1.12E-01	4.11E-117	5.47E-12	1.81E-11	7.37E-06	1.30E-28
	Std	1.04E-09	2.55E-01	3.54E-08	5.39E-08	6.08E-07	0.00E+00	2.79E-15	1.11E-03	9.93E-02	1.50E-116	1.57E-12	4.30E-11	1.65E-06	6.39E-28
	Rank	7	14	9	8	10	1	4	12	13	2	5	6	11	3
	Stat	+	+	+	+	+	-	+	+	+	-	+	+	+	NA
F3	Mean	1.02E+02	7.47E-01	1.18E+04	5.28E-13	9.51E+02	0.00E+00	1.54E+03	2.19E+04	2.61E+01	6.16E-151	1.28E-01	3.35E+03	5.07E+04	1.79E-05
	Std	5.75E+01	8.95E-01	1.78E+03	9.20E-13	1.08E+03	0.00E+00	3.39E+03	3.15E+03	4.63E+01	3.13E-150	5.15E-02	1.92E+03	4.67E+03	4.41E-05
	Rank	8	6	12	3	9	1	10	13	7	2	5	11	14	4
	Stat	+	+	+	-	+	-	+	+	+	-	+	+	+	NA
F4	Mean	1.34E+00	1.11E+00	3.51E+00	1.95E-07	8.72E+00	6.75E-162	1.03E+01	3.07E+01	3.30E-02	2.11E-97	2.61E-03	2.13E+00	8.42E+01	3.79E-03
	Std	3.86E-01	1.27E+00	2.98E-01	2.06E-07	8.24E+00	3.24E-161	2.31E+01	3.02E+00	2.98E-02	6.72E-97	1.34E-03	1.55E+00	3.80E+00	7.77E-03
	Rank	8	7	10	3	11	1	12	13	6	2	4	9	14	5
	Stat	+	+	+	-	+	-	+	+	+	-	≈	+	+	NA
F5	Mean	1.71E+02	6.31E+01	4.32E+01	2.82E+01	2.97E+01	2.89E+01	2.83E+01	1.30E+02	8.23E-01	1.42E+02	6.55E-01	2.63E+01	3.63E+01	2.55E+01
	Std	5.50E+02	8.85E+01	7.99E+00	1.66E-01	4.78E+00	2.36E-02	3.50E-01	3.34E+01	1.19E+00	3.66E+02	1.39E+00	4.84E-01	2.07E+01	3.78E-01
	Rank	14	11	10	5	8	7	6	12	2	13	1	4	9	3
	Stat	+	+	+	+	+	+	+	+	-	+	-	+	+	NA
F6	Mean	4.82E-14	7.08E-09	2.19E-11	9.95E-02	3.87E+00	5.82E+00	4.49E+00	5.37E-02	7.83E-02	5.88E+00	2.20E-23	1.13E-02	5.54E-02	1.32E-01
	Std	1.45E-13	1.38E-09	6.21E-12	2.56E-02	3.89E-01	5.21E-01	3.93E-01	1.15E-02	1.60E-01	8.32E+00	1.34E-23	2.05E-03	8.13E-03	1.68E-01
	Rank	2	4	3	9	11	13	12	6	8	14	1	5	7	10
	Stat	-	-	-	≈	+	+	+	≈	≈	+	-	-	≈	NA
F7	Mean	1.43E-02	2.63E-02	2.69E-02	1.01E-04	1.36E-02	4.81E-05	5.50E-03	1.38E-01	8.82E-03	1.16E-04	1.33E-02	1.49E-02	1.13E-01	1.00E-02
	Std	4.30E-03	8.76E-03	5.42E-03	7.51E-05	1.43E-02	3.81E-05	2.77E-03	3.14E-02	8.24E-03	5.30E-05	4.74E-03	6.65E-03	2.41E-02	5.23E-03
	Rank	9	11	12	2	8	1	4	14	5	3	7	10	13	6
	Stat	+	+	+	-	≈	-	-	+	≈	-	+	+	+	NA
F8	Mean	-9.57E+03	-7.57E+03	-1.26E+04	-1.06E+04	-4.17E+03	-4.47E+03	-4.12E+03	-1.12E+04	-1.26E+04	-7.20E+03	-1.12E+04	-7.88E+03	-7.22E+03	-5.88E+03
	Std	5.22E+02	6.62E+02	5.44E-08	2.83E+02	2.36E+02	3.51E+02	3.15E+02	1.48E+02	3.74E-02	6.03E+02	6.28E+02	2.31E+02	1.44E+02	4.77E+02
	Rank	6	8	1	5	13	12	14	4	2	10	3	7	9	11
	Stat	-	-	-	-	+	+	+	-	-	-	-	-	-	NA
F9	Mean	3.39E+01	4.12E+01	9.87E+00	6.82E-14	8.41E+00	0.00E+00	1.88E-11	5.00E+00	2.44E-02	8.37E-01	7.51E-02	1.17E+00	1.36E+02	9.93E-01
	Std	9.50E+00	1.84E+01	1.84E+00	1.49E-13	1.72E+01	0.00E+00	9.36E-11	1.29E+00	2.76E-02	4.40E+00	3.89E-02	3.71E+00	9.04E+00	4.01E+00
	Rank	12	13	11	2	10	1	3	9	4	6	5	8	14	7
	Stat	+	+	+	≈	+	≈	≈	+	-	≈	-	+	+	NA
F10	Mean	4.46E-08	1.32E+00	1.22E-06	5.70E-08	1.09E+01	8.88E-16	1.00E+01	4.34E-02	4.50E-02	1.13E-15	5.71E-02	1.45E+00	1.97E+01	1.16E-12
	Std	5.62E-08	9.23E-01	1.89E-07	6.58E-08	9.81E+00	0.00E+00	1.02E+01	2.26E-02	5.93E-02	9.01E-16	1.66E-01	5.06E+00	3.40E-01	4.61E-12
	Rank	4	10	6	5	13	1	12	7	8	2	9	11	14	3
	Stat	+	+	+	+	+	-	+	+	+	-	+	+	+	NA
F11	Mean	1.07E-02	1.08E-02	8.46E-10	1.75E-13	7.02E-02	0.00E+00	1.83E-04	7.84E-03	6.90E-02	3.70E-18	8.61E-03	1.46E-09	2.93E-05	1.35E-03
	Std	1.35E-02	1.30E-02	1.07E-09	2.42E-13	1.45E-01	0.00E+00	9.05E-04	9.37E-03	8.90E-02	2.03E-17	9.77E-03	6.34E-09	1.50E-04	5.82E-03
	Rank	11	12	4	3	14	1	7	9	13	2	10	5	6	8
	Stat	+	+	+	-	+	≈	-	+	+	≈	+	-	-	NA
F12	Mean	6.91E-03	2.13E+00	7.02E-13	5.34E-04	5.71E-01	7.34E-01	5.32E-01	1.47E-03	4.00E-04	4.25E-01	1.24E-04	4.91E-04	2.45E-03	1.72E-02
	Std	2.63E-02	1.61E+00	2.04E-13	1.33E-04	3.02E-01	1.64E-01	1.03E-01	3.70E-04	6.60E-04	6.81E-01	6.24E-04	1.12E-04	4.65E-04	2.26E-02
	Rank	8	14	1	5	12	13	11	6	3	10	2	4	7	9
	Stat	-	+	-	-	+	+	+	-	-	+	-	-	-	NA
F13	Mean	2.56E-03	5.85E-03	4.42E-12	1.60E-02	2.86E+00	2.99E+00	2.45E+00	3.04E-02	1.40E-02	1.93E+00	3.66E-14	1.08E-02	5.38E-02	9.18E-01
	Std	4.73E-03	1.07E-02	1.34E-12	5.11E-03	2.53E+00	4.31E-02	1.29E-01	8.33E-03	2.57E-02	8.66E-01	1.90E-13	2.73E-03	8.32E-03	3.17E-01
	Rank	3	4	2	7	13	14	12	8	6	11	1	5	9	10
	Stat	-	-	-	-	+	+	+	-	-	+	-	-	-	NA

Table 2 (continued)

Problem	Results	PSO	SSA	DE	TS	SCA	mSCA	OBSCA	DDSCA	ASCA	NBOLDE	RSDE	HDESC	HSCA	ISCA
F14	Mean	9.98E-01	9.98E-01	9.98E-01	9.98E-01	1.06E+00	1.85E+00	1.33E+00	9.98E-01	1.84E+00	1.69E+00	9.98E-01	9.98E-01	9.98E-01	9.98E-01
	Std	4.12E-17	2.00E-16	0.00E+00	1.51E-16	3.62E-01	1.32E+00	7.52E-01	2.44E-11	2.59E+00	1.14E+00	3.45E-14	2.19E-12	4.12E-17	1.40E-16
	Rank	2	5	2	5	10	14	11	9	13	12	7	8	2	5
	Stat	≈	≈	≈	≈	+	+	+	+	+	+	+	+	≈	NA
F15	Mean	1.07E-03	7.79E-04	4.94E-03	3.16E-04	8.06E-04	1.06E-03	1.10E-03	6.62E-04	3.87E-03	2.16E-03	1.71E-03	4.15E-04	6.53E-04	7.96E-04
	Std	3.66E-03	3.41E-04	7.18E-05	1.54E-05	4.56E-04	7.63E-04	3.63E-04	1.52E-04	6.97E-03	5.21E-03	5.08E-03	1.14E-04	6.35E-05	4.65E-04
	Rank	9	5	14	1	7	8	10	4	13	12	11	2	3	6
	Stat	≈	≈	+	≈	≈	≈	+	≈	+	+	≈	-	-	NA
F16	Mean	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-9.94E-01	-1.03E+00	-1.03E+00	-6.45E-01	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00
	Std	6.78E-16	3.58E-15	6.78E-16	5.05E-16	8.54E-06	3.56E-02	3.73E-06	1.62E-08	2.98E-01	6.65E-16	1.18E-10	7.23E-10	6.78E-16	4.70E-16
	Rank	3.5	7	3.5	3.5	12	13	11	10	14	3.5	8	9	3.5	3.5
	Stat	≈	+	≈	+	+	+	+	+	+	≈	+	+	≈	NA
F17	Mean	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	4.31E-01	3.99E-01	3.98E-01	6.71E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01
	Std	0.00E+00	6.71E-15	0.00E+00	0.00E+00	2.09E-04	3.07E-02	5.58E-04	8.05E-07	1.92E-01	0.00E+00	7.10E-11	3.86E-08	0.00E+00	0.00E+00
	Rank	3.5	7	3.5	3.5	11	13	12	10	14	3.5	8	9	3.5	3.5
	Stat	≈	+	≈	≈	+	+	+	+	+	≈	+	+	≈	NA
F18	Mean	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	1.32E+01	3.00E+00	3.00E+00	1.36E+01	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00
	Std	2.12E-15	3.73E-14	1.73E-15	1.86E-15	1.77E-06	6.50E+00	2.31E-05	5.06E-07	1.09E+01	2.03E-15	1.36E-09	6.90E-09	1.91E-15	1.80E-15
	Rank	3	7	3	6	11	13	12	10	14	3	8	9	3	3
	Stat	≈	+	≈	+	+	+	+	+	+	≈	+	+	≈	NA
F19	Mean	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.70E+00	-3.86E+00	-3.86E+00	-3.64E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00
	Std	2.71E-15	7.75E-15	2.71E-15	2.18E-15	2.56E-03	7.47E-02	4.20E-03	2.31E-06	1.61E-01	1.37E-11	1.95E-11	4.94E-08	2.21E-15	2.26E-15
	Rank	3	6	3	3	12	13	11	10	14	7	8	9	3	3
	Stat	≈	+	≈	≈	+	+	+	+	+	≈	+	+	≈	NA
F20	Mean	-3.28E+00	-3.23E+00	-3.32E+00	-3.32E+00	-3.06E+00	-2.51E+00	-3.10E+00	-3.32E+00	-2.25E+00	-3.28E+00	-3.27E+00	-3.32E+00	-3.31E+00	-3.25E+00
	Std	6.23E-02	4.84E-02	1.36E-15	3.63E-11	1.06E-01	2.26E-01	2.30E-02	2.98E-05	3.76E-01	6.00E-02	6.03E-02	1.19E-05	3.02E-02	5.99E-02
	Rank	7	10	1	2	12	13	11	4	14	6	8	3	5	9
	Stat	≈	+	-	-	+	+	+	-	+	≈	≈	-	-	NA
F21	Mean	-7.63E+00	-8.30E+00	-1.02E+01	-6.07E+00	-4.37E+00	-5.12E+00	-6.63E+00	-1.01E+01	-1.02E+01	-9.65E+00	-8.73E+00	-1.02E+01	-1.00E+01	-1.02E+01
	Std	3.23E+00	2.72E+00	7.12E-09	2.07E+00	2.04E+00	9.26E-01	1.51E+00	3.86E-03	4.84E-03	1.91E+00	2.70E+00	1.22E-03	1.13E-01	4.95E-15
	Rank	10	9	2	12	14	13	11	5	4	7	8	3	6	1
	Stat	+	+	+	+	+	+	+	+	+	+	+	+	+	NA
F22	Mean	-8.74E+00	-1.02E+01	-1.04E+01	-6.68E+00	-5.22E+00	-5.32E+00	-7.24E+00	-1.04E+01	-1.04E+01	-9.42E+00	-1.01E+01	-1.04E+01	-1.04E+01	-1.04E+01
	Std	2.85E+00	9.63E-01	2.20E-07	2.47E+00	1.76E+00	9.14E-01	1.45E+00	7.14E-03	6.55E-03	2.57E+00	1.34E+00	8.09E-04	2.87E-02	1.36E-15
	Rank	10	7	2	12	14	13	11	5	4	9	8	3	6	1
	Stat	+	+	+	+	+	+	+	+	+	+	+	+	+	NA
F23	Mean	-9.92E+00	-9.49E+00	-1.05E+01	-7.07E+00	-5.77E+00	-5.61E+00	-7.54E+00	-1.05E+01	-1.05E+01	-8.13E+00	-1.04E+01	-1.05E+01	-1.05E+01	-1.05E+01
	Std	1.90E+00	2.43E+00	3.06E-09	2.60E+00	1.18E+00	1.20E+00	1.22E+00	5.61E-03	3.18E-03	3.51E+00	9.79E-01	1.40E-03	2.31E-02	2.42E-15
	Rank	8	9	2	12	13	14	11	5	3	10	7	4	6	1
	Stat	≈	+	+	+	+	+	+	+	+	+	+	+	+	NA
	Mean rank	6.83	8.52	5.48	5.39	11.30	8.43	9.83	8.61	8.61	6.61	5.96	6.48	7.78	5.17
	Overall rank	7	10	3	2	14	9	13	11	11	6	4	5	8	1
	+	11	18	12	9	21	14	20	17	16	9	13	16	11	NA
	-	4	3	6	8	0	6	2	4	5	7	6	7	6	NA
	≈	8	2	5	6	2	3	1	2	2	7	4	0	6	NA

Table 3: Comparison of average errors recorded over 30 independent runs

Problem	PSO	SSA	DE	TS	SCA	mSCA	OBSCA	DDSCA	ASCA	NBOLDE	RSDE	HDESC	HSCA	ISCA
F1	5.93E-13	1.20E-07	4.03E-10	1.81E-12	1.65E-03	0.00E+00	9.63E-12	2.95E-03	1.00E+00	1.36E-219	4.20E-22	4.31E-13	1.97E-06	3.71E-21
F2	6.56E-09	1.00E+00	1.55E-06	3.69E-07	3.93E-06	4.01E-166	6.00E-15	2.50E-02	8.28E-01	3.04E-116	4.05E-11	1.34E-10	5.46E-05	9.62E-28
F3	2.02E-03	1.47E-05	2.33E-01	1.04E-17	1.88E-02	0.00E+00	3.03E-02	4.33E-01	5.15E-04	1.22E-155	2.52E-06	6.61E-02	1.00E+00	3.54E-10
F4	1.60E-02	1.32E-02	4.17E-02	2.32E-09	1.04E-01	8.02E-164	1.22E-01	3.65E-01	3.92E-04	2.51E-99	3.10E-05	2.53E-02	1.00E+00	4.50E-05
F5	1.00E+00	3.68E-01	2.52E-01	1.65E-01	1.73E-01	1.69E-01	1.65E-01	7.62E-01	4.80E-03	8.28E-01	3.83E-03	1.54E-01	2.12E-01	1.49E-01
F6	8.20E-15	1.20E-09	3.73E-12	1.69E-02	6.57E-01	9.89E-01	7.64E-01	9.13E-03	1.33E-02	1.00E+00	3.74E-24	1.92E-03	9.41E-03	2.24E-02
F7	1.03E-01	1.90E-01	1.94E-01	7.31E-04	9.79E-02	3.47E-04	3.97E-02	1.00E+00	6.37E-02	8.40E-04	9.63E-02	1.07E-01	8.14E-01	7.23E-02
F8	7.28E-01	1.21E+00	9.26E-08	4.66E-01	2.04E+00	1.96E+00	2.05E+00	3.24E-01	5.89E-06	1.30E+00	3.23E-01	1.14E+00	1.30E+00	1.62E+00
F9	2.48E-01	3.02E-01	7.24E-02	5.00E-16	6.17E-02	0.00E+00	1.38E-13	3.67E-02	1.79E-04	6.14E-03	5.50E-04	8.61E-03	1.00E+00	7.28E-03
F10	2.26E-09	6.69E-02	6.17E-08	2.89E-09	5.54E-01	4.50E-17	5.06E-01	2.20E-03	2.28E-03	5.70E-17	2.89E-03	7.33E-02	1.00E+00	5.89E-14
F11	1.53E-01	1.54E-01	1.20E-08	2.49E-12	1.00E+00	0.00E+00	2.60E-03	1.12E-01	9.84E-01	5.27E-17	1.23E-01	2.08E-08	4.18E-04	1.92E-02
F12	3.25E-03	1.00E+00	3.30E-13	2.51E-04	2.68E-01	3.45E-01	2.50E-01	6.92E-04	1.88E-04	2.00E-01	5.83E-05	2.31E-04	1.15E-03	8.07E-03
F13	8.59E-04	1.96E-03	1.48E-12	5.37E-03	9.58E-01	1.00E+00	8.20E-01	1.02E-02	4.68E-03	6.47E-01	1.23E-14	3.62E-03	1.80E-02	3.07E-01
F14	0.00E+00	6.01E-17	0.00E+00	6.01E-17	3.58E-02	4.59E-01	1.79E-01	1.47E-11	4.59E-01	3.76E-01	8.42E-15	1.27E-12	0.00E+00	6.01E-17
F15	1.54E-01	9.56E-02	9.38E-01	1.78E-03	1.01E-01	1.52E-01	1.61E-01	7.20E-02	7.21E-01	3.75E-01	2.83E-01	2.18E-02	7.02E-02	9.91E-02
F16	0.00E+00	5.16E-15	0.00E+00	0.00E+00	1.37E-05	5.82E-02	6.36E-06	2.15E-08	5.99E-01	0.00E+00	1.00E-10	9.21E-10	0.00E+00	0.00E+00
F17	0.00E+00	6.62E-15	0.00E+00	0.00E+00	4.10E-04	4.99E-02	1.28E-03	1.23E-06	4.07E-01	0.00E+00	1.09E-10	5.90E-08	0.00E+00	0.00E+00
F18	0.00E+00	6.25E-15	0.00E+00	2.93E-16	1.07E-07	7.50E-01	1.76E-06	1.74E-08	7.80E-01	0.00E+00	6.69E-11	3.44E-10	0.00E+00	0.00E+00
F19	4.93E-11	4.93E-11	4.93E-11	4.93E-11	1.97E-03	4.38E-02	1.80E-03	7.05E-07	6.26E-02	5.02E-11	5.71E-11	1.75E-08	4.93E-11	4.93E-11
F20	2.03E-02	4.23E-02	1.85E-10	2.00E-10	1.16E-01	3.63E-01	9.85E-02	2.55E-05	4.76E-01	1.99E-02	2.47E-02	7.53E-06	3.52E-03	2.99E-02
F21	5.76E-01	4.23E-01	7.48E-10	9.33E-01	1.32E+00	1.15E+00	8.06E-01	1.48E-03	6.10E-04	1.15E-01	3.26E-01	2.84E-04	3.29E-02	2.15E-10
F22	3.18E-01	3.37E-02	2.22E-08	7.14E-01	9.94E-01	9.73E-01	6.07E-01	1.82E-03	6.35E-04	1.89E-01	6.74E-02	2.27E-04	4.04E-03	6.10E-10
F23	1.09E-01	1.87E-01	7.48E-10	6.18E-01	8.49E-01	8.77E-01	5.34E-01	1.37E-03	2.60E-04	4.30E-01	3.18E-02	3.19E-04	4.00E-03	5.89E-10
Mean rank	6.83	8.52	5.48	5.39	11.30	8.43	9.83	8.61	8.61	6.61	5.96	6.48	7.78	5.17
Overall rank	7	10	3	2	13	9	12	11	11	6	4	5	8	1

4.1.1. Sensitivity analysis

This subsection analyzes the impact of different parameters involved in the ISCA on its performance quality using their sensitivity analysis.

To perform the sensitivity analysis of parameters, we have tuned their setting and analyzed the solution quality of the ISCA on diverse category of problems. We have discussed the setting of parameters namely, maximum iterations t_{max}, population size N, and switch parameter S_P. The other parameters such as r_1, r_2 and r_3 are fixed same as set in the classical SCA. When we perform the sensitivity analysis of one parameter then settings of other parameter is fixed same as suggested in the ISCA.

1. Maximum iterations: The proposed ISCA is run for different number of iterations. The values of t_{max} used in the experiments are 100, 200, 500 and 1000. The experimental results are shown in Table 4, which indicate that ISCA converges towards the global optima with increasing number of iterations.
2. Population size: Different population sizes such as 20, 50, 70 and 100 are used to analyze their impacts on the performance of the ISCA. The numerical results are provided in Table 5. From this table, it can be seen that when the population increases, the algorithm performs very well on almost all the category of optimization problems.
3. Switch parameter: In the proposed ISCA, we have set a time-varying setting for the switch parameter S_P to maintain an appropriate between exploration and exploitation. We have fixed four different settings such as random number between 0 and 1 (i.e. rand(0,1)), time-varying function as given by eq. (22) with $S_{P,start} = 1$ and $S_{P,end} = 0$ represented by $S_P \in [0, 1]$, time-varying function as given by eq. (22) with $S_{P,start} = 0.8$ and $S_{P,end} = 0.2$ represented by $S_P \in [0.2, 0.8]$, and time-varying function as given by eq. (22) with $S_{P,start} = 0.6$ and $S_{P,end} = 0.4$ represented by $S_P \in [0.4, 0.6]$. The results with these settings are shown in Table 6. The table verifies our setting that we have set in the ISCA to provide sufficient exploration and exploitation during the optimization procedure.

4.2. Application of the binary ISCA for solving MKP

In this subsection, we use our proposed binary version of the ISCA, named BISCA, for solving the MKP. A well-known collection of MKP benchmark sets provided in the OR-library

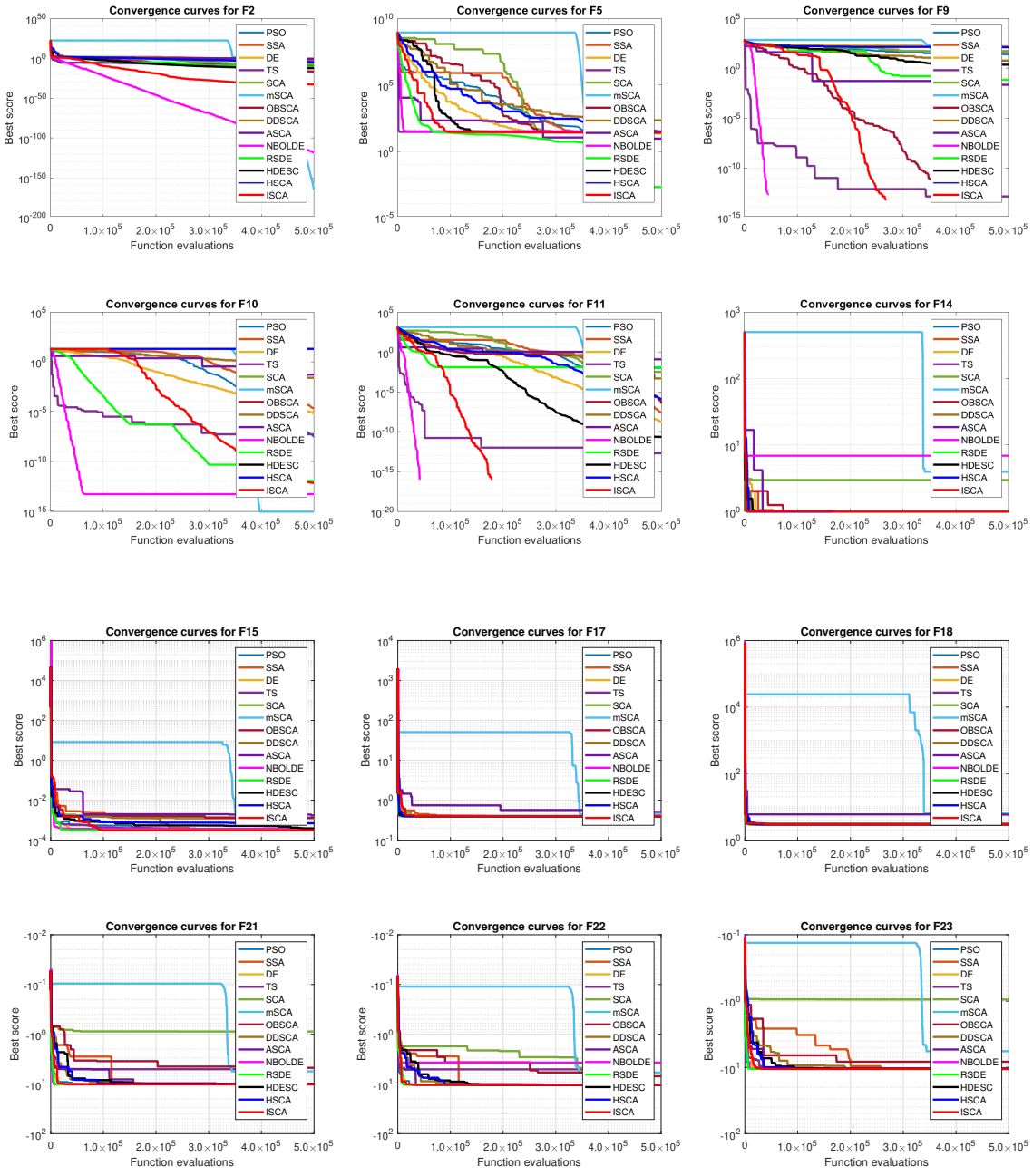

Figure 3: Convergence behavior comparison on benchmark problems

Table 4: Results comparison using different numbers of maximum iterations

Problem	t_{max} = **100**		t_{max} = **200**		t_{max} = **500**		t_{max} = **1000**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
F1	3.01E+01	2.37E+01	1.18E-01	1.59E-01	1.14E-09	3.57E-09	2.15E-22	7.84E-22
F2	2.14E-01	1.95E-01	1.82E-04	2.43E-04	2.54E-12	1.35E-11	1.30E-28	6.39E-28
F3	5.61E+02	2.38E+02	7.08E+01	5.20E+01	4.56E-01	1.04E+00	1.79E-05	4.41E-05
F4	5.97E+00	1.49E+00	2.72E+00	1.40E+00	3.28E-01	7.10E-01	3.79E-03	7.77E-03
F5	2.81E+03	1.91E+03	5.98E+01	4.21E+01	2.64E+01	5.50E-01	2.55E+01	3.78E-01
F6	5.06E+01	3.79E+01	1.61E+00	6.21E-01	3.55E-01	3.02E-01	1.32E-01	1.68E-01
F7	5.92E-02	2.71E-02	3.86E-02	1.48E-02	1.75E-02	6.44E-03	1.00E-02	5.23E-03
F8	-4.57E+03	3.20E+02	-4.98E+03	3.82E+02	-5.30E+03	3.44E+02	-5.88E+03	4.77E+02
F9	1.03E+02	5.12E+01	2.35E+01	3.55E+01	1.80E+00	3.83E+00	9.93E-01	4.01E+00
F10	5.11E+00	4.00E+00	4.62E-01	8.13E-01	1.10E-04	5.77E-04	1.16E-12	4.61E-12
F11	1.32E+00	4.04E-01	1.44E-01	1.16E-01	1.17E-02	3.02E-02	1.35E-03	5.82E-03
F12	4.80E+00	2.05E+00	3.76E-01	4.28E-01	5.56E-02	5.41E-02	1.72E-02	2.26E-02
F13	2.03E+01	6.02E+00	1.79E+00	6.86E-01	8.70E-01	3.28E-01	9.18E-01	3.17E-01
F14	1.19350	4.71E-01	1.03121	1.81E-01	0.99800	2.31E-16	0.99800	1.40E-16
F15	0.00082	2.40E-04	0.00056	4.10E-04	0.00043	3.17E-04	0.00080	4.65E-04
F16	-1.03162	4.50E-06	-1.03163	5.68E-08	-1.03163	1.75E-15	-1.03163	4.70E-16
F17	0.39795	8.63E-05	0.39789	8.54E-07	0.39789	1.59E-14	0.39789	0.00E+00
F18	3.00000	1.27E-07	3.00000	1.88E-15	3.00000	9.03E-16	3.00000	1.80E-15
F19	-3.86278	5.26E-11	-3.86278	2.29E-15	-3.86278	2.27E-15	-3.86278	2.26E-15
F20	-3.25757	5.93E-02	-3.26255	6.05E-02	-3.23877	5.54E-02	-3.25462	5.99E-02
F21	-10.15317	1.30E-04	-10.15320	6.70E-15	-10.15320	5.27E-15	-10.15320	4.95E-15
F22	-10.40294	3.08E-07	-10.40294	1.19E-15	-10.40294	8.08E-16	-10.40294	1.36E-15
F23	-10.53641	5.44E-07	-10.53641	2.97E-15	-10.53641	1.95E-15	-10.53641	2.42E-15
rank	3.91		2.76		1.91		1.41	
+/≈/-	21/2/0		17/6/0		13/9/1		NA	

Table 5: Results comparison using different population sizes

Problem	N = **20**		N=**50**		N=**70**		N=**100**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
F1	3.84E-26	1.71E-25	1.30E-23	4.73E-23	4.97E-23	2.41E-22	2.15E-22	7.84E-22
F2	3.50E-27	1.37E-26	2.44E-30	1.04E-29	3.05E-30	8.15E-30	1.30E-28	6.39E-28
F3	7.41E-02	2.72E-01	4.02E-04	8.36E-04	1.08E-04	2.78E-04	1.79E-05	4.41E-05
F4	4.80E-02	1.91E-01	9.31E-02	2.42E-01	2.51E-02	5.17E-02	3.79E-03	7.77E-03
F5	2.71E+01	9.06E-01	2.65E+01	8.02E-01	2.69E+01	4.46E+00	2.55E+01	3.78E-01
F6	1.96E+00	5.07E-01	8.00E-01	3.24E-01	3.69E-01	3.21E-01	1.32E-01	1.68E-01
F7	1.02E-02	4.98E-03	1.28E-02	6.27E-03	1.08E-02	5.41E-03	1.00E-02	5.23E-03
F8	-6.53E+03	6.84E+02	-6.92E+03	8.78E+02	-6.24E+03	8.14E+02	-5.88E+03	4.77E+02
F9	9.01E-01	3.07E+00	3.84E+00	1.17E+01	5.58E+00	1.35E+01	9.93E-01	4.01E+00
F10	2.00E+00	6.09E+00	6.96E-13	1.66E-12	8.91E-13	1.92E-12	1.16E-12	4.61E-12
F11	4.33E-03	1.13E-02	6.84E-04	3.75E-03	5.50E-04	2.09E-03	1.35E-03	5.82E-03
F12	1.12E-01	5.13E-02	4.10E-02	2.14E-02	4.15E-02	4.29E-02	1.72E-02	2.26E-02
F13	1.71E+00	3.32E-01	1.35E+00	3.34E-01	1.10E+00	2.64E-01	9.18E-01	3.17E-01
F14	1.13028	5.03E-01	0.99800	3.86E-16	0.99800	2.39E-16	0.99800	1.40E-16
F15	0.00104	3.73E-04	0.00070	4.62E-04	0.00070	4.62E-04	0.00080	4.65E-04
F16	-1.03163	4.14E-16	-1.03163	4.55E-16	-1.03163	4.46E-16	-1.03163	4.70E-16
F17	0.39789	0.00E+00	0.39789	0.00E+00	0.39789	0.00E+00	0.39789	0.00E+00
F18	3.00000	2.23E-15	3.00000	2.10E-15	3.00000	2.21E-15	3.00000	1.80E-15
F19	-3.86278	1.98E-15	-3.86278	2.21E-15	-3.86278	2.22E-15	-3.86278	2.26E-15
F20	-3.22652	5.71E-02	-3.23877	5.54E-02	-3.25859	6.03E-02	-3.25462	5.99E-02
F21	-7.79172	2.81E+00	-10.15320	4.66E-15	-10.15320	4.72E-15	-10.15320	4.95E-15
F22	-8.99800	2.63E+00	-10.40294	8.08E-16	-10.40294	1.48E-15	-10.40294	1.36E-15
F23	-9.38474	2.65E+00	-10.53641	3.09E-15	-10.53641	2.60E-15	-10.53641	2.42E-15
rank	3.33		2.30		2.24		2.13	
+/≈/-	12/9/2		7/14/2		5/18/0		NA	

Table 6: Results comparison using different ranges for switch parameter S_P

Problem	$S_P = rand(0,1)$		$S_P \in [0,1]$		$S_P \in [0.2, 0.8]$		$S_P \in [0.4, 0.6]$	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
F1	2.05E-23	7.41E-23	7.55E-38	4.07E-37	2.15E-22	7.84E-22	3.57E-23	1.42E-22
F2	4.25E-29	1.18E-28	8.62E-30	1.44E-29	1.30E-28	6.39E-28	9.68E-30	2.67E-29
F3	2.52E-04	5.89E-04	4.73E-04	1.14E-03	1.79E-05	4.41E-05	7.84E-05	2.68E-04
F4	2.63E-02	7.48E-02	5.33E-02	1.02E-01	3.79E-03	7.77E-03	5.57E-03	1.13E-02
F5	2.60E+01	4.75E-01	2.61E+01	5.62E-01	2.55E+01	3.78E-01	2.58E+01	5.28E-01
F6	1.13E-01	1.22E-01	2.02E-01	1.81E-01	1.32E-01	1.68E-01	1.91E-01	2.40E-01
F7	1.09E-02	3.87E-03	5.11E-03	3.28E-03	1.00E-02	5.23E-03	9.16E-03	3.61E-03
F8	-5.62E+03	5.95E+02	-5.58E+03	4.51E+02	-5.88E+03	4.77E+02	-5.72E+03	5.63E+02
F9	1.60E+00	5.60E+00	7.31E-01	2.84E+00	9.93E-01	4.01E+00	1.87E+00	6.24E+00
F10	2.62E-12	8.37E-12	3.98E-09	1.67E-08	1.16E-12	4.61E-12	1.11E-13	1.77E-13
F11	8.30E-03	3.42E-03	0.00E+00	0.00E+00	1.35E-03	5.82E-03	3.51E-03	1.92E-03
F12	1.15E-01	5.49E-01	2.73E-02	2.78E-02	1.72E-02	2.26E-02	3.85E-02	1.11E-01
F13	1.26E+00	2.20E-01	9.32E-01	2.39E-01	9.18E-01	3.17E-01	1.19E+00	3.10E-01
F14	0.99800	1.91E-16	0.99800	1.62E-16	0.99800	1.40E-16	0.99800	2.39E-16
F15	0.00040	2.79E-04	0.00070	4.62E-04	0.00080	4.65E-04	0.00052	3.94E-04
F16	-1.03163	4.97E-16	-1.03163	4.65E-16	-1.03163	4.70E-16	-1.03163	4.79E-16
F17	0.39789	0.00E+00	0.39789	0.00E+00	0.39789	0.00E+00	0.39789	0.00E+00
F18	3.00000	1.94E-15	3.00000	1.76E-15	3.00000	1.80E-15	3.00000	1.68E-15
F19	-3.86278	2.25E-15	-3.86278	2.26E-15	-3.86278	2.26E-15	-3.86278	2.23E-15
F20	-3.25859	6.03E-02	-3.21499	3.63E-02	-3.25462	5.99E-02	-3.28633	5.54E-02
F21	-10.15320	5.27E-15	-9.81333	1.29E+00	-10.15320	4.95E-15	-10.15320	5.32E-15
F22	-10.40294	1.09E-15	-10.40294	1.28E-15	-10.40294	1.36E-15	-10.40294	1.40E-15
F23	-10.53641	2.56E-15	-10.53641	2.31E-15	-10.53641	2.42E-15	-10.53641	2.53E-15
rank	2.70		2.74		2.22		2.35	
+/≈/-	4/18/1		7/14/2		NA		2/20/1	

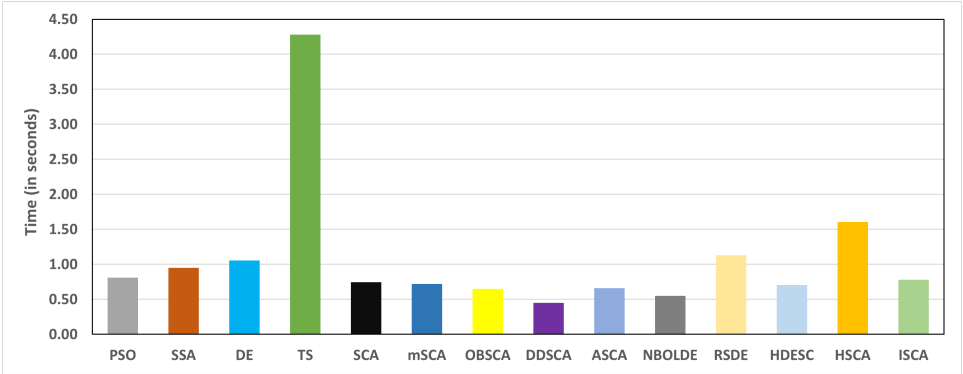

Figure 4: Average computational time recorded by all the compared algorithms on benchmark set

(http://people.brunel.ac.uk/ mastjjb/jeb/orlib/mknapinfo.html) is collected to verify the efficiency of the BISCA. In this paper, the experiments are carried out on the MKP by diving the considered benchmark MKPs in two sets. The first set consists of 38 problems from Chu's instances and Weing instances, while the second problem set includes 11 problem sets from GK's instances. The optimal profit for the first problem set is known and therefore all the results are also compared with the optimal solution. However, on most of the problems of the second set, the optimal solution is unknown due to their larger dimensionality. Therefore, the performance comparison on this set is measured based on the solutions received corresponding to the linear programming problem by relaxing all the functions into linear forms.

To solve these MKPs, the proposed BISCA is applied and its results are compared with other algorithms such as binary PSO (BPSO), binary SSA (BSSA), binary DE (BDE), binary TS (BTS), binary SCA (BSCA), binary mSCA (BmSCA), binary OBSCA (BOBSCA), binary DDSCA (BDDSCA), binary ASCA (BASCA), binary NBOLDE (BNBOLDE), binary RSDE (BRSDE), binary HDESC (BHDESC), binary HSCA (BHSCA), and binary tabu search (BTabu), which are used in the previous section for comparison on continuous problems. The binary versions of these algorithms are designed using the same initialization procedure, same transformation function, and the same repair operator as used by the BISCA. For a fair comparison, the population size in the BISCA and all these algorithms is set to 100. Furthermore, some other algorithms, binary fruit fly optimization algorithm (BFOA) [49], hybrid harmony search-based algorithm (BHHS) [51], binary grey wolf optimizer (BGWO) [50] and hybrid quantum particle swarm optimization (QPSO) [47], are picked to compare the search accuracy of the BISCA as these are binary algorithms specially designed for solving the MKPs in the literature. The reason for selecting the BGWO for comparison is that our BISCA embeds the same initialization and repair operators. However, the search mechanism and the transformation function are different in the BISCA. Collectively, all the parameter details are provided in Table 7.

The stopping criteria in all the algorithms in set to 100000 function evaluations and 30 independent runs are executed to record the results on a PC with configuration: Intel i7-10510U, 2.30GHz CPU, and 16GB RAM. The solution of the relaxed dual programming problem of each knapsack is reported in [50], we have used them for the comparison of algorithms on the second set of MKPs.

4.2.1. Comparison of results

The proposed BISCA is applied on the first MKP set with the parameter settings as discussed in the previous section and the produced results are reported in Table 8. In this table, 38 instances of the MKP are taken and the known optimal solutions are presented to compare the performance of the algorithm. The results are shown in terms of Mean and Std of profits produced in 30 independent runs of algorithm. In the last two rows, the sum of ranks and overall ranks are provided for each compared algorithm based on ranking the produced solution on each instance. This ranking is provided in descending order of mean profit produced by algorithms. The comparison indicates that the proposed BISCA provides better profit than the BPSO, BSSA, BDE, BTS, BSCA, BmSCA, BOBSCA, BDDSCA, BASCA, BNBOLDE, BRSDE, BHDESC, BHSCA, BTabu, BFOA, BHHS, BGWO, and QPSO on 21, 21, 24, 24, 25, 28, 37, 38, 30, 37, 31, 35, 36, 28, 32, 30, 15, and 24 instances, respectively, however, the BISCA performs poor than these algorithms on 6, 4, 4, 3, 4, 2, 0, 0, 1, 0, 0, 0, 0, 3, 1, 1, 9 and 6 instances, respectively. In the remaining instances, all the algorithms provide similar results to the BISCA. This comparison indicates the superior search performance of the BISCA in providing better profits for the MKPs. The sum of ranks (Rank-sum) and overall ranks also support this fact as the BISCA has the best ranking, followed by the BGWO, BSSA, BPSO, QPSO, BSCA, BDE, BTS, BmSCA, BRSDE, BTabu, BHSCA, BHHS, BASCA, BHDESC, BFOA, BOBSCA, BNBOLDE, and BDDSCA in sequential order.

Table 7: Details of algorithm parameters

Algorithm	Parameters
BISCA	$N = 100, fes_{max} = 100000, T(y) = y/\sqrt{1+y^2}$
BPSO	$N = 100, fes_{max} = 100000, c_1 = c_2 = 2.0, w_{min} = 0.2, w_{max} = 0.9,$ $w = w_{max} - (w_{max} - w_{min} \times (t/t_{max}), T(y) = y/\sqrt{1+y^2}$
BSSA	$N = 100, fes_{max} = 100000, c_1 = 2 \cdot exp(-(4 \cdot t/t_{max})), T(y) = y/\sqrt{1+y^2}$
BDE	$N = 100, fes_{max} = 100000, F = CR = 0.5, T(y) = y/\sqrt{1+y^2}$
BTS	$n_s, 10, SN = 10, fes_{max} = 100000, T(y) = y/\sqrt{1+y^2}$
BSCA	$N = 100, fes_{max} = 100000, r_1 = 2 - 2 \times (t/t_{max}), T(y) = y/\sqrt{1+y^2}$
BmSCA	$N = 100, fes_{max} = 100000, w_{start} = 2, w_{end} = 0,$ $w = w_{end} + (w_{start} - w_{end}) \times (t_{max} - t)/t, T(y) = y/\sqrt{1+y^2}$
BOBSCA	$N = 100, fes_{max} = 100000, r_1 = 2 - 2 \times (t/t_{max}), T(y) = y/\sqrt{1+y^2}$
BDDSCA	$N = 100, fes_{max} = 100000, r_1 = 2 - 2 \times (t/t_{max}), r_3 = 2 \cdot r_5 - r_6,$ $r_5 \in [0.5, 1.5], r_6 = 2 \cdot (t/t_{max}), T(y) = y/\sqrt{1+y^2}$
BASCA	$N = 100, fes_{max} = 100000, r_1 = 4 \cdot (1 - t/t_{max}) \cdot (1 - 2^{(t/t_{max}-1)}),$ $y_{k+1} = 4 \cdot y_k \cdot (1 - y_k), \lambda = (t_{max} - t + 1)/t_{max}, T(y) = y/\sqrt{1+y^2}$
BNBOLDE	$N = 100, fes_{max} = 100000, F_1 = F_2 = 0.4, CR = 0.9, Jr = 0.3, T(y) = y/\sqrt{1+y^2}$
BRSDE	$N = 100, fes_{max} = 100000, F = 0.65, CR = 0.9, \alpha = 5 \cdot d, \beta = 10 \cdot d, \gamma = 3,$ $\delta = 10, \epsilon = 10^{-4}, l_t = (fes/10^5)^3, T(y) = y/\sqrt{1+y^2}$
BHDESC	$N = 100, fes_{max} = 100000, r_1 = 2 - 2 \times (t/t_{max}), CR = 0.3, T(y) = y/\sqrt{1+y^2}$
BHSCA	$N = 100, fes_{max} = 100000, F \in [0.2, 0.8], CR = 0.2, r_1 = 2 - 2 \times (t/t_{max}),$ $CR = 0.3, T(y) = y/\sqrt{1+y^2}$
BTabu	$N = 100, fes_{max} = 100000, T(y) = y/\sqrt{1+y^2}$
BFOA	$N = 40, fes_{max} = 100000, S = L = 3, b = 30$
BHHS	$HMS = 40, fes_{max} = 100000, HMCR = 0.74, gn = 20, S = 2, L = 3$
BGWO	$N = 20, fes_{max} = 100000, T(y) = \lvert tanh(y) \rvert$
QPSO	$N = 20, fes_{max} = 100000, c_1 = c_3 = 0.4, c_2 = 0.2, \alpha = 0.1, \beta = 0.9$

The success rate on these 38 problems is shown in Table 9 for all the algorithms, which indicates the competitive performance of the BISCA as compared to other algorithms. The average success rate determined on all the 38 problems for BPSO, BSSA, BDE, BTS, BSCA, BmSCA, BOBSCA, BDDSCA, BASCA, BNBOLDE, BRSDE, BHDESC, BHSCA, BTabu, BFOA, BHHS, BGWO, QPSO and BISCA are 50.00%, 56.67%, 44.91%, 43.51%, 50.35%, 38.95%, 10.00%, 2.54%, 36.75%, 13.95%, 36.32%, 30.79%, 32.72%, 36.84%, 33.33%, 38.33%, 60.18%, 51.84%, and 58.95%, respectively. This illustrates the outperform search efficiency of the ISCA as compared to all other algorithms in providing optimal solutions to the problem. The success rate percentage for the BGWO is slightly better than the BISCA, however, the previous section has demonstrated that the profits provided by the BISCA are still better than the BGWO. Hence, in terms of robustness, the BISCA is better than the BGWO.

Although the Mean and Std of profits and success rates comparison have manifested the better accuracy of the solutions provided by the BISCA than other algorithms, statistical analysis of results should be conducted to verify the difference in results statistically. Therefore, the Wilcoxon signed-rank test [77] is used at a 5% significance level. The obtained p-values and the statistical outcomes (Stat) are shown in Table 10, where the meaning of "+/-/≈" is the better, worst, or statistically similar performance of the BISCA to its competitive algorithm. A collective summary of results is shown at the ending three rows, which indicates that better results have been achieved by the BISCA as compared to other algorithms. Furthermore, the computational time is also measured for all the compared algorithms, which has been shown in Figure 5. In this figure, an average computational time is shown in solving all the problems. The figure indicates that the BTS has the highest computational time among all the other algorithms. The lowest computation time is taken by the BTabu. The average computational times for the BPSO, BSCA, BmSCA, BOBSCA, BASCA,

BNBOLDE, BRSDE, BHDESC, BHSCA, BQPSO, and BISCA are very close to each other. This comparison indicates that the BISCA does not consume very high computational time as compared to other algorithms, but provides very good results within this competitive computational time.

The convergence curves on some selected problems are shown in Figure 6. In this figure, the average relative error in profits is shown which has been calculated over 30 independent runs of each algorithm. The curves indicate the better convergence behavior of the BISCA and fast convergence speed than all other algorithms on most of the problems.

The results on the second set of MKPs, i.e., on large-scale MKP, are shown in Table 11, where the Rank-sum and overall rank for each compared algorithm are presented based on their performances on each problem. These results are produced using the same parameter settings as mentioned in Table 7. In this table, the scale of the problem is presented in the second column and the best-known solutions to each instance are shown in the third column, which are obtained from the corresponding linear programming relaxation. To compare the results of the BISCA on this problem set, the same set of algorithms, i.e. BPSO, BSSA, BDE, BTS, BSCA, BmSCA, BOBSCA, BDDSCA, BASCA, BNBOLDE, BRSDE, BHDESC, BHSCA, BTabu, BFOA, BHHS, BGWO, and QPSO are used. Mean and Std values of profits show the better solution accuracy of the BISCA than the BPSO, BSSA, BTS, BSCA, BmSCA, BOBSCA, BDDSCA, BRSDE, BTabu, BFOA and BHHS on all the problems, than the QPSO in 10 problems, than the BNBOLDE in 9 problems, than the BDE and BASCA in 8 problems, and than the BHDESC, BHSCA and BGWO in only 6 problems. The rank of algorithms shown in the last two rows of table indicates that the BISCA is the winner optimizer on this problem set. Furthermore, the statistical comparison is provided in Table 12 by conducting the Wilcoxon signed-rank test, which verifies the significantly better performance of the BISCA than other optimization algorithms on most of the problems. However, the performances of the BHSCA and BGWO are very competitive with the BISCA.

Based on all the above performance measures, the effectiveness of the BISCA is verified in efficiently solving the MKP. The comparison with other algorithms based on the statistical test indicates the better solution accuracy of the BISCA than most of the other compared algorithms. The overall rank calculated on both of the problem sets manifests that the BISCA is the winner on these sets to provide better profit in solving the MKPs.

Hence, based on the performance of the ISCA on benchmark test problems and MKP, it can be concluded that all the embedded strategies in the ISCA have enhanced the search efficiency of the ISCA. As we have discussed earlier that the modified search mechanism of the SCA together with its hybridization with the DE has successfully maintained the balance of exploration and exploitation during the search procedure of the ISCA. On one side, where the modified search scheme of the SCA enhances the exploitation skills and speeds up the convergence rate of the algorithm, on the other side, hybridization of this modified SCA scheme with DE maintains the diversity of the population to discover new promising search regions of the search space. The proposed settings of control parameters also play an important role in maintaining the balance between exploration and exploitation in the algorithm. All these strategies jointly enhance the performance of the ISCA and are desired to keep the balance between diversity and convergence in the ISCA. The computational results produced by the ISCA for MKPs manifest its significantly better search efficiency compared to other metaheuristic algorithms.

Although the conducted experiments have verified the better search efficiency of the ISCA than other algorithms in solving diverse categories of benchmark optimization problems and MKP, there are some limitations to our study. For very large dimensional MKP, the ISCA is not efficient enough to locate near-to-optimal results. Increasing the complexity of the problems due to the large search space degrades the exploration and convergence ability of the algorithm in these instances. Moreover,

in this study, we have developed the ISCA for solving only single-objective optimization problems with continuous search space, however, there are many optimization problems having multiple objective functions and discrete search space.

Table 8: Mean and Std value of profits produced by the proposed BISCA and other algorithms on first set of the MKP over 30 independent runs

Problem set	Size	Optimal solution	BPSO		BSSA		BDE		BTS		BSCA		BmSCA		BOBSCA		BDDSCA		BASCA	
			Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Chu-1	**5 × 100**	**24381**	24381.00	0.00	24375.93	13.14	24373.57	15.83	24352.33	21.92	24365.80	18.93	24367.47	23.15	24296.40	49.24	24003.00	0.00	24334.17	32.13
Chu-2	**5 × 100**	**24274**	24274.00	0.00	24274.00	0.00	24256.03	36.60	24256.80	21.12	24269.60	13.43	24272.53	8.03	24117.67	3.65	23970.00	0.00	24233.07	57.57
Chu-3	**5 × 100**	**23551**	23534.00	6.75	23530.17	6.55	23523.37	2.01	23524.87	5.96	23523.87	3.34	23524.50	4.58	23523.00	0.00	23368.00	0.00	23524.27	9.18
Chu-4	**5 × 100**	**23534**	23513.27	18.89	23481.60	17.72	23465.20	24.66	23465.00	11.04	23478.53	15.47	23467.40	23.66	23439.20	8.30	22775.00	0.00	23470.90	26.01
Chu-5	**5 × 100**	**23991**	23963.23	13.31	23956.53	14.33	23960.73	11.37	23944.00	20.09	23952.77	30.71	23947.53	11.92	23847.97	20.11	23187.00	0.00	23929.57	31.00
Chu-6	**5 × 100**	**24613**	24590.33	17.65	24608.20	5.98	24606.20	6.05	24601.40	2.19	24603.13	11.40	24576.67	34.10	24529.00	0.00	24042.00	0.00	24563.27	42.16
Chu-7	**5 × 100**	**25591**	25588.67	12.78	25588.67	12.78	25567.67	33.56	25549.00	34.88	25588.67	12.78	25560.67	35.28	25490.33	51.72	24854.00	0.00	25533.67	30.05
Chu-8	**5 × 100**	**23410**	23368.17	2.65	23384.80	16.78	23375.87	9.73	23389.60	18.14	23386.97	17.87	23363.37	16.60	23340.33	42.76	23013.57	35.97	23366.07	16.10
Chu-9	**5 × 100**	**24216**	24204.00	0.00	24210.90	6.55	24205.60	4.15	24212.80	5.40	24201.30	4.19	24204.00	0.00	24173.33	6.34	23551.33	56.60	24196.03	16.27
Chu-10	**5 × 100**	**24411**	24391.40	24.47	24402.80	16.80	24382.33	32.69	24363.80	17.49	24399.10	19.61	24378.07	32.25	24281.40	18.30	23929.00	0.00	24330.90	44.42
Chu-11	**5 × 100**	**42757**	42715.40	21.16	42722.33	24.93	42706.43	9.59	42702.93	7.46	42717.13	22.37	42708.47	13.19	42681.50	49.67	42481.00	0.00	42688.13	18.57
Chu-12	**5 × 100**	**42545**	42481.73	29.67	42472.30	4.57	42460.50	6.03	42458.00	3.21	42487.63	31.39	42467.73	6.02	42453.13	21.18	42256.00	0.00	42456.87	9.65
Chu-13	**5 × 100**	**41968**	41948.07	16.19	41956.50	5.92	41946.47	12.57	41939.53	12.15	41944.13	15.24	41942.60	17.18	41922.00	0.00	41604.00	0.00	41936.27	14.23
Chu-14	**5 × 100**	**45090**	45069.60	3.54	45070.60	0.50	45070.30	0.47	45070.70	3.65	45071.53	3.50	45070.17	0.38	45042.63	50.88	44744.00	0.00	45066.30	11.52
Chu-15	**5 × 100**	**42218**	42218.00	0.00	42207.60	12.10	42191.77	26.93	42153.53	18.22	42203.33	11.43	42212.40	10.32	42125.97	23.10	41610.00	0.00	42159.83	33.32
Chu-16	**5 × 100**	**42927**	42927.00	0.00	42927.00	0.00	42927.00	0.00	42927.00	0.00	42927.00	0.00	42927.00	0.00	42899.47	26.90	42673.00	0.00	42920.97	15.88
Chu-17	**5 × 100**	**42009**	42009.00	0.00	42009.00	0.00	42009.00	0.00	42009.00	0.00	42009.00	0.00	42009.00	0.00	41980.97	5.29	41657.60	401.61	42008.03	5.29
Chu-18	**5 × 100**	**45020**	45004.03	5.69	45005.67	2.67	45008.50	3.09	45006.07	3.67	45004.27	14.43	45006.80	6.05	44895.93	53.46	44704.00	0.00	44983.83	36.57
Chu-19	**5 × 100**	**43441**	43362.23	33.08	43376.17	27.53	43378.63	26.42	43372.27	39.29	43371.93	22.52	43359.30	18.35	43305.60	20.49	42849.00	0.00	43335.03	17.42
Chu-20	**5 × 100**	**44554**	44510.87	10.18	44522.07	14.82	44515.73	12.75	44515.30	13.12	44515.93	11.44	44512.63	10.41	44505.93	1.95	44072.00	0.00	44510.87	0.51
Chu-21	**5 × 100**	**59822**	59822.00	0.00	59822.00	0.00	59822.00	0.00	59822.00	0.00	59822.00	0.00	59803.60	10.32	59821.20	4.38	59798.00	0.00	59822.00	0.00
Chu-22	**5 × 100**	**62081**	61996.20	25.58	62052.43	44.60	61987.10	18.31	61996.80	38.84	62017.33	46.21	62054.87	44.08	61916.40	22.58	61608.00	0.00	61964.10	33.02
Chu-23	**5 × 100**	**59802**	59745.73	6.41	59746.47	5.88	59749.23	15.24	59745.03	10.76	59745.77	11.05	59746.13	8.53	59743.00	0.00	59334.00	0.00	59747.37	15.04
Chu-24	**5 × 100**	**60479**	60478.93	0.25	60479.00	0.00	60476.00	6.82	60458.50	25.03	60479.00	0.00	60477.80	4.57	60404.87	12.61	60193.00	0.00	60450.77	26.54
Chu-25	**5 × 100**	**61091**	61083.80	14.65	61091.00	0.00	61091.00	0.00	61091.00	0.00	61089.80	6.57	61091.00	0.00	61026.67	39.77	60746.00	0.00	61083.33	19.31
Chu-26	**5 × 100**	**58959**	58931.40	9.35	58941.00	10.96	58933.70	8.67	58925.17	7.51	58924.53	7.65	58921.13	4.31	58920.00	0.00	58920.00	0.00	58921.13	4.31
Chu-27	**5 × 100**	**61538**	61528.10	15.38	61529.97	13.07	61519.57	13.64	61515.63	13.12	61517.10	16.17	61507.70	7.53	61505.00	0.00	61105.00	122.08	61511.43	20.75
Chu-28	**5 × 100**	**61520**	61497.17	15.37	61507.00	14.17	61518.97	5.66	61515.87	10.72	61502.40	15.08	61501.40	15.45	61442.17	48.76	61039.00	0.00	61492.80	25.33
Chu-29	**5 × 100**	**59453**	59451.27	9.49	59453.00	0.00	59453.00	0.00	59453.00	0.00	59453.00	0.00	59453.00	0.00	59448.90	22.46	59124.50	97.13	59453.00	0.00
Chu-30	**5 × 100**	**59965**	59960.00	0.00	59960.00	0.00	59960.00	0.00	59958.87	6.21	59960.00	0.00	59960.00	0.00	59879.43	73.63	59650.00	0.00	59955.63	11.86
Weing-1	**2 × 28**	**141278**	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00	140503.70	146.24	140477.00	0.00	141278.00	0.00
Weing-2	**2 × 28**	**130883**	130883.00	0.00	130883.00	0.00	130883.00	0.00	130883.00	0.00	130877.67	29.21	130744.33	55.32	130723.00	0.00	130382.60	692.44	130883.00	0.00
Weing-3	**2 × 28**	**95677**	95658.67	24.51	95677.00	0.00	95633.67	17.29	95677.00	0.00	95672.00	15.26	95642.00	23.30	95147.67	344.73	95018.47	244.38	95677.00	0.00
Weing-4	**2 × 28**	**119337**	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00	117484.80	985.44	107681.80	3436.05	119337.00	0.00
Weing-5	**2 × 28**	**98796**	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98499.47	1128.49	98796.00	0.00
Weing-6	**2 × 28**	**130623**	130623.00	0.00	130623.00	0.00	130571.00	134.84	130623.00	0.00	130597.00	98.95	130259.00	98.95	130233.00	0.00	130244.33	71.62	130610.00	71.20
Weing-7	**2 × 105**	**1095445**	1095405.10	30.88	1095386.20	15.98	1095384.10	11.50	1095384.10	11.50	1095394.60	25.63	1095388.30	19.22	1095299.83	136.58	1094757.00	0.00	1095382.00	0.00
Weing-8	**2 × 105**	**624319**	624319.00	0.00	624319.00	0.00	623887.93	1117.80	624319.00	0.00	624319.00	0.00	624319.00	0.00	619915.00	607.58	619101.00	0.00	624319.00	0.00
Rank-sum			6.40		4.90		6.80		7.80		6.70		8.60		16.80		19.00		11.50	
Overall rank			4		3		7		8		6		9		17		19		14	

Table 8 (continued)

Problem set	Size	Optimal solution	BNBOLDE		BRSDE		BHDESC		BHSCA		BTabu		BFOA		BHHS		BGWO		QPSO		BISCA	
			Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Chu-1	**5 × 100**	**24381**	24291.13	35.20	24341.63	27.20	24343.13	26.90	24359.03	26.25	24343.30	27.13	24331.73	48.50	24363.17	32.23	24377.90	12.01	24381.00	0.00	24379.73	6.94
Chu-2	**5 × 100**	**24274**	24137.43	47.66	24246.27	48.95	24224.50	65.59	24209.13	66.16	24248.60	22.73	24222.20	53.53	24249.30	33.16	24274.00	0.00	24271.00	16.43	24274.00	0.00
Chu-3	**5 × 100**	**23551**	23523.87	3.34	23523.87	3.34	23523.00	0.00	23523.37	2.01	23523.00	0.00	23517.27	17.34	23527.00	6.75	23527.43	7.81	23532.50	7.35	23535.33	5.14
Chu-4	**5 × 100**	**23534**	23422.20	31.36	23452.37	26.10	23452.27	27.36	23453.63	23.73	23460.77	7.07	23460.50	19.86	23464.93	11.15	23494.23	17.14	23486.13	13.65	23498.87	19.36
Chu-5	**5 × 100**	**23991**	23861.07	60.55	23945.20	25.48	23911.50	47.83	23944.57	26.23	23942.07	27.85	23944.10	35.43	23942.50	15.47	23970.23	12.60	23956.50	11.18	23959.90	9.89
Chu-6	**5 × 100**	**24613**	24498.63	76.03	24591.00	24.63	24580.40	40.63	24593.80	24.86	24601.40	2.19	24559.30	30.93	24564.87	23.99	24602.60	18.39	24598.10	18.15	24611.80	3.66
Chu-7	**5 × 100**	**25591**	25439.10	60.89	25535.67	32.35	25572.33	31.48	25545.17	43.35	25520.33	23.63	25521.43	43.12	25530.33	24.20	25591.00	0.00	25539.67	31.48	25591.00	0.00
Chu-8	**5 × 100**	**23410**	23308.33	55.26	23356.17	23.53	23344.13	40.67	23367.07	16.85	23380.00	13.65	23352.80	24.59	23350.63	18.93	23371.80	18.04	23373.83	10.39	23375.73	12.02
Chu-9	**5 × 100**	**24216**	24172.80	22.62	24200.77	8.76	24197.63	12.89	24204.40	2.19	24209.77	11.02	24187.60	27.87	24191.80	23.68	24207.60	5.59	24216.00	0.00	24206.80	5.16
Chu-10	**5 × 100**	**24411**	24294.40	39.37	24338.37	37.38	24339.20	45.16	24362.70	41.78	24328.10	24.98	24318.53	37.36	24343.70	38.49	24407.00	5.75	24379.27	33.69	24405.10	12.09
Chu-11	**5 × 100**	**42757**	42652.50	42.09	42703.67	11.81	42703.33	39.47	42703.67	17.76	42703.00	1.44	42701.57	15.86	42704.80	0.76	42705.87	13.92	42708.47	13.19	42740.70	25.33
Chu-12	**5 × 100**	**42545**	42449.17	23.86	42457.97	3.55	42458.90	4.83	42460.73	6.30	42458.43	3.94	42453.90	21.60	42459.33	9.36	42477.27	28.60	42470.70	15.34	42478.60	25.01
Chu-13	**5 × 100**	**41968**	41906.90	39.92	41936.90	14.59	41932.23	13.96	41943.50	14.39	41928.37	11.48	41920.83	22.68	41943.23	20.34	41953.80	10.91	41958.23	4.20	41957.67	6.10
Chu-14	**5 × 100**	**45090**	45014.33	52.69	45070.17	0.38	45072.20	6.05	45072.37	6.00	45059.83	22.89	45028.83	30.89	45023.90	29.99	45070.90	0.31	45055.77	18.69	45072.53	5.94
Chu-15	**5 × 100**	**42218**	42130.30	37.17	42181.17	29.70	42152.00	36.21	42148.03	33.03	42153.10	22.72	42188.27	46.70	42211.03	15.23	42218.00	0.00	42216.43	8.58	42213.20	9.76
Chu-16	**5 × 100**	**42927**	42878.30	63.21	42927.00	0.00	42914.13	20.21	42922.90	12.51	42922.80	23.00	42865.30	52.01	42879.77	54.45	42927.00	0.00	42917.17	26.47	42927.00	0.00
Chu-17	**5 × 100**	**42009**	41973.13	63.33	42009.00	0.00	42003.20	11.80	42007.07	7.36	42009.00	0.00	41986.27	47.02	41998.80	31.12	42009.00	0.00	42009.00	0.00	42009.00	0.00
Chu-18	**5 × 100**	**45020**	44926.57	71.45	45001.90	12.70	45003.83	3.02	45006.03	3.87	44994.67	26.26	44969.57	49.97	44996.83	34.93	45008.40	4.34	45017.67	7.28	45007.00	2.49
Chu-19	**5 × 100**	**43441**	43308.17	51.88	43363.40	35.34	43352.40	33.95	43361.00	31.91	43358.30	28.25	43352.37	52.82	43358.27	45.23	43374.63	15.23	43390.50	45.01	43406.47	26.89
Chu-20	**5 × 100**	**44554**	44466.17	52.05	44506.20	14.91	44508.67	2.73	44507.03	12.77	44511.00	0.00	44467.43	38.67	44490.17	27.68	44529.60	18.45	44513.53	9.78	44535.70	13.64
Chu-21	**5 × 100**	**59822**	59818.00	9.10	59822.00	0.00	59822.00	0.00	59822.00	0.00	59822.00	0.00	59806.20	20.52	59816.63	9.89	59822.00	0.00	59822.00	0.00	59822.00	0.00
Chu-22	**5 × 100**	**62081**	61939.30	35.39	61974.30	22.68	61973.90	26.29	61974.33	19.29	61983.00	48.98	61986.27	59.64	61972.33	43.57	62014.80	27.53	61991.87	31.75	62016.07	26.66
Chu-23	**5 × 100**	**59802**	59728.03	24.14	59748.90	13.56	59743.47	12.28	59748.03	15.05	59743.73	2.79	59742.93	28.71	59734.73	20.10	59782.30	24.14	59752.80	15.83	59776.00	24.00
Chu-24	**5 × 100**	**60479**	60409.17	16.23	60460.33	26.24	60456.40	25.67	60464.50	24.27	60460.70	22.33	60433.50	28.38	60434.63	26.85	60479.00	0.00	60445.97	28.41	60479.00	0.00
Chu-25	**5 × 100**	**61091**	61011.47	36.90	61074.73	30.42	61055.63	42.09	61050.93	41.82	61085.00	13.65	61060.80	19.57	61070.83	23.86	61091.00	0.00	61085.80	12.48	61091.00	0.00
Chu-26	**5 × 100**	**58959**	58921.70	5.19	58927.87	8.56	58922.80	6.37	58926.70	8.35	58920.53	2.92	58921.77	13.77	58923.97	7.31	58938.47	5.58	58934.43	13.65	58945.07	10.78
Chu-27	**5 × 100**	**61538**	61469.03	54.94	61517.37	14.11	61505.57	25.23	61511.43	12.59	61477.77	41.97	61488.43	38.26	61494.40	36.70	61538.00	0.00	61516.93	15.55	61536.40	4.88
Chu-28	**5 × 100**	**61520**	61433.77	45.48	61500.37	15.19	61490.17	34.70	61511.73	13.94	61502.77	19.32	61468.80	21.57	61482.70	22.83	61512.23	13.36	61505.00	15.49	61516.90	9.46
Chu-29	**5 × 100**	**59453**	59351.03	72.28	59449.53	13.19	59451.27	9.49	59446.07	17.98	59447.93	27.75	59438.37	42.59	59453.00	0.00	59453.00	0.00	59448.07	27.02	59453.00	0.00
Chu-30	**5 × 100**	**59965**	59900.93	52.61	59960.00	0.00	59952.20	16.66	59960.00	0.00	59957.37	10.55	59945.83	21.41	59947.67	19.96	59960.00	0.00	59960.00	0.00	59960.17	0.91
Weing-1	**2 × 28**	**141278**	141091.10	344.58	141278.00	0.00	141278.00	0.00	141251.30	146.24	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00	141278.00	0.00
Weing-2	**2 × 28**	**130883**	130797.67	81.19	130883.00	0.00	130824.33	78.42	130845.67	68.83	130883.00	0.00	130883.00	0.00	130883.00	0.00	130883.00	0.00	130883.00	0.00	130883.00	0.00
Weing-3	**2 × 28**	**95677**	95459.23	309.30	95632.00	15.26	95534.47	250.24	95508.83	273.45	95677.00	0.00	95677.00	0.00	95677.00	0.00	95677.00	0.00	95677.00	0.00	95633.67	17.29
Weing-4	**2 × 28**	**119337**	118613.00	1221.14	119337.00	0.00	118779.03	1275.76	118481.17	1457.45	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00	119337.00	0.00
Weing-5	**2 × 28**	**98796**	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00	98796.00	0.00
Weing-6	**2 × 28**	**130623**	130285.00	134.84	130610.00	71.20	130350.00	181.78	130350.00	181.78	130623.00	0.00	130623.00	0.00	130623.00	0.00	130623.00	0.00	130623.00	0.00	130623.00	0.00
Weing-7	**2 × 105**	**1095445**	1095382.00	0.00	1095382.00	0.00	1095382.00	0.00	1095382.00	0.00	1095382.00	0.00	1095384.10	11.50	1095390.40	21.78	1095438.70	19.22	1095388.30	19.22	1095405.10	30.88
Weing-8	**2 × 105**	**624319**	620883.40	1501.02	623893.10	1299.54	622321.13	1938.60	622286.93	1969.94	622379.20	1610.92	623564.63	1390.78	624319.00	0.00	624319.00	0.00	624319.00	0.00	624319.00	0.00
Rank-sum			16.80		10.40		12.80		11.10		11.00		13.70		11.30		4.30		6.40		3.80	
Overall rank			18		10		15		12		11		16		13		2		5		1	

Table 9: Comparison of success rates (in percentage) on first set of the MKPs over 30 independent runs

Problem set	BPSO	BSSA	BDE	BTS	BSCA	BmSCA	BOBSCA	BDDSCA	BASCA	BNBOLDE	BRSDE	BHDESC	BHSCA	BTabu	BFOA	BHHS	BGWO	BQPSO	BISCA
Chu-1	100.00	86.67	76.67	20.00	60.00	73.33	0.00	0.00	26.67	0.00	23.33	20.00	56.67	20.00	40.00	73.33	93.33	100.00	96.67
Chu-2	100.00	100.00	73.33	56.67	90.00	96.67	0.00	0.00	56.67	6.67	66.67	56.67	43.33	43.33	33.33	50.00	100.00	96.67	100.00
Chu-3	0.00	0.00	0.00	3.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.33	0.00	0.00
Chu-4	0.00	0.00	3.33	0.00	0.00	0.00	0.00	0.00	6.67	0.00	0.00	3.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Chu-5	13.33	13.33	6.67	3.33	26.67	0.00	0.00	0.00	3.33	0.00	3.33	3.33	3.33	0.00	16.67	6.67	23.33	0.00	3.33
Chu-6	23.33	60.00	43.33	3.33	40.00	16.67	0.00	0.00	6.67	0.00	20.00	13.33	26.67	3.33	0.00	0.00	73.33	43.33	90.00
Chu-7	96.67	96.67	66.67	40.00	96.67	56.67	0.00	0.00	20.00	0.00	23.33	73.33	40.00	6.67	16.67	13.33	100.00	26.67	100.00
Chu-8	0.00	30.00	6.67	43.33	36.67	0.00	0.00	0.00	0.00	0.00	0.00	3.33	0.00	16.67	10.00	0.00	13.33	6.67	10.00
Chu-9	0.00	60.00	13.33	73.33	0.00	0.00	0.00	0.00	3.33	3.33	6.67	3.33	3.33	73.33	16.67	16.67	30.00	100.00	23.33
Chu-10	50.00	80.00	43.33	3.33	66.67	46.67	0.00	0.00	10.00	3.33	10.00	16.67	23.33	0.00	6.67	13.33	66.67	40.00	73.33
Chu-11	20.00	33.33	3.33	0.00	23.33	6.67	0.00	0.00	0.00	0.00	3.33	23.33	6.67	0.00	3.33	0.00	6.67	6.67	70.00
Chu-12	16.67	0.00	0.00	0.00	20.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	13.33	3.33	10.00
Chu-13	0.00	0.00	0.00	3.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Chu-14	0.00	0.00	0.00	3.33	3.33	0.00	0.00	0.00	0.00	0.00	0.00	10.00	10.00	3.33	0.00	3.33	0.00	0.00	10.00
Chu-15	100.00	56.67	40.00	3.33	36.67	76.67	0.00	0.00	16.67	0.00	30.00	10.00	10.00	6.67	63.33	80.00	100.00	96.67	80.00
Chu-16	100.00	100.00	100.00	100.00	100.00	100.00	46.67	0.00	86.67	43.33	100.00	70.00	90.00	96.67	33.33	53.33	100.00	86.67	100.00
Chu-17	100.00	100.00	100.00	100.00	100.00	100.00	3.33	0.00	96.67	30.00	100.00	80.00	93.33	100.00	80.00	90.00	100.00	100.00	100.00
Chu-18	3.33	0.00	0.00	0.00	0.00	13.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	33.33	60.00	3.33	90.00	0.00
Chu-19	10.00	10.00	10.00	23.33	3.33	0.00	0.00	0.00	0.00	3.33	13.33	6.67	6.67	10.00	20.00	20.00	0.00	40.00	36.67
Chu-20	0.00	6.67	6.67	10.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	20.00	0.00	13.33
Chu-21	100.00	100.00	100.00	100.00	100.00	23.33	96.67	0.00	100.00	83.33	100.00	100.00	100.00	100.00	43.33	76.67	100.00	100.00	100.00
Chu-22	6.67	70.00	3.33	16.67	33.33	73.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	16.67	23.33	10.00	13.33	10.00	13.33
Chu-23	0.00	0.00	6.67	3.33	3.33	0.00	0.00	0.00	6.67	0.00	3.33	0.00	6.67	0.00	10.00	0.00	56.67	6.67	40.00
Chu-24	93.33	100.00	83.33	33.33	100.00	93.33	0.00	0.00	33.33	0.00	50.00	46.67	70.00	33.33	10.00	6.67	100.00	26.67	100.00
Chu-25	80.00	100.00	100.00	100.00	96.67	100.00	26.67	0.00	83.33	10.00	73.33	56.67	50.00	83.33	23.33	50.00	100.00	83.33	100.00
Chu-26	3.33	23.33	3.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.33	0.00	6.67	16.67	36.67
Chu-27	70.00	70.00	30.00	20.00	36.67	3.33	0.00	0.00	13.33	3.33	26.67	6.67	16.67	3.33	13.33	10.00	100.00	33.33	90.00
Chu-28	16.67	50.00	96.67	86.67	40.00	40.00	0.00	0.00	26.67	3.33	36.67	40.00	73.33	46.67	0.00	10.00	73.33	50.00	90.00
Chu-29	96.67	100.00	100.00	100.00	100.00	100.00	96.67	0.00	100.00	20.00	93.33	96.67	86.67	96.67	86.67	100.00	100.00	96.67	100.00
Chu-30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.33
Weing-1	100.00	100.00	100.00	100.00	100.00	100.00	3.33	0.00	100.00	76.67	100.00	100.00	96.67	100.00	100.00	100.00	100.00	100.00	100.00
Weing-2	100.00	100.00	100.00	100.00	96.67	13.33	0.00	0.00	100.00	46.67	100.00	63.33	76.67	100.00	100.00	100.00	100.00	100.00	100.00
Weing-3	63.33	100.00	13.33	100.00	90.00	30.00	0.00	0.00	100.00	0.00	10.00	6.67	3.33	100.00	100.00	100.00	100.00	100.00	13.33
Weing-4	100.00	100.00	100.00	100.00	100.00	100.00	6.67	0.00	100.00	73.33	100.00	83.33	73.33	100.00	100.00	100.00	100.00	100.00	100.00
Weing-5	100.00	100.00	100.00	100.00	100.00	100.00	100.00	93.33	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Weing-6	100.00	100.00	86.67	100.00	93.33	6.67	0.00	3.33	96.67	13.33	96.67	30.00	30.00	100.00	100.00	100.00	100.00	100.00	100.00
Weing-7	36.67	6.67	3.33	3.33	20.00	10.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.33	13.33	90.00	10.00	36.67
Weing-8	100.00	100.00	86.67	100.00	100.00	100.00	0.00	0.00	100.00	10.00	90.00	46.67	46.67	40.00	76.67	100.00	100.00	100.00	100.00
Average	50.00	56.67	44.91	43.51	50.35	38.95	10.00	2.54	36.75	13.95	36.32	30.79	32.72	36.84	33.33	38.33	60.18	51.84	58.95

Table 10: Statistical results obtained by the Wilcoxon signed-rank test on first set of the MKPs over 30 independent runs

Problem set	BPSO		BSSA		BDE		BTS		BSCA		BmSCA		BOBSCA		BDDSCA		BASCA		BNBOLDE		BRSDE		BHDESC		BHSCA		BTabu		BFOA		BHHS		BGWO		QPSO	
	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat
Chu-1	1.00E+00	≈	3.75E-01	≈	7.03E-02	≈	4.14E-05	+	3.42E-03	+	7.81E-03	+	3.34E-07	+	6.80E-08	+	5.21E-05	+	1.37E-06	+	3.51E-05	+	1.51E-05	+	7.32E-04	+	1.58E-05	+	1.50E-04	+	1.17E-02	+	7.50E-01	≈	1.00E+00	≈
Chu-2	1.00E+00	≈	1.00E+00	≈	7.81E-03	+	2.44E-04	+	2.50E-01	≈	1.00E+00	≈	6.80E-08	+	4.32E-08	+	2.44E-04	+	3.26E-06	+	1.95E-03	+	2.44E-04	+	2.70E-04	+	5.70E-05	+	8.61E-05	+	6.10E-05	+	1.00E+00	≈	1.00E+00	≈
Chu-3	4.72E-01	≈	3.30E-03	+	2.34E-06	+	1.44E-05	+	3.81E-06	+	2.43E-05	+	1.87E-06	+	5.92E-07	+	1.93E-05	+	3.81E-06	+	2.88E-06	+	1.87E-06	+	2.34E-06	+	1.87E-06	+	1.29E-05	+	4.86E-04	+	3.36E-03	+	8.30E-02	≈
Chu-4	1.02E-02	-	7.22E-04	+	2.78E-04	+	1.30E-05	+	6.79E-04	+	1.19E-04	+	1.44E-06	+	1.36E-06	+	1.21E-03	+	1.70E-06	+	4.13E-06	+	1.37E-05	+	1.02E-05	+	1.60E-06	+	1.82E-06	+	3.70E-06	+	3.56E-01	≈	5.17E-03	+
Chu-5	2.43E-01	≈	3.16E-01	≈	6.00E-01	≈	5.62E-04	+	5.60E-01	≈	1.06E-04	+	1.48E-06	+	8.66E-07	+	8.71E-05	+	2.77E-06	+	1.03E-02	+	2.26E-05	+	1.67E-02	+	4.14E-03	+	5.84E-02	≈	5.01E-04	+	3.00E-03	-	1.53E-01	≈
Chu-6	2.60E-05	+	3.52E-02	+	4.65E-04	+	3.41E-07	+	6.10E-05	+	1.31E-05	+	1.44E-07	+	1.44E-07	+	7.13E-06	+	1.65E-06	+	3.03E-05	+	8.61E-06	+	4.07E-05	+	3.41E-07	+	1.60E-06	+	1.62E-06	+	1.17E-02	+	3.92E-04	+
Chu-7	1.00E+00	≈	1.00E+00	≈	1.95E-03	+	2.21E-05	+	1.00E+00	≈	2.44E-04	+	4.77E-07	+	4.32E-08	+	1.50E-06	+	1.41E-06	+	2.52E-06	+	7.81E-03	+	3.38E-05	+	5.36E-07	+	2.44E-06	+	3.41E-07	+	1.00E+00	≈	2.73E-06	+
Chu-8	1.09E-04	+	1.08E-02	-	7.13E-01	≈	3.06E-03	-	5.41E-03	-	3.46E-03	+	1.33E-04	+	7.75E-07	+	3.11E-02	+	2.07E-05	+	1.19E-04	+	1.70E-04	+	2.90E-02	+	1.21E-01	≈	6.61E-04	+	3.27E-05	+	2.26E-01	≈	3.52E-01	≈
Chu-9	1.56E-02	+	8.22E-03	-	5.08E-01	≈	2.75E-04	-	2.44E-04	+	1.56E-02	+	5.80E-07	+	4.87E-07	+	1.41E-03	+	6.48E-06	+	2.66E-03	+	2.93E-03	+	3.13E-02	+	7.40E-02	≈	1.16E-03	+	4.31E-03	+	7.27E-01	≈	1.62E-06	-
Chu-10	7.34E-03	+	5.45E-01	≈	1.87E-03	+	3.92E-06	+	1.90E-01	≈	6.40E-04	+	1.19E-06	+	4.99E-07	+	5.61E-06	+	1.90E-06	+	7.47E-06	+	1.11E-05	+	1.33E-04	+	1.51E-06	+	3.10E-06	+	1.01E-05	+	6.88E-01	≈	5.80E-04	+
Chu-11	2.84E-03	+	3.02E-02	+	4.25E-05	+	1.50E-05	+	3.40E-02	+	3.60E-04	+	5.53E-06	+	5.80E-07	+	4.43E-06	+	1.66E-06	+	1.86E-05	+	2.54E-03	+	2.06E-04	+	1.82E-05	+	1.11E-04	+	2.13E-05	+	1.75E-05	+	8.11E-05	+
Chu-12	7.54E-01	≈	6.57E-01	≈	1.30E-04	+	7.60E-06	+	2.52E-01	≈	9.58E-02	≈	7.10E-06	+	1.48E-06	+	2.73E-05	+	7.18E-06	+	1.59E-05	+	3.40E-05	+	4.36E-04	+	1.52E-05	+	1.42E-05	+	5.52E-05	+	5.18E-01	≈	1.39E-01	≈
Chu-13	5.98E-03	+	4.75E-01	≈	1.02E-03	+	1.16E-05	+	6.59E-04	+	4.94E-04	+	2.61E-07	+	2.61E-07	+	3.28E-05	+	3.55E-06	+	3.45E-05	+	1.21E-05	+	6.18E-04	+	2.08E-06	+	5.22E-06	+	1.74E-03	+	8.58E-02	≈	8.13E-01	≈
Chu-14	2.40E-03	+	2.94E-01	≈	1.23E-02	+	1.57E-03	+	6.35E-01	≈	1.15E-03	+	6.91E-05	+	9.55E-07	+	7.07E-05	+	3.44E-06	+	1.66E-03	+	1.21E-01	≈	3.93E-01	≈	3.48E-04	+	1.22E-05	+	3.65E-06	+	7.61E-01	≈	6.52E-05	+
Chu-15	3.13E-02	-	6.54E-02	+	1.80E-03	+	1.40E-06	+	2.11E-03	+	1.00E+00	≈	8.08E-07	+	3.26E-07	+	1.51E-05	+	1.43E-06	+	1.36E-04	+	4.55E-06	+	3.74E-06	+	1.50E-06	+	8.30E-03	+	6.56E-01	≈	3.13E-02	-	3.59E-01	≈
Chu-16	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	2.73E-04	+	4.32E-08	+	1.25E-01	≈	2.57E-04	+	1.00E+00	≈	3.91E-03	+	2.50E-01	≈	1.00E+00	≈	8.47E-05	+	1.22E-04	+	1.00E+00	≈	1.25E-01	≈
Chu-17	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	7.24E-08	+	7.45E-07	+	1.00E+00	≈	1.02E-05	+	1.00E+00	≈	3.13E-02	+	5.00E-01	≈	1.00E+00	≈	3.13E-02	+	2.50E-01	≈	1.00E+00	≈	1.00E+00	≈
Chu-18	1.99E-02	+	8.81E-02	≈	1.32E-02	-	3.80E-01	≈	6.62E-01	≈	7.19E-01	≈	1.41E-06	+	7.45E-07	+	3.31E-05	+	2.46E-06	+	1.18E-02	+	2.29E-03	+	3.24E-01	≈	1.57E-01	≈	5.95E-04	+	6.08E-01	≈	8.68E-02	≈	1.86E-05	-
Chu-19	7.12E-05	+	2.00E-04	+	5.55E-05	+	5.07E-03	+	1.55E-04	+	4.97E-06	+	1.43E-06	+	1.16E-06	+	2.50E-06	+	2.83E-06	+	7.36E-06	+	8.14E-06	+	1.42E-04	+	8.67E-05	+	2.01E-04	+	5.35E-04	+	3.49E-05	+	1.34E-01	≈
Chu-20	5.30E-06	+	3.57E-03	+	5.93E-05	+	1.59E-05	+	3.56E-06	+	1.03E-05	+	1.26E-06	+	7.84E-07	+	7.50E-06	+	5.20E-06	+	9.30E-06	+	4.99E-06	+	1.09E-05	+	5.07E-06	+	8.11E-06	+	1.13E-05	+	1.07E-01	≈	1.02E-05	+
Chu-21	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.62E-06	+	1.00E+00	≈	4.32E-08	+	1.00E+00	≈	6.25E-02	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	5.70E-05	+	1.56E-02	+	1.00E+00	≈	1.00E+00	≈
Chu-22	2.73E-03	+	1.33E-03	-	1.94E-05	+	2.36E-02	+	9.89E-01	≈	2.55E-04	-	5.06E-07	+	3.32E-07	+	3.45E-06	+	3.22E-06	+	9.64E-07	+	9.63E-07	+	1.78E-06	+	8.47E-03	+	1.88E-02	+	7.35E-04	+	8.24E-01	≈	3.11E-03	+
Chu-23	1.24E-05	+	1.89E-05	+	1.34E-04	+	4.80E-05	+	8.55E-05	+	1.23E-05	+	9.25E-06	+	1.41E-06	+	1.06E-04	+	2.41E-06	+	5.29E-05	+	9.99E-06	+	2.34E-05	+	8.58E-06	+	3.34E-04	+	7.18E-06	+	3.40E-01	≈	1.06E-03	+
Chu-24	5.00E-01	≈	1.00E+00	≈	6.25E-02	≈	7.71E-05	+	1.00E+00	≈	5.00E-01	≈	9.63E-07	+	4.32E-08	+	8.02E-05	+	1.39E-06	+	6.10E-05	+	2.73E-04	+	3.91E-03	+	7.18E-05	+	5.18E-06	+	3.37E-06	+	1.00E+00	≈	3.37E-05	+
Chu-25	3.13E-02	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	6.12E-06	+	4.32E-08	+	6.25E-02	≈	2.84E-06	+	7.81E-03	+	2.44E-04	+	6.10E-05	+	6.25E-02	≈	3.66E-06	+	6.10E-05	+	1.00E+00	≈	6.25E-02	≈
Chu-26	3.29E-04	+	2.13E-01	≈	4.98E-04	+	1.70E-06	+	1.10E-05	+	7.42E-07	+	6.91E-07	+	6.91E-07	+	1.13E-06	+	1.78E-06	+	3.11E-05	+	1.90E-06	+	8.48E-06	+	7.81E-07	+	2.89E-05	+	4.68E-06	+	2.25E-02	+	1.43E-02	+
Chu-27	3.91E-03	+	3.91E-03	+	3.28E-05	+	1.22E-05	+	3.81E-05	+	5.40E-07	+	1.44E-07	+	9.04E-07	+	5.76E-06	+	1.82E-06	+	3.16E-05	+	2.83E-06	+	3.11E-06	+	1.32E-06	+	7.34E-06	+	4.11E-06	+	2.50E-01	≈	2.19E-05	+
Chu-28	1.60E-05	+	3.05E-03	+	6.25E-01	≈	1.00E+00	≈	4.33E-04	+	6.10E-05	+	2.73E-06	+	1.44E-07	+	1.17E-04	+	3.03E-06	+	3.47E-04	+	3.33E-04	+	1.80E-01	≈	2.44E-04	+	2.70E-06	+	1.15E-05	+	2.27E-01	≈	7.12E-03	+
Chu-29	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	8.21E-07	+	1.00E+00	≈	1.66E-05	+	5.00E-01	≈	1.00E+00	≈	1.25E-01	≈	1.00E+00	≈	1.25E-01	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Chu-30	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	5.00E-01	≈	1.00E+00	≈	1.00E+00	≈	3.82E-06	+	6.80E-08	+	9.38E-02	≈	9.92E-06	+	1.00E+00	≈	1.56E-02	+	1.00E+00	≈	2.50E-01	≈	1.95E-03	+	4.88E-04	+	1.00E+00	≈	1.00E+00	≈
Weing-1	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	7.24E-08	+	4.32E-08	+	1.00E+00	≈	1.56E-02	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Weing-2	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	3.41E-07	+	4.32E-08	+	3.26E-07	+	1.00E+00	≈	6.33E-05	+	1.00E+00	≈	9.77E-04	+	1.56E-02	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Weing-3	5.79E-04	-	3.41E-07	-	1.00E+00	≈	3.41E-07	-	1.62E-06	-	1.80E-01	≈	2.93E-05	+	6.78E-07	+	3.41E-07	-	9.77E-04	+	1.00E+00	≈	9.38E-02	≈	1.95E-02	+	3.41E-07	-	3.41E-07	-	3.41E-07	-	3.41E-07	-	3.41E-07	-
Weing-4	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	2.91E-06	+	1.60E-06	+	1.00E+00	≈	7.81E-03	+	1.00E+00	≈	6.25E-02	≈	7.81E-03	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Weing-5	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	5.00E-01	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Weing-6	1.00E+00	≈	1.00E+00	≈	1.25E-01	≈	1.00E+00	≈	5.00E-01	≈	1.21E-07	+	4.32E-08	+	4.23E-07	+	1.00E+00	≈	3.41E-07	+	1.00E+00	≈	4.59E-06	+	4.59E-06	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
Weing-7	1.00E+00	≈	1.17E-02	+	1.95E-03	+	6.35E-03	+	2.27E-01	≈	2.15E-02	+	1.78E-04	+	6.91E-07	+	9.77E-04	+	9.77E-04	+	9.77E-04	+	9.77E-04	+	9.77E-04	+	9.77E-04	+	6.35E-03	+	9.23E-02	≈	1.62E-04	-	3.86E-02	+
Weing-8	1.00E+00	≈	1.00E+00	≈	1.25E-01	≈	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈	1.05E-06	+	4.32E-08	+	1.00E+00	≈	3.76E-06	+	2.50E-01	≈	2.95E-04	+	2.90E-04	+	2.21E-05	+	1.56E-02	+	1.00E+00	≈	1.00E+00	≈	1.00E+00	≈
+	16		11		18		21		14		22		35		37		25		36		26		31		28		23		30		26		5		14	
-	3		4		1		3		2		1		0		0		1		0		0		0		0		1		1		1		4		3	
≈	19		23		19		14		22		15		3		1		12		2		12		7		10		14		7		11		29		21	

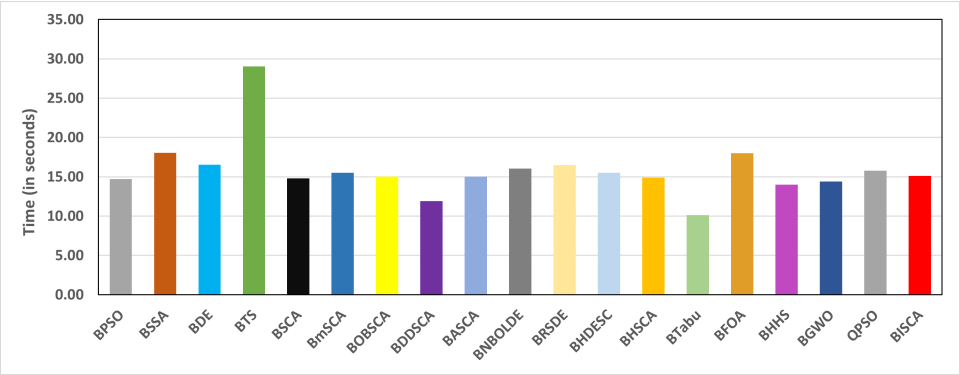

Figure 5: Average computational time recorded by all the compared algorithms on first set of MKPs over 30 independent runs

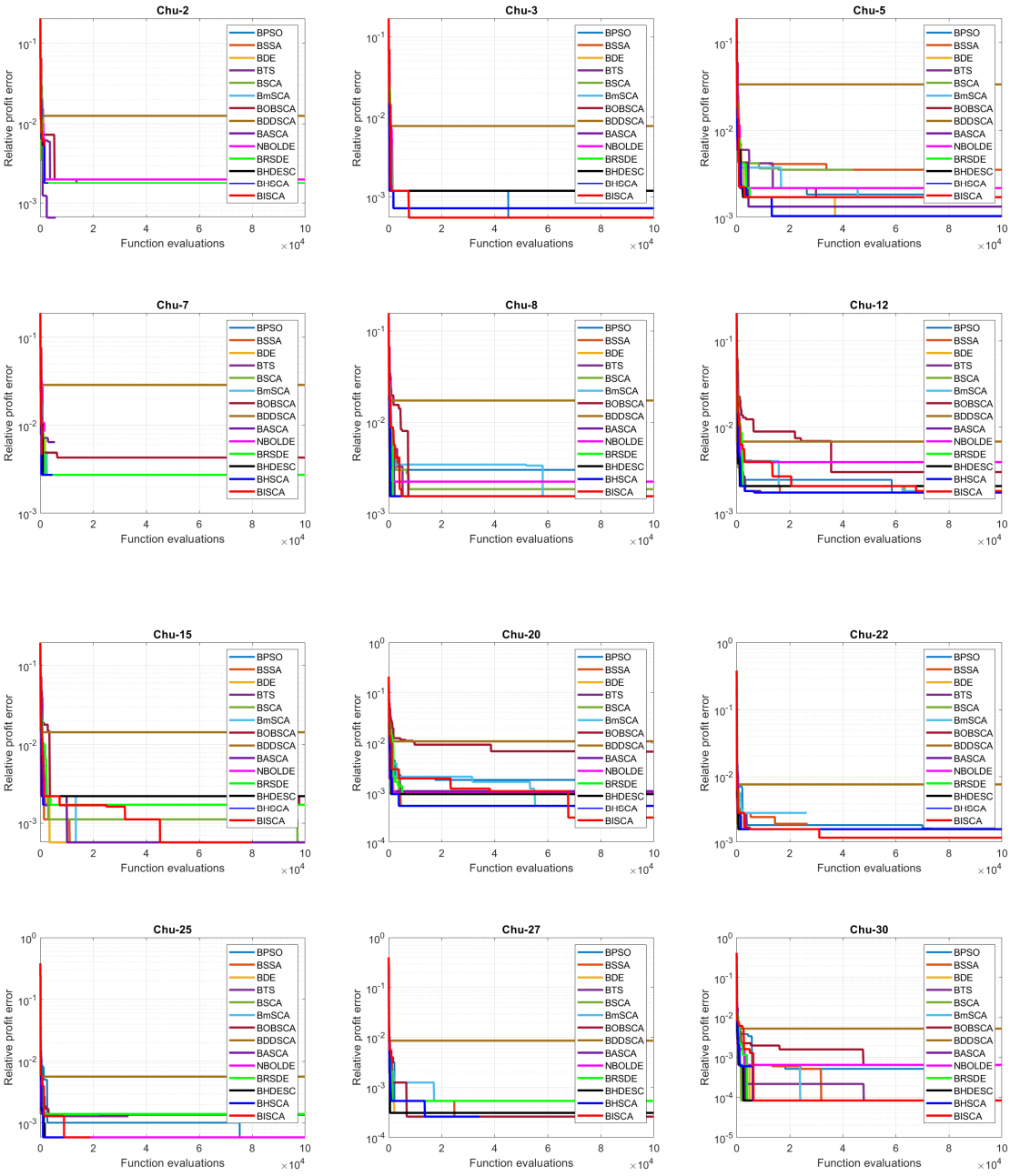

Figure 6: Convergence behavior comparison on MKPs

Table 11: Mean and Std value of profits produced by the proposed BISCA and other algorithms on second set of MKPs over 30 independent runs

Problem set	Size	Known best-solution	BPSO		BSSA		BDE		BTS		BSCA		BmSCA		BOBSCA		BDDSCA		BASCA	
			Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
GK-1	**15 × 100**	**3766**	3756.67	1.42	3755.87	3.77	3762.67	2.86	3756.07	2.05	3754.90	2.84	3753.50	2.27	3754.60	2.21	3707.17	12.60	3755.43	3.16
GK-2	**25 × 100**	**3958**	3946.43	3.16	3939.80	4.25	3946.17	1.53	3941.97	3.24	3941.47	3.60	3940.43	1.77	3940.33	3.82	3862.00	0.00	3941.63	4.48
GK-3	**25 × 150**	**5656**	5638.27	3.65	5636.30	3.08	5642.13	3.25	5638.63	2.24	5637.40	3.64	5633.17	2.35	5636.27	3.54	5528.00	0.00	5637.20	4.84
GK-4	**50 × 150**	**5767**	5735.93	5.76	5726.93	7.80	5746.07	3.81	5728.87	5.19	5730.13	8.03	5717.77	7.25	5724.83	9.39	5569.00	4.76	5745.73	6.72
GK-5	**25 × 200**	**7560**	7543.00	4.09	7539.70	4.28	7548.87	1.91	7539.77	3.80	7541.77	5.26	7539.47	4.34	7540.67	3.83	7470.13	6.86	7544.90	5.33
GK-6	**50 × 200**	**7677**	7639.87	4.24	7627.80	5.07	7636.73	3.54	7627.10	5.20	7630.83	5.19	7629.67	4.21	7629.43	4.28	7587.00	0.00	7640.47	5.89
GK-7	**25 × 500**	**19220**	19198.97	3.24	19197.77	3.05	19204.73	0.91	19195.20	3.60	19200.77	2.66	19198.30	2.20	19200.93	4.30	19148.00	0.00	19199.83	4.10
GK-8	**50 × 500**	**18806**	18764.53	4.91	18753.73	8.54	18765.93	3.27	18752.80	4.44	18752.47	8.11	18760.10	5.00	18754.40	6.19	18615.00	0.00	18769.03	5.59
GK-9	**25 × 1500**	**58087**	58062.43	5.41	58062.40	3.77	58065.70	2.31	58058.47	3.30	58068.10	3.41	58065.27	2.61	58067.53	3.35	57992.00	0.00	58069.57	3.62
GK-10	**50 × 1500**	**57295**	57254.37	3.95	57245.20	7.02	57248.10	4.08	57240.87	4.42	57251.80	4.00	57251.27	5.11	57250.87	4.73	57124.00	0.00	57259.80	5.54
GK-11	**100 × 2500**	**95237**	95132.00	7.40	95118.83	8.94	95069.07	13.03	95101.07	6.34	95125.70	6.96	95130.37	8.02	95126.07	5.99	94990.00	0.00	95142.23	7.72
Rank-sum			8.36		13.45		5.82		12.91		10.91		12.77		12.00		17.73		6.64	
Overall rank			7		15		5		14		11		13		12		19		6	

Table 11 (continued)

Problem set	Size	Known best-solution	BNBOLDE		BRSDE		BHDESC		BHSCA		BTabu		BFOA		BHHS		BGWO		QPSO		BISCA	
			Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
GK-1	**15 × 100**	**3766**	3750.20	6.73	3759.50	3.36	3756.83	3.71	3759.87	2.98	3756.97	3.31	3752.10	5.26	3753.50	3.77	3757.77	3.33	3756.70	1.73	3761.60	3.50
GK-2	**25 × 100**	**3958**	3935.27	5.69	3946.10	2.76	3943.00	2.53	3945.63	1.71	3942.50	2.61	3945.40	3.98	3947.00	4.32	3944.97	2.80	3950.57	5.16	3947.97	2.89
GK-3	**25 × 150**	**5656**	5632.73	5.41	5641.63	3.33	5639.27	2.36	5642.23	3.18	5638.43	5.99	5629.40	10.50	5633.10	8.27	5641.87	3.10	5630.40	6.30	5642.40	3.09
GK-4	**50 × 150**	**5767**	5741.03	7.94	5744.90	5.05	5741.77	6.87	5743.47	3.47	5736.90	8.81	5739.40	10.63	5742.07	8.16	5746.67	4.33	5737.63	5.26	5747.47	3.36
GK-5	**25 × 200**	**7560**	7541.90	4.00	7548.60	3.31	7546.97	2.79	7548.57	1.94	7543.93	5.15	7527.80	10.51	7527.27	10.75	7547.77	3.71	7519.73	11.55	7548.73	3.06
GK-6	**50 × 200**	**7677**	7638.50	10.58	7631.73	5.06	7635.57	7.35	7635.13	2.90	7632.57	8.17	7629.63	17.39	7637.50	12.34	7640.60	5.04	7630.00	8.27	7641.10	4.82
GK-7	**25 × 500**	**19220**	19193.73	4.28	19192.23	8.67	19202.00	3.19	19204.90	1.30	19199.77	3.85	19108.40	27.87	19114.47	27.01	19202.73	3.96	19061.87	39.50	19201.53	2.52
GK-8	**50 × 500**	**18806**	18766.87	5.66	18747.10	5.88	18769.27	4.13	18767.87	3.73	18761.90	7.43	18661.97	27.07	18658.17	24.93	18770.87	5.04	18627.67	27.69	18767.40	4.06
GK-9	**25 × 1500**	**58087**	58065.17	4.65	57991.23	20.89	58072.87	2.40	58072.97	1.81	58066.50	4.45	57679.10	69.39	57620.20	74.82	58072.30	3.81	57638.93	71.04	58069.83	2.26
GK-10	**50 × 1500**	**57295**	57255.83	5.10	57131.83	28.97	57260.30	4.65	57261.50	2.67	57251.93	6.01	56813.83	65.35	56760.73	70.10	57260.73	4.88	56896.53	59.03	57255.80	3.97
GK-11	**100 × 2500**	**95237**	95140.90	5.62	94960.00	22.27	95146.27	6.34	95141.67	4.40	95118.53	11.67	94283.73	78.28	94184.27	98.69	95145.27	4.70	94568.43	84.00	95136.37	8.08
Rank-sum			10.64		10.09		5.18		3.82		9.18		15.55		14.05		3.45		14.27		3.18	
Overall rank			10		9		4		3		8		18		16		2		17		1	

Table 12: Statistical results obtained by the Wilcoxon signed-rank test on second set of MKPs over 30 independent runs

Problem set	BPSO		BSSA		BDE		BTS		BSCA		BmSCA		BOBSCA		BDDSCA		BASCA		BNBOLDE		BRSDE		BHDESC		BHSCA		BTabu		BFOA		BHHS		BGWO		QPSO	
	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat	p-value	Stat
GK-1	1.42E-05	+	3.68E-05	+	2.07E-01	≈	3.03E-06	+	6.88E-06	+	1.67E-06	+	1.66E-06	+	1.58E-06	+	1.08E-05	+	2.52E-06	+	1.95E-02	+	9.71E-05	+	4.99E-02	+	1.28E-04	+	2.08E-06	+	2.94E-06	+	2.34E-04	+	1.16E-05	+
GK-2	1.41E-02	+	1.95E-06	+	5.85E-03	+	8.36E-06	+	6.90E-06	+	1.66E-06	+	3.69E-06	+	1.50E-06	+	4.63E-06	+	1.71E-06	+	1.11E-02	+	1.34E-05	+	1.21E-03	+	8.26E-06	+	7.55E-03	+	2.21E-01	≈	3.42E-04	+	2.93E-02	-
GK-3	1.06E-04	+	2.65E-06	+	8.00E-01	≈	1.01E-05	+	3.63E-06	+	2.48E-06	+	4.71E-06	+	1.61E-06	+	1.84E-04	+	2.65E-06	+	1.26E-01	≈	2.13E-04	+	8.69E-01	≈	5.38E-03	+	3.71E-06	+	1.59E-05	+	4.92E-01	≈	2.45E-06	+
GK-4	1.71E-06	+	1.71E-06	+	1.97E-01	≈	1.89E-06	+	1.71E-06	+	1.71E-06	+	1.71E-06	+	1.70E-06	+	2.47E-01	≈	9.48E-04	+	7.12E-02	≈	4.18E-04	+	4.16E-04	+	2.61E-05	+	3.69E-04	+	3.13E-03	+	5.64E-01	≈	4.38E-06	+
GK-5	4.71E-06	+	2.50E-06	+	7.91E-01	≈	2.93E-06	+	1.99E-05	+	2.96E-06	+	2.96E-06	+	1.67E-06	+	4.89E-03	+	6.35E-06	+	7.27E-01	≈	1.12E-02	+	7.69E-01	≈	2.78E-04	+	1.73E-06	+	2.33E-06	+	2.48E-01	≈	1.72E-06	+
GK-6	1.54E-01	≈	1.71E-06	+	1.66E-04	+	2.50E-06	+	4.92E-06	+	2.19E-06	+	2.08E-06	+	1.67E-06	+	5.09E-01	≈	4.25E-01	≈	4.43E-06	+	1.79E-03	+	3.23E-05	+	5.48E-05	+	3.67E-03	+	3.47E-01	≈	4.68E-01	≈	3.39E-05	+
GK-7	3.99E-04	+	3.22E-05	+	5.93E-06	-	3.59E-06	+	1.44E-01	≈	4.54E-05	+	3.06E-01	≈	1.58E-06	+	1.26E-01	≈	1.88E-06	+	3.46E-06	+	9.09E-01	≈	8.35E-06	-	4.70E-02	+	1.73E-06	+	1.73E-06	+	1.57E-01	≈	1.73E-06	+
GK-8	1.88E-02	+	4.19E-06	+	1.70E-01	≈	1.70E-06	+	2.57E-06	+	5.45E-05	+	1.72E-06	+	1.66E-06	+	1.14E-01	≈	5.02E-01	≈	1.71E-06	+	1.04E-01	≈	5.78E-01	≈	1.94E-03	+	1.73E-06	+	1.73E-06	+	6.16E-03	-	1.73E-06	+
GK-9	2.76E-06	+	3.44E-06	+	1.13E-05	+	1.66E-06	+	3.27E-02	+	6.27E-06	+	2.73E-03	+	1.64E-06	+	7.91E-01	≈	1.27E-04	+	1.72E-06	+	7.70E-05	-	1.17E-04	-	2.16E-03	+	1.73E-06	+	1.73E-06	+	1.01E-02	-	1.73E-06	+
GK-10	1.54E-01	≈	3.84E-06	+	5.08E-06	+	1.71E-06	+	6.00E-04	+	3.25E-03	+	1.82E-04	+	1.69E-06	+	3.99E-03	-	9.59E-01	≈	1.73E-06	+	6.65E-04	-	3.47E-05	-	4.83E-03	+	1.73E-06	+	1.73E-06	+	2.54E-04	-	1.73E-06	+
GK-11	1.51E-02	+	2.97E-06	+	1.72E-06	+	1.71E-06	+	5.25E-05	+	1.65E-02	+	1.75E-05	+	1.67E-06	+	1.59E-02	-	5.07E-03	-	1.73E-06	+	3.28E-04	-	1.79E-03	-	1.42E-05	+	1.73E-06	+	1.73E-06	+	9.06E-05	-	1.73E-06	+
+	9		11		5		11		10		11		10		11		4		7		8		6		4		11		11		9		2		10	
-	0		0		1		0		0		0		0		0		2		1		0		3		4		0		0		0		4		1	
≈	2		0		5		0		1		0		1		0		5		3		3		2		3		0		0		2		5		0	

5. Conclusion

In this paper, an enhanced version of the SCA named ISCA has been developed based on improving the exploitation, diversity skills and convergence rate. The ISCA attempts to reduce the shortcomings of the classical SCA including premature convergence and local optima stagnation due to insufficient diversity and improper balance between the exploration and exploration features of the search. In the ISCA, the evolutionary state of each candidate solution is utilized to perform the search procedure. Based on these evolutionary states, the exploitation and exploration search features are added through the algorithm's search procedure. The exploitation feature is enhanced by modifying the SCA search mechanism through exploration around the best-obtained candidate solution so far in the search procedure. To enrich the algorithm with sufficient diversity skills, the DE search procedure has been executed. The balance between the exploration and exploitation is managed by introducing the switch parameter, which decides that a particular candidate solution will follow the modified SCA search procedure or DE search procedure. To verify the impact of embedded search strategies in the ISCA, first, it has been validated and compared with other metaheuristics on a set of 23 continuous benchmark optimization problems. The comparison is analyzed based on different performance measures such as Mean and Std of fitness values, ranking procedure, statistical analysis, and convergence analysis. All these measures manifest the search efficacy of the ISCA.

As the second objective of this paper, the proposed ISCA has been extended to its binary version denoted by the BISCA based on transformation function and repair operators. The transformation function transforms the continuous variables evolved by the ISCA to a binary variable and the repair operator maintains the feasibility of the produced solution. The proposed BISCA has been used to solve two sets of MKPs including 38 small-dimensional and 11 large-dimensional instances, respectively. Performance comparison of the BISCA based on Mean and Std of profits, success rate, ranking procedure, statistical analysis, and convergence behavior has demonstrated the outperform search-ability of the BISCA over other algorithms.

In the future, we will investigate the proposed ISCA for solving other NP-hard and combinatorial optimization problems using compatible repair operators. We will develop the multi-objective variant of the ISCA to deal with multi-objective real-world applications. We will also focus on further improving the search performance of the ISCA and BISCA to make them more computationally efficient optimizers.

Acknowledgment

This research is supported by A*STAR under its RIE2020 Advanced Manufacturing and Engineering (AME) Industry Alignment Fund - PrePositioning (IAF-PP) (Award A19D6a0053). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of A*STAR.

References

[1] X.-S. Yang, Nature-inspired metaheuristic algorithms, Luniver press, 2010.

[2] A. H. Gandomi, X.-S. Yang, S. Talatahari, A. H. Alavi, Metaheuristic algorithms in modeling and optimization, Metaheuristic applications in structures and infrastructures (2013) 1–24.

[3] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-international conference on neural networks, volume 4, IEEE, pp. 1942–1948.

[4] X.-S. Yang, Firefly algorithm, stochastic test functions and design optimisation, International journal of bio-inspired computation 2 (2010) 78–84.

[5] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE computational intelligence magazine 1 (2006) 28–39.

[6] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (1997) 341–359.

[7] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE transactions on evolutionary computation 1 (1997) 67–82.

[8] N. Bacanin, I. Brajevic, M. Tuba, Firefly algorithm applied to integer programming problems, Recent Advances in Mathematics 888 (2013) 999.

[9] A. Fetanat, E. Khorasaninejad, Size optimization for hybrid photovoltaic–wind energy system using ant colony optimization for continuous domains based integer programming, Applied Soft Computing 31 (2015) 196–209.

[10] K. Deep, K. P. Singh, M. L. Kansal, C. Mohan, A real coded genetic algorithm for solving integer and mixed integer optimization problems, Applied Mathematics and Computation 212 (2009) 505–518.

[11] S. Gupta, K. Deep, An efficient grey wolf optimizer with opposition-based learning and chaotic local search for integer and mixed-integer optimization problems., Arabian Journal for Science & Engineering (Springer Science & Business Media BV) 44 (2019).

[12] S. Mirjalili, A. Lewis, S-shaped versus v-shaped transfer functions for binary particle swarm optimization, Swarm and Evolutionary Computation 9 (2013) 1–14.

[13] Q.-K. Pan, M. F. Tasgetiren, P. N. Suganthan, T. J. Chua, A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem, Information sciences 181 (2011) 2455–2468.

[14] A.-l. Chen, G.-k. Yang, Z.-m. Wu, Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem, Journal of Zhejiang University-Science A 7 (2006) 607–614.

[15] Q.-K. Pan, M. F. Tasgetiren, Y.-C. Liang, A discrete differential evolution algorithm for the permutation flowshop scheduling problem, Computers & Industrial Engineering 55 (2008) 795–816.

[16] G. K. Jati, et al., Evolutionary discrete firefly algorithm for travelling salesman problem, in: International conference on adaptive and intelligent systems, Springer, pp. 393–403.

[17] S. Mirjalili, Sca: a sine cosine algorithm for solving optimization problems, Knowledge-based systems 96 (2016) 120–133.

[18] S. Li, H. Fang, X. Liu, Parameter optimization of support vector regression based on sine cosine algorithm, Expert systems with Applications 91 (2018) 63–77.

[19] A.-F. Attia, R. A. El Sehiemy, H. M. Hasanien, Optimal power flow solution in power systems using a novel sine-cosine algorithm, International Journal of Electrical Power & Energy Systems 99 (2018) 331–343.

[20] M. Belazzoug, M. Touahria, F. Nouioua, M. Brahimi, An improved sine cosine algorithm to select features for text categorization, Journal of King Saud University-Computer and Information Sciences 32 (2020) 454–464.

[21] M. E. Abd Elaziz, A. A. Ewees, D. Oliva, P. Duan, S. Xiong, A hybrid method of sine cosine algorithm and differential evolution for feature selection, in: International conference on neural information processing, Springer, pp. 145–155.

[22] S. Gupta, K. Deep, Improved sine cosine algorithm with crossover scheme for global optimization, Knowledge-Based Systems 165 (2019) 374–406.

[23] S. Bureerat, N. Pholdee, Adaptive sine cosine algorithm integrated with differential evolution for structural damage detection, in: International Conference on Computational Science and Its Applications, Springer, pp. 71–86.

[24] S. Das, A. Bhattacharya, A. K. Chakraborty, Solution of short-term hydrothermal scheduling using sine cosine algorithm, Soft Computing 22 (2018) 6409–6427.

[25] A. Daoui, H. Karmouni, M. Sayyouri, H. Qjidaa, M. Maaroufi, B. Alami, New robust method for image copyright protection using histogram features and sine cosine algorithm, Expert Systems with Applications 177 (2021) 114978.

[26] M. Abd Elaziz, D. Oliva, S. Xiong, An improved opposition-based sine cosine algorithm for global optimization, Expert Systems with Applications 90 (2017) 484–500.

[27] D. R. Nayak, R. Dash, B. Majhi, S. Wang, Combining extreme learning machine with modified sine cosine algorithm for detection of pathological brain, Computers & Electrical Engineering 68 (2018) 366–380.

[28] C. Qu, Z. Zeng, J. Dai, Z. Yi, W. He, A modified sine-cosine algorithm based on neighborhood search and greedy levy mutation, Computational intelligence and neuroscience 2018 (2018).

[29] S. Gupta, K. Deep, A hybrid self-adaptive sine cosine algorithm with opposition based learning, Expert Systems with Applications 119 (2019) 210–230.

[30] Y. Li, Y. Zhao, J. Liu, Dimension by dimension dynamic sine cosine algorithm for global optimization problems, Applied Soft Computing 98 (2021) 106933.

[31] N. Singh, S. Singh, A novel hybrid gwo-sca approach for optimization problems, Engineering Science and Technology, an International Journal 20 (2017) 1586–1601.

[32] M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedan, A. Alzohairy, Asca-pso: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment, Expert Systems with Applications 99 (2018) 56–70.

[33] H. Nenavath, R. K. Jatoth, S. Das, A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking, Swarm and evolutionary computation 43 (2018) 1–30.

[34] L. Abualigah, A. J. Dulaimi, A novel feature selection method for data mining tasks using hybrid sine cosine algorithm and genetic algorithm, Cluster Computing (2021) 1–16.

[35] L. Abualigah, A. Diabat, Advances in sine cosine algorithm: a comprehensive survey, Artificial Intelligence Review (2021) 1–42.

[36] S. M. Mirjalili, S. Z. Mirjalili, S. Saremi, S. Mirjalili, Sine cosine algorithm: theory, literature review, and application in designing bend photonic crystal waveguides, in: Nature-Inspired Optimizers, Springer, 2020, pp. 201–217.

[37] H. Nenavath, R. K. Jatoth, Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking, Applied Soft Computing 62 (2018) 1019–1043.

[38] E. V. Altay, B. Alatas, Differential evolution and sine cosine algorithm based novel hybrid multi-objective approaches for numerical association rule mining, Information Sciences 554 (2021) 198–221.

[39] H. Kellerer, U. Pferschy, D. Pisinger, Multidimensional knapsack problems, in: Knapsack problems, Springer, 2004, pp. 235–283.

[40] A. Fréville, The multidimensional 0–1 knapsack problem: An overview, European Journal of Operational Research 155 (2004) 1–21.

[41] S. Balev, N. Yanev, A. Fréville, R. Andonov, A dynamic programming based reduction procedure for the multidimensional 0–1 knapsack problem, European Journal of Operational Research 186 (2008) 63–76.

[42] V. C. Li, Y.-C. Liang, H.-F. Chang, Solving the multidimensional knapsack problems with generalized upper bound constraints by the adaptive memory projection method, Computers & operations research 39 (2012) 2111–2121.

[43] J. Puchinger, G. R. Raidl, U. Pferschy, The multidimensional knapsack problem: Structure and algorithms, INFORMS Journal on Computing 22 (2010) 250–265.

[44] Y. Vimont, S. Boussier, M. Vasquez, Reduced costs propagation in an efficient implicit enumeration for the 01 multidimensional knapsack problem, Journal of Combinatorial Optimization 15 (2008) 165–178.

[45] D. Du, K.-I. Ko, X. Hu, et al., Design and analysis of approximation algorithms, volume 62, Springer, 2012.

[46] R. Motwani, P. Raghavan, Randomized algorithms, Cambridge university press, 1995.

[47] B. Haddar, M. Khemakhem, S. Hanafi, C. Wilbaut, A hybrid quantum particle swarm optimization for the multidimensional knapsack problem, Engineering Applications of Artificial Intelligence 55 (2016) 1–13.

[48] M. Vasquez, Y. Vimont, Improved results on the 0–1 multidimensional knapsack problem, European Journal of Operational Research 165 (2005) 70–81.

[49] L. Wang, X.-l. Zheng, S.-y. Wang, A novel binary fruit fly optimization algorithm for solving the multidimensional knapsack problem, Knowledge-Based Systems 48 (2013) 17–23.

[50] K. Luo, Q. Zhao, A binary grey wolf optimizer for the multidimensional knapsack problem, Applied Soft Computing 83 (2019) 105645.

[51] B. Zhang, Q.-K. Pan, X.-L. Zhang, P.-Y. Duan, An effective hybrid harmony search-based algorithm for solving multidimensional knapsack problems, Applied Soft Computing 29 (2015) 288–297.

[52] J. Lampinen, I. Zelinka, Mechanical engineering design optimization by differential evolution, in: New ideas in optimization, 1999, pp. 127–146.

[53] W. F. Sacco, A. A. de Moura Meneses, N. Henderson, Some studies on differential evolution variants for application to nuclear reactor core design, Progress in Nuclear Energy 63 (2013) 49–56.

[54] P. Georgilakis, Differential evolution solution to transformer no-load loss reduction problem, IET generation, transmission & distribution 3 (2009) 960–969.

[55] L. Arya, A. Koshti, S. Choube, Distributed generation planning using differential evolution accounting voltage stability consideration, International journal of electrical power & energy systems 42 (2012) 196–207.

[56] E. Cuevas, D. Zaldivar, M. Pérez-Cisneros, A novel multi-threshold segmentation approach based on differential evolution optimization, Expert Systems with Applications 37 (2010) 5265–5271.

[57] S. L. Sabat, K. S. Kumar, P. Rangababu, Differential evolution algorithm for motion estimation, in: International Workshop on Multi-disciplinary Trends in Artificial Intelligence, Springer, pp. 309–316.

[58] M. Pant, H. Zaheer, L. Garcia-Hernandez, A. Abraham, et al., Differential evolution: A review of more than two decades of research, Engineering Applications of Artificial Intelligence 90 (2020) 103479.

[59] A. W. Mohamed, H. Z. Sabry, M. Khorshid, An alternative differential evolution algorithm for global optimization, Journal of advanced research 3 (2012) 149–165.

[60] S. Das, A. Abraham, U. K. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE transactions on evolutionary computation 13 (2009) 526–553.

[61] S. Das, S. S. Mullick, P. N. Suganthan, Recent advances in differential evolution–an updated survey, Swarm and evolutionary computation 27 (2016) 1–30.

[62] L. Cui, G. Li, Z. Zhu, Q. Lin, K.-C. Wong, J. Chen, N. Lu, J. Lu, Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism, Information Sciences 422 (2018) 122–143.

[63] P. C. Chu, J. E. Beasley, A genetic algorithm for the multidimensional knapsack problem, Journal of heuristics 4 (1998) 63–86.

[64] M. A. K. Azad, A. M. A. Rocha, E. M. Fernandes, Improved binary artificial fish swarm algorithm for the 0–1 multidimensional knapsack problems, Swarm and evolutionary computation 14 (2014) 66–75.

[65] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary computation 3 (1999) 82–102.

[66] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future generation computer systems 97 (2019) 849–872.

[67] W. Zhou, P. Wang, A. A. Heidari, M. Wang, X. Zhao, H. Chen, Multi-core sine cosine optimization: Methods and inclusive analysis, Expert Systems with Applications 164 (2021) 113974.

[68] M. Wang, A. A. Heidari, M. Chen, H. Chen, X. Zhao, X. Cai, Exploratory differential ant lion-based optimization, Expert Systems with Applications 159 (2020) 113548.

[69] A. A. Heidari, I. Aljarah, H. Faris, H. Chen, J. Luo, S. Mirjalili, An enhanced associative learning-based exploratory whale optimizer for global optimization, Neural Computing and Applications 32 (2020) 5185–5211.

[70] M. A. Arasomwan, A. O. Adewumi, On the performance of linear decreasing inertia weight particle swarm optimization for global optimization, The Scientific World Journal 2013 (2013).

[71] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Advances in Engineering Software 114 (2017) 163–191.

[72] M. Mirrashid, H. Naderpour, Transit search: An optimization algorithm based on exoplanet exploration, Results in Control and Optimization 7 (2022) 100127.

[73] W. Long, T. Wu, X. Liang, S. Xu, Solving high-dimensional global optimization problems using an improved sine cosine algorithm, Expert systems with applications 123 (2019) 108–126.

[74] Y. Ji, J. Tu, H. Zhou, W. Gui, G. Liang, H. Chen, M. Wang, An adaptive chaotic sine cosine algorithm for constrained and unconstrained optimization, Complexity 2020 (2020).

[75] W. Deng, S. Shang, X. Cai, H. Zhao, Y. Song, J. Xu, An improved differential evolution algorithm and its application in optimization problem, Soft Computing 25 (2021) 5277–5298.

[76] C. Xu, H. Huang, S. Ye, A differential evolution with replacement strategy for real-parameter numerical optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, pp. 1617–1624.

[77] J. Carrasco, S. García, M. Rueda, S. Das, F. Herrera, Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review, Swarm and Evolutionary Computation 54 (2020) 100665.