Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms☆

Hamdi Tolga Kahraman *, Sefa Aras, Eyüp Gedikli

Department of Software Engineering, Faculty of Technology, Karadeniz Technical University, Trabzon, Turkey

ARTICLE INFO

Article history:
Received 16 May 2019
Received in revised form 24 October 2019
Accepted 29 October 2019
Available online xxxx

Keywords:
FDB selection method
FDB-based symbiotic organism search
Meta-heuristic search algorithms
Benchmark problems
Optimization

ABSTRACT

Selection methods have an important role in the meta-heuristic search (MHS) process. However, apart from a few successful methods developed in the past, new and effective studies have not been found in recent years. It is known that solution candidates selecting from the population during the search process directly affects the direction and success of the search. In this study, a new selection method based on fitness-distance balance (FDB) is developed in order to solve the premature convergence problem in the MHS process. Thanks to the developed method, solution candidates with the highest potential to improve the search process can be determined effectively and consistently from the population. Experimental studies have been conducted to test and verify the developed FDB selection method. For this purpose, 90 benchmark functions with different types and complexity levels have been used. In order to test the developed FDB method, numerous variants have been formed. These variants have been compared to each other to determine the most effective FDB variant. In the validation study, the FDB-SOS (FDB-based symbiotic organism search) algorithm is compared with thirteen well-known and up-to-date MHS techniques. The search performance of the algorithms has been analyzed by the Wilcoxon Rank Sum Test. The results show that the developed selection method makes a significant contribution to the meta-heuristic search process.

© 2019 Elsevier B.V. All rights reserved.

1. Introduction

The meta-heuristic search (MHS) process, a population-based search method, consists generally of two main steps. In the first step, the solution candidates (reference positions in the search space) are selected from the population depending on the selection method adopted. In the second step, the search process is executed using the selected reference positions. The second step is performed by the search operators of the MHS algorithms. The task of the search operators in the MHS process is to meet the exploitation and exploration requirements. A search close to the reference positions (solution candidates from the selection process) is called a neighborhood search (exploitation). The aim is to discover the best position (solution) near the chosen solution candidates. Diversity is the task of rescuing the population from local solution traps in cases where the improvement in the search process has stopped completely or successful solutions cannot be obtained. An operation, such as a mutation, is applied to provide an effective change on the reference positions in order to fulfill the diversity task [1–4]. The process of diversity should enable exploring locations that are more successful than reference positions in the search space.

Selection methods are classified into three categories as non-deterministic, deterministic and probabilistic [1,5–7]. In the non-deterministic method, solution candidates are selected randomly from the population [5,7]. This selection method is usually used to contribute to the diversity of the search process. In the majority of deterministic methods, fitness values of solution candidates are taken into consideration. This selection is also referred to as the greedy or elitist method. The best solution candidates are selected and the direction of the search is determined by these candidates. In this respect, the greedy or elite selection method serves to search and converge in the neighborhood of successful positions in the search space [8,9]. Probabilistic selection methods have the characteristics of elitist and random methods [10]. The most commonly used examples of the probabilistic selection method are the roulette wheel and tournament methods [4,11–14]. These two methods calculate the probability of being selected depending on the suitability values of the candidates. The selection process in the roulette wheel takes place in one-step depending on the possibilities. The tournament method consists of two steps. The first step of the tournament method is the random selection of the candidates who will participate in the

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105169.

* Corresponding author.
E-mail address: htolgakahraman@ktu.edu.tr (H.T. Kahraman).

https://doi.org/10.1016/j.knosys.2019.105169
0950-7051/© 2019 Elsevier B.V. All rights reserved.

tournament, and the second step is where the tournament takes place between the selected candidates. The candidate with the highest fitness value out of the two candidates wins the tournament. In the MHS process, what is expected of the selection method is that it can determine the solution candidates that can make the most contribution to the search process in a stable and conscious manner [4,15]. The selection method should effectively determine the direction of the search process and make it easier for search operators [6,16]. This brings up various issues: Do the current selection methods mentioned so far have this function? For example, in a method based on randomness, can a stable or conscious choice occur? How successful is it to use a random selection method to select a solution candidate that can contribute the most to the search process from a population of 50 members? On the other hand, is it possible to claim that the candidate, who can offer the most contributions to the search process, is the solution candidate with the best fitness value in the population?

In answering these questions, the solution candidates that are randomly distributed to the search space are considered at the beginning of the search process. In the later stages of the search process, the position of the solution candidates in the search space will change depending on the interactions in the population and the operating characteristics of the search operators. In the algorithms with a high convergence rate, the solution candidates are located near the best candidate within a short time. In particular, algorithms with insufficient diversity often face a premature convergence problem [6,9,17,18]. It is known that in the algorithms that have a premature convergence problem, the solution candidates are oriented towards the position of the best solution candidate in the population [16,19–21]. In this case, the fitness values and positions of the solution candidates become very close to each other in the later stages of the MHS process. In other words, the solution candidates become similar in a short period of time. In this case, the search process ends with a local optima trap. Although the fitness values are high, this brings up other issues; such as, can the candidates who are very similar to each other make important contributions to the search process? Can these candidates prevent the population from being trapped in a local optima? Can they contribute to diversity? The answers to these issues seem to be clear. Solution candidates who are very similar to each other can only contribute to exploitation (a sensitive neighborhood search) in the search process. These candidates cannot prevent the population from being trapped in a local optima. Moreover, they can even lead to these traps [18,22,23]. In such a case, the only mechanism that could resist the premature convergence problem is the diversity operator.

Local optima traps, especially in large-scale and complex search spaces, are the most difficult optimization problems to solve for even the most successful MHS algorithms and diversity operators [9,24–26]. Local solution traps can only be overcome by the combined use of effective selection methods and powerful search operators that support each other. However, when such a problem is encountered, it is not possible to achieve success in the search process by using elitist selection methods based solely on the fitness value of the solution candidates [4,7,11]. In order to avoid the problem of premature convergence, the solution candidates who can make the greatest contribution to the search process from the population should be determined to be the most stable and effective selection method.

Finally, if the probabilistic selection method is considered, it can be said that it consists of a combination of two other methods. This method becomes more stable than the non-deterministic method by taking into account the fitness values of the candidates, and by using randomness, it provides diversity more effectively than the elitist method [7]. Nevertheless, it cannot be said that by using probabilistic methods, the most appropriate solution candidates for the success of the search process can be determined consciously and decisively. This is because, in order to determine the candidates that can make the strongest contribution to the search process, there is a need for a robust method that takes into account the current situation and needs of the population at every step of the search process.

Hundreds of new MHS algorithms have been developed over the last 20 years in order to perform exploitation and exploration tasks more effectively and thus, to achieve more successful search performance [27–29]. The majority of these studies focus on the design of search operators or the development of variants of well-known MHS algorithms [8,17,30–33]. However, the success of the search process is not only dependent on search operators of the MHS algorithms [1,5]. From the literature, it is clearly observed that researchers have paid less attention to the topic on selection methods even though it has a direct effect on the performance of MHS algorithms [10,28,34]. The main condition for the success of the MHS process is that the selection methods and search operators work in ways that support each other [11,19]. In order to perform their tasks, search operators take solution candidates/reference positions as input parameters. In other words, the success of the search process depends on the capabilities of the search operators as well as the position of the solution candidates determined by the selection method in the search space [16]. Therefore, this clearly demonstrates the important role of selection methods in the process of MHS.

In this study, a new selection method based on the Fitness-Distance Balance (FDB) of the solution candidates is developed. The FDB method developed to improve the search performance of MHS algorithms consists of easy-to-implement steps. In the FDB method, the score of each candidate for the selection process is calculated and with this score the fitness values and the distance values of candidates are used. While the value of fitness indicates the success of the solution candidate, the distance value indicates the difference from the best solution candidate. A long distance between the two solution candidates is an indication that these two candidates do not resemble each other, and have the potential to complete each other's shortcomings. It is strong evidence that a candidate with a high fitness value can make a significant contribution to the search process if the distance value is also high. In other words, candidates with a high score have the potential to eliminate the handicap of the best solution in the current population. Although the fitness value is high, a solution candidate that is very close to the best candidate in the population has no potential to improve the search process. This candidate cannot be a solution for the premature convergence problem described above, and cannot contribute to population diversity, but may partly contribute to the neighborhood search. As the distance value of such a candidate will be close to zero, the score value will also be very low. Thanks to the score, it is possible to identify and eliminate the solution candidates which cannot contribute to the search process or even cause premature convergence. The calculation of the score using the FDB selection method is a powerful solution to avoid local optima traps.

The FDB selection method provides two critical information about the status of population members. These include identifying members (candidates) which are very similar to the best solution in the population, and identifying members with a high fitness value, even though they do not resemble the best solution. Taking into account the first information, members who are very similar to each other are prevented from being elected together during the selection process. In other words, candidates with positions very close to each other in the search space cannot be selected at the same time. Taking the second information into consideration, it is ensured that the members who can overcome the deficiencies of the best solution in the population are selected.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

In summary, these two important pieces of information used in the score calculation provide a conscious, stable and effective selection method. The developed FDB selection method is applied to Symbiotic Organism Search (SOS) [35], a powerful and modern MHS technique. A comprehensive test is conducted to explore and strengthen the exploration and exploitation capabilities of the FDB-based SOS algorithm (FDB-SOS). As a result of test studies, a more powerful FDB-SOS algorithm has been developed in comparison with the base SOS algorithm. Verification studies are conducted to compare the FDB-SOS algorithm with the most recent and effective MHS techniques developed in the last decade. Test and verification studies show that the FDB selection method significantly increases the search performance of the SOS algorithm. The developed FDB-SOS algorithm has demonstrated a superior search performance compared to the original SOS algorithm and other powerful and state-of-the-art algorithms.

This article consists of four chapters and various subdivisions belonging to each chapter. Accordingly, the second section of the article titled "2. Materials and Methods" includes three subsections these subsections describe the preliminaries of MHS algorithms, the proposed FDB method, and the FDB-SOS algorithm, respectively. Firstly, 2.1. The sub-section titled "Preliminaries" describes "MHS Process" and "Selection Methods". Examples of related studies in the literature are discussed in the selection methods. Secondly, in the subheading "2.2. Proposed Method: FDB Selection" the proposed selection method is defined and explained in detail. This topic provides the implementation steps and pseudo-code of the FDB method. An example case study is also described showing how the FDB method is applied. Thirdly, "2.3. FDB-SOS" describes the implementation of the FDB method in the SOS algorithm. For this purpose, the SOS algorithm describes the working steps, and then the pseudo-code is given to explain how the FDB method is applied to the selection process in the SOS algorithm (Algorithm 3). Thus, the second chapter of the article 2 describes the general steps of the MHS process in the proposed FDB selection method and the FDB-SOS algorithm.

In the third part of the article, experimental studies on the FDB method are developed. Experimental studies have been gathered under three subheadings. These are "Experimental Settings", "Test Studies" and "Validation Studies". Information on benchmark problems used in experimental studies, introduction of competing MHS algorithms and the number of experiment iterations are explained under sub-heading "3.1. Experimental Settings". Under the sub-heading "3.2. Test Studies", the effects of the FDB selection method on the search process lifecycle of the SOS algorithm are investigated. For this purpose, various FDB-SOS combinations were created and case studies were designed. Each case study (alternative FDB-SOS methods) were attempted in benchmark problems and each of them has been investigated for their search performances. The most effective FDB-SOS method has been determined by statistical analysis of experimental data from the test studies. Under the subheading "3.3. Validation Studies", the proposed FDB-SOS algorithm compares the search performances with competing MHS algorithms in the literature. Data obtained from verification studies are analyzed statistically, approximation curves and computational complexity of algorithms are given.

Briefly, the FDB method has been applied to the SOS algorithm with different strategies and the most effective FDB-SOS case is determined in the test studies. In verification studies, the most effective FDB-SOS algorithm was performed with competing algorithms. The information obtained from experimental studies clearly reveals the success of the FDB selection method and the FDB-SOS algorithm. Finally, in the fourth part, the FDB selection method is indicated for future studies.

2. Materials and methods

This section focuses on three subjects. Firstly, the preliminaries of MHS process is described where the steps of the MHS process are handled independently of the algorithm. While describing the search process lifecycle steps, the selection methods and search operators are discussed in detail. Secondly, the definition and steps of the FDB selection method are described. Finally, the application of the FDB-based method applied to the SOS algorithm, which is a powerful and modern MHS technique, is discussed.

2.1. Preliminaries

In this subsection, general form of continuous valued optimization problems, general steps of search process of MHS algorithms and selection methods are discussed.

2.1.1. MHS process

The parameters that define any continuous value and unconstrained optimization problems include the following: The parameter m is the problem dimension, the design variables are represented by the $X \equiv [x_1, x_2, \ldots, x_m], m \in N^+, \forall_{i=1}^{m} x_i \in X$, and the bounds of the design variables are $[a, b]$ for $a, b \in R$ and $-\infty < (a, b) < +\infty$. The objective function G can be represented as given in Eq. (1). The P-population, which is created based on these definitions and consists of n-solution candidates, is given in Eq. (2). The P-population (set of solution candidates) consists of m-members. Each member is an n-dimensional vector that is created between the lower (a) and upper (b) bounds of design variables and is random. The vector F, which represents the fitness values of the solution candidates in this population, is created as given in Eq. (3) using the values obtained from the objective function (G) of the problem.

$$G = f(x_1, x_2, \ldots, x_m) \tag{1}$$

$$P \equiv \begin{bmatrix} x_{11} & \ldots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \ldots & x_{nm} \end{bmatrix}_{n \times m} \tag{2}$$

$$F \equiv \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}_{n \times 1} \tag{3}$$

The process of modeling optimization problems using MHS algorithms is given in Algorithm 1. The first stage of this process is the creation of the P-population consisting of n-members. Once the population is generated, an objective function value (G_i) is calculated for each solution candidate. Eq. (4) is used for this.

$$G_i = F_G(P_i), n \in N^+, \forall_{i=1}^{n} -\infty < G_i < +\infty \tag{4}$$

In the next step, the fitness values of each candidate in P are calculated. Normalized G_i values ($normG_i$) are used for this operation. Fitness values of solution candidates are calculated as given in Eq. (5).

$$n \in N^+, \forall_{i=1}^{n} F_i \equiv \begin{cases} \text{if goal is minimization} \rightarrow F_i = & 1 - normG_i \\ \text{if goal is maximization} \rightarrow F_i = & normG_i \end{cases} \tag{5}$$

In the second stage of the modeling of optimization problems, the search process lifecycle is initiated (Step 6 of Algorithm 1). At this stage, the behavior of algorithms in nature is imitated. The search characteristics and performance of MHS algorithms appear at this stage. In this process, algorithms try to explore the optimum values of the design parameters of the problem.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Algorithm 1: General steps of the search process in MHS algorithms

```
1  begin
2      P: create a population of solution candidates randomly
3      for i = 1 : n (number of solution candidates) do
4          F: evaluate fitness value of the candidates
5      end
6      while (search process lifecycle) do
7          Selection method: select the solution candidates
           from the members in the P-population to direct the
           search process, P_k ∈ P,   (k < n, (k, n) ∈ N^+)
8          Search operators:
9              Exploitation (neighborhood search around P_k)
10             Exploration (diversity in the search space using
               P_k)
11     end
12 end
```

The search process lifecycle of MHS algorithms can be summarized in two steps [10,36] (see Algorithm 1). The first of these steps is the selection process. In the selection process, P_k candidates are selected from the P-population, where k is the number of candidates to be selected ($k < n, (k, n) \in N^+$). For example, in GA [37], k is equal to the number of parents. The number of parents is usually two. In PSO [38] and GSA [39], $k = 1$. In the SOS (X_{best}, X_i and X_j organisms) [35] and in GWO [40] (α, β, δ) wolves, $k = 3$. Similarly, in the ABC algorithm [41], the number of employed bees (k) varies between 2 and 6. When the search process lifecycle given in Algorithm 1 is examined, it is understood that the outputs (P_k members) obtained from the selection method are the inputs of the search operators. Therefore, the selection process directly affects the success of both intensification and diversification [5].

According to Algorithm 1, search operations are performed after selection. These operations are performed by the search operators of the MHS algorithms. Search operators should basically meet two requirements, which are, exploitation and exploration tasks. During the exploitation process, a search is made in the local neighborhood of a reference position (obtained from the selection process) in the search space. The aim is to find the best position in the local search around the reference location. There is a need for diversity when the neighborhood search leads to premature convergence and the optimum solution cannot be reached. The purpose of the diversity process is to help the algorithm jump out of local optima traps when the search process stops improving. MHS techniques can perform these two tasks in one or more steps with a different number of search operators [19,25,26,42,43]. For example, in PSO [38], GSA [39], CGSA [44], MFO [45], SSA [11] and MS [46], neighborhood search (exploitation) and diversity (exploration) tasks are performed by the same operator. Thanks to random and dynamic parameters, both tasks are achieved by the same operator [31]. That is, the search operator has a dynamic mathematical expression.

Exploitation or exploration is provided depending on the values of the dynamic parameters. For example, in PSO, GSA and CGSA algorithms, if the velocity and acceleration of the particle are high, a diversity effect occurs, otherwise, the exploitation effect is created. The adaptive values of parameters α and A allow GWO [40] to smoothly transition between exploration and exploitation. In the squirrel search algorithm [29], the balance between exploration and exploitation is achieved with the help of a gliding constant in the mathematical model. In MFO [45], a logarithmic spiral is chosen as the main update mechanism of moths. The t parameter in the spiral equation defines how much the next position of the moth should be close to the flame (is the closest position to the flame, while showing the farthest). The coefficient c_1 is the most important parameter in SSA [11] because it balances exploration and exploitation. Similarly, in the MS [46], λ is a scale factor which can control the convergence speed of the algorithm and improve the diversity of the population. In the CSA algorithm [47], the flight length of the crow is a randomly determined dynamic parameter. When this parameter takes values between 0 and 1, an exploitation effect is generated, and a diversity effect is generated when it is higher than 1. In the BOA algorithm [48], diversification and intensification processes are performed in two steps: global search phase and local search phase. It is decided by probability values which steps to apply in the search process. In GA, the function of neighborhood search is provided by the crossover operator and the diversity task by the mutation operator. The crossover and mutation processes are two steps that operate independently of one another but are a continuation of each other. In the ABC algorithm [20,41], exploiting is performed by an onlooker bee who tries to find a food source by means of the information given by the employed bee and a diversification task is carried out by a scout bee who searches the environment randomly. These tasks are performed in two different steps. In the SOS algorithm, search operations are carried out in three different phases by using mutualism, commensalism, and parasitism operators.

In the next subsection, the current selection methods are discussed.

2.1.2. Selection methods

In this subsection, we first discuss the selection methods in the literature in order to better explain the FDB method. In MHS algorithms, the selection process is applied before search operators (please see step 7 in Algorithm 1). Different methods are used in the literature for selection processes [4]. The first of these methods is that the solution candidates are randomly selected from the population. In the three search phases of the SOS algorithm (mutualism, commensalism, and parasitism), the X_j organism (solution candidate) is randomly selected from the ecosystem [35]. Similarly, BSA [49] randomly chooses the direction individual (solution candidate) from individuals of a randomly chosen previous generation. In the GWO [40] the wolves alpha, beta, and delta estimate the position of the prey, and the other wolves update their positions randomly around the prey. In BOA [48], a switching probability is used which decides whether the butterfly will move toward the best butterfly or perform a random walk. In SMONM [50] if it does not achieve any improvement in the Local Leaders' position in a predefined number of iterations, then each member is reinitialized randomly to explore the search space. In the squirrel search algorithm [29], in order to improve the exploration ability of the proposed method, the concept of random relocation of some flying squirrels are used.

The second method is that the candidates are chosen based on their fitness values by using greedy selection. X_{best} in the SOS algorithm, elite bees in the base ABC algorithm, the best position of the particles in the algorithms in PSO and hybrid PSO methods (MPSO [30], RMPSO [3,7,12]), the best nest position in the CS algorithm [51], the best leader position in the LSA algorithm [52], the best mass/agent in the GSA [39] and CGSA [44] algorithms, and the fittest solution in GWO (α) [40] are only determined based on fitness values. Besides, the second and third wolves, which are named beta (β) and delta (δ) in GWO respectively, are selected by using the greedy method. In SSA [11], the position of the best salp to be chased by the salp chain, in BOA [48] the fittest butterfly (solution candidate), in the predator prey model the best plants [53], and the best optimal search agent in the emperor penguin algorithm [54] are determined only based on fitness values.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

The third method is to select the solution candidates from the population, in an ordinal manner. Ordinal-based selection only takes into account the relative order of individuals according to their indexes in the population. In the MSA [46], LSA [52], CS [51], SOS [35], BOA [48] and in almost all of the other MHS algorithms, solution candidates are ordinally included in the search process.

The fourth, being the probabilistic selection method. The probabilistic method is widely used in MHS algorithms, especially in IAFOA [33], variants of ABC [15,20] and variants of DE [5,16]. In the selection process, the probability of selection of candidates based on their fitness values is calculated. In this way, the possibility of selection of the candidates who have high fitness values is also high [10]. Moreover, candidates with low fitness values due to randomness also have the possibility (although the probability is low). Roulette wheel and tournament strategies are examples of the probabilistic selection method.

The fifth method is the combined selection method where at least two of the other selection methods are applied together. In MHS algorithms where the combined selection method is applied, solution candidates can be determined by random, greedy, probabilistic or adaptive methods [29,33,50,53]. In cases where the combined selection strategy is used, the other one is usually the greedy method. The greedy method is suitable for neighborhood searching and convergence tasks. However, using only the greedy method in the selection process may cause the algorithms to easily become trapped in the local optima. This especially occurs in algorithms with low diversity, in the early stages of the search process lifecycle, the population is under the dominant influence of solution candidates whose fitness value is higher than other members [28,39,51,52]. In this case, MHS algorithms that have a strong ability of exploitation, results in providing worse diversity and premature convergence when solving complex multimodal problems. The selection methods currently used are insufficient to solve the premature convergence problem. In the exploration process, it is necessary to change the direction of the search to support diversity, in order to prevent all the solution candidates in the population to be overly similar to each other in the exploitation process. As a result, the greedy selection method is a powerful solution for neighborhood search tasks and a new selection method is needed for diversity tasks.

In the next subsection, the proposed FDB method in this study is explained in detail.

2.2. Proposed method: The FDB (Fitness-Distance Balance) selection

The purpose of developing the FDB selection method is to discover the candidate or candidates who will make the most contribution to the search process within a population in a stable and effective way. Consequently, it will be ensured that the variation process, which enables exploring different regions of the search space, and the intensification process, which ensures the exploitation of previous knowledge about the fitness landscape for MHS algorithms, will be carried out in a balanced manner.

In the first step of the FDB method, the distance of the solution candidates from the best solution (P_{best}) is calculated. Some of the distance measurement metrics include Euclidean (EU), Manhattan (MA) and Minkowski (MI), which can be used in distance calculations. Using the objective function given in Eq. (1), the P-population given in Eq. (2), and the EU metric, the distance value of each member from P_{best} is calculated as given in Eq. (6). The distance vector D_P created for solution candidates is given in Eq. (7).

$$\forall_{i=1}^{n}, P_i \neq P_{best}, D_{P_i} = \sqrt{(x_{1P_i} - x_{1P_{best}})^2 + (x_{2P_i} - x_{2P_{best}})^2 + \cdots + (x_{mP_i} - x_{mP_{best}})^2} \quad (6)$$

$$D_P \equiv \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix}_{n\times 1} \quad (7)$$

In the second step of the FDB method, the score is calculated for the solution candidates. Normalized fitness values ($normF$) and normalized distance values ($normD_P$) of the candidates are used in the score calculation. The reason for using normalized numerical quantities is to prevent these two parameters to dominate each other in the score calculation. In calculating the scores of the members, a weight coefficient (w) is used, which also determines the impact of fitness and distance values. According to these explanations and $0 < w < 1$, the score of each candidate in the P-population is calculated. In this study, w is taken as 0.5. Eqs. ((8) or (9)) can be used for the score calculation in the FDB selection method.

$$\forall_{i=1}^{n} P_i, S_{FDB^1 P_i} = w * normF_{P_i} + (1 - w) * normD_{P_i} \quad (8)$$

$$\forall_{i=1}^{n} P_i, S_{FDB^2 P_i} = normF_{P_i} * normD_{P_i} \quad (9)$$

The score vector (S_p) of the population is given in Eq. (10).

$$S_P \equiv \begin{bmatrix} s_1 \\ \vdots \\ s_n \end{bmatrix}_{n\times 1} \quad (10)$$

The steps of the FDB selection method are given in Algorithm 2.

Algorithm 2: Steps of FDB selection method

```
begin
    Select a distance metric
    Generate a random number in the range [0.4, 0.6] to the
    w-weight coefficient
    for i = 1 : n do
        calculate distance between the P_i and P_best using Eq.
        (6)
        create or update the D_P distance vector as given in
        Eq. (7)
    end
    for i = 1 : n do
        Normalize distance and fitness vectors within the
        range [0, 1]
        Calculate the score for each solution candidate as
        given in Eqs. (8 or 9)
        Create the FDB-based score vector as given in Eq.
        (10)
    end
    Select the solution candidates to be included in the
    search process based on their scores (S_P)
end
```

A simple case study example has been prepared to explain the impact of the proposed FDB selection method with concrete data. A continuous and unconstrained optimization problem, G_q is given in Eq. (11). Let the optimum values of design variables (x_1, x_2, x_3) be searched in order to minimize given objective function (G_q).

$$G_q = (x_1 - 5)^2 + (x_2 + 4)^2 + (x_3 - 10)^2 \quad (11)$$

Suppose that a population (P_q) consisting of 7-solution candidates is created to optimize the objective function (G_q). Each of the solution candidates in the population ($p_1, p_2, p_3, \ldots, p_7$) is represented by a 3-dimensional vector. Each row of the P_q-vector given in Eq. (12) represents a solution candidate, while columns represent the position information of the candidate. The objective function value of each candidate is calculated using

Eq. (11) depending on the information given in the P_q. Eq. (5) ($F_{(i)} = 1 - normG_i$) is used to calculate the fitness values of the solution candidates in the P_q population. The obtained fitness values information is represented by the F_q vector (see Eq. (12)).

$$P_q \equiv \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \end{bmatrix}_{7\times 1} \equiv \begin{bmatrix} 5 & -4 & 4 \\ 5 & -4 & 3 \\ 3 & 0 & 2 \\ 10 & 4 & 10 \\ -4 & 4 & 0 \\ 1 & -4 & -2 \\ -10 & 3 & -5 \end{bmatrix}_{7\times 3} \quad F_q \equiv \begin{bmatrix} 1 \\ 0,97 \\ 0,90 \\ 0,89 \\ 0,55 \\ 0,73 \\ 0,00 \end{bmatrix}_{7\times 1} \tag{12}$$

When the greedy selection method is used in the MHS process, the candidates with the highest fitness values in the population are selected. The positions of the selected candidates are used by the search operators as a reference in the search process. According to the information given in Eq. (12), if the greedy selection method is used, p_1 and p_2 solution candidates are determined as parents in the GA algorithm, and as elite bees in the ABC algorithm. In the GWO algorithm alpha, beta and the delta gray wolves will be p_1, p_2 ve p_3 respectively. It should be noted that the optimum values of the design variables for the optimization problem given in Eq. (11) are $\langle x_{1-opt}, x_{2-opt}, x_{3-opt}\rangle$ respectively $\langle 5, -4, 10\rangle$. Accordingly, it is understood that p_1 and p_2 candidates discover the optimum values (x_{1-opt}, x_{2-opt}) for x_1 and x_2 design variables. The value of the x_3 design variable for both candidates (p_1 and p_2) is far from the x_{3-opt} value. According to the design variables, p_1 and p_2 candidates represent two positions very close to each other in the search space (see Fig. 1). In this respect, it is unlikely that a search in the neighborhood of these two solution candidates will be able to move the x_3-design variable to x_{3-opt}. For example, when these two solution candidates are selected as parents in the genetic algorithm, there is no possibility to improve the x_3-design variable of the child individuals that the crossover operator will produce. This is because the value of the x_3-design variable of both parents represents a bad position for the optimum solution (far from the x_{3-opt} value). With the mutation process that will take place after the crossover operator in GA, there is a possibility that the x_3-design variable may deteriorate as much as the probability of improvement. It is also unclear which design variable (x_1, x_2, x_3) to apply the mutation process.

As the problem dimension (the number of design variables) increases, uncertainty will also increase. In addition, the information that will improve the x_3-design variable, which is the weakness of both solution candidates (p_1 and p_2), is in the p_4 candidate. Therefore, what is expected of a successful selection method is that it can decisively determine the solution candidates who can improve each other's weaknesses. In this case, p_1 can be chosen by using the greedy selection method to effectively perform the exploitation. At the same time, p_4 should be chosen to recover the process from local optima traps and to assure diversity because it is clear that p_4 is the candidate who can make the most contribution to p_1 (P_{best}) in the population conditions given in Eq. (12).

The aim of developing the FDB selection method is to determine the candidate(s) who will make the greatest contribution to P_{best} and the search process in an effective and determined way. The scores of the solution candidates should be calculated to apply the FDB selection method. For score calculations, normalized fitness ($normF_{P_i}$) and distance ($normD_{P_i}$) values of the solution candidates are needed. The $normF_{P_i}$ vector is given by F_q at Eq. (12).$normD_{P_i}$ vector is obtained by using Eq. (6). Eq. (8) is used in the score calculation and the steps given in Algorithm 2 are followed. The value of the parameter w is taken as 0.5 in the calculations. The fitness values of the solution candidates given in Eq. (12) according to their distance from P_{best} (p_1) are given in Fig. 1(a). The scores of the candidates ($S_{p_1}, S_{p_2}, \ldots, S_{p_7}$) according to their distance from P_{best} are given in Fig. 1(b). The distance of the solution candidates to P_{best}, which is the best candidate in the P_q-population, is given on the horizontal axis of Fig. 1.

When the solution candidates are ranked according to their fitness values, the first candidate selected by the greedy method is p_1 (see Fig. 1-a). When the candidates are ranked according to their scores calculated by FDB method, p_4 having the highest score (0.74) is selected (see Fig. 1-b). For this case study example, p_4 can also be selected by using random or probabilistic methods. However, there is little probability that the mentioned case will occur for multidimensional and complex problems. Moreover, the case study given in Eq. (11) is too simple to be encountered in real applications and engineering problems. In some optimization problems such as Convolutional Neural Networks, the number of design variables is more than one thousand [55]. In addition, the objective function of the problem is often not composed of such simple mathematical expressions [47]. The number of candidates in the population in real-world applications range between approximately 20–200 [56–58]. Therefore, it is a difficult task to get rid of the local optimum traps in complex optimization problems by using the solution candidates determined by greedy or random selection methods [59]. Furthermore, the greatest advantage of the FDB selection method compared to other selection methods is that it can determine the candidate (p_4 for this case study) who can make the most contribution to the solution process in a determined and conscious way, not randomly or probabilistically. This feature of FDB describes the ability to effectively maintain diversity and prevent premature convergence even in the later stages of the search process lifecycle.

2.3. FDB-SOS: Symbiotic organism search with fitness-distance balance

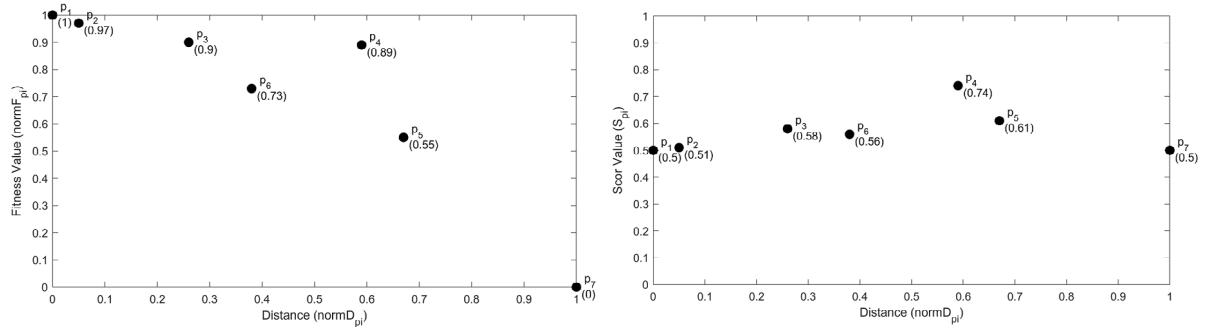
The SOS algorithm was developed in 2014 by Cheng and Prayogo [35]. The theoretical basis of the algorithm is the behavior of organisms in nature. The extraordinary performance of SOS has made it one of the most successful among the more than two hundred MHS algorithms in the literature. In the SOS algorithm, three organisms, P_{best}, P_i and P_j, are selected from the ecosystem (population) in the first step of the search process lifecycle. In the selection process, both P_i and P_{best} organisms are determined by the deterministic method. P_{best} is the solution candidate with the highest fitness value in the ecosystem and selected by the greedy method. P_i represents the solution candidates selected from the ecosystem in order of their index by the ordinal selection method. P_j is randomly selected from the ecosystem.

The second step of the search process lifecycle of the SOS algorithm consists of three phases: mutualism, commensalism and parasitism. In these three stages, the same P_{best} and P_i organisms are used. In other words, the P_{best} and P_i organisms in the mutualism phase and those used in commensalism, and parasitism stages are the same. P_j is determined randomly in three stages.

In the FDB-SOS algorithm, P_j, unlike the SOS algorithm, is determined by FDB-based selection method in the stages of mutualism and commensalism (see Algorithm 2). According to these explanations, the search process lifecycle of the FDB-SOS algorithm is given in Algorithm 3.

On the ninth line of Algorithm 3, the mutualism phase begins. In this phase, *Mutual_Vector* is created using Eq. (13), and the positions of P_inew and P_jnew organisms are updated using the Eqs. (14 and 15). The P_j candidate is included in these three equations. As in the mutualism phase, the P_j candidate is involved in the search process along with the other two candidates, P_i

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.



(a) Fitness and distance values of solution candidates

(b) Scores and distance values of solution candidates

Fig. 1. Presentation of $normF_{P_i}$, $normD_{P_i}$ Sp_i and information of solution candidates given in Eq. (12).

Algorithm 3: Steps of the FDB-SOS

```
begin
    P: create an ecosystem of randomly generated organisms
    for i = 1 : n (number of solution candidates) do
        F: evaluate the fitness of the organisms (solution candidates)
    end
    while (search process lifecycle) do
        for i = 1 : n (number of solution candidates) do
            Selection process: select the P_best and P_i (same as in SOS) select P_j (use the FDB selection method given in Algorithm 2)
            Mutualism phase:
                Mutual_Vector = (P_i + P_j)/2                                   (13)
                P_inew = P_i + rand(0, 1) * (P_best − Mutual_Vector * BF_1)      (14)
                P_jnew = P_j + rand(0, 1) * (P_best − Mutual_Vector * BF_2)      (15)
            Selection process: select P_j (use the FDB selection method given in Algorithm 2)
            Commensalism phase:
                P_inew = P_i + rand(−1, 1) * (P_best − P_j)                      (16)
            Selection process: select P_j (same as in SOS)
            Parasitism phase:
                Create a Parasite_Vector from organism P_i
                if Parasite_Vector fitter than organism P_j then P_j = Parasite_Vector end
        end
    end
end
```

and P_{best}, during the commensalism phase. In the phase of the commensalism, Eq. (16) is used for searching. Considering that the P_j candidate is selected by using the FDB method, the effect of the method on mutualism and commensalism is revealed. In this respect, it can be said that the FDB selection method has an important role on the success of the SOS algorithm.

In the next section, a comprehensive comparison is made between the FDB-SOS and SOS methods. As a result of these comparisons, the contribution of the FDB selection method to the symbiotic search process is proven by experimental study results.

3. Experimental study and discussion

A comprehensive experimental study has been conducted to test and verify the effect of the proposed FDB selection method. Experimental studies are presented under three sub-sections. Information about test problems, algorithms and parameter settings used in experimental studies are given in the following subsections: The first being "3.1. Experimental Settings", and in the second subsection "3.2. Test Studies", the results obtained from the experimental studies in which the FDB method has been tested are presented. For this purpose, variants of the SOS algorithm are obtained using various combinations of the FDB selection method. In order to determine the most effective FDB-SOS method, comparison studies between developed variants and the original SOS algorithm are performed. In the third subsection, "3.3. Validation Studies", the performance of the FDB-SOS method has been compared with the latest and most effective MHS algorithms in the literature. As a result of the comparisons, the success of the FDB-SOS method developed in this study is confirmed by experimental study results.

Experimental studies have been also conducted to examine the effect of the FDB selection method on algorithm complexity. For this purpose, the complexity of the algorithm is compared between the SOS variants in which the FDB method has been applied, and the base SOS algorithm. In addition, the time complexity of all competing algorithms used in the experimental study have been calculated. In the calculations, the problem dimensions are set to 30, 50 and 100. Furthermore, to determine whether the results obtained by FDB-based variants of SOS are significantly different from the results generated by the base SOS algorithm and the other MHS algorithms, the nonparametric Wilcoxon rank-sum tests [60] have been executed. In Wilcoxon tests, the search performance of the algorithms on 90 test functions and three different problem dimensions are compared. To examine the exploitation and exploration abilities of the FDB-based method, convergence curves are shown for unimodal, multimodal, hybrid and composition functions.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 1
Classical test problems.

Name	Function	Type	Range	Min
Ackley	$f_1(x) = 20 - 20exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{D} x_i^2}{D}}\right) - exp\left(\frac{\sum_{i=1}^{D} \cos(2\pi x_i)}{D}\right) + e$	M	$[-100, 100]$	0
Alpine	$f_2(x) = \sum_{i=1}^{D} \mid x_i \sin(x_i) + 0.1x_i \mid$	M	$[-100, 100]$	0
Cigar	$f_3(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$	U	$[-100, 100]$	0
DixonPrice	$f_4(x) = (x_1 - 1)^2 + \sum_{i=2}^{D} i(2x_i^2 - x_{i-1})^2$	U	$[-10, 10]$	0
Elliptic	$f_5(x) = \sum_{i=1}^{D} \left((10^6)^{\frac{i-1}{D-1}} x_i^2\right)$	U	$[-100, 100]$	0
Exponential	$f_6(x) = exp\left(0.5 \sum_{i=1}^{D} x_i^2\right)$	M	$[-10, 10]$	0
Griewank	$f_7(x) = 1 + \frac{\sum_{i=1}^{D} x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right)$	M	$[-600, 600]$	0
I.C.M.[a]	$f_8(x) = \frac{D}{10} - \left(\frac{\sum_{i=1}^{D} \cos(5\pi x_i)}{10} - \sum_{i=1}^{D} x_i^2\right)$	M	$[-100, 100]$	0
Levy	$f_9(x) = \sin^2(3\pi x_1) + \mid x_D - 1 \mid * (1 + \sin^2(3\pi x_D)) + \sum_{i=1}^{D-1} [(x_i - 1)^2 (1 + \sin^2(3\pi x_{i+1}))]$	M	$[-10, 10]$	0
Michalewicz	$f_{10}(x) = D - \sum_{i=1}^{D} \sin(x_i) \sin^{20}\left(\frac{ix_i^2}{\pi}\right)$	M	$[0, \pi]$	0
Penalized-1	$f_{11}(x) = \frac{\pi}{D}\left[10 \sin^2(\pi y_1) + \sum_{i=1}^{D-1} [(y_i - 1)^2 (1 + 10 \sin^2(\pi y_{i+1}))] + (y_D - 1)^2\right] + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $y_i(x_i) = 1 + \frac{x_i + 1}{4}$ $u_i(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > 0 \\ 0 & a \geq x_i \geq -a \\ k(-x_i - a)^m & -a > x_i \end{cases}$	M	$[-50, 50]$	0
Penalized-2	$f_{12}(x) = 0.1\left[\sin^2(3\pi x_1) + \sum_{i=1}^{D} [(x_i - 1)^2 (1 + \sin^2(3\pi x_i + 1))] + (x_D - 1)^2 (1 + sin^2(2\pi x_D))\right] + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$ $u_i(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > 0 \\ 0 & a \geq x_i \geq -a \\ k(-x_i - a)^m & -a > x_i \end{cases}$	M	$[-50, 50]$	0
Powell	$f_{13}(x) = \sum_{i=1}^{D/4} [(x_{4i-3} - 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^2]$	U	$[-4, 5]$	0
Rastrigin	$f_{14}(x) = 10D + \sum_{i=1}^{D} x_i^2 - 10 \cos(2\pi x_i)$	M	$[-100, 100]$	0
Rosenbrock	$f_{15}(x) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	M	$[-10, 10]$	0
R.H.E. [b]	$f_{16}(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	U	$[-100, 100]$	0
Salomon	$f_{17}(x) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2}$	M	$[-100, 100]$	0
Schaffer	$f_{18}(x) = 0.5 + \frac{\sin^2(\sum_{i=1}^{D} x_i^2) - 0.5}{(1 + 0.001 \sum_{i=1}^{D} x_i^2)^2}$	M	$[-100, 100]$	0
Schwefel	$f_{19}(x) = 418.982887272434 * D - \sum_{i=1}^{D} x_i \sin(\sqrt{\mid x_i \mid})$	U	$[-500, 500]$	0
Schwefel 1.20	$f_{20}(x) = \sum_{i=1}^{D} \mid x_i \mid$	U	$[-100, 100]$	0
Schwefel 2.21	$f_{21}(x) = \max_{i=1,\ldots,D} \mid x_i \mid$	U	$[-100, 100]$	0
Schwefel 2.22	$f_{22}(x) = \sum_{i=1}^{D} \mid x_i \mid + \prod_{i=1}^{D} \mid x_i \mid$	U	$[-10, 10]$	0
Sphere	$f_{23}(x) = \sum_{i=1}^{D} x_i^2$	U	$[-100, 100]$	0
Step	$f_{24}(x) = \sum_{i=1}^{D} (x_i + 0.5)^2$	U	$[-100, 100]$	0
Styblinski–Tang	$f_{25}(x) = 39.1661657037714 * D + 0.5 \sum_{i=1}^{D} x_i^4 - 16x_i^2 + 5x_i$	M	$[-5, 5]$	0
SumPower	$f_{26}(x) = \sum_{i=1}^{D} \mid x_i \mid^{i+1}$	M	$[-10, 10]$	0
SumSquares	$f_{27}(x) = \sum_{i=1}^{D} ix_i^2$	U	$[-10, 10]$	0
Quartic	$f_{28}(x) = \sum_{i=1}^{D} ix_i^4$	U	$[-10, 10]$	0
Weierstrass	$f_{29}(x) = \sum_{i=1}^{D} \left[\sum_{j=0}^{k} 0.5^j \cos\left(2\pi 3^j (x_i + 0.5)\right)\right] - D \sum_{j=0}^{k} 0.5^j \cos(\pi 3^j)$ $k = \begin{cases} 20 & D \geq 20 \\ D & D < 20 \end{cases}$	M	$[-1, 1]$	0
Zakharov	$f_{30}(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^4$	M	$[-5, 10]$	0

M: Multimodal U: Unimodal.
[a] Inverted cosine mixture.
[b] Rotated hyper ellipsoid.

Table 2
Parameter settings for all algorithms.

Algorithm	Parameters
SOS	–
GSA	Initial gravitational constant = 100 Decreasing coefficient = 20
BSA	$Mixrate = 1.00$ $F = 3 * randn$
HSS	$N_{SC} = N/10$ $r^1_{min} = 0.0$ $r^1_{max} = 1.1$ $Pr^1_{angle} = 0.1$ $r^2_{min} = 0.0$ $r^2_{max} = 1.1$ $Pr^2_{angle} = 0.05$
GWO	A linearly decreases from 2 to 0
MFO	A linearly decreases from −1 to -2
LSA	Max channel time = 10
SCA	r_1 linearly decreases from 2 to 0
CSA	Flight length = 2 awareness probability = 0.1
CGSA	Initial gravitational constant = 100 Decreasing coefficient = 20 Chaos index = 2
SSA	–
MS	Number of kept moths at each generation = 2 $\beta = 1.5$ $(S_{max}) = 1.0$ Acceleration factor = $5^{1/2} - 1$
BOA	Power exponent increases from 0.1 to 0.3 Modular modality = 0.01 $p = 0.8$

Table 3
Information on the FDB-SOS variants prepared for testing.

Abbreviation of FDB-SOS variations	Method used in the selection of P_j solution candidate		
	Mutualism	Commensalism	Parasitism
DSOS-1	$P_j = S_{FDB^2 P_i}$	P_j (Random)	P_j (Random)
DSOS-2	$P_j = S_{FDB^1 P_i}$	P_j (Random)	P_j (Random)
DSOS-3	$P_j = S_{FDB^1 P_i}$	P_j (Random)	$P_j = S_{FDB^2 P_i}$
DSOS-4	$P_j = S_{FDB^1 P_i}$	$P_j = S_{FDB^1 P_i}$	P_j (Random)
DSOS-5	$P_j = S_{FDB^1 P_i}$	$P_j = S_{FDB^1 P_i}$	$P_j = S_{FDB^2 P_i}$
DSOS-6	$P_j = S_{FDB^1 P_i}$	$P_j = S_{FDB^1 P_i}$	$P_j = S_{FDB^1 P_i}$

3.1. Experimental settings

The set of benchmark functions used in test studies: A set of classical benchmark problems was prepared to be used in the test studies in which the FDB-SOS method was developed. In this set consisting of the most frequently used problems in the literature, there are 30 unconstrained optimization problems. Seventeen of these benchmark problems are multi-modal types and the others are unimodal. Note that all the test functions are shown in Table 1, including their names, expressions, types, search space (ranges), and global optimum values, respectively.

Benchmark test suites used in validation studies: In order to compare the FDB-SOS method with the powerful and up-to-date MHS techniques in the literature, and to verify the search performance, 90 benchmark problems were used. These are 30 classic benchmark problems (see Table 1), CEC 2014 (30) [61] and CEC 2017 (30) [62] test suites, respectively. In the CEC 2014 test suite, there are 3 unimodal ($f_1 - f_3$), 13 simple multimodal ($f_4 - f_{16}$), 6 hybrid ($f_{17} - f_{22}$) and 8 composition ($f_{23} - f_{30}$) type functions. In the CEC 2017 test suite, there are 3 unimodal ($f_1 - f_3$), 7 simple multimodal ($f_4 - f_{10}$), 10 hybrid ($f_{11} - f_{20}$) and 10 composition ($f_{20} - f_{30}$) type functions. All test functions are minimization problems defined in Eq. (17) [61]:

$$Min f(z),\ z = [z_1, z_2, \ldots, z_D]^T \tag{17}$$

Hybrid functions are used to effectively test both convergence and diversity capabilities of algorithms. A hybrid function is created by taking into account the real-world optimization problems and it is a set of multiple functions. In a set of hybrid functions, variables are randomly divided into some sub-components where different basic functions are used for different sub-components. The hybrid functions are defined in Eq.(18) [61]:

$$F(x) = g_1(M_1 z_1) + g_2(M_2 z_2) + \cdots + g_N(M_N z_N) + F^*(x) \tag{18}$$

$F(x)$ is the hybrid function and $g_a(x)$ is the *ath* basic function used to construct the hybrid function and N is the number of basic functions.

Composition functions are used to test the balance between the exploitation and exploration ability of the algorithms. The composition functions have a massive number of local optima, and are defined in Eq. (19) [61]:

$$F(x) = \sum_{a=1}^{N} \omega_a * [\lambda_a g_a(x) + bias_a] + F^* \tag{19}$$

$F(x)$: composition function, $g_a(x)$: *ath* basic function used to construct the composition function, N: number of basic functions, $bias_a$: defines which optimum is global optimum, λ_a: used to control each $g_a(x)$'s height, ω_a: the normalized weight value for each $g_a(x)$.

Dimensions of test functions and search range: All of the test functions used in experimental studies are scalable, as given in Eq. (17). In order to test how the performance of FDB changes with scaling the search space, we test all functions in 30, 50 and 100 dimensions. In this way, the effect of the FDB selection method on low, middle and high-dimensional functions can be examined in detail. The search range for classical benchmark problems is given in the "range" column of Table 1. The search range for the CEC 2014 [61] and CEC 2017 [62] test suites is taken as $[-100, 100]^D$, as indicated in the description documents of these problems.

Competing algorithms and parameter settings: In experimental studies, 13 well-known MHS techniques were used to compare the search performance of competing algorithms. The abbreviations of these algorithms and the years they were developed are GSA (2009) [39], BSA (2013) [49], SOS (2014) [35], HSS (2014) [63], GWO (2014) [40], MFO (2015) [45], LSA (2015) [52], SCA (2016) [57], CSA (2016) [47], CGSA (2017) [44], SSA (2017) [11], MS (2018) [46], and BOA (2018) [48] respectively. In experimental studies, the parameter settings in the original papers of the competing algorithms were followed. Accordingly, the parameter settings of the algorithms are given in Table 2. All compared algorithms are implemented in MATLAB®R2018a and run on an Intel(R) Core(TM) i7-4770K CPU @3.50 GHz with 16 GB of RAM and a x64-based processor.

Termination criteria and statistical analysis: To make a fair comparison, the parameters of all competing algorithms are set to be the same as in their original papers. Furthermore, the

Table 4
Comparison of mean error and standard deviation in objective function value for 30 dimension of classical test problems.

F_n	SOS	DSOS-1	DSOS-2	DSOS-3	DSOS-4	DSOS-5	DSOS-6
f_{01}	1.97E+01(2.81E+00)	1.96E+00(6.01E+00) +	2.00E+01(9.12E−02) +	2.01E+01(9.70E−02) =	2.01E+01(1.44E−01) =	2.01E+01(1.45E−01) −	2.01E+01(1.67E−01) −
f_{02}	9.67E−19(6.01E−19)	2.78E−38(9.30E−38) +	6.47E−22(8.93E−22) +	6.46E−17(4.53E−16) +	3.63E−21(1.62E−20) +	5.40E−21(2.19E−20) +	2.28E−21(5.63E−21) +
f_{03}	9.45E−31(1.31E−30)	7.64E−63(3.98E−62) +	5.76E−33(5.84E−33) +	4.51E−33(4.66E−33) +	1.78E−33(2.24E−33) +	1.39E−33(2.54E−33) +	1.87E−33(2.66E−33) +
f_{04}	6.67E−01(1.35E−12)	6.67E−01(5.63E−11) −	6.67E−01(5.97E−15) +	6.67E−01(8.09E−15) +	6.67E−01(2.42E−14) +	6.67E−01(2.53E−14) +	6.67E−01(1.28E−08) +
f_{05}	4.43E−33(5.33E−33)	3.20E−65(1.54E−64) +	8.81E−35(9.73E−35) +	7.34E−35(9.75E−35) +	2.09E−35(2.10E−35) +	1.62E−35(2.11E−35) +	2.20E−35(3.00E−35) +
f_{06}	7.18E−66(4.08E−78)	7.18E−66(3.19E−81) =	7.18E−66(3.19E−81) =	7.18E−66(3.19E−81) =	7.18E−66(3.19E−81) =	7.18E−66(3.19E−81) =	7.18E−66(3.19E−81) =
f_{07}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	1.32E−03(4.45E−03) −	9.93E−04(4.03E−03) =	3.32E−04(1.66E−03) =	1.25E−03(3.94E−03) −	2.01E−04(1.43E−03) =
f_{08}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{09}	7.89E−02(1.29E−01)	1.94E+00(4.06E+00) −	4.67E−02(1.09E−01) +	4.08E−02(1.16E−01) +	2.35E−02(7.34E−02) +	4.94E−02(1.54E−01) +	4.08E−02(8.46E−02) +
f_{10}	1.29E+01(1.02E+00)	1.06E+01(1.03E+00) +	1.77E+01(8.63E−01) −	1.36E+01(8.57E−01) −	1.77E+01(8.02E−01) −	1.45E+01(1.19E+00) −	1.31E+01(1.43E+00) =
f_{11}	4.14E−11(6.26E−11)	6.35E−08(1.24E−07) −	2.15E−13(7.49E−13) +	8.96E−14(2.57E−13) +	5.71E−14(1.73E−13) +	1.11E−13(4.40E−13) +	2.00E−13(4.15E−13) +
f_{12}	8.75E−02(8.57E−02)	3.30E−01(1.67E−01) −	4.06E−02(5.83E−02) +	4.49E−02(5.98E−02) +	3.14E−02(4.13E−02) +	5.25E−02(8.04E−02) +	4.42E−02(6.31E−02) +
f_{13}	1.29E−16(8.77E−16)	4.30E−70(3.06E−69) +	1.08E−04(5.16E−04) +	6.21E−07(4.09E−06) +	2.85E−05(1.21E−04) +	3.44E−09(2.45E−08) +	1.36E−05(6.77E−05) +
f_{14}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	7.04E+00(2.41E+01) −	7.65E+00(2.36E+01) −	5.85E+00(2.44E+01) −	7.22E+00(2.67E+01) −	3.81E+00(1.81E+01) −
f_{15}	2.35E+01(3.99E−01)	2.42E+01(4.68E−01) −	2.27E+01(3.98E−01) +	2.26E+01(3.79E−01) +	2.28E+01(5.29E−01) +	2.28E+01(4.21E−01) +	2.29E+01(5.01E−01) +
f_{16}	1.10E−09(1.99E−09)	2.30E−56(1.44E−55) +	1.15E−07(2.49E−07) −	7.81E−08(1.53E−07) −	4.51E−08(6.23E−08) −	1.05E−07(3.80E−07) −	6.62E−08(1.25E−07) −
f_{17}	9.99E−02(3.76E−08)	9.60E−02(1.96E−02) =	9.99E−02(6.59E−08) =	9.99E−02(6.34E−08) =	9.99E−02(9.79E−08) =	9.99E−02(6.68E−08) =	9.99E−02(5.19E−08) =
f_{18}	3.13E−03(1.76E−07)	4.29E−04(1.09E−03) +	3.13E−03(1.97E−07) =	3.13E−03(2.66E−07) =	3.13E−03(1.65E−07) =	3.13E−03(1.98E−07) =	3.13E−03(1.73E−07) =
f_{19}	4.88E+03(4.71E+02)	4.39E+03(4.72E+02) +	7.06E+03(4.63E+02) −	5.34E+03(3.67E+02) −	6.96E+03(5.00E+02) −	5.69E+03(4.37E+02) −	5.49E+03(6.37E+02) −
f_{20}	5.84E−13(1.08E−12)	1.84E−57(9.86E−57) +	4.15E−12(9.53E−12) −	1.10E−11(3.39E−11) −	1.77E−12(3.63E−12) −	1.46E−12(2.48E−12) −	2.16E−12(5.46E−12) −
f_{21}	2.79E−15(1.85E−15)	7.28E−32(2.76E−31) +	2.90E−16(1.96E−16) +	2.15E−16(1.46E−16) +	1.20E−16(7.59E−17) +	9.90E−17(6.08E−17) +	1.24E−16(7.87E−17) +
f_{22}	1.48E−19(9.09E−20)	7.57E−38(3.69E−37) +	3.51E−23(2.02E−23) +	4.18E−23(3.24E−23) +	3.69E−23(4.06E−23) +	3.38E−23(3.36E−23) +	3.12E−23(2.65E−23) +
f_{23}	1.17E−37(1.12E−37)	4.18E−68(2.93E−67) +	1.50E−39(1.41E−39) +	1.10E−39(1.00E−39) +	3.04E−40(2.38E−40) +	2.50E−40(2.18E−40) +	3.34E−40(4.31E−40) +
f_{24}	8.45E−10(9.87E−10)	1.63E−06(2.13E−06) −	1.52E−13(1.83E−13) +	4.38E−13(1.22E−12) +	1.81E−13(2.48E−13) +	1.66E−13(2.81E−13) +	1.92E−13(2.19E−13) +
f_{25}	3.26E+01(2.54E+01)	6.45E+01(3.15E+01) −	5.75E+01(6.39E+01) −	6.26E+01(6.78E+01) =	4.77E+01(3.26E+01) −	9.01E+01(9.40E+01) −	2.59E+01(2.28E+01) =
f_{26}	5.15E−91(1.70E−90)	9.52E−107(6.33E−106) +	3.02E−91(8.63E−91) =	7.56E−91(3.74E−90) +	2.86E−93(6.97E−93) +	5.27E−93(2.13E−92) +	1.54E−92(7.25E−92) +
f_{27}	1.21E−38(9.19E−39)	5.27E−69(3.62E−68) +	2.13E−40(2.06E−40) +	1.53E−40(1.45E−40) +	5.60E−41(4.98E−41) +	3.75E−41(3.40E−41) +	4.65E−41(4.28E−41) +
f_{28}	3.86E−73(1.26E−72)	1.82E−134(1.10E−133) +	5.27E−74(1.25E−73) =	1.84E−74(4.14E−74) +	3.01E−75(5.24E−75) +	5.01E−75(3.11E−74) +	1.02E−75(1.47E−75) +
f_{29}	8.99E−03(4.43E−02)	3.21E−04(1.64E−03) =	2.78E−03(1.99E−02) +	1.73E−02(7.27E−02) =	8.72E−02(2.55E−01) =	2.71E−02(1.17E−01) =	4.02E−02(1.33E−01) =
f_{30}	7.34E−05(9.28E−05)	2.41E−35(1.51E−34) +	1.31E−03(2.78E−03) −	1.06E−03(1.78E−03) −	7.75E−04(1.09E−03) −	6.91E−04(7.63E−04) −	6.31E−04(8.20E−04) −
+/= /−		17/6/7	16/6/8	16/8/6	16/7/7	16/5/9	16/8/6

maximum number of function evaluations (*maxFEs*) is used as the termination condition, which was set to $1000 * D$. Convergence to the global optimum point to 10^{-8} is also accepted to solve problems [62]. However, reaching the tolerance value is not used as a termination criterion. The search process termination criterion is only *maxFEs*. The mean and the standard deviation of the best-of-run errors for 51 independent runs of all of the contestant algorithms on the benchmark problems for $D = 30$, $D = 50$ and $D = 100$ are given in Tables 4–6 and Tables 9–17. The error is the absolute value of the difference between the real optimum value of the objective function (f_{opt}) and the best result f (P_{best}). In order to test the statistical validity of the results obtained from test and validation studies, non-parametric pairwise Wilcoxon test has been applied. The test has been conducted for 5% level of significance. The results are presented in Tables and "+", " = ", and "−" denote that the performance of the corresponding algorithm is better than, similar and worse than to that of FDB-SOS, respectively. We also show the total number of the aforementioned cases at the end of the Tables 4–6 and Tables 9–17 of each dimension, for each of the competitor algorithms as (+/= /−).

3.2. Test studies: Comparison of SOS and FDB-SOS

In this subsection, firstly the variants of the FDB-SOS algorithm are introduced. The steps given in Algorithm 3 have been followed to create FDB-SOS variants, and Table 3 provides information on these variants prepared for test studies. In the first column of Table 3, abbreviations of the FDB-SOS variants are given and in the other three columns, the methods used in the selection of P_j solution candidate are presented. The P_j solution candidate is chosen randomly among the individuals in the population in the base model of the SOS algorithm, while in the FDB-SOS method, it is chosen based on the score values of the solution candidates. The $S_{FDB^1P_i}$ used in the selection of P_j represents the FDB method given in Eq. (8) and the $S_{FDB^2P_i}$ represents the FDB method given in Eq. (9).

In the test studies, comparisons are made between the base model of the SOS algorithm and the FDB-SOS variants given in Table 3. Classical benchmark functions given in Table 1 are used in these comparisons. Multimodal problems are suitable for evaluating the exploration ability of any search algorithm, while unimodal problems are appropriate for evaluating the exploitation of any search algorithm. In Table 1, the functions f_3, f_4, f_5, f_{13}, f_{16}, f_{19}, f_{20}, f_{21}, f_{22}, f_{23}, f_{24}, f_{27}, f_{28} are unimodal and the functions f_1, f_2, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{14}, f_{15}, f_{17}, f_{18}, f_{25}, f_{26}, f_{29}, f_{30} are multimodal. Table 4 shows the results obtained by performing SOS and FDB-SOS variants on test functions for 30-dimensions. Tables 5 and 6 provide the results of the same experiment conducted for 50 and 100 dimensions. In Tables 4–6, the mean and standard deviation of the absolute error value for the test functions are reported.

Evaluation of the results given in Table 4 for 30-dimensional problems: Table 4 reports the experimental results obtained by performing SOS and FDB-SOS variants on test functions for 30-dimensions. For the functions f_6, f_7, f_8, all algorithms find the global optimum. For the functions f_{17}, f_{29} all algorithms except DSOS-2 exhibits similar performance. It discovers a better solution. For the functions $f_1, f_2, f_3, f_5, f_{13}, f_{21}, f_{22}, f_{23}, f_{26}, f_{27}, f_{28}$ all of the FDB-SOS variants shows better performance. For the functions $f_4, f_9, f_{11}, f_{12}, f_{15}, f_{24}$ except for DSOS-1 from FDB-SOS variants, all perform better than SOS algorithm. For the functions f_{10}, f_{14}, f_{16}, f_{18}, f_{19}, f_{20}, f_{30} only DSOS-1 perform better than SOS algorithm. The SOS algorithm for the f_{25} function performs similarly with DSOS-3 and DSOS-6, while it performs better than the other variants of the FDB-SOS algorithm.

According to the results, the FDB selection method achieves a significant improvement over most unimodal problems. This means that the FDB method has the ability to apply knowledge of previous good solutions to find better solutions. For multimodal problems, FDB variants are more successful than the base SOS algorithm. This result expresses the ability of the FDB method to explore various unknown regions in the search space to explore the global optimum. Further consideration of the results show that the DSOS-1 method is more successful than other methods

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 5
Comparison of mean error and standard deviation in objective function value for 50 dimension of classical test problems.

F_n	SOS	DSOS-1	DSOS-2	DSOS-3	DSOS-4	DSOS-5	DSOS-6
f_{01}	2.00E+01(8.65E−02)	3.92E+00(8.02E+00) +	2.00E+01(9.30E−03) +	2.01E+01(1.17E−01) =	2.00E+01(4.15E−02) +	2.00E+01(1.07E−01) =	2.01E+01(1.64E−01) =
f_{02}	3.45E−31(2.71E−31)	1.66E−64(7.27E−64) +	2.83E−37(4.01E−37) +	3.97E−37(8.98E−37) +	1.33E−36(2.71E−36) +	1.03E−36(2.45E−36) +	5.89E−37(7.69E−37) +
f_{03}	4.56E−54(6.08E−54)	5.70E−111(2.95E−110) +	5.61E−58(1.20E−57) +	4.14E−58(5.92E−58) +	1.96E−58(3.50E−58) +	2.48E−58(5.06E−58) +	2.48E−58(4.13E−58) +
f_{04}	6.67E−01(5.88E−13)	6.67E−01(2.85E−11) −	6.67E−01(1.18E−15) +	6.67E−01(1.18E−15) +	6.67E−01(3.92E−15) +	6.67E−01(1.17E−14) +	6.67E−01(3.28E−15) +
f_{05}	2.48E−56(4.88E−56)	1.13E−111(8.03E−111) +	6.63E−60(7.87E−60) +	6.51E−60(1.79E−59) +	2.31E−60(3.55E−60) +	2.24E−60(4.85E−60) +	3.00E−60(4.26E−60) +
f_{06}	2.67E−109(2.01E−120)	2.67E−109(8.66E−114) −	2.67E−109(9.55E−125) +	2.67E−109(9.55E−125) +	2.67E−109(9.55E−125) +	2.67E−109(9.55E−125) +	2.67E−109(9.55E−125) +
f_{07}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	7.71E−05(5.51E−04) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{08}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{09}	4.20E−01(5.06E−01)	3.27E+01(1.21E+01) −	1.88E−01(2.98E−01) +	1.69E−01(2.28E−01) +	2.16E−01(3.66E−01) +	2.82E−01(4.17E−01) +	1.91E−01(3.34E−01) +
f_{10}	2.37E+01(1.43E+00)	2.04E+01(1.57E+00) +	3.20E+01(1.03E+00) −	2.54E+01(1.09E+00) −	3.22E+01(1.31E+00) −	2.72E+01(1.15E+00) −	2.46E+01(2.01E+00) −
f_{11}	3.96E−10(6.77E−10)	7.56E−07(1.92E−06) −	1.22E−03(8.71E−03) +	4.55E−11(3.22E−10) +	1.77E−11(7.08E−11) +	3.91E−12(2.23E−11) +	1.21E−10(8.26E−10) +
f_{12}	6.72E−01(2.44E−01)	1.45E+00(4.93E−01) −	4.60E−01(2.45E−01) +	4.85E−01(2.23E−01) +	4.43E−01(2.19E−01) +	3.99E−01(2.09E−01) +	4.37E−01(1.59E−01) +
f_{13}	1.46E−57(3.14E−57)	3.56E−115(2.54E−114) +	2.11E−08(1.51E−07) +	1.22E−63(3.00E−63) +	4.04E−63(1.01E−62) +	3.49E−63(5.95E−63) +	3.70E−63(7.86E−63) +
f_{14}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	2.18E−02(1.56E−01) =
f_{15}	4.22E+01(6.32E−01)	4.30E+01(7.48E−01) −	4.06E+01(7.53E−01) +	4.06E+01(7.52E−01) +	4.13E+01(8.32E−01) +	4.12E+01(7.78E−01) +	4.10E+01(8.01E−01) +
f_{16}	1.61E−13(3.86E−13)	4.29E−89(3.02E−88) +	3.41E−10(6.72E−10) −	3.31E−10(1.21E−09) −	1.86E−10(4.75E−10) −	2.62E−10(7.28E−10) −	5.17E−10(2.13E−09) −
f_{17}	9.99E−02(1.91E−08)	9.99E−02(8.17E−08) −	9.99E−02(3.39E−08) =	9.99E−02(1.44E−08) =	9.99E−02(3.03E−08) −	9.99E−02(4.95E−09) =	9.99E−02(2.46E−08) =
f_{18}	3.13E−03(1.43E−08)	8.58E−04(1.41E−03) +	3.13E−03(2.45E−08) =	3.13E−03(9.21E−08) =	3.13E−03(5.40E−08) =	3.13E−03(4.23E−08) =	3.13E−03(2.58E−08) =
f_{19}	9.01E+03(7.30E+02)	8.25E+03(9.10E+02) +	1.30E+04(6.93E+02) −	9.86E+03(4.95E+02) −	1.31E+04(7.02E+02) −	1.05E+04(4.66E+02) −	9.57E+03(1.27E+03) −
f_{20}	4.50E−18(1.69E−17)	5.78E−95(4.12E−94) +	1.05E−16(2.60E−16) −	2.11E−16(8.10E−16) −	4.49E−17(1.37E−16) −	4.51E−17(1.54E−16) −	1.40E−16(5.43E−16) −
f_{21}	5.76E−24(5.22E−24)	9.88E−53(7.04E−52) +	6.15E−26(5.90E−26) +	4.74E−26(3.75E−26) +	2.29E−26(1.66E−26) +	2.23E−26(1.45E−26) +	2.10E−26(1.91E−26) +
f_{22}	1.29E−31(8.22E−32)	1.61E−62(1.08E−61) +	3.27E−38(2.67E−38) +	4.44E−38(4.59E−38) +	1.21E−37(1.25E−37) +	1.47E−37(3.35E−37) +	9.53E−38(9.32E−38) +
f_{23}	7.68E−61(1.41E−60)	7.95E−115(5.56E−114) +	9.22E−65(1.34E−64) +	5.15E−65(7.91E−65) +	5.10E−65(1.11E−64) +	1.40E−65(1.75E−65) +	5.36E−65(9.84E−65) +
f_{24}	6.57E−08(2.16E−07)	8.82E−05(4.59E−04) −	4.05E−12(1.03E−11) +	5.82E−12(1.49E−11) +	4.37E−12(1.53E−11) +	3.72E−12(1.12E−11) +	7.80E−12(2.41E−11) +
f_{25}	9.68E+01(3.93E+01)	1.48E+02(3.98E+01) −	1.29E+02(3.88E+01) −	1.21E+02(8.32E+01) =	1.41E+02(4.68E+01) −	1.35E+02(1.42E+02) =	8.70E+01(3.96E+01) =
f_{26}	2.24E−154(8.77E−154)	9.18E−177(0.00E+00) +	8.79E−155(2.41E−154) =	5.38E−154(1.88E−153) =	5.26E−156(3.21E−155) +	5.66E−155(3.85E−154) +	1.82E−156(8.63E−156) +
f_{27}	1.54E−61(2.34E−61)	1.14E−115(7.48E−115) +	1.71E−65(2.39E−65) +	1.38E−65(1.70E−65) +	6.93E−66(1.10E−65) +	4.94E−66(6.58E−66) +	7.62E−66(9.68E−66) +
f_{28}	1.14E−116(2.97E−116)	1.57E−221(0.00E+00) +	1.59E−118(5.07E−118) +	1.09E−118(3.20E−118) +	1.15E−119(1.76E−119) +	1.75E−119(8.06E−119) +	2.16E−119(6.06E−119) +
f_{29}	1.34E−01(4.57E−01)	3.29E−03(1.91E−02) +	1.56E−04(9.21E−04) +	2.22E−01(6.33E−01) =	1.31E−01(3.49E−01) =	1.22E−01(3.79E−01) =	1.55E−01(4.61E−01) =
f_{30}	1.17E−03(1.51E−03)	6.34E−38(4.19E−37) +	2.26E−02(4.51E−02) −	1.42E−02(2.10E−02) −	1.30E−02(1.84E−02) −	9.36E−03(1.17E−02) −	1.56E−02(2.54E−02) −
+/= /−		18/3/9	18/6/3	16/9/5	18/5/7	17/8/5	17/8/5

Table 6
Comparison of mean error and standard deviation in objective function value for 100 dimension of classical test problems.

F_n	SOS	DSOS-1	DSOS-2	DSOS-3	DSOS-4	DSOS-5	DSOS-6
f_{01}	2.00E+01(2.89E−02)	1.06E+01(1.01E+01) +	2.00E+01(3.50E−02) +	2.01E+01(7.65E−02) −	2.00E+01(1.01E−02) +	2.00E+01(6.05E−02) =	2.00E+01(5.14E−02) +
f_{02}	6.73E−62(8.73E−62)	2.49E−134(1.76E−133) +	4.25E−75(9.76E−75) +	4.26E−75(6.27E−75) +	3.12E−74(5.29E−74) +	4.35E−74(7.15E−74) +	4.69E−74(1.12E−73) +
f_{03}	2.17E−112(4.11E−112)	2.44E−240(0.00E+00) +	3.95E−121(1.35E−120) +	2.34E−121(5.14E−121) +	4.43E−120(2.86E−119) +	2.15E−121(4.17E−121) +	3.40E−121(9.54E−121) +
f_{04}	6.67E−01(1.50E−11)	6.67E−01(1.61E−10) −	6.67E−01(9.67E−13) +	6.67E−01(1.45E−12) +	6.67E−01(5.12E−13) +	6.67E−01(8.76E−12) +	6.67E−01(1.77E−11) +
f_{05}	1.12E−114(3.64E−114)	5.95E−244(0.00E+00) +	1.40E−123(4.78E−123) +	8.34E−124(1.61E−123) +	1.08E−123(2.66E−123) +	7.10E−124(1.40E−123) +	5.65E−124(1.02E−123) +
f_{06}	7.12E−218(0.00E+00)	7.12E−218(0.00E+00) −	7.12E−218(0.00E+00) +	7.12E−218(0.00E+00) +	7.12E−218(0.00E+00) +	7.12E−218(0.00E+00) +	7.12E−218(0.00E+00) +
f_{07}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{08}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{09}	7.44E+00(1.20E+01)	8.63E+01(3.45E+00) −	1.16E+00(9.80E−01) +	1.39E+00(1.26E+00) +	1.73E+00(1.71E+00) +	1.50E+00(1.53E+00) +	2.14E+00(3.67E+00) +
f_{10}	5.42E+01(1.52E+00)	4.76E+01(2.11E+00) +	6.87E+01(1.54E+00) −	5.69E+01(1.65E+00) −	6.98E+01(1.69E+00) −	6.04E+01(1.56E+00) −	5.61E+01(3.14E+00) −
f_{11}	2.62E−05(1.11E−04)	5.53E−04(9.95E−04) −	5.32E−07(2.98E−06) +	4.09E−07(1.96E−06) +	6.13E−04(4.35E−03) +	2.31E−07(6.89E−07) +	1.47E−06(7.71E−06) +
f_{12}	3.90E+00(5.86E−01)	5.26E+00(8.44E−01) −	3.35E+00(4.76E−01) +	3.47E+00(6.82E−01) +	3.20E+00(5.75E−01) +	3.43E+00(6.96E−01) +	3.21E+00(6.32E−01) +
f_{13}	1.11E−117(2.11E−117)	9.46E−269(0.00E+00) +	2.02E−126(5.25E−126) +	6.72E−127(9.28E−127) +	3.61E−126(6.51E−126) +	1.84E−126(3.50E−126) +	6.62E−126(1.70E−125) +
f_{14}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{15}	9.09E+01(1.35E+00)	9.24E+01(1.29E+00) −	8.83E+01(1.54E+00) +	8.81E+01(1.37E+00) +	8.92E+01(1.41E+00) +	8.92E+01(1.39E+00) +	8.90E+01(1.62E+00) +
f_{16}	2.63E−23(7.91E−23)	6.59E−189(0.00E+00) +	1.99E−15(1.34E−14) −	7.60E−13(4.74E−12) −	1.30E−15(6.74E−15) −	1.20E−14(8.34E−14) −	1.49E−14(1.02E−13) −
f_{17}	9.99E−02(1.65E−09)	9.60E−02(1.96E−02) −	9.99E−02(4.82E−09) −	9.99E−02(6.65E−09) =	9.99E−02(7.39E−09) −	9.99E−02(3.44E−09) −	9.99E−02(3.22E−09) =
f_{18}	3.13E−03(1.96E−09)	1.23E−03(1.54E−03) +	3.13E−03(5.89E−09) =	3.13E−03(5.46E−09) =	3.13E−03(2.18E−08) −	3.13E−03(6.15E−09) =	3.13E−03(1.04E−08) =
f_{19}	1.99E+04(2.12E+03)	1.81E+04(1.78E+03) +	2.91E+04(1.06E+03) −	2.24E+04(8.78E+02) −	2.92E+04(1.02E+03) −	2.43E+04(7.31E+02) −	2.07E+04(2.35E+03) =
f_{20}	2.38E−31(9.72E−31)	5.27E−203(0.00E+00) +	3.78E−25(2.64E−24) −	5.43E−27(1.94E−26) −	2.84E−27(1.44E−26) −	2.05E−26(1.42E−25) −	2.47E−24(1.76E−23) −
f_{21}	9.88E−46(1.14E−45)	2.03E−108(1.45E−107) +	3.28E−50(3.28E−50) +	2.17E−50(3.12E−50) +	8.60E−51(1.18E−50) +	8.43E−51(7.55E−51) +	5.73E−51(4.09E−51) +
f_{22}	3.17E−62(3.03E−62)	2.33E−132(1.66E−131) +	1.77E−75(2.86E−75) +	1.59E−75(3.08E−75) +	1.90E−74(5.00E−74) +	1.44E−74(3.14E−74) +	1.79E−74(3.94E−74) +
f_{23}	2.69E−119(7.43E−119)	6.06E−240(0.00E+00) +	6.11E−129(1.14E−128) +	2.61E−129(2.66E−129) +	7.69E−129(2.20E−128) +	4.07E−129(7.43E−129) +	6.68E−129(1.22E−128) +
f_{24}	4.76E−03(2.35E−02)	1.04E−01(1.32E−01) −	1.11E−05(3.19E−05) +	6.13E−04(4.22E−03) +	1.63E−04(1.02E−03) +	4.42E−05(1.02E−04) +	7.23E−05(3.98E−04) +
f_{25}	3.13E+02(4.85E+01)	4.16E+02(5.71E+01) −	3.66E+02(5.58E+01) −	3.26E+02(9.81E+01) =	3.69E+02(7.50E+01) −	2.92E+02(9.30E+01) +	3.05E+02(6.90E+01) =
f_{26}	0.00E+00(0.00E+00)	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =	0.00E+00(0.00E+00) =
f_{27}	6.27E−119(1.92E−118)	1.79E−248(0.00E+00) +	2.04E−129(3.06E−129) +	2.30E−129(4.26E−129) +	2.74E−129(3.80E−129) +	1.29E−129(1.77E−129) +	2.05E−129(3.49E−129) +
f_{28}	1.78E−225(0.00E+00)	0.00E+00(0.00E+00) +	2.14E−230(0.00E+00) +	9.14E−231(0.00E+00) +	1.62E−230(0.00E+00) +	7.52E−232(0.00E+00) +	3.69E−231(0.00E+00) +
f_{29}	1.75E−01(5.02E−01)	1.96E−05(1.27E−04) +	1.38E−03(9.81E−03) +	5.77E−01(1.33E+00) =	9.85E−02(3.06E−01) +	1.87E−01(3.69E−01) =	2.87E−01(1.06E+00) +
f_{30}	4.60E−02(8.64E−02)	1.90E−37(1.36E−36) +	6.98E−01(7.80E−01) −	4.40E−01(5.47E−01) −	4.24E−01(3.84E−01) −	4.73E−01(5.25E−01) −	4.33E−01(6.65E−01) −
+/= /−		17/4/9	18/5/7	16/8/6	18/4/8	17/7/6	18/8/4

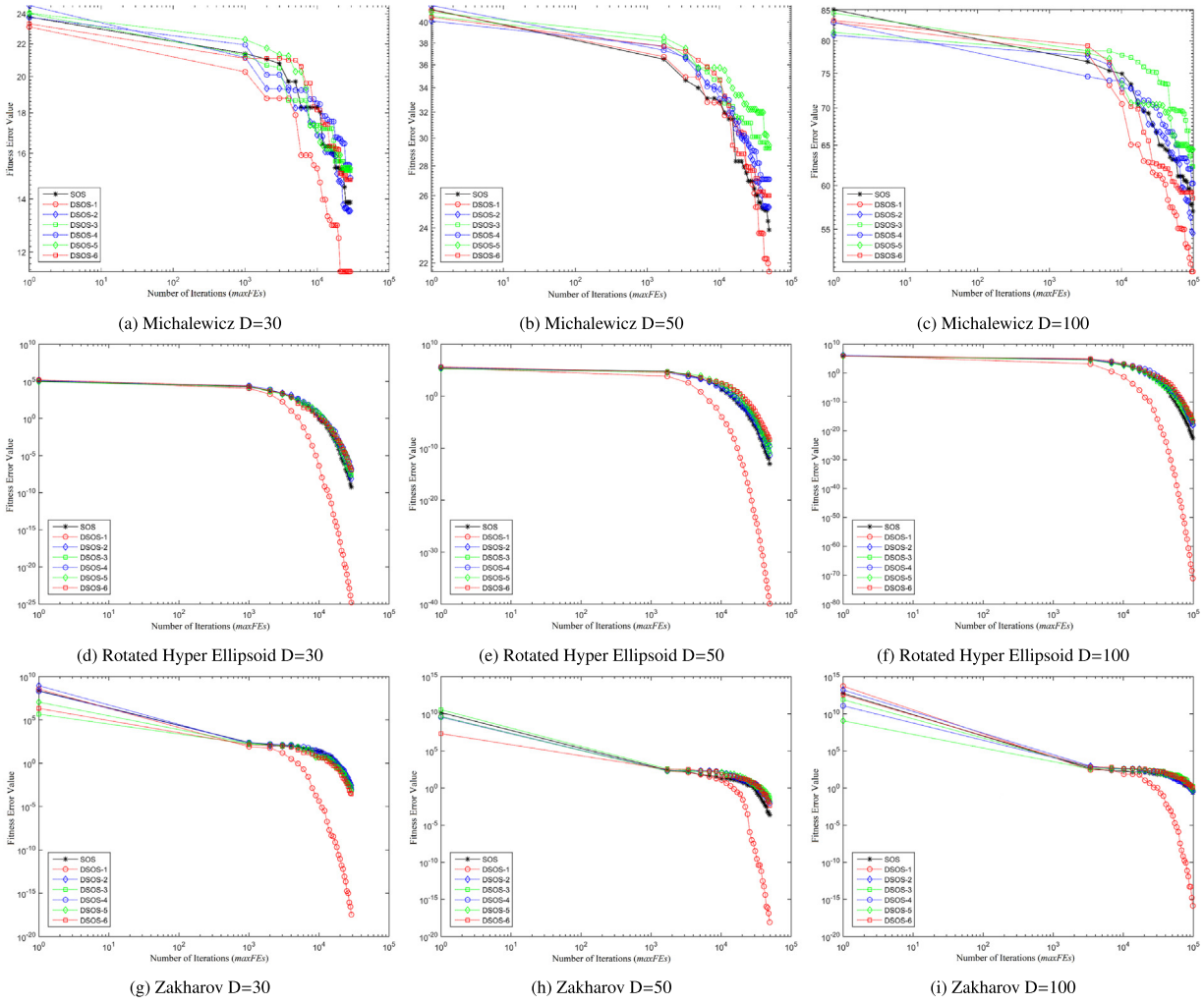
among the FDB-SOS variants in terms of convergence to the global optimum point. This is clearly observed for the functions f_7, f_{13}, f_{14}, f_{16}, f_{20}, f_{21}, f_{22}, f_{23}, f_{28}, f_{30}. According to the results of the Wilcoxon Test given at the end of Table 4, the FDB-based SOS variants have a distinct superiority over the base SOS algorithm. Moreover, all variants increased the performance of the SOS algorithm in at least 16 of the 30 test functions. Even in the worst result, there is the superiority of an FDB-based variant (DSOS-5) on sixteen test functions, similar results on five test functions and superiority of the SOS algorithm on nine test functions. For 30-dimensional problems, there are small differences in the experimental results of the FDB-SOS variants (see Table 4). This indicates the stability of the proposed selection method.

Evaluation of the results given in Tables 5 and 6 for 50 and 100-dimensional problems: When the results of the experimental studies given in Tables 5 and 6 are evaluated, it is seen that the comparison results obtained are consistent with those in Table 4. There are two main reasons for this. The first is that the number of *maxFEs* is increased depending on the problem dimension and the second is the stability of the algorithm. That is,

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 7
Statistical comparison results of SOS variants.

vs. SOS	D = 30			D = 50			D = 100		
	better	similar	worse	better	similar	worse	better	similar	worse
DSOS-1	17	6	7	18	3	9	17	4	9
DSOS-2	16	6	8	18	6	6	18	5	7
DSOS-3	16	8	6	16	9	5	16	8	6
DSOS-4	16	7	7	18	5	7	18	4	8
DSOS-5	16	5	9	17	8	5	17	7	6
DSOS-6	16	8	6	17	8	5	18	8	4



(a) Michalewicz D=30 (b) Michalewicz D=50 (c) Michalewicz D=100

(d) Rotated Hyper Ellipsoid D=30 (e) Rotated Hyper Ellipsoid D=50 (f) Rotated Hyper Ellipsoid D=100

(g) Zakharov D=30 (h) Zakharov D=50 (i) Zakharov D=100

Fig. 2. Convergence curves on multi-dimensional (30-50-100) classical test problems.

the search performance of the FDB-SOS variants does not change much, although the dimensions of the problems are raised to 30, 50, and 100, respectively. Moreover, the FDB method enhanced the performance of SOS in both unimodal and multimodal test functions.

The results in Table 4 indicate that the FDB selection method increases the exploration and convergence capabilities of the SOS algorithm, while the results in Tables 5 and 6 confirm this information.

3.2.1. Statistical analysis

The statistical comparison results of the algorithms are given in Table 7, which is a summary of Tables 4–6. According to these results, as the problem dimension increases, it is seen that the FDB-SOS variants increase the search performance, compared to the SOS algorithm. For example, when the problem dimension is 30, the DSOS-6 algorithm has superiority in 16 test functions compared to the SOS algorithm where it has similar results in 8 test functions and has a worse search performance in 6 test functions. This is an obvious advantage. When the problem dimension is 50, the DSOS-6 algorithm has superiority in 17 test functions compared to the SOS algorithm, which has similar results in 8 test functions and has a worse search performance in 5 test functions. When the problem dimension is 100, the DSOS-6 algorithm has superiority in 18 test functions compared to the SOS algorithm, which has similar results in 8 test functions and has worse search performance in 4 test functions.

Using the FDB method increases the diversity of the population and enables the effective exploration of the large search space. These results suggest that the FDB selection method provides better guidance to the SOS algorithm.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 8
Algorithm complexity of SOS variants.

Dimension	T0	T1	SOS	D-SOS1	D-SOS2	D-SOS3	D-SOS4	D-SOS5	D-SOS6
D = 30	0,0216	0,8197	160,5057	196,6848	181,9865	208,7687	223,5328	203,3043	216,4475
D = 50	0,0216	1,3630	176,6348	198,7254	200,7512	236,3448	258,9150	233,2411	255,3711
D = 100	0,0216	4,6609	214,3267	252,2109	249,6468	301,8156	337,2465	297,5197	336,6986

3.2.2. Convergence analysis

Some representative cases of convergence curves of SOS and FDB-SOS variants are shown in Fig. 2. To show the speed of convergence of algorithms on 30, 50 and 100-dimensional problems, the Michalewicz, Rotated Hyper Ellipsoid and Zakharov test functions are used. The Michalewicz function is a multimodal test problem with local optima up to the factorial of the number of design variables ($n!$). Therefore, as the dimension of the function increases, the complexity of the problem increases much more. In the Michalewicz function, the area with this global solution is very small compared to the entire search space. Therefore, the Michalewicz function is one of the most appropriate problems that can be used to test an algorithm's ability to escape any local minima.

When the curves of Fig. 2(a, b, c) are investigated, it is seen that the highest exploration ability of the FDB-SOS variants is DSOS-1. Although the DSOS-1 algorithm cannot discover the global solution, it is seen that it makes better convergence than other methods. Two examples of convex/unimodal problems, Rotated Hyper-Ellipsoid and Zakharov test functions are used to examine the algorithms' ability to exploitation. When the curves in Fig. 2(d, e, f, g, h, i) are investigated, it is seen that in this case, DSOS-1 performs the neighborhood search precisely. While DSOS-1 converges to the global minimum point with a value smaller than the tolerance value (10^{-8}), it is seen that the other methods converge with a value greater than the tolerance value.

3.2.3. Algorithm complexity

Experimental studies are conducted to test the effect of the FDB selection method on algorithm complexity. For this purpose, the IEEE CEC 2014 [61] definition document is taken as a guide. $T0$, $T1$, and $T2$, which are the parameters used to calculate the algorithm complexity, are defined in IEEE CEC 2014. T0 is the time to calculate the test program defined in CEC 2014. $T1$, is the computing time of only one algorithm just for Function 18 (f_{18} in CEC 2014). $T1$ is obtained by evaluating 200,000 iterations in the defined D dimensions of f_{18}. The complete computing time for the algorithm with 200,000 evaluations of the same D dimensional f_{18} is $T2$. The computational complexities of the algorithms based on these explanations are given in Table 8. The results of time complexity show that the most complex FDB-SOS variant is DSOS-4. The least time complexity is observed with SOS. The time complexity of SOS, DSOS-1, and DSOS-2 are similar.

Fig. 3 is plotted to provide for the computational complexity to be more visible. While the computational complexity of the SOS algorithm is in the range of 160–214, the complexity of the proposed methods is in the range of 204–295. The FDB selection method slightly increases the calculation cost of the algorithm.

3.3. Validation studies: Comparison of FDB-SOS and competing MHS algorithms

As a result of the test studies, it is determined that the most effective FDB-SOS variant is DSOS-1. In this subsection, 13 competing MHS algorithms and 90 test functions are used to verify the performance of the FDB-SOS algorithm. The results obtained from the experiments are evaluated by using statistical analysis and convergence analysis. Finally, the computational complexities of the algorithms are presented.

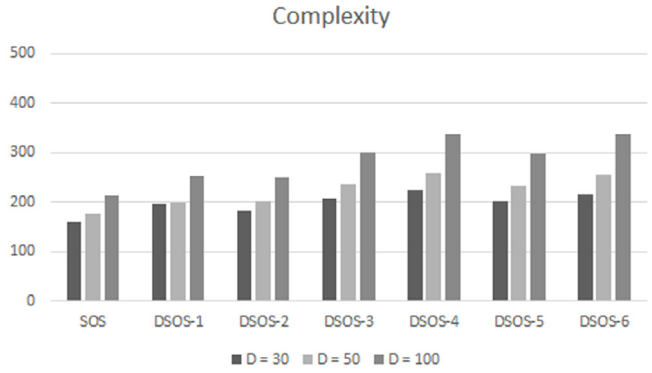


Fig. 3. Algorithm complexity of SOS variants for dimension 30, 50 and 100.

Comparison on classical benchmark functions: The first comparison is made using the classical benchmark functions given in Table 1. Table 9 reports the experimental results obtained by performing DSOS-1, and the competing algorithms, on test functions for 30-dimensions. According to the results given in Table 9, the DSOS-1 algorithm performs global optimum on 17 test functions. It provides better results in 17 of 30 test functions compared to its competitors. Considering the types of functions, 7 of these functions are multimodal (f_1, f_2, f_7, f_{11}, f_{14}, f_{15}, f_{30}) and 10 are unimodal (f_3, f_5, f_{13}, f_{16}, f_{20}, f_{21}, f_{22}, f_{23}, f_{27} f_{28}). DSOS-1 performs better than others, except for GSA on function f_4, MS on functions f_{17} and f_{18}, BSA on function f_{25}, and GWO on function f_{26}. The DSOS-1 for the f_8 function performs the global optimum with GWO, while it performs better than the other contestants do. According to the statistical results given in the last line of Table 9, the optimization performance of DSOS-1 on 30-dimensional test functions is quantitatively far superior to that of its counterparts. The closest competitor of DSOS-1 is BSA. However, DSOS-1 surpassed BSA on twenty functions. According to the results given in Tables 10 and 11, while the problem dimensions are 50 and 100, DSOS-1 finds the global solution on 18 and 17 test functions, respectively. The results given in Tables 9–11 are similar. DSOS-1 outperforms its competitors in both the low, middle and high-dimensional classic benchmark sets and in the unimodal and multi-modal test functions.

Comparison on CEC2014 benchmark suite: This benchmark set is suitable to investigate the performance of MHS algorithms. It includes 30 different optimization problems in unimodal, multimodal, hybrid and composite types at different difficulty levels [61]. The numerical performance of DSOS-1 and the 12 competing algorithms for 30, 50 and 100 dimensional test functions are given in Tables 12–14, respectively. The pair-wise comparisons between DSOS-1 and the competing algorithms on test functions are given in the last lines of Tables 12–14.

The performance of the competing thirteen algorithms for 30-dimensional problems was investigated. For unimodal function f_1, DSOS-1 and LSA perform similar performance and are superior to the other competing algorithms. For the unimodal function f_2, LSA and SSA perform similar performance and are superior to other competing algorithms. Furthermore, DSOS-1 is ranked third among the thirteen algorithms.

For unimodal function f_3, the performance of BSA is the best among the competing algorithms, and DSOS-1 is the second most

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 9
Comparison of mean error and standard deviation in objective function value for 30 dimension of classical test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	1.96E+00(6.01E+00)	4.36E−09(5.58E−10) −	2.00E+01(5.68E−09) −	2.02E+01(1.05E−01) −	2.08E+01(7.98E−02) −	2.00E+01(4.31E−07) −	2.04E+01(3.47E−01) −
f_{02}	2.78E−38(9.30E−38)	4.52E−01(7.07E−01) −	3.13E+01(8.32E+00) −	2.06E+02(4.95E+01) −	4.99E−04(6.67E−04) −	7.37E+01(6.15E+01) −	3.01E+00(4.85E+00) −
f_{03}	7.64E−63(3.98E−62)	1.06E+02(1.38E+02) −	1.27E+07(8.14E+06) −	5.32E+10(2.54E+10) −	8.48E−34(2.12E−33) −	3.92E+09(1.96E+10) −	1.31E+02(3.84E+02) −
f_{04}	6.67E−01(5.63E−11)	6.93E−01(6.20E−02) +	5.74E+00(1.91E+00) −	2.75E+04(2.39E+04) −	6.67E−01(7.39E−06) −	4.01E+04(9.82E+04) −	2.03E+00(1.48E+00) −
f_{05}	3.20E−65(1.54E−64)	1.83E+06(1.71E+06) −	4.65E+04(5.15E+04) −	2.75E+08(1.83E+08) −	8.70E−37(1.86E−36) −	1.66E+08(2.39E+08) −	7.86E−01(1.79E+00) −
f_{06}	7.18E−66(3.19E−81)	1.51E−33(1.08E−32) −	9.56E−66(1.87E−66) −	1.47E−55(8.17E−55) −	0.00E+00(0.00E+00) +	7.18E−66(3.19E−81) =	1.10E−63(3.93E−63) −
f_{07}	0.00E+00(0.00E+00)	1.44E+01(4.02E+00) −	9.16E−01(1.41E−01) −	6.63E+01(2.45E+01) −	2.97E−03(6.20E−03) −	1.97E+01(4.53E+01) −	9.71E−03(1.36E−02) −
f_{08}	0.00E+00(0.00E+00)	5.47E−02(1.06E−01) −	4.77E+00(2.76E+00) −	6.12E+03(2.61E+03) −	0.00E+00(0.00E+00) =	1.37E+03(4.01E+03) −	1.42E+00(5.14E−01) −
f_{09}	1.94E+00(4.06E+00)	1.72E−02(4.04E−02) +	5.81E−02(5.22E−02) +	1.19E+02(3.69E+01) −	3.35E+00(1.99E+00) −	2.38E+01(4.58E+01) −	7.59E−01(1.23E+00) =
f_{10}	1.06E+01(1.03E+00)	2.97E+00(1.02E+00) +	8.11E+00(7.15E−01) +	1.80E+01(1.35E+00) −	2.34E+01(8.14E−01) −	8.75E+00(2.37E+00) +	4.84E+00(9.26E−01) +
f_{11}	6.35E−08(1.24E−07)	2.67E−01(3.38E−01) −	2.41E−02(2.29E−02) −	1.38E+06(1.99E+06) −	2.99E−02(1.56E−02) −	2.18E+00(1.65E+00) −	5.20E−01(8.40E−01) −
f_{12}	3.30E−01(1.67E−01)	8.84E−01(1.37E+00) =	1.90E−01(1.36E−01) +	1.61E+07(2.48E+07) −	4.65E−01(2.18E−01) −	3.65E+00(2.88E+00) −	7.30E−01(1.10E+00) =
f_{13}	4.30E−70(3.06E−69)	3.10E−01(2.09E−01) −	1.20E+00(6.40E−01) −	2.80E+02(2.01E+02) −	1.22E−03(1.47E−03) −	1.42E+02(1.12E+02) −	1.23E+00(1.29E+00) −
f_{14}	0.00E+00(0.00E+00)	3.43E+01(1.31E+01) −	1.38E+02(2.46E+01) −	7.69E+03(2.60E+03) −	1.58E+00(3.03E+00) −	7.21E+02(2.37E+03) −	7.03E+01(2.09E+01) −
f_{15}	2.42E+01(4.68E−01)	2.99E+01(1.38E+01) −	1.17E+02(3.58E+01) −	7.52E+04(8.27E+04) −	2.68E+01(7.26E−01) −	1.62E+03(3.42E+03) −	7.37E+01(4.81E+01) −
f_{16}	2.30E−56(1.44E−55)	4.76E+02(1.60E+02) −	1.84E+03(7.09E+02) −	2.09E+04(5.85E+03) −	2.43E−10(6.19E−10) −	1.64E+04(1.05E+04) −	7.43E+02(3.34E+02) −
f_{17}	9.60E−02(1.96E−02)	2.22E+00(4.87E−01) −	1.82E+00(2.62E−01) −	9.65E+00(1.54E+00) −	1.76E−01(4.73E−02) −	3.40E+00(3.21E+00) −	6.79E−01(1.75E−01) −
f_{18}	4.29E−04(1.09E−03)	4.32E−01(2.44E−02) −	4.21E−01(3.48E−02) −	4.96E−01(2.24E−03) −	3.13E−03(2.68E−10) −	3.28E−01(1.07E−01) −	7.09E−02(2.44E−02) −
f_{19}	4.39E+03(4.72E+02)	9.77E+03(4.84E+02) −	2.43E+03(3.09E+02) +	7.19E+03(4.38E+02) −	1.13E+04(9.92E+01) −	3.66E+03(8.34E+02) +	4.56E+03(6.31E+02) =
f_{20}	1.84E−57(9.86E−57)	5.16E+03(1.77E+03) −	2.21E+04(9.52E+03) −	2.77E+05(1.20E+05) −	1.68E−12(4.68E−12) −	2.29E+05(1.92E+05) −	2.81E+03(1.46E+03) −
f_{21}	7.28E−32(2.76E−31)	2.03E+00(1.24E+00) −	8.15E+00(1.38E+00) −	4.00E+01(7.62E+00) −	3.03E−10(3.00E−10) −	6.06E+01(8.89E+00) −	6.07E+00(3.17E+00) −
f_{22}	7.57E−38(3.69E−37)	2.84E−08(6.31E−09) −	4.72E−01(1.81E−01) −	5.39E+01(1.64E+01) −	6.21E−23(5.23E−23) −	3.09E+01(1.96E+01) −	7.90E−02(2.56E−01) −
f_{23}	4.18E−68(2.93E−67)	3.09E−17(1.01E−17) −	1.86E+00(1.74E+00) −	5.93E+03(2.71E+03) −	7.35E−41(9.53E−41) −	2.36E+03(5.51E+03) −	1.88E−04(1.21E−03) −
f_{24}	1.63E−06(2.13E−06)	3.09E−17(8.11E−18) +	1.51E+00(1.14E+00) −	6.06E+03(2.62E+03) −	4.30E−01(3.08E−01) −	1.57E+03(3.68E+03) −	1.26E−04(5.74E−04) =
f_{25}	6.45E+01(3.15E+01)	7.62E+01(2.77E+01) =	1.90E+01(1.11E+01) +	3.44E+02(4.71E+01) −	2.85E+02(5.87E+01) −	1.44E+02(4.16E+01) −	1.20E+02(3.37E+01) −
f_{26}	9.52E−107(6.33E−106)	7.32E−06(3.60E−05) −	6.93E−04(3.46E−03) −	3.82E+08(2.01E+09) −	1.27E−115(9.06E−115)+	2.45E+16(1.54E+17) −	9.87E−19(2.59E−18) −
f_{27}	5.27E−69(3.62E−68)	3.00E−16(1.14E−16) −	2.43E−01(3.35E−01) −	3.69E+02(1.02E+02) −	1.99E−39(5.67E−39) −	4.67E+02(4.80E+02) −	5.16E−04(2.15E−03) −
f_{28}	1.82E−134(1.10E−133)	6.53E−34(3.42E−34) −	1.82E−03(4.56E−03) −	7.26E+03(6.01E+03) −	3.94E−64(2.06E−63) −	7.65E+03(1.52E+04) −	2.06E−11(5.98E−11) −
f_{29}	3.21E−04(1.64E−03)	1.59E−01(3.21E−01) −	0.00E+00(0.00E+00) +	2.09E+00(1.21E+00) −	1.42E+01(1.42E+00) −	0.00E+00(0.00E+00) +	5.21E+00(1.64E+00) −
f_{30}	2.41E−35(1.51E−34)	6.15E+01(1.42E+01) −	8.02E+01(2.02E+01) −	4.31E+02(1.37E+02) −	4.69E−04(1.32E−03) −	2.71E+02(1.03E+02) −	2.85E+01(1.12E+01) −
+/= /−		4/2/24	6/0/24	0/0/30	2/1/27	3/1/26	1/4/25

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	2.03E+01(6.05E−02) −	3.68E+00(9.89E−01) −	1.79E+01(7.83E−01) −	6.39E+00(7.64E+00) −	3.52E−08(2.98E−08) −	9.72E−07(1.22E−06) −
f_{02}	1.14E+01(1.30E+01) −	2.02E+01(5.91E+00) −	1.12E+02(1.48E+01) −	3.16E+01(1.28E+01) −	7.48E−09(8.41E−09) −	2.46E−07(6.63E−08) −
f_{03}	4.28E+06(1.09E+07) −	5.96E+06(2.86E+06) −	2.56E+10(6.24E+09) −	1.00E+03(1.20E+03) −	4.30E−08(8.53E−08) −	1.78E−09(1.16E−10) −
f_{04}	1.70E+01(3.70E+01) −	2.66E+00(1.23E+00) −	1.79E+04(7.10E+03) −	7.24E+02(1.44E+03) −	6.67E−01(1.10E−07) −	9.80E−01(8.79E−03) −
f_{05}	1.02E+02(2.34E+02) −	4.15E+06(2.11E+06) −	9.81E+08(5.75E+08) −	2.96E+07(2.09E+07) −	3.14E−10(5.61E−10) −	1.53E−09(1.56E−10) −
f_{06}	1.46E−43(8.70E−43) −	1.32E−29(7.08E−29) −	7.14E−12(3.35E−11) −	0.00E+00(0.00E+00) +	6.62E−56(3.73E−55) −	4.13E−10(2.00E−09) −
f_{07}	5.21E−01(3.20E−01) −	7.29E−01(1.30E−01) −	2.48E+01(5.61E+00) −	9.75E−03(1.05E−02) −	2.11E−16(4.72E−16) −	1.04E−09(2.58E−10) −
f_{08}	1.73E+00(3.51E+00) −	4.19E+00(7.86E−01) −	2.85E+03(5.97E+02) −	1.90E+00(6.18E−01) −	7.07E−14(1.26E−13) −	1.30E−09(2.82E−10) −
f_{09}	2.33E+01(1.80E+00) −	2.86E+00(4.21E+00) −	1.27E+02(2.17E+01) −	2.71E+01(2.20E+01) −	3.62E+00(1.10E+00) −	2.78E+01(2.50E+00) −
f_{10}	2.22E+01(7.65E−01) −	9.88E+00(1.68E+00) +	2.13E+01(6.56E−01) −	1.97E+01(1.52E+00) −	9.42E+00(1.74E+00) +	2.06E+01(5.54E−01) −
f_{11}	1.28E+01(5.41E+01) −	2.56E+00(1.05E+00) −	4.94E+03(2.98E+04) −	5.58E+00(3.02E+00) −	6.59E−03(3.20E−03) −	5.63E−01(1.29E−01) −
f_{12}	6.29E+03(2.47E+04) −	5.28E−01(5.39E−01) =	2.72E+05(2.67E+05) −	3.40E+00(6.82E+00) +	1.86E−01(7.51E−02) +	2.95E+00(3.13E−01) −
f_{13}	1.45E+00(3.09E+00) −	8.40E−01(3.39E−01) −	2.45E+03(5.13E+02) −	3.43E+03(2.80E+03) −	5.85E−14(9.90E−14) −	1.04E−09(1.16E−10) −
f_{14}	8.04E+01(6.19E+01) −	2.28E+02(6.45E+01) −	2.98E+03(6.30E+02) −	1.84E+02(5.96E+01) −	5.93E−13(1.18E−12) −	2.53E+00(1.81E+01) −
f_{15}	1.37E+02(2.07E+02) −	3.72E+01(1.89E+01) −	3.60E+04(1.26E+04) −	5.21E+02(1.48E+03) −	2.65E+01(1.67E−01) −	2.89E+01(2.66E−02) −
f_{16}	4.66E+03(4.36E+03) −	1.31E+02(4.51E+01) −	1.84E+04(1.06E+04) −	2.95E+02(1.59E+02) −	3.07E−14(5.12E−14) −	1.29E−09(1.24E−10) −
f_{17}	5.34E−01(3.39E−01) −	1.19E+00(1.39E−01) −	5.46E+00(5.35E−01) −	1.10E+00(1.87E−01) −	4.69E−09(4.02E−09) +	3.30E−01(3.37E−02) −
f_{18}	1.03E−01(7.64E−02) −	1.69E−01(3.59E−02) −	4.33E−01(2.22E−02) −	9.25E−02(3.29E−02) −	4.35E−18(3.11E−17) +	6.24E−03(3.38E−05) −
f_{19}	8.68E+03(2.63E+02) −	5.61E+03(7.09E+02) −	9.83E+03(4.40E+02) −	1.11E+04(1.00E+02) −	6.65E+03(5.56E+02) −	8.94E+03(3.87E+02) −
f_{20}	3.29E+04(3.58E+04) −	1.38E+03(4.77E+02) −	2.69E+05(1.47E+05) −	4.57E+03(3.23E+03) −	4.27E−13(9.28E−13) −	1.40E−09(1.27E−10) −
f_{21}	2.34E+01(8.56E+00) −	4.16E+00(1.39E+00) −	1.79E+01(2.67E+00) −	6.18E+00(2.56E+00) −	4.06E−08(4.61E−08) −	4.28E−07(2.87E−08) −
f_{22}	1.52E−03(1.62E−03) −	1.89E+00(7.11E−01) −	1.01E+12(4.65E+12) −	1.84E+15(1.12E+16) −	8.54E−08(6.95E−08) −	2.55E−07(5.68E−08) −
f_{23}	8.54E−01(2.47E+00) −	8.07E−01(3.97E−01) −	2.62E+03(5.45E+02) −	1.48E−08(3.30E−09) −	6.69E−15(2.03E−14) −	1.32E−09(8.70E−11) −
f_{24}	7.65E+00(6.85E+00) −	7.04E−01(2.89E−01) −	2.72E+03(6.30E+02) −	1.52E−08(3.90E−09) +	4.98E−02(2.54E−02) −	5.51E+00(5.40E−01) −
f_{25}	5.67E+02(3.69E+01) −	1.75E+02(3.94E+01) −	6.88E+02(4.82E+01) −	1.83E+02(3.72E+01) −	1.74E+02(3.20E+01) −	4.28E+02(1.89E+01) −
f_{26}	1.93E+01(6.24E+01) −	1.46E−05(3.49E−05) −	1.53E+21(8.31E+21) −	5.89E+13(2.96E+14) −	9.76E−22(1.73E−21) −	1.02E+22(3.90E+22) −
f_{27}	9.50E−02(2.62E−01) −	1.22E+00(7.64E−01) −	8.07E+02(1.69E+02) −	3.73E+01(5.55E+01) −	3.25E−14(7.04E−14) −	1.19E−09(8.58E−11) −
f_{28}	3.00E+00(9.80E+00) −	1.61E−03(1.49E−03) −	4.16E+03(1.58E+03) −	2.80E−15(7.09E−15) −	2.21E−27(8.98E−27) −	1.63E−12(2.03E−13) −
f_{29}	4.48E+00(1.05E+00) −	1.43E+01(1.67E+00) −	0.00E+00(0.00E+00) +	5.26E+00(7.80E−01) −	0.00E+00(0.00E+00) +	1.72E+01(1.62E+00) −
f_{30}	1.63E+01(1.27E+01) −	7.12E+01(2.34E+01) −	1.05E+09(2.56E+09) −	4.09E+03(4.63E+03) −	9.46E−14(2.36E−13) −	9.53E−10(9.47E−11) −
+/= /−	0/0/30	1/1/28	1/0/29	3/0/27	5/0/25	0/0/30

successful algorithm after BSA. According to the numerical comparison results given in the first three lines of Table 12, two successful algorithms on the 30-dimensional uniform functions (f_1–f_3) are DSOS-1 and LSA. In all of the multimodal test functions (f_4–f_{16}), the DSOS-1 algorithm is superior to the SCA, CGSA and BOA algorithms.

Among the fourteen test functions, only the HSS algorithm found the optimal result with the DSOS-1 algorithm for f_5 and failed more than DSOS-1 in all other test functions. DSOS-1 and GWO exhibit similar performance on multimodal functions. From Table 12, it can be seen that DSOS-1 outperformed GSA, BSA, MFO, LSA, CSA, and MS on 9, 3, 9, 4, 6, and 8 multimodal test functions, respectively. In contrast, BSA and LSA performed better than DSOS-1 on 10, and 5 multimodal test functions, respectively. BSA performs extremely well on multimodal functions.

The DSOS-1 algorithm has achieved superiority in all hybrid problems (f_{17}–f_{22}) for the HSS, MFO, SCA, CGSA, MS and BOA algorithms. From Table 12, it can be seen that the DSOS-1 algorithm performed better in three problems than the LSA algorithm in four problems compared to the GSA, BSA, GWO and SSA algorithms for hybrid test functions. For hybrid functions, DSOS-1 and CSA perform extremely well.

DSOS-1 performs better in all composition test functions (f_{23}–f_{30}) than HSS, GWO, SCA, CSA, CGSA and SSA. GSA, BSA, MFO, and LSA performed better than DSOS-1 on 1 (f_{29}), 1 (f_{26}), 1 (f_{28}), and 2 (f_{29} and f_{30}) test functions, respectively. In composition problems,

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 10
Comparison of mean error and standard deviation in objective function value for 50 dimension of classical test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	3.92E+00(8.02E+00)	1.72E−02(1.23E−01) −	2.00E+01(3.26E−09) −	2.03E+01(1.01E−01) −	2.11E+01(5.74E−02) −	2.00E+01(3.00E−07) −	2.04E+01(4.12E−01) −
f_{02}	1.66E−64(7.27E−64)	8.33E−01(1.36E+00) −	4.36E+01(1.29E+01) −	4.09E+02(5.51E+01) −	9.35E−05(2.81E−04) −	2.04E+02(1.49E+02) −	1.47E+01(1.25E+01) −
f_{03}	5.70E−111(2.95E−110)	7.30E+01(1.02E+02) −	1.17E+07(8.90E+06) −	1.17E+11(2.62E+10) −	1.39E−44(2.31E−44) −	7.45E+10(9.76E+10) −	9.25E+04(3.30E+05) −
f_{04}	6.67E−01(2.85E−11)	7.02E−01(7.32E−02) +	1.15E+01(3.72E+00) −	6.55E+04(2.82E+04) −	6.67E−01(2.45E−07) −	2.08E+05(3.46E+05) −	9.01E+00(4.05E+00) −
f_{05}	1.13E−111(8.03E−111)	1.61E+06(1.02E+06) −	1.53E+04(9.74E+03) −	9.17E+08(4.48E+08) −	7.23E−48(1.93E−47) −	1.34E+09(1.49E+09) −	4.03E+03(1.52E+04) −
f_{06}	2.67E−109(8.66E−114)	2.34E−55(1.66E−54) −	1.40E−108(7.67E−109) −	3.72E−89(2.65E−88) −	0.00E+00(0.00E+00) +	2.31E−106(1.15E−105)+	2.90E−101(1.19E−100) −
f_{07}	0.00E+00(0.00E+00)	1.63E+01(4.34E+00) −	7.13E−01(2.08E−01) −	1.25E+02(2.90E+01) −	7.46E−04(3.14E−03) =	5.91E+01(6.70E+01) −	7.86E−03(8.92E−03) −
f_{08}	0.00E+00(0.00E+00)	3.73E−01(3.18E−01) −	5.14E+00(1.20E+00) −	1.30E+04(2.80E+03) −	0.00E+00(0.00E+00) =	6.87E+03(7.34E+03) −	2.28E+00(6.88E−01) −
f_{09}	3.27E+01(1.21E+01)	1.50E−01(5.07E−01) +	2.66E−02(1.90E−02) +	2.53E+02(5.93E+01) −	1.26E+01(2.96E+00) +	1.37E+02(1.24E+02) −	4.33E+00(6.59E+00) +
f_{10}	2.04E+01(1.57E+00)	5.38E+00(1.52E+00) +	1.53E+01(9.28E−01) +	3.32E+01(1.26E+00) −	4.19E+01(7.26E−01) −	1.59E+01(2.84E+00) +	1.03E+01(1.79E+00) +
f_{11}	7.56E−07(1.92E−06)	4.14E−01(3.22E−01) −	4.10E−03(3.46E−03) −	1.92E+07(3.03E+07) −	5.88E−02(2.06E−02) −	2.51E+07(9.24E+07) −	9.66E−01(1.48E+00) −
f_{12}	1.45E+00(4.93E−01)	6.72E+00(4.50E+00) −	1.07E−01(6.58E−02) +	3.21E+07(3.15E+07) −	1.49E+00(4.49E−01) =	3.78E+01(2.53E+01) −	7.20E+00(5.81E+00) −
f_{13}	3.56E−115(2.54E−114)	5.31E−01(2.91E−01) −	1.67E+00(6.19E−01) −	7.06E+02(3.43E+02) −	3.38E−04(5.48E−04) −	4.02E+02(3.78E+02) −	4.40E+00(3.39E+00) −
f_{14}	0.00E+00(0.00E+00)	7.32E+01(2.53E+01) −	2.02E+02(2.86E+01) −	1.57E+04(2.84E+03) −	1.62E+00(9.19E+00) −	8.98E+03(8.90E+03) −	1.40E+02(3.72E+01) −
f_{15}	4.30E+01(7.48E−01)	4.67E+01(1.02E+00) −	1.49E+02(5.27E+01) −	1.88E+05(1.25E+05) −	4.69E+01(7.56E−01) −	7.29E+04(2.48E+05) −	1.56E+02(6.38E+01) −
f_{16}	4.29E−89(3.02E−88)	1.06E+03(2.46E+02) −	3.47E+03(1.21E+03) −	6.46E+04(2.24E+04) −	3.84E−09(2.07E−08) −	3.88E+04(1.61E+04) −	6.15E+03(1.74E+03) −
f_{17}	9.99E−02(8.17E−08)	3.23E+00(5.22E−01) −	2.53E+00(3.35E−01) −	1.41E+01(1.56E+00) −	1.84E−01(4.18E−02) −	1.12E+01(4.90E+00) −	1.06E+00(2.26E−01) −
f_{18}	8.58E−04(1.41E−03)	4.74E−01(1.05E−02) −	4.47E−01(2.03E−02) −	4.99E−01(5.39E−04) −	3.13E−03(1.08E−10) −	4.86E−01(1.97E−02) −	2.43E−01(4.93E−02) −
f_{19}	8.25E+03(9.10E+02)	1.74E+04(5.95E+02) −	4.83E+03(3.34E+02) +	1.33E+04(7.00E+02) −	1.92E+04(1.22E+02) −	7.63E+03(1.43E+03) +	8.57E+03(8.65E+02) =
f_{20}	5.78E−95(4.12E−94)	1.87E+04(5.58E+03) −	6.53E+04(2.43E+04) −	1.30E+06(4.13E+05) −	3.99E−11(2.71E−10) −	9.88E+05(6.00E+05) −	6.88E+04(2.11E+04) −
f_{21}	9.88E−53(7.04E−52)	3.98E+00(1.27E+00) −	1.00E+01(1.89E+00) −	4.61E+01(7.37E+00) −	2.62E−11(3.75E−11) −	7.82E+01(4.92E+00) −	2.26E+01(4.31E+00) −
f_{22}	1.61E−62(1.08E−61)	5.62E−08(8.80E−09) −	4.21E−01(1.64E−01) −	1.06E+02(4.90E+01) −	1.59E−29(1.48E−29) −	6.57E+01(3.11E+01) −	8.30E−01(1.03E+00) −
f_{23}	7.95E−115(5.56E−114)	6.96E−17(1.75E−17) −	1.24E+00(9.46E−01) −	1.26E+04(3.36E+03) −	1.04E−51(1.56E−51) −	7.65E+03(9.07E+03) −	8.02E−03(3.71E−02) −
f_{24}	8.82E−05(4.59E−04)	6.90E−17(1.63E−17) +	1.08E+00(5.00E−01) −	1.32E+04(3.43E+03) −	1.67E+00(5.84E−01) −	7.35E+03(1.00E+04) −	6.17E−03(2.45E−02) −
f_{25}	1.48E+02(3.98E+01)	1.48E+02(3.93E+01) =	4.90E+01(2.65E+01) +	6.88E+02(8.07E+01) −	6.70E+02(8.69E+01) −	2.72E+02(4.41E+01) −	2.23E+02(3.90E+01) −
f_{26}	9.18E−177(0.00E+00)	1.13E−02(7.53E−02) −	4.51E−08(2.16E−07) −	1.39E+19(4.44E+19) −	7.37E−190(0.00E+00) =	2.66E+33(1.41E+34) −	5.68E−09(2.23E−08) −
f_{27}	1.14E−115(7.48E−115)	1.04E−15(2.99E−16) −	2.43E−01(1.71E−01) −	1.13E+03(2.67E+02) −	1.88E−50(2.96E−50) −	2.57E+03(2.24E+03) −	1.30E−01(5.33E−01) −
f_{28}	1.57E−221(0.00E+00)	2.82E−33(1.48E−33) −	5.30E−04(6.43E−04) −	1.93E+04(7.46E+03) −	5.25E−82(1.15E−81) −	4.66E+04(5.63E+04) −	2.37E−05(1.55E−04) −
f_{29}	3.29E−03(1.91E−02)	5.31E−01(4.98E−01) −	0.00E+00(0.00E+00) +	4.93E+00(1.92E+00) −	2.87E+01(1.69E+00) −	0.00E+00(0.00E+00) +	1.23E+01(2.98E+00) −
f_{30}	6.34E−38(4.19E−37)	1.06E+02(2.06E+01) −	1.53E+02(4.06E+01) −	8.92E+02(2.11E+02) −	2.70E−01(4.73E−01) −	7.44E+02(1.58E+02) −	1.58E+02(3.54E+01) −
+/= /−		4/1/25	6/0/24	0/0/30	2/4/24	4/0/26	2/1/27

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	2.04E+01(6.35E−02) −	5.17E+00(8.98E−01) −	1.80E+01(6.50E−01) −	7.15E+00(7.52E+00) −	2.16E−08(2.04E−08) −	6.62E−10(5.10E−10) −
f_{02}	4.35E+01(3.87E+01) −	5.39E+01(1.55E+01) −	2.03E+02(2.52E+01) −	6.45E+01(1.65E+01) −	6.31E−09(5.50E−09) −	1.09E−09(3.54E−10) −
f_{03}	2.41E+08(4.05E+08) −	1.17E+07(3.79E+06) −	4.82E+10(9.81E+09) −	7.04E+02(8.11E+02) −	1.74E−08(3.18E−08) −	1.51E−11(1.00E−12) −
f_{04}	2.26E+03(4.56E+03) −	6.00E+00(3.07E+00) −	1.36E+04(5.41E+03) −	7.56E+02(1.10E+03) −	6.67E−01(1.09E−07) −	9.93E−01(3.15E−03) −
f_{05}	3.50E+03(5.08E+03) −	8.96E+06(3.00E+06) −	2.34E+09(1.07E+09) −	5.36E+07(3.03E+07) −	6.65E−11(1.24E−10) −	1.40E−11(8.76E−13) −
f_{06}	4.97E−56(3.55E−55) −	7.81E−38(5.39E−37) −	2.89E−16(1.39E−15) −	0.00E+00(0.00E+00) +	7.84E−96(5.59E−95) −	2.45E−16(1.16E−15) −
f_{07}	1.25E+00(7.96E−01) −	7.88E−01(9.95E−02) −	4.16E+01(8.05E+00) −	7.83E−03(9.12E−03) −	1.31E−17(4.24E−17) −	7.49E−12(2.61E−12) −
f_{08}	3.74E+01(5.22E+01) −	8.14E+00(1.43E+00) −	4.64E+03(7.21E+02) −	3.68E+00(7.78E−01) −	1.45E−14(2.86E−14) −	1.19E−11(3.11E−12) −
f_{09}	4.87E+01(4.91E+00) −	2.25E+01(1.36E+01) +	1.51E+02(2.11E+01) −	3.08E+02(9.29E+02) −	7.66E+00(1.83E+00) +	4.96E+01(1.23E+00) −
f_{10}	3.92E+01(1.06E+00) −	2.01E+01(2.37E+00) =	3.73E+01(6.53E−01) −	3.57E+01(2.07E+00) −	1.66E+01(1.69E+00) +	3.62E+01(7.03E−01) −
f_{11}	7.53E+05(1.72E+06) −	3.57E+00(1.14E+00) −	1.18E+04(1.69E+04) −	1.00E+01(4.43E+00) −	9.49E−03(9.06E−03) −	8.66E−01(1.23E−01) −
f_{12}	3.17E+06(6.35E+06) −	1.61E+01(2.35E+01) −	9.46E+05(4.79E+05) −	4.38E+01(2.15E+01) −	3.94E−01(1.24E−01) +	5.61E+00(3.86E−01) −
f_{13}	1.04E+01(1.66E+01) −	1.49E+00(4.16E−01) −	3.84E+03(7.37E+02) −	6.70E+03(3.64E+03) −	2.47E−14(5.31E−14) −	1.13E−11(7.65E−13) −
f_{14}	2.99E+02(2.03E+02) −	5.31E+02(1.26E+02) −	4.97E+03(9.09E+02) −	4.12E+02(1.12E+02) −	6.04E−13(1.16E−12) −	2.25E−07(1.54E−06) −
f_{15}	5.01E+03(6.11E+03) −	6.25E+01(1.66E+01) −	1.90E+04(5.64E+03) −	2.13E+03(8.23E+03) −	4.63E+01(1.51E−01) −	4.89E+01(2.86E−02) −
f_{16}	2.88E+04(1.07E+04) −	2.89E+02(6.63E+01) −	5.30E+04(3.07E+04) −	1.99E+03(1.30E+03) −	2.22E−14(2.95E−14) −	1.22E−11(9.15E−13) −
f_{17}	1.66E+00(9.06E−01) −	1.57E+00(1.75E−01) −	7.07E+00(6.77E−01) −	1.79E+00(2.48E−01) −	2.85E−09(3.13E−09) +	3.24E−01(2.69E−02) −
f_{18}	3.35E−01(1.17E−01) −	2.30E−01(3.15E−02) −	4.74E−01(6.44E−03) −	1.60E−01(3.91E−02) −	0.00E+00(0.00E+00) +	6.23E−03(1.55E−06) −
f_{19}	1.58E+04(3.11E+02) −	9.73E+03(1.11E+03) −	1.74E+04(5.69E+02) −	1.85E+04(1.27E+02) −	1.17E+04(8.56E+02) −	1.62E+04(4.69E+02) −
f_{20}	4.15E+05(2.21E+05) −	5.43E+03(1.65E+03) −	1.14E+06(6.14E+05) −	3.12E+04(1.58E+04) −	3.30E−13(4.55E−13) −	1.31E−11(9.91E−13) −
f_{21}	5.49E+01(8.26E+00) −	5.56E+00(1.02E+00) −	2.02E+01(2.17E+00) −	1.30E+01(2.35E+00) −	2.48E−08(1.96E−08) −	5.87E−09(3.55E−10) −
f_{22}	6.63E−03(1.28E−02) −	3.81E+00(8.94E−01) −	3.17E+22(1.70E+23) −	3.12E+29(1.63E+30) −	7.33E−08(7.69E−08) −	4.98E+22(1.85E+23) −
f_{23}	2.52E+01(4.06E+01) −	1.37E+00(5.16E−01) −	4.54E+03(1.00E+03) −	4.21E−08(9.62E−09) −	1.68E−15(3.03E−15) −	1.25E−11(6.62E−13) −
f_{24}	3.98E+01(5.10E+01) −	1.31E+00(3.58E−01) −	4.71E+03(9.54E+02) −	4.29E−08(9.20E−09) +	1.15E−01(3.56E−02) −	1.02E+01(8.16E−01) −
f_{25}	1.07E+03(6.02E+01) −	3.18E+02(4.63E+01) −	1.21E+03(5.85E+01) −	3.27E+02(4.56E+01) −	3.31E+02(5.57E+01) −	8.03E+02(2.80E+01) −
f_{26}	8.31E+10(5.77E+11) −	5.92E−06(6.20E−06) −	3.62E+38(1.63E+39) −	2.19E+28(1.21E+29) −	4.19E−23(6.60E−23) −	5.04E+39(2.76E+40) −
f_{27}	2.99E+00(5.48E+00) −	3.59E+00(1.69E+00) −	1.23E+03(2.50E+02) −	1.76E+02(1.85E+02) −	4.14E−14(6.68E−14) −	1.21E−11(6.98E−13) −
f_{28}	9.27E+02(2.04E+03) −	9.02E−03(4.95E−03) −	3.19E+03(1.51E+03) −	9.85E−14(1.59E−13) −	1.46E−28(3.54E−28) −	1.43E−14(1.13E−15) −
f_{29}	1.12E+01(1.38E+00) −	2.89E+01(2.27E+00) −	8.19E−01(5.11E−01) −	1.18E+01(1.06E+00) −	0.00E+00(0.00E+00) +	2.54E+01(4.38E+00) −
f_{30}	7.10E+01(3.13E+01) −	2.95E+02(9.21E+01) −	1.70E+11(2.76E+11) −	1.47E+04(1.13E+04) −	3.40E−14(5.26E−14) −	1.04E−11(8.69E−13) −
+/= /−	0/0/30	1/1/28	0/0/30	2/0/28	6/0/24	0/0/30

DSOS-1 and BOA have a similar performance. DSOS-1 in three of the eight problems and MS in five problems performed better. The most successful algorithm for composition test problems was MS.

According to the pair-wise Wilcoxon rank sum test results (see the last line in Table 12), DSOS-1 performs a better convergence and obtains significantly better statistical results than the others. Among the competitors, the only algorithm that performs close to DSOS-1 in the CEC 2014 benchmark suite is BSA.

The numerical performance of the algorithms on the 50, 100 dimensional test functions, and the results of the pair-wise comparisons are given in Tables 13 and 14. When the statistical and numerical results given in Tables 13 and 14 are investigated, it is seen that they are similar to the algorithm performance given in Table 12. According to the pair-wise Wilcoxon rank sum test results, the DSOS-1 algorithm is superior to all other algorithms in 50-dimensional test functions (see the last line in Table 13). From Table 14, it can be seen that DSOS-1 and BSA outperformed the other algorithms on 100-dimensional test functions.

Comparison on CEC2017 benchmark suite: The numerical performance of DSOS-1 and twelve competing algorithms for 30, 50 and 100-dimensional test functions are given in Tables 15, 16 and 17, respectively. Pair-wise comparison results are given in the last lines of the same tables. Firstly, the performance of thirteen algorithms competing for 30-dimensional problems is investigated. The DSOS-1 algorithm is superior to all competing algorithms except LSA, CSA and SSA algorithms for all of the unimodal test

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 11
Comparison of mean error and standard deviation in objective function value for 100 dimension of classical test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	1.06E+01(1.01E+01)	1.26E+00(9.31E−01) =	2.00E+01(1.58E−09) −	2.03E+01(9.18E−02) −	2.12E+01(3.71E−02) −	2.00E+01(3.13E−07) =	2.04E+01(3.80E−01) −
f_{02}	2.49E−134(1.76E−133)	2.24E+00(1.97E+00) −	5.91E+01(9.18E+00) −	9.12E+02(1.14E+02) −	2.71E−05(1.13E−04) −	3.46E+02(2.77E+02) −	1.00E+02(4.30E+01) −
f_{03}	2.44E−240(0.00E+00)	9.95E+02(6.60E+03) −	2.29E+07(1.04E+07) −	2.11E+11(2.57E+10) −	4.93E−66(1.06E−65) −	2.07E+11(1.32E+11) −	7.10E+06(8.96E+06) −
f_{04}	6.67E−01(1.61E−10)	8.72E+00(8.89E+00) −	4.97E+01(1.19E+01) −	1.53E+05(4.16E+04) −	6.67E−01(1.17E−07) −	1.75E+06(1.87E+06) −	1.16E+02(7.61E+01) −
f_{05}	5.95E−244(0.00E+00)	2.90E+06(1.49E+06) −	1.65E+04(8.00E+03) −	3.23E+09(8.63E+08) −	1.64E−69(4.63E−69) −	8.93E+09(5.11E+09) −	8.11E+04(1.55E+05) −
f_{06}	7.12E−218(0.00E+00)	1.69E−102(8.22E−102) −	7.39E−215(0.00E+00) −	1.73E−178(0.00E+00) −	0.00E+00(0.00E+00) +	1.85E−214(0.00E+00) +	3.78E−188(0.00E+00) −
f_{07}	0.00E+00(0.00E+00)	2.53E+01(5.61E+00) −	7.06E−01(1.62E−01) −	2.39E+02(3.45E+01) −	1.57E−04(1.12E−03) =	2.02E+02(1.20E+02) −	5.76E−02(6.03E−02) −
f_{08}	0.00E+00(0.00E+00)	2.18E+00(7.66E−01) −	1.09E+01(1.77E+00) −	2.53E+04(3.49E+03) −	0.00E+00(0.00E+00) =	1.91E+04(1.41E+04) −	8.01E+00(2.18E+00) −
f_{09}	8.63E+01(3.45E+00)	7.93E−02(1.81E−01) +	6.60E−02(4.55E−02) +	5.99E+02(1.06E+02) −	5.03E+01(4.94E+00) +	4.28E+02(2.68E+02) −	4.29E+01(3.14E+01) +
f_{10}	4.76E+01(2.11E+00)	1.09E+01(2.23E+00) +	3.51E+01(1.39E+00) +	7.22E+01(2.20E+00) −	8.79E+01(1.37E+00) −	3.39E+01(4.38E+00) +	2.82E+01(3.67E+00) +
f_{11}	5.53E−04(9.95E−04)	7.17E−01(3.44E−01) −	3.86E−03(6.19E−03) −	5.00E+07(5.83E+07) −	1.79E−01(4.54E−02) −	6.04E+07(1.31E+08) −	6.55E−01(6.76E−01) −
f_{12}	5.26E+00(8.44E−01)	2.99E+01(1.06E+01) −	1.81E−01(8.45E−02) +	1.44E+08(1.27E+08) −	5.93E+00(4.83E−01) −	1.10E+08(1.98E+08) −	3.38E+01(1.29E+01) −
f_{13}	9.46E−269(0.00E+00)	1.40E+00(6.86E−01) −	3.67E+00(1.29E+00) −	2.07E+03(4.91E+02) −	3.37E−05(7.66E−05) −	1.58E+03(9.23E+02) −	2.43E+01(1.01E+01) −
f_{14}	0.00E+00(0.00E+00)	3.66E+02(1.17E+02) −	4.39E+02(3.88E+01) −	2.98E+04(3.47E+03) −	2.36E−01(1.69E+00) −	2.42E+04(1.12E+04) −	2.98E+02(4.92E+01) −
f_{15}	9.24E+01(1.29E+00)	1.08E+02(1.99E+01) −	2.95E+02(7.38E+01) −	5.44E+05(2.54E+05) −	9.69E+01(7.53E−01) −	4.39E+05(6.41E+05) −	6.14E+02(1.61E+02) −
f_{16}	6.59E−189(0.00E+00)	2.99E+03(6.31E+02) −	8.23E+03(1.90E+03) −	2.81E+05(8.66E+04) −	2.06E−07(6.46E−07) −	1.49E+05(6.09E+04) −	4.34E+04(7.79E+03) −
f_{17}	9.60E−02(1.96E−02)	5.97E+00(5.37E−01) −	4.56E+00(3.46E−01) −	1.97E+01(1.50E+00) −	2.10E−01(4.13E−02) −	2.71E+01(4.24E+00) −	1.77E+00(2.96E−01) −
f_{18}	1.23E−03(1.54E−03)	4.92E−01(1.82E−03) −	4.85E−01(6.04E−03) −	5.00E−01(9.16E−05) −	3.31E−03(7.36E−04) −	5.00E−01(2.91E−05) −	4.91E−01(5.69E−03) −
f_{19}	1.81E+04(1.78E+03)	3.69E+04(8.02E+02) −	1.07E+04(6.62E+02) +	2.84E+04(1.17E+03) −	3.90E+04(2.34E+02) −	1.73E+04(2.96E+03) =	1.86E+04(1.38E+03) =
f_{20}	5.27E−203(0.00E+00)	1.17E+05(2.24E+04) −	3.38E+05(8.29E+04) −	1.05E+07(3.89E+06) −	2.03E−07(1.44E−06) −	6.24E+06(2.46E+06) −	1.47E+06(2.12E+05) −
f_{21}	2.03E−108(1.45E−107)	7.64E+00(1.19E+00) −	1.44E+01(2.10E+00) −	5.30E+01(8.33E+00) −	1.89E−10(8.92E−10) −	9.15E+01(2.05E+00) −	5.12E+01(4.14E+00) −
f_{22}	2.33E−132(1.66E−131)	5.86E−02(1.58E−01) −	5.20E−01(1.10E−01) −	4.35E+09(3.10E+10) −	2.26E−42(2.03E−42) −	1.45E+02(5.33E+01) −	6.24E+00(2.95E+00) −
f_{23}	6.06E−240(0.00E+00)	2.46E−16(6.69E−17) −	2.29E+00(8.31E−01) −	2.51E+04(3.59E+03) −	3.07E−73(5.53E−73) −	2.20E+04(1.56E+04) −	7.43E−01(8.30E−01) −
f_{24}	1.04E−01(1.32E−01)	2.50E−16(7.18E−17) +	2.08E+00(7.84E−01) −	2.54E+04(3.34E+03) −	7.84E+00(7.41E−01) −	2.05E+04(1.60E+04) −	8.59E−01(1.26E+00) −
f_{25}	4.16E+02(5.71E+01)	3.32E+02(5.89E+01) +	1.02E+02(4.88E+01) +	1.59E+03(1.85E+02) −	1.88E+03(1.42E+02) −	5.88E+02(8.42E+01) −	4.81E+02(6.49E+01) −
f_{26}	0.00E+00(0.00E+00)	2.41E+02(1.38E+03) −	3.44E−09(1.73E−08) −	3.82E+51(2.57E+52) −	0.00E+00(0.00E+00) =	2.05E+78(1.41E+79) −	8.95E+06(2.83E+07) −
f_{27}	1.79E−248(0.00E+00)	8.25E−15(3.90E−15) −	8.39E−01(2.82E−01) −	5.34E+03(9.82E+02) −	3.45E−71(1.08E−70) −	1.42E+04(8.95E+03) −	2.73E+01(3.92E+01) −
f_{28}	0.00E+00(0.00E+00)	1.20E−04(8.56E−04) −	6.13E−03(3.93E−03) −	4.63E+04(1.11E+04) −	4.03E−114(1.16E−113) −	3.51E+05(3.37E+05) −	4.15E−01(4.88E−01) −
f_{29}	1.96E−05(1.27E−04)	3.13E+00(1.05E+00) −	0.00E+00(0.00E+00) +	1.17E+01(4.30E+00) −	6.73E+01(2.49E+00) −	0.00E+00(0.00E+00) +	3.85E+01(8.99E+00) −
f_{30}	1.90E−37(1.36E−36)	2.44E+02(2.57E+01) −	3.60E+02(6.80E+01) −	1.99E+03(3.88E+02) −	5.72E+02(9.15E+02) −	2.12E+03(2.09E+02) −	6.39E+02(9.15E+01) −
+/= /−		4/1/25	6/0/24	0/0/30	2/3/25	3/2/25	2/1/27

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	2.06E+01(5.53E−02) −	7.23E+00(1.10E+00) =	1.80E+01(4.84E−01) =	1.06E+01(7.30E+00) =	4.61E−09(4.74E−09) =	1.53E−15(2.49E−15) +
f_{02}	1.55E+02(8.03E+01) −	1.35E+02(2.29E+01) −	4.20E+02(3.21E+01) −	1.66E+02(3.06E+01) −	2.71E−09(2.83E−09) −	3.32E−13(2.76E−13) −
f_{03}	1.39E+10(1.42E+10) −	2.41E+07(5.73E+06) −	9.40E+10(1.29E+10) −	4.21E+02(4.34E+02) −	3.10E−09(6.29E−09) −	1.95E−14(1.16E−15) −
f_{04}	3.06E+05(3.15E+05) −	2.00E+01(7.06E+00) −	3.15E+05(5.41E+04) −	1.08E+03(1.45E+03) −	6.67E−01(1.13E−07) −	9.99E−01(6.61E−04) −
f_{05}	5.72E+05(1.12E+06) −	1.57E+07(4.29E+06) −	5.65E+09(2.28E+09) −	9.98E+07(3.79E+07) −	1.23E−11(2.25E−11) −	1.89E−14(1.05E−15) −
f_{06}	1.40E−88(9.98E−88) −	1.10E−59(4.27E−59) −	3.79E−21(1.58E−20) −	0.00E+00(0.00E+00) +	3.38E−189(0.00E+00) −	6.74E−23(4.24E−22) −
f_{07}	1.40E+01(1.11E+01) −	7.81E−01(7.64E−02) −	7.61E+01(1.24E+01) −	3.46E−03(6.85E−03) −	2.18E−18(1.55E−17) =	1.18E−14(5.24E−15) −
f_{08}	1.97E+03(2.24E+03) −	2.11E+01(3.24E+00) −	9.32E+03(1.57E+03) −	9.41E+00(1.44E+00) −	5.71E−15(9.56E−15) −	1.16E−14(2.37E−15) −
f_{09}	1.40E+02(2.80E+01) −	9.30E+01(2.18E+01) =	5.08E+02(4.47E+01) −	8.95E+03(5.46E+03) −	1.58E+01(2.27E+00) +	9.99E+01(6.62E−02) −
f_{10}	8.16E+01(1.16E+00) −	4.68E+01(3.32E+00) =	7.88E+01(1.08E+00) −	7.83E+01(2.96E+00) −	3.52E+01(3.69E+00) +	7.60E+01(1.09E+00) −
f_{11}	4.33E+07(3.72E+07) −	3.72E+00(8.68E−01) −	3.44E+04(5.78E+04) −	1.59E+01(3.75E+00) −	1.07E−02(2.57E−03) −	1.00E+00(7.64E−02) −
f_{12}	9.91E+07(8.51E+07) −	9.31E+01(3.80E+01) −	1.88E+06(1.14E+06) −	1.49E+02(1.80E+01) −	1.01E+00(2.10E−01) +	1.25E+01(4.34E−01) −
f_{13}	1.17E+02(1.26E+02) −	3.79E+00(6.77E−01) −	6.33E+03(7.71E+02) −	1.22E+04(5.83E+03) −	9.55E−15(1.96E−14) −	1.70E−14(8.36E−16) −
f_{14}	1.99E+03(1.18E+03) −	1.37E+03(2.84E+02) −	1.03E+04(1.24E+03) −	1.28E+03(2.35E+02) −	6.24E−14(1.69E−13) −	0.00E+00(0.00E+00) =
f_{15}	2.80E+05(3.01E+05) −	1.14E+02(1.42E+01) −	1.77E+05(3.21E+04) −	5.94E+02(1.25E+03) −	9.57E+01(1.70E−01) −	9.89E+01(3.60E−02) −
f_{16}	1.46E+05(3.70E+04) −	7.85E+02(1.15E+02) −	2.19E+05(1.29E+05) −	1.21E+04(6.18E+03) −	1.03E−14(2.14E−14) −	1.78E−14(9.56E−16) −
f_{17}	5.52E+00(2.31E+00) −	2.45E+00(1.84E−01) −	9.54E+00(6.43E−01) −	3.44E+00(3.62E−01) −	1.28E−09(1.13E−09) +	3.22E−01(2.19E−02) −
f_{18}	4.94E−01(7.80E−03) −	3.26E−01(2.13E−02) −	4.92E−01(2.03E−03) −	3.45E−01(4.37E−02) −	0.00E+00(0.00E+00) +	6.22E−03(1.89E−07) −
f_{19}	3.42E+04(5.00E+02) −	2.12E+04(1.44E+03) −	3.69E+04(5.70E+02) −	3.71E+04(1.88E+02) −	2.45E+04(1.39E+03) −	3.46E+04(6.36E+02) −
f_{20}	5.42E+06(1.64E+06) −	2.99E+04(5.20E+03) −	7.34E+06(3.70E+06) −	3.90E+05(1.51E+05) −	2.74E−13(5.83E−13) −	1.81E−14(8.90E−16) −
f_{21}	8.02E+01(4.36E+00) −	7.28E+00(9.95E−01) −	2.21E+01(2.31E+00) −	1.95E+01(2.58E+00) −	1.11E−08(1.05E−08) −	1.17E−11(5.46E−13) −
f_{22}	4.57E−02(8.76E−02) −	7.88E+00(1.32E+00) −	9.43E+45(6.67E+46) −	1.31E+73(9.29E+73) −	3.43E−08(3.42E−08) −	1.34E+48(3.71E+48) −
f_{23}	1.73E+03(2.12E+03) −	2.34E+00(5.51E−01) −	9.16E+03(1.26E+03) −	1.76E−07(2.51E−08) −	3.05E−16(4.73E−16) −	1.78E−14(7.31E−16) −
f_{24}	2.25E+03(2.25E+03) −	2.50E+00(6.41E−01) −	9.25E+03(1.33E+03) −	1.79E−07(2.53E−08) +	4.01E−01(8.39E−02) −	2.28E+01(8.54E−01) −
f_{25}	2.45E+03(1.04E+02) −	7.09E+02(6.87E+01) −	2.62E+03(7.12E+01) −	6.72E+02(7.47E+01) −	6.72E+02(7.61E+01) −	1.80E+03(4.41E+01) −
f_{26}	6.95E+48(4.96E+49) −	1.09E−06(9.30E−07) −	1.76E+84(1.26E+85) −	2.33E+75(1.62E+76) −	3.62E−25(5.80E−25) −	3.17E+85(1.12E+86) −
f_{27}	5.16E+02(5.70E+02) −	9.58E+00(2.61E+00) −	1.22E+04(1.25E+03) −	1.01E+03(6.86E+02) −	2.12E−14(4.89E−14) −	1.77E−14(9.34E−16) −
f_{28}	6.78E+04(5.33E+04) −	3.07E−02(1.44E−02) −	7.60E+04(1.49E+04) −	1.82E−11(8.06E−11) −	9.62E−29(4.99E−28) −	1.82E−17(1.05E−18) −
f_{29}	2.94E+01(2.40E+00) −	7.01E+01(3.31E+00) −	8.93E+00(1.49E+00) −	3.03E+01(1.61E+00) −	0.00E+00(0.00E+00) +	1.17E+01(9.99E+00) −
f_{30}	2.90E+02(7.10E+01) −	7.28E+10(2.99E+11) −	1.51E+14(8.90E+13) −	6.18E+04(3.57E+04) −	5.89E−14(1.05E−13) −	1.63E−14(1.01E−15) −
+/= /−	0/0/30	0/3/27	0/1/29	2/1/27	6/2/22	1/1/28

functions (f_1–f_3). For the function f_1, LSA and SSA have a similar performance and are superior to other competing algorithms, and DSOS-1 is ranked third among the thirteen algorithms. For the function f_2, the performance of LSA is the best among the competitors and DSOS-1 is ranked second. For the function f_3, LSA and CSA show a similar performance, are superior to other competing algorithms, and DSOS-1 is ranked third.

The performance of algorithms on problems of the unimodal type in CEC 2014 and CEC 2017 is similar. In all of the multimodal test functions (f_4–f_{10}), the DSOS-1 algorithm is superior to the HSS, SCA, CGSA and BOA algorithms. For the function f_{10}, GSA, MFO, and MS performed better than DSOS-1. For the functions f_7 and f_{10}, LSA and SSA showed a similar performance, are superior to other competing algorithms, and DSOS-1 is ranked third. For the functions f_8 and f_{10}, CSA performed better than DSOS-1. Moreover, BSA and GWO performed better than DSOS-1 in multimodal problems.

In all of the hybrid test functions (f_{11}–f_{20}), the DSOS-1 algorithm was superior to the GSA, BSA, HSS, MFO, SCA, CGSA, SSA, MS and BOA algorithms (see Table 15). That is, the DSOS-1 algorithm is more successful than nine of the twelve competing algorithms, on ten hybrid functions. From Table 15, it can be seen that DSOS-1 outperformed GWO, LSA, and CSA on 8, 4, and 8 hybrid test functions, respectively. Furthermore, DSOS-1 exhibits similar performance to GWO, LSA, and CSA on 2, 6, and 2 hybrid

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 12
Comparison of mean error and standard deviation in objective function value for 30 dimension of CEC2014 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	8.38E+06(4.57E+06)	2.96E+08(6.74E+07) −	4.79E+07(1.28E+07) −	1.69E+08(5.59E+07) −	7.68E+07(4.31E+07) −	7.46E+07(7.23E+07) −	7.87E+06(5.08E+06) =
f_{02}	2.15E+04(1.26E+04)	1.55E+10(2.90E+09) −	2.01E+07(1.21E+07) −	1.46E+10(4.59E+09) −	1.65E+09(1.64E+09) −	7.58E+09(5.25E+09) −	1.05E+04(8.95E+03) +
f_{03}	6.15E+03(3.22E+03)	8.73E+04(8.41E+03) −	1.47E+03(1.28E+03) +	2.36E+05(6.17E+04) −	4.62E+04(1.11E+04) −	9.11E+04(3.86E+04) −	1.19E+04(1.02E+04) −
f_{04}	1.23E+02(3.57E+01)	1.45E+03(2.54E+02) −	1.76E+02(1.84E+01) −	1.47E+03(5.64E+02) −	2.37E+02(6.67E+01) −	7.28E+02(6.21E+02) −	1.13E+02(4.27E+01) =
f_{05}	2.08E+01(1.03E−01)	2.00E+01(4.65E−04) +	2.06E+01(5.04E−02) +	2.08E+01(1.05E−01) =	2.10E+01(6.83E−02) −	2.03E+01(1.81E−01) +	2.10E+01(2.03E−01) −
f_{06}	1.72E+01(2.98E+00)	3.04E+01(2.24E+00) −	2.42E+01(1.80E+00) −	3.60E+01(2.74E+00) −	1.49E+01(2.95E+00) +	2.18E+01(3.88E+00) −	2.24E+01(4.45E+00) −
f_{07}	6.89E−02(4.58E−02)	1.72E+02(2.99E+01) −	1.15E+00(6.47E−02) −	1.82E+02(4.92E+01) −	1.67E+01(1.22E+01) −	7.68E+01(5.20E+01) −	1.98E−02(2.36E−02) +
f_{08}	1.17E+02(2.37E+01)	1.45E+02(1.14E+01) −	5.48E+01(6.59E+00) +	2.44E+02(2.82E+01) −	8.66E+01(1.97E+01) +	1.37E+02(3.73E+01) −	1.46E+02(3.52E+01) −
f_{09}	1.85E+02(4.00E+01)	1.69E+02(1.69E+01) +	1.42E+02(1.67E+01) +	3.00E+02(4.02E+01) −	1.10E+02(4.13E+01) +	1.97E+02(4.83E+01) =	2.18E+02(5.04E+01) −
f_{10}	3.06E+03(5.07E+02)	3.89E+03(5.70E+02) −	1.43E+03(2.15E+02) +	6.21E+03(5.20E+02) −	2.33E+03(4.88E+02) +	3.44E+03(1.01E+03) −	2.68E+03(5.38E+02) +
f_{11}	4.81E+03(7.52E+02)	4.53E+03(4.48E+02) +	4.56E+03(2.37E+02) +	6.30E+03(5.19E+02) −	3.50E+03(1.44E+03) +	4.18E+03(8.74E+02) +	4.08E+03(6.65E+02) +
f_{12}	1.37E+00(3.02E−01)	2.05E−02(1.04E−02) +	9.63E−01(1.33E−01) +	1.83E+00(4.77E−01) −	2.74E+00(1.01E+00) −	4.06E−01(2.21E−01) +	7.84E−01(3.91E−01) +
f_{13}	4.91E−01(1.12E−01)	3.80E+00(3.27E−01) −	3.46E−01(6.62E−02) +	3.43E+00(6.52E−01) −	4.88E−01(1.02E−01) =	1.33E+00(1.00E+00) −	5.05E−01(1.29E−01) =
f_{14}	3.38E−01(1.25E−01)	7.28E+01(1.63E+01) −	2.82E−01(3.93E−02) +	8.03E+01(1.34E+01) −	2.71E+00(5.39E+00) −	1.66E+01(1.61E+01) −	3.02E−01(6.00E−02) =
f_{15}	2.74E+01(6.98E+00)	1.54E+03(1.00E+03) −	1.61E+01(1.88E+00) +	6.01E+04(5.71E+04) −	7.32E+01(1.12E+02) −	1.39E+05(3.73E+05) −	1.46E+01(5.29E+00) +
f_{16}	1.22E+01(4.63E−01)	1.37E+01(2.61E−01) −	1.19E+01(2.28E−01) +	1.30E+01(3.39E−01) −	1.19E+01(6.37E−01) +	1.26E+01(5.74E−01) −	1.21E+01(5.40E−01) =
f_{17}	7.00E+05(6.76E+05)	2.41E+07(7.36E+06) −	3.01E+06(1.47E+06) −	1.16E+07(8.12E+06) −	2.77E+06(4.10E+06) −	3.18E+06(2.88E+06) −	9.83E+05(7.12E+05) −
f_{18}	7.79E+03(8.16E+03)	4.38E+02(2.99E+02) +	2.32E+05(3.44E+05) −	4.05E+07(6.56E+07) −	1.07E+07(2.03E+07) −	2.02E+07(8.55E+07) −	3.19E+03(3.69E+03) +
f_{19}	2.76E+01(3.27E+01)	1.80E+02(2.75E+01) −	1.99E+01(6.77E+00) −	1.59E+02(6.71E+01) −	4.55E+01(2.86E+01) −	5.30E+01(5.34E+01) −	2.25E+01(2.73E+01) =
f_{20}	1.28E+04(6.85E+03)	2.26E+05(1.09E+05) −	5.85E+03(4.15E+03) +	2.43E+05(2.60E+05) −	2.93E+04(1.85E+04) −	7.36E+04(9.71E+04) −	2.07E+04(1.33E+04) −
f_{21}	1.64E+05(1.41E+05)	1.12E+07(4.16E+06) −	3.87E+05(2.64E+05) −	8.48E+06(6.97E+06) −	1.48E+06(2.64E+06) −	1.21E+06(1.52E+06) −	4.56E+05(4.02E+05) −
f_{22}	4.56E+02(1.55E+02)	1.17E+03(2.95E+02) −	5.34E+02(1.32E+02) −	1.20E+03(3.09E+02) −	4.43E+02(1.98E+02) =	8.00E+02(2.37E+02) −	6.53E+02(2.57E+02) −
f_{23}	2.00E+02(0.00E+00)	2.53E+02(8.74E+01) −	3.19E+02(1.33E+00) −	4.52E+02(4.31E+01) −	3.38E+02(1.16E+01) −	3.54E+02(2.42E+01) −	3.15E+02(4.36E−02) −
f_{24}	2.00E+02(2.80E−04)	2.16E+02(5.86E+00) −	2.35E+02(4.04E+00) −	2.99E+02(1.42E+01) −	2.00E+02(1.59E−02) −	2.77E+02(2.77E+01) −	2.47E+02(7.59E+00) −
f_{25}	2.00E+02(0.00E+00)	2.05E+02(2.88E+00) −	2.14E+02(1.63E+00) −	2.47E+02(1.41E+01) −	2.13E+02(4.00E+00) −	2.15E+02(6.53E+00) −	2.21E+02(8.10E+00) −
f_{26}	1.01E+02(1.27E−01)	1.91E+02(2.47E+01) −	1.00E+02(5.75E−02) +	1.30E+02(4.62E+01) −	1.22E+02(4.14E+01) −	1.02E+02(1.06E+00) −	1.46E+02(5.01E+01) −
f_{27}	6.20E+02(2.09E+02)	1.89E+03(2.60E+02) −	5.02E+02(6.37E+01) =	1.00E+03(2.69E+02) −	6.91E+02(1.20E+02) =	8.67E+02(2.18E+02) −	7.61E+02(2.54E+02) −
f_{28}	1.16E+03(3.09E+02)	2.53E+03(9.24E+02) −	1.08E+03(2.49E+01) =	2.67E+03(6.85E+02) −	1.17E+03(2.88E+02) =	1.02E+03(1.09E+02) +	3.39E+03(7.49E+02) −
f_{29}	3.11E+06(4.26E+06)	2.06E+07(7.63E+07) +	2.69E+04(1.42E+04) −	2.22E+07(1.22E+07) −	9.75E+05(2.97E+06) −	1.74E+06(2.93E+06) −	1.40E+03(6.08E+02) +
f_{30}	6.68E+03(6.19E+03)	1.60E+06(5.74E+05) −	1.70E+04(4.84E+03) −	6.84E+05(3.33E+05) −	8.32E+04(5.76E+04) −	3.11E+04(2.69E+04) −	4.46E+03(2.16E+03) +
+/= /−		6/0/24	13/2/15	0/1/29	6/4/20	4/1/25	9/6/15

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	4.11E+08(1.20E+08) −	4.76E+07(1.81E+07) −	2.13E+09(5.48E+08) −	2.72E+07(1.14E+07) −	2.22E+08(8.19E+07) −	1.28E+09(2.85E+08) −
f_{02}	2.59E+10(4.91E+09) −	1.59E+08(6.52E+07) −	9.69E+10(1.18E+10) −	1.04E+04(1.09E+04) +	9.88E+09(3.43E+09) −	6.19E+10(8.24E+09) −
f_{03}	6.27E+04(1.26E+04) −	4.24E+04(8.47E+03) −	2.57E+05(2.47E+05) −	7.11E+04(2.07E+04) −	3.43E+04(6.42E+03) −	7.99E+04(4.58E+03) −
f_{04}	1.78E+03(5.44E+02) −	2.59E+02(6.74E+01) −	2.25E+04(4.39E+03) −	1.42E+02(4.04E+01) −	7.31E+02(2.26E+02) −	1.46E+04(1.95E+03) −
f_{05}	2.10E+01(5.29E−02) −	2.10E+01(8.65E−02) −	2.11E+01(5.25E−02) −	2.01E+01(1.13E−01) +	2.00E+01(3.56E−02) +	2.11E+01(5.18E−02) −
f_{06}	3.73E+01(2.48E+00) −	2.71E+01(3.43E+00) −	4.58E+01(1.64E+00) −	2.13E+01(3.88E+00) −	2.95E+01(2.85E+00) −	3.86E+01(1.59E+00) −
f_{07}	2.23E+02(4.35E+01) −	2.76E+00(8.13E−01) −	9.42E+02(1.03E+02) −	1.17E−02(1.15E−02) +	8.98E+01(3.55E+01) −	7.47E+02(7.16E+01) −
f_{08}	2.72E+02(2.43E+01) −	1.29E+02(2.04E+01) −	3.95E+02(2.39E+01) −	1.52E+02(3.45E+01) −	1.36E+02(1.55E+01) −	3.22E+02(1.56E+01) −
f_{09}	3.03E+02(2.20E+01) −	1.42E+02(2.40E+01) +	4.39E+02(2.78E+01) −	1.61E+02(3.89E+01) +	1.98E+02(1.78E+01) =	3.48E+02(1.91E+01) −
f_{10}	6.70E+03(5.28E+02) −	3.49E+03(6.56E+02) −	8.36E+03(4.59E+02) −	3.69E+03(7.24E+02) −	3.13E+03(5.15E+02) =	7.67E+03(3.70E+02) −
f_{11}	7.59E+03(3.39E+02) −	3.82E+03(6.47E+02) +	8.73E+03(4.03E+02) −	4.06E+03(7.08E+02) +	4.60E+03(6.65E+02) =	7.94E+03(3.33E+02) −
f_{12}	3.10E+00(4.62E−01) −	8.17E−01(3.35E−01) +	4.68E+00(6.50E−01) −	6.90E−01(3.74E−01) +	6.63E−01(2.48E−01) +	3.58E+00(4.95E−01) −
f_{13}	3.77E+00(4.47E−01) −	4.87E−01(9.39E−02) =	9.76E+00(8.32E−01) −	5.02E−01(1.35E−01) =	1.80E+00(9.93E−01) −	8.47E+00(4.70E−01) −
f_{14}	7.08E+01(1.31E+01) −	3.09E−01(8.68E−02) =	3.54E+02(5.27E+01) −	4.38E−01(1.97E−01) −	3.14E+01(1.28E+01) −	2.83E+02(3.23E+01) −
f_{15}	1.72E+04(1.24E+04) −	2.45E+01(7.48E+00) =	2.86E+06(1.51E+06) −	1.22E+01(3.73E+00) +	1.27E+03(1.27E+03) −	2.80E+05(9.96E+04) −
f_{16}	1.32E+01(2.60E−01) −	1.22E+01(4.39E−01) =	1.38E+01(2.39E−01) −	1.24E+01(4.81E−01) −	1.26E+01(4.24E−01) −	1.33E+01(1.81E−01) −
f_{17}	1.39E+07(6.24E+06) −	7.67E+05(6.75E+05) =	2.65E+08(1.08E+08) −	1.45E+06(1.01E+06) −	1.55E+07(1.02E+07) −	9.71E+07(5.58E+07) −
f_{18}	2.89E+08(1.43E+08) −	1.32E+03(1.17E+03) +	8.90E+09(2.60E+09) −	1.01E+04(8.86E+03) =	9.19E+06(1.14E+07) −	3.28E+09(1.57E+09) −
f_{19}	1.29E+02(2.89E+01) −	3.03E+01(2.07E+01) −	8.03E+02(2.20E+02) −	2.04E+01(1.72E+01) −	9.51E+01(2.81E+01) −	4.85E+02(6.53E+01) −
f_{20}	4.64E+04(1.95E+04) −	1.35E+04(8.49E+03) =	6.11E+06(7.59E+06) −	3.31E+04(1.61E+04) −	3.74E+04(1.75E+04) −	3.47E+05(2.96E+05) −
f_{21}	3.80E+06(2.16E+06) −	1.84E+05(1.93E+05) =	1.48E+08(8.35E+07) −	4.75E+05(3.41E+05) −	4.05E+06(3.46E+06) −	2.63E+07(2.15E+07) −
f_{22}	1.09E+03(1.53E+02) −	6.08E+02(1.84E+02) −	3.41E+04(7.26E+04) −	6.16E+02(1.93E+02) −	8.16E+02(2.63E+02) −	1.20E+04(1.29E+04) −
f_{23}	4.10E+02(2.31E+01) −	3.40E+02(7.26E+00) −	1.55E+03(3.05E+02) −	3.37E+02(1.21E+01) −	2.00E+02(1.28E−06) −	2.00E+02(0.00E+00) =
f_{24}	2.18E+02(1.71E+01) −	2.25E+02(5.27E+00) −	4.03E+02(4.54E+01) −	2.37E+02(7.19E+00) −	2.00E+02(6.78E−05) +	2.00E+02(0.00E+00) +
f_{25}	2.40E+02(1.12E+01) −	2.08E+02(2.64E+00) −	2.44E+02(1.44E+01) −	2.14E+02(4.39E+00) −	2.00E+02(1.97E−08) −	2.00E+02(0.00E+00) =
f_{26}	1.04E+02(3.53E−01) −	1.01E+02(1.53E+00) =	2.18E+02(2.21E+01) −	1.01E+02(1.36E−01) =	1.77E+02(4.24E+01) −	1.71E+02(3.76E+01) −
f_{27}	1.04E+03(3.23E+02) −	6.21E+02(3.01E+02) =	1.91E+03(2.96E+02) −	8.29E+02(1.95E+02) −	2.00E+02(2.36E−07) +	7.27E+02(1.23E+02) −
f_{28}	2.72E+03(5.34E+02) −	3.76E+03(7.21E+02) −	8.79E+03(1.03E+03) −	1.27E+03(3.26E+02) =	2.00E+02(4.93E−07) +	2.18E+03(1.14E+03) −
f_{29}	3.36E+07(1.18E+07) −	8.70E+06(3.26E+07) −	9.15E+08(2.31E+08) −	4.53E+06(8.85E+06) −	2.02E+02(1.93E+00) +	2.00E+02(0.00E+00) +
f_{30}	4.70E+05(1.64E+05) −	8.25E+04(5.28E+04) −	1.06E+07(5.38E+06) −	4.55E+04(3.39E+04) −	2.00E+02(1.01E−01) +	2.00E+02(0.00E+00) +
+/= /−	0/0/30	4/9/17	0/0/30	7/4/19	7/3/20	3/2/25

functions, and has an outstanding performance against all other competitors on 30-dimensional hybrid functions.

For all of the composition functions (f_{21}–f_{30}), DSOS-1 performs a better convergence and obtains better statistical results than the GSA, HSS, SCA, CGSA, MS and BOA algorithms. From Table 15, it can be seen that DSOS-1 outperformed BSA, GWO, MFO, LSA, CSA, and SSA on 5, 6, 8, 7, 9, and 6 composition functions, respectively.

For the composition functions, DSOS-1 performs significantly better than its opponents. According to the pair-wise Wilcoxon rank sum test results (see the last line in Table 15), DSOS-1 performs a better convergence and obtains significantly better statistical results than the others on 30-dimensional test functions in the CEC 2017 benchmark suite. The numerical performance of the algorithms on the 50 and 100 dimensional test functions and the results of the pair-wise comparisons are given in Tables 16 and 17.

When the statistical and numerical results given in Tables 16 and 17 are investigated, it is seen that they are similar to the algorithm performance given in Table 15. According to the pair-wise Wilcoxon rank sum test results, the DSOS-1 algorithm is superior to all other algorithms in 50-dimensional test functions (see the last line in Table 16). From Table 17, it can be seen that DSOS-1 and BSA outperformed the other algorithms on 100-dimensional test functions.

In summary, this study obtained similar results to the CEC 2014 and CEC 2017 benchmark suites.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 13
Comparison of mean error and standard deviation in objective function value for 50 dimension of CEC2014 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	1.19E+07(5.64E+06)	4.59E+08(2.88E+08) −	3.47E+07(8.57E+06) −	4.65E+08(1.95E+08) −	1.14E+08(6.23E+07) −	2.74E+08(1.88E+08) −	8.19E+06(3.51E+06) +
f_{02}	6.11E+05(1.23E+06)	3.03E+10(4.61E+09) −	4.09E+07(1.75E+07) −	3.69E+10(9.71E+09) −	8.27E+09(4.13E+09) −	4.33E+10(1.78E+10) −	8.44E+04(3.70E+05) +
f_{03}	1.86E+04(7.31E+03)	1.57E+05(1.41E+04) −	8.93E+03(2.53E+03) +	3.25E+05(7.81E+04) −	7.62E+04(1.36E+04) −	1.68E+05(8.88E+04) −	3.38E+04(1.35E+04) −
f_{04}	1.96E+02(5.32E+01)	4.29E+03(9.70E+02) −	2.65E+02(3.10E+01) −	5.35E+03(2.06E+03) −	1.01E+03(4.62E+02) −	3.88E+03(2.80E+03) −	1.47E+02(5.35E+01) +
f_{05}	2.09E+01(8.40E−02)	2.00E+01(1.16E−04) +	2.07E+01(3.88E−02) +	2.10E+01(7.17E−02) −	2.12E+01(3.82E−02) −	2.03E+01(1.71E−01) +	2.12E+01(9.55E−02) −
f_{06}	3.92E+01(4.91E+00)	5.53E+01(3.09E+00) −	4.54E+01(1.83E+00) −	6.82E+01(3.26E+00) −	3.29E+01(4.01E+00) +	4.43E+01(5.42E+00) −	4.67E+01(7.24E+00) −
f_{07}	4.84E−01(2.17E−01)	2.96E+02(4.59E+01) −	1.35E+00(1.71E−01) −	5.23E+02(8.91E+01) −	6.40E+01(3.41E+01) −	3.55E+02(1.81E+02) −	9.70E−02(1.31E−01) +
f_{08}	2.37E+02(3.11E+01)	2.81E+02(1.68E+01) −	9.98E+01(1.09E+01) +	4.82E+02(4.40E+01) −	2.06E+02(3.43E+01) +	2.97E+02(6.73E+01) −	3.06E+02(5.77E+01) −
f_{09}	2.98E+02(7.62E+01)	3.61E+02(2.34E+01) −	2.70E+02(2.17E+01) =	6.48E+02(6.37E+01) −	2.17E+02(3.04E+01) +	4.66E+02(7.83E+01) −	4.69E+02(9.30E+01) −
f_{10}	6.44E+03(7.90E+02)	7.75E+03(7.39E+02) −	3.01E+03(3.27E+02) +	1.25E+04(6.72E+02) −	6.22E+03(1.20E+03) +	7.07E+03(1.46E+03) −	5.34E+03(7.04E+02) +
f_{11}	9.14E+03(1.46E+03)	8.57E+03(9.53E+02) +	8.67E+03(3.47E+02) =	1.24E+04(7.56E+02) −	6.94E+03(2.32E+03) +	7.47E+03(1.13E+03) +	7.31E+03(9.15E+02) +
f_{12}	1.58E+00(3.45E−01)	1.34E−02(5.25E−03) +	1.05E+00(1.11E−01) +	2.46E+00(4.96E−01) −	3.36E+00(1.23E+00) −	4.93E−01(2.17E−01) +	9.31E−01(3.69E−01) +
f_{13}	6.21E−01(1.04E−01)	3.63E+00(2.46E−01) −	4.53E−01(5.28E−02) +	4.95E+00(4.81E−01) −	8.17E−01(3.38E−01) −	3.66E+00(1.46E+00) −	6.22E−01(1.41E−01) =
f_{14}	3.99E−01(1.44E−01)	6.51E+01(1.28E+01) −	3.37E−01(2.57E−02) +	1.36E+02(2.30E+01) −	1.77E+01(1.23E+01) −	9.99E+01(4.64E+01) −	3.38E−01(7.65E−02) +
f_{15}	8.34E+01(2.20E+01)	2.06E+04(1.22E+04) −	3.52E+01(5.23E+00) +	6.37E+05(7.93E+05) −	2.35E+03(3.00E+03) −	1.66E+06(5.27E+06) −	3.30E+01(9.11E+00) +
f_{16}	2.12E+01(5.78E−01)	2.27E+01(3.19E−01) −	2.07E+01(3.41E−01) +	2.27E+01(3.31E−01) −	2.11E+01(7.15E−01) =	2.21E+01(5.41E−01) −	2.13E+01(7.44E−01) =
f_{17}	1.72E+06(1.28E+06)	4.01E+07(2.71E+07) −	6.93E+06(2.52E+06) −	5.84E+07(3.76E+07) −	7.42E+06(7.10E+06) −	1.40E+07(1.50E+07) −	1.85E+06(1.15E+06) =
f_{18}	2.59E+03(1.68E+03)	7.24E+08(1.07E+09) −	5.64E+03(7.82E+03) −	4.17E+08(3.74E+08) −	1.09E+08(1.83E+08) −	1.68E+08(2.98E+08) −	1.49E+03(1.33E+03) +
f_{19}	6.48E+01(3.26E+01)	2.02E+02(4.19E+01) −	6.70E+01(1.67E+01) =	2.31E+02(8.29E+01) −	9.98E+01(3.02E+01) −	1.63E+02(8.99E+01) −	6.56E+01(2.55E+01) =
f_{20}	1.52E+04(5.33E+03)	1.27E+05(4.37E+04) −	1.33E+04(4.83E+03) =	1.03E+06(1.25E+06) −	2.97E+04(1.30E+04) −	1.46E+05(9.71E+04) −	3.55E+04(1.63E+04) −
f_{21}	6.75E+05(3.74E+05)	4.20E+06(1.25E+06) −	2.13E+06(9.87E+05) −	2.02E+07(1.08E+07) −	5.07E+06(3.56E+06) −	8.76E+06(8.63E+06) −	1.76E+06(1.33E+06) −
f_{22}	1.19E+03(3.86E+02)	2.62E+03(2.17E+03) −	1.40E+03(1.74E+02) −	3.06E+03(9.73E+02) −	1.00E+03(4.12E+02) +	1.70E+03(4.57E+02) −	1.43E+03(3.72E+02) −
f_{23}	2.00E+02(0.00E+00)	2.39E+02(1.23E+02) −	3.54E+02(4.43E+00) −	5.89E+02(5.80E+01) −	4.49E+02(5.26E+01) −	5.07E+02(1.08E+02) −	3.44E+02(3.02E−02) −
f_{24}	2.00E+02(1.01E−04)	2.45E+02(1.23E+01) −	2.75E+02(2.44E+00) −	4.32E+02(2.72E+01) −	2.00E+02(2.59E−02) −	3.77E+02(5.19E+01) −	3.00E+02(9.23E+00) −
f_{25}	2.00E+02(0.00E+00)	2.04E+02(4.37E+00) −	2.29E+02(3.05E+00) −	2.90E+02(1.89E+01) −	2.26E+02(1.50E+01) −	2.35E+02(1.53E+01) −	2.44E+02(1.59E+01) −
f_{26}	1.81E+02(3.94E+01)	2.00E+02(7.52E−02) −	1.00E+02(5.38E−02) +	2.03E+02(5.57E+01) −	1.84E+02(4.81E+01) −	1.25E+02(7.71E+01) +	1.76E+02(4.29E+01) −
f_{27}	1.37E+03(1.24E+02)	3.31E+03(6.98E+02) −	1.43E+03(1.52E+02) −	2.13E+03(1.14E+02) −	1.17E+03(1.24E+02) +	1.58E+03(1.30E+02) −	1.53E+03(1.71E+02) −
f_{28}	2.36E+03(8.66E+02)	5.67E+03(1.07E+03) −	1.74E+03(8.81E+01) +	6.75E+03(1.38E+03) −	2.53E+03(6.65E+02) =	1.84E+03(5.41E+02) +	7.67E+03(1.38E+03) −
f_{29}	2.28E+07(1.94E+07)	2.00E+02(5.94E−02) +	1.64E+05(1.23E+05) =	1.10E+08(7.67E+07) −	6.58E+06(1.15E+07) =	3.35E+07(1.12E+07) =	2.05E+03(6.88E+02) +
f_{30}	1.52E+04(6.56E+03)	4.07E+06(4.23E+06) −	2.37E+04(5.68E+03) −	4.03E+06(1.94E+06) −	2.22E+05(1.28E+05) −	1.39E+05(1.52E+05) −	1.98E+04(5.60E+03) −
+/= /−		4/0/26	11/5/14	0/0/30	7/3/20	5/1/24	11/4/15

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	9.82E+08(2.09E+08) −	9.07E+07(3.69E+07) −	9.08E+09(2.72E+09) −	2.93E+07(9.98E+06) −	2.33E+08(7.73E+07) −	4.33E+09(1.17E+09) −
f_{02}	6.72E+10(8.53E+09) −	5.40E+08(1.86E+08) −	1.96E+11(1.48E+10) −	6.29E+03(7.48E+03) +	2.48E+10(7.43E+09) −	1.38E+11(1.75E+10) −
f_{03}	1.25E+05(1.67E+04) −	6.33E+04(9.26E+03) −	5.42E+05(4.83E+05) −	9.88E+04(2.89E+04) −	6.59E+04(1.16E+04) −	1.63E+05(1.95E+04) −
f_{04}	1.16E+04(2.27E+03) −	4.95E+02(1.60E+02) −	6.80E+04(9.86E+03) −	1.75E+02(5.28E+01) +	3.21E+03(9.26E+02) −	4.59E+04(4.30E+03) −
f_{05}	2.12E+01(4.25E−02) −	2.09E+01(3.27E−01) =	2.13E+01(4.27E−02) −	2.01E+01(8.19E−02) +	2.00E+01(3.80E−02) +	2.12E+01(3.80E−02) −
f_{06}	7.01E+01(2.65E+00) −	5.58E+01(4.11E+00) −	7.94E+01(2.13E+00) −	4.51E+01(4.94E+00) −	5.39E+01(2.97E+00) −	6.90E+01(2.49E+00) −
f_{07}	6.60E+02(7.36E+01) −	7.38E+00(2.26E+00) −	1.86E+03(1.77E+02) −	1.21E−02(1.02E−02) +	2.14E+02(5.92E+01) −	1.51E+03(7.53E+01) −
f_{08}	5.51E+02(3.10E+01) −	2.58E+02(2.28E+01) −	7.22E+02(3.39E+01) −	2.90E+02(3.63E+01) −	2.65E+02(2.65E+01) −	6.12E+02(2.16E+01) −
f_{09}	6.05E+02(3.19E+01) −	3.31E+02(4.29E+01) −	8.63E+02(3.95E+01) −	3.15E+02(6.41E+01) =	3.71E+02(3.69E+01) −	7.00E+02(2.58E+01) −
f_{10}	1.30E+04(6.79E+02) −	6.56E+03(8.04E+02) =	1.53E+04(5.17E+02) −	6.70E+03(9.32E+02) =	6.22E+03(5.11E+02) =	1.41E+04(4.47E+02) −
f_{11}	1.41E+04(4.54E+02) −	6.98E+03(9.50E+02) +	1.59E+04(4.84E+02) −	6.96E+03(9.94E+02) +	8.28E+03(8.76E+02) +	1.47E+04(3.97E+02) −
f_{12}	3.91E+00(3.85E−01) −	1.17E+00(3.69E−01) +	5.75E+00(5.17E−01) −	9.33E−01(4.85E−01) +	6.24E−01(1.54E−01) +	4.37E+00(4.35E−01) −
f_{13}	5.35E+00(4.08E−01) −	5.59E−01(9.40E−02) +	9.62E+00(6.17E−01) −	6.21E−01(1.24E−01) =	2.85E+00(8.77E−01) −	8.47E+00(3.35E−01) −
f_{14}	1.64E+02(1.88E+01) −	3.68E−01(1.25E−01) =	4.69E+02(4.42E+01) −	5.04E−01(2.63E−01) =	5.82E+01(1.46E+01) −	3.75E+02(2.19E+01) −
f_{15}	3.36E+05(1.83E+05) −	8.73E+01(2.45E+01) =	2.45E+07(1.39E+07) −	3.07E+01(7.33E+00) +	1.93E+04(1.42E+04) −	3.85E+06(1.36E+06) −
f_{16}	2.29E+01(2.56E−01) −	2.12E+01(6.28E−01) =	2.35E+01(2.82E−01) −	2.14E+01(5.91E−01) =	2.15E+01(4.89E−01) −	2.29E+01(2.02E−01) −
f_{17}	7.83E+07(2.46E+07) −	2.41E+06(1.83E+06) −	1.12E+09(4.38E+08) −	2.52E+06(1.35E+06) −	4.64E+07(2.53E+07) −	5.15E+08(2.57E+08) −
f_{18}	2.17E+09(4.91E+08) −	1.66E+05(1.06E+06) =	2.82E+10(5.10E+09) −	2.50E+03(1.47E+03) =	4.15E+08(2.79E+08) −	1.46E+10(3.92E+09) −
f_{19}	3.81E+02(8.21E+01) −	6.65E+01(2.33E+01) =	4.52E+03(1.23E+03) −	5.78E+01(2.26E+01) =	1.83E+02(4.16E+01) −	2.80E+03(7.66E+02) −
f_{20}	6.81E+04(2.88E+04) −	1.42E+04(5.55E+03) =	6.31E+06(8.94E+06) −	4.18E+04(1.89E+04) −	3.55E+04(1.08E+04) −	2.61E+05(1.43E+05) −
f_{21}	2.01E+07(1.03E+07) −	5.95E+05(3.57E+05) =	2.84E+08(1.33E+08) −	1.43E+06(9.14E+05) −	1.05E+07(4.23E+06) −	7.43E+07(5.72E+07) −
f_{22}	2.74E+03(2.80E+02) −	1.50E+03(4.28E+02) −	6.82E+05(5.16E+05) −	1.29E+03(3.53E+02) =	1.94E+03(5.91E+02) −	3.78E+05(3.82E+05) −
f_{23}	8.70E+02(1.24E+02) −	4.12E+02(2.58E+01) −	2.29E+03(3.01E+02) −	3.79E+02(1.21E+01) −	2.00E+02(9.67E−07) −	2.00E+02(0.00E+00) =
f_{24}	3.15E+02(5.11E+01) −	2.65E+02(8.99E+00) −	6.50E+02(9.01E+01) −	2.95E+02(7.67E+00) −	2.00E+02(7.32E−05) −	2.00E+02(0.00E+00) +
f_{25}	2.87E+02(2.23E+01) −	2.05E+02(4.82E+00) −	2.97E+02(2.85E+01) −	2.32E+02(7.18E+00) −	2.00E+02(2.03E−08) −	2.00E+02(0.00E+00) =
f_{26}	1.13E+02(5.15E+01) +	1.73E+02(4.08E+01) −	2.63E+02(3.75E+01) −	1.01E+02(1.27E−01) +	1.98E+02(1.39E+01) −	1.99E+02(5.51E+00) −
f_{27}	2.18E+03(1.01E+02) −	2.05E+03(2.63E+02) −	3.68E+03(5.22E+02) −	1.46E+03(1.78E+02) −	2.00E+02(1.21E−07) +	2.19E+03(3.10E+02) −
f_{28}	7.21E+03(8.32E+02) −	8.17E+03(1.07E+03) −	1.83E+04(1.85E+03) −	2.71E+03(8.89E+02) =	2.00E+02(1.19E−06) +	3.35E+03(4.27E+03) =
f_{29}	2.97E+08(5.34E+07) −	8.30E+07(2.62E+08) =	2.85E+09(5.40E+08) −	7.32E+07(7.63E+07) −	2.01E+02(1.60E+00) +	2.00E+02(0.00E+00) +
f_{30}	3.55E+06(1.28E+06) −	3.03E+05(1.79E+05) −	6.78E+07(2.26E+07) −	6.61E+04(2.78E+04) −	2.00E+02(7.83E−02) +	2.00E+02(0.00E+00) +
+/= /−	1/0/29	3/10/17	0/0/30	8/9/13	7/1/22	3/3/24

3.3.1. Statistical analysis

The Wilcoxon rank-sum test was performed to determine the statistical differences systematically and to test the performance of thirteen algorithms on classical benchmark problems, CEC 2014 and CEC 2017 benchmark suites. Table 18 summarizes the results of the validation studies given in Tables 9–17, for all pairwise comparisons involving DSOS-1.

As seen from Table 18, the proposed FDB-based SOS algorithm (DSOS-1) outperforms all of its competitors on three different benchmark sets and 90 different test functions. Moreover, this success is verified in 30, 50, and 100-dimensional search spaces. Considering the performance of algorithms in the classical benchmark set, CEC 2014 and CEC 2017 benchmark suites, MS and GSA are the most unstable algorithms among the thirteen competing algorithms. The performance of these two algorithms on the CEC 2017 test suite and their performance on the other two test sets are inconsistent.

The BSA algorithm among competitors attracts attention with its successful performance. BSA, which predominates its competitors especially in multimodal problem types, cannot achieve the same success in other problem types such as, in unimodal, hybrid and composition. Unlike BSA, the proposed DSOS-1 has shown a stable performance in different problem types. The performance of SSA, LSA and GWO algorithms in the CEC 2014 and CEC 2017 benchmark sets is superior to their performance in the classical benchmark problems.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 14
Comparison of mean error and standard deviation in objective function value for 100 dimension of CEC2014 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	7.33E+07(2.23E+07)	3.12E+08(1.82E+08) −	1.14E+08(1.62E+07) −	1.39E+09(4.55E+08) −	4.13E+08(1.56E+08) −	1.07E+09(6.15E+08) −	6.77E+07(2.12E+07) =
f_{02}	2.16E+08(2.51E+08)	5.77E+10(6.51E+09) −	1.97E+08(5.32E+07) −	1.49E+11(1.90E+10) −	4.33E+10(8.42E+09) −	1.40E+11(4.99E+10) −	9.79E+05(8.53E+05) +
f_{03}	3.54E+04(8.60E+03)	2.87E+05(1.76E+04) −	2.12E+04(3.86E+03) +	6.46E+05(1.40E+05) −	1.89E+05(2.16E+04) −	3.64E+05(1.61E+05) −	5.06E+04(1.72E+04) −
f_{04}	5.33E+02(9.34E+01)	6.82E+03(2.55E+03) −	5.23E+02(2.78E+01) =	2.12E+04(5.56E+03) −	3.49E+03(1.17E+03) −	2.04E+04(1.22E+04) −	3.78E+02(6.66E+01) +
f_{05}	2.11E+01(5.11E−02)	2.00E+01(1.09E−04) +	2.09E+01(2.63E−02) +	2.12E+01(4.62E−02) −	2.14E+01(2.58E−02) −	2.04E+01(1.46E−01) +	2.13E+01(1.31E−01) −
f_{06}	1.08E+02(7.06E+00)	1.21E+02(4.34E+00) −	1.05E+02(2.52E+00) +	1.53E+02(4.97E+00) −	8.74E+01(5.46E+00) +	1.14E+02(8.59E+00) −	1.13E+02(9.98E+00) −
f_{07}	3.67E+00(3.34E+00)	6.14E+02(7.30E+01) −	2.97E+00(5.26E−01) =	1.35E+03(1.54E+02) −	3.95E+02(1.00E+02) −	1.43E+03(4.46E+02) −	4.96E−01(2.07E−01) +
f_{08}	5.59E+02(6.96E+01)	5.76E+02(2.27E+01) −	2.28E+02(1.88E+01) +	1.18E+03(7.89E+01) −	6.22E+02(7.40E+01) −	7.07E+02(1.38E+02) −	7.78E+02(9.31E+01) −
f_{09}	6.99E+02(6.32E+01)	7.21E+02(3.73E+01) =	6.88E+02(4.35E+01) =	1.43E+03(8.01E+01) −	6.15E+02(7.10E+01) +	1.29E+03(1.41E+02) −	1.18E+03(1.48E+02) −
f_{10}	1.59E+04(1.24E+03)	1.64E+04(8.00E+02) −	7.05E+03(5.04E+02) +	2.86E+04(1.13E+03) −	1.66E+04(1.57E+03) −	1.45E+04(3.08E+03) +	1.25E+04(1.30E+03) +
f_{11}	1.66E+04(2.72E+03)	1.41E+04(1.06E+03) +	2.01E+04(5.50E+02) −	2.89E+04(1.01E+03) −	1.61E+04(4.96E+03) +	1.69E+04(1.86E+03) =	1.59E+04(1.47E+03) =
f_{12}	1.89E+00(3.89E−01)	9.99E−03(3.06E−03) +	1.26E+00(9.73E−02) +	3.21E+00(4.82E−01) −	3.78E+00(1.55E+00) −	8.01E−01(3.26E−01) +	1.28E+00(4.79E−01) +
f_{13}	6.69E−01(6.61E−02)	4.11E+00(2.37E−01) −	4.43E−01(3.32E−02) +	6.66E+00(4.40E−01) −	2.62E+00(1.08E+00) −	6.19E+00(1.05E+00) −	6.58E−01(7.42E−02) =
f_{14}	3.66E−01(4.31E−02)	2.27E+02(5.09E+01) −	3.10E−01(1.59E−02) +	4.07E+02(4.89E+01) −	1.10E+02(2.67E+01) −	4.24E+02(1.34E+02) −	3.37E−01(8.58E−02) +
f_{15}	4.99E+02(2.61E+02)	6.28E+04(9.25E+04) −	1.26E+02(1.55E+01) +	7.67E+06(5.61E+06) −	3.93E+04(3.47E+04) −	7.25E+06(9.81E+06) −	9.16E+01(2.15E+01) +
f_{16}	4.46E+01(5.97E−01)	4.58E+01(3.34E−01) −	4.36E+01(3.86E−01) +	4.68E+01(4.67E−01) −	4.54E+01(9.40E−01) −	4.59E+01(6.12E−01) −	4.45E+01(8.25E−01) =
f_{17}	6.23E+06(3.44E+06)	1.05E+08(1.04E+08) −	2.93E+07(7.79E+06) −	1.26E+08(5.78E+07) −	4.06E+07(2.30E+07) −	7.16E+07(7.24E+07) −	8.17E+06(2.93E+06) −
f_{18}	6.03E+03(2.50E+04)	1.32E+05(5.47E+05) =	1.11E+03(6.08E+02) =	2.85E+09(1.42E+09) −	9.07E+08(7.25E+08) −	4.96E+09(3.86E+09) −	1.61E+05(7.41E+05) =
f_{19}	1.81E+02(3.59E+01)	7.63E+02(2.77E+02) −	1.60E+02(2.15E+01) +	1.70E+03(5.13E+02) −	3.86E+02(9.69E+01) −	1.07E+03(4.50E+02) −	1.55E+02(3.35E+01) +
f_{20}	5.73E+04(1.42E+04)	2.80E+05(5.85E+04) −	6.70E+04(1.28E+04) −	1.44E+06(1.31E+06) −	8.30E+04(2.71E+04) −	3.33E+05(2.37E+05) −	9.43E+04(2.87E+04) −
f_{21}	3.89E+06(2.00E+06)	1.40E+07(5.36E+06) −	1.14E+07(3.97E+06) −	1.16E+08(4.62E+07) −	1.77E+07(1.10E+07) −	3.44E+07(4.20E+07) −	4.54E+06(1.86E+06) −
f_{22}	2.66E+03(7.18E+02)	3.48E+03(5.87E+02) −	3.08E+03(2.59E+02) −	1.55E+04(8.12E+03) −	2.04E+03(5.27E+02) +	4.20E+03(7.27E+02) −	3.19E+03(5.93E+02) −
f_{23}	2.00E+02(0.00E+00)	2.24E+02(9.92E+01) −	3.57E+02(1.96E+00) −	1.16E+03(1.85E+02) −	6.84E+02(8.63E+01) −	1.27E+03(4.79E+02) −	3.50E+02(1.36E+00) −
f_{24}	2.00E+02(6.37E−05)	3.20E+02(1.48E+01) −	3.78E+02(5.72E+00) −	8.17E+02(5.74E+01) −	2.00E+02(8.48E−03) −	8.57E+02(1.53E+02) −	4.55E+02(1.92E+01) −
f_{25}	2.00E+02(0.00E+00)	2.11E+02(7.69E+00) −	2.46E+02(9.90E+00) −	4.53E+02(3.69E+01) −	2.05E+02(2.18E+01) −	3.25E+02(4.97E+01) −	3.41E+02(2.77E+01) −
f_{26}	2.00E+02(0.00E+00)	2.00E+02(7.44E−02) −	1.75E+02(4.43E+01) −	2.62E+02(1.76E+01) −	2.00E+02(4.99E−02) −	3.17E+02(1.81E+02) −	2.03E+02(8.90E−01) −
f_{27}	3.01E+03(2.05E+02)	7.55E+03(1.28E+03) −	2.69E+03(2.36E+02) +	4.30E+03(1.75E+02) −	2.43E+03(1.59E+02) +	3.34E+03(1.77E+02) −	3.18E+03(2.12E+02) −
f_{28}	6.18E+03(9.97E+02)	1.07E+04(1.69E+03) −	4.55E+03(6.68E+02) +	1.78E+04(2.26E+03) −	8.17E+03(1.34E+03) −	5.44E+03(9.51E+02) +	1.91E+04(2.36E+03) −
f_{29}	1.09E+07(3.02E+07)	7.05E+07(2.26E+08) −	1.43E+05(1.02E+05) −	6.58E+08(2.84E+08) −	8.88E+07(4.51E+07) −	9.91E+07(1.01E+07) −	5.86E+03(2.81E+03) −
f_{30}	2.93E+04(1.89E+04)	1.56E+07(2.37E+07) −	1.03E+05(2.57E+04) −	2.45E+07(1.48E+07) −	3.92E+06(1.55E+06) −	2.64E+06(1.82E+06) −	1.11E+05(5.63E+04) −
+/= /−		3/2/25	13/4/13	0/0/30	5/0/25	4/1/25	8/5/17

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	3.59E+09(6.53E+08) −	1.83E+08(3.02E+07) −	1.43E+10(2.63E+09) −	1.20E+08(3.24E+07) −	4.85E+08(1.15E+08) −	7.83E+09(1.65E+09) −
f_{02}	1.97E+11(1.28E+10) −	1.40E+09(3.36E+08) −	3.64E+11(1.94E+10) −	1.93E+04(2.93E+04) +	5.88E+10(9.54E+09) −	2.84E+11(1.31E+10) −
f_{03}	2.97E+05(2.77E+04) −	1.19E+05(1.71E+04) −	6.42E+05(2.11E+05) −	1.60E+05(2.93E+04) −	1.22E+05(1.32E+04) −	3.09E+05(1.38E+04) −
f_{04}	3.64E+04(4.60E+03) −	9.69E+02(1.60E+02) −	1.45E+05(2.00E+04) −	3.43E+02(6.58E+01) +	5.77E+03(1.59E+03) −	1.00E+05(8.61E+03) −
f_{05}	2.14E+01(2.78E−02) −	2.04E+01(2.17E−01) +	2.14E+01(3.14E−02) −	2.00E+01(4.49E−02) +	2.01E+01(4.30E−02) +	2.14E+01(2.40E−02) −
f_{06}	1.56E+02(4.79E+00) −	1.33E+02(6.21E+00) −	1.68E+02(2.49E+00) −	1.09E+02(9.37E+00) =	1.21E+02(5.77E+00) −	1.51E+02(3.77E+00) −
f_{07}	1.89E+03(1.17E+02) −	1.56E+01(3.34E+00) −	3.60E+03(1.70E+02) −	1.58E−02(7.79E−03) +	5.85E+02(7.99E+01) −	3.06E+03(1.00E+02) −
f_{08}	1.29E+03(5.34E+01) −	5.86E+02(2.94E+01) −	1.50E+03(4.28E+01) −	6.79E+02(7.37E+01) −	5.43E+02(3.48E+01) =	1.34E+03(2.56E+01) −
f_{09}	1.41E+03(5.21E+01) −	7.10E+02(4.60E+01) =	1.71E+03(4.85E+01) −	7.81E+02(9.25E+01) −	8.25E+02(4.65E+01) −	1.47E+03(3.40E+01) −
f_{10}	2.94E+04(1.06E+03) −	1.49E+04(1.35E+03) +	3.33E+04(7.44E+02) −	1.46E+04(1.59E+03) +	1.32E+04(1.05E+03) +	3.17E+04(7.73E+02) −
f_{11}	3.14E+04(6.72E+02) −	1.48E+04(1.31E+03) +	3.31E+04(7.29E+02) −	1.45E+04(1.54E+03) +	1.58E+04(1.29E+03) =	3.16E+04(7.32E+02) −
f_{12}	4.51E+00(2.58E−01) −	1.99E+00(3.56E−01) =	5.81E+00(4.75E−01) −	1.36E+00(5.19E−01) +	7.49E−01(1.99E−01) +	4.87E+00(3.15E−01) −
f_{13}	7.22E+00(2.20E−01) −	5.54E−01(6.20E−02) +	1.04E+01(3.59E−01) −	6.49E−01(9.31E−02) =	3.96E+00(4.86E−01) −	9.50E+00(2.07E−01) −
f_{14}	5.43E+02(4.34E+01) −	3.63E−01(9.88E−02) +	1.07E+03(5.24E+01) −	4.20E−01(1.77E−01) =	1.61E+02(2.63E+01) −	9.20E+02(2.29E+01) −
f_{15}	4.20E+06(2.26E+06) −	3.06E+02(7.32E+01) +	9.02E+07(3.51E+07) −	1.02E+02(2.02E+01) +	8.56E+04(4.15E+04) −	1.91E+07(5.40E+06) −
f_{16}	4.71E+01(2.96E−01) −	4.40E+01(7.20E−01) +	4.80E+01(2.87E−01) −	4.44E+01(7.99E−01) =	4.42E+01(7.51E−01) +	4.72E+01(2.27E−01) −
f_{17}	4.17E+08(9.90E+07) −	1.05E+07(5.43E+06) −	2.67E+09(5.61E+08) −	6.34E+06(2.97E+06) =	1.04E+08(3.12E+07) −	1.46E+09(3.64E+08) −
f_{18}	9.58E+09(1.95E+09) −	1.18E+06(5.60E+06) −	5.72E+10(6.38E+09) −	6.46E+03(1.98E+04) −	1.39E+09(7.43E+08) −	3.59E+10(5.37E+09) −
f_{19}	1.75E+03(3.19E+02) −	2.24E+02(3.79E+01) −	1.49E+04(2.26E+03) −	1.24E+02(2.13E+01) +	6.12E+02(1.16E+02) −	9.79E+03(9.71E+02) −
f_{20}	2.75E+05(9.78E+04) −	4.55E+04(1.33E+04) +	1.11E+07(8.91E+06) −	9.21E+04(2.75E+04) −	1.19E+05(1.81E+04) −	8.98E+05(2.73E+05) −
f_{21}	1.52E+08(4.98E+07) −	3.65E+06(1.59E+06) =	9.80E+08(2.91E+08) −	4.48E+06(2.05E+06) =	5.82E+07(1.56E+07) −	4.40E+08(1.29E+08) −
f_{22}	6.20E+03(4.59E+02) −	3.12E+03(5.42E+02) −	1.04E+06(6.40E+05) −	2.64E+03(5.07E+02) =	3.49E+03(5.72E+02) −	2.69E+05(1.51E+05) −
f_{23}	1.59E+03(1.63E+02) −	2.78E+02(6.41E+01) −	2.73E+03(2.95E+02) −	4.30E+02(1.69E+01) −	2.00E+02(3.78E−07) −	2.00E+02(0.00E+00) =
f_{24}	6.79E+02(4.13E+01) −	3.31E+02(1.96E+01) −	1.09E+03(1.24E+02) −	4.45E+02(1.35E+01) −	2.00E+02(6.50E−05) −	2.00E+02(0.00E+00) +
f_{25}	4.47E+02(7.88E+01) −	2.04E+02(9.23E−01) −	3.87E+02(2.74E+01) −	2.83E+02(1.69E+01) −	2.00E+02(1.19E−08) −	2.00E+02(0.00E+00) =
f_{26}	3.63E+02(2.83E+02) =	2.00E+02(6.42E−02) −	3.04E+02(3.85E+01) −	1.77E+02(4.48E+01) −	2.00E+02(4.07E−11) −	2.00E+02(0.00E+00) =
f_{27}	4.56E+03(1.34E+02) −	4.62E+03(3.15E+02) −	8.24E+03(1.43E+03) −	3.14E+03(2.50E+02) −	2.00E+02(7.26E−08) +	4.92E+03(4.45E+02) −
f_{28}	2.03E+04(1.43E+03) −	2.21E+04(2.25E+03) −	4.14E+04(2.60E+03) −	6.87E+03(1.14E+03) −	2.00E+02(8.01E−07) +	1.92E+04(8.10E+03) −
f_{29}	1.50E+09(1.85E+08) −	4.87E+07(8.55E+07) −	8.24E+09(9.25E+08) −	8.56E+07(2.68E+08) −	2.01E+02(8.58E−01) −	2.00E+02(0.00E+00) +
f_{30}	3.80E+07(9.56E+06) −	1.19E+06(4.77E+05) −	6.03E+08(2.03E+08) −	5.80E+05(2.77E+05) −	2.00E+02(2.37E−01) +	2.00E+02(0.00E+00) +
+/= /−	0/1/29	8/3/19	0/0/30	9/7/14	7/2/21	3/3/24

According to the paired comparison results between these three algorithms and DSOS-1, it is seen that as the problem dimension increases, DSOS-1 increases the search performance. As Table 18 shows, DSOS-1 consistently obtains highly stable results on all the benchmark sets for all dimensions compared with the twelve other algorithms.

3.3.2. Convergence analysis

In this subsection, the convergence behaviors of algorithms in the unimodal, multimodal, hybrid and composition problem types are investigated. These functions are used to compare the algorithms' exploitation, exploration, convergence/diversity balance and balanced search abilities for high complexity problems. The aim is to test the rate of convergence of algorithms on different problem types, different function dimensions and to evaluate their ability to rid local optimum traps. For this purpose, functions f_1, f_6, f_{22} and f_{30} are selected from the CEC 2014 benchmark suite. The properties of the functions and the problem types are as follows [61]. f_1: Rotated High Conditioned Elliptic Function, unimodal, non-separable, quadratic ill-conditioned. f_6: Shifted and Rotated Weierstrass Function, multi-modal, non-separable, continuous but differentiable only on a set of points. f_{22}: Hybrid Function-6 (N = 5, contains five functions). f_{30}: Composition Function-8, multi-modal, non-separable, asymmetrical, different properties around different local optima, and different properties for different variable subcomponents.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 15
Comparison of mean error and standard deviation in objective function value for 30 dimension of CEC2017 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	9.08E+03(6.74E+03)	9.96E+09(2.07E+09) −	1.37E+07(9.57E+06) −	2.56E+10(1.05E+10) −	1.67E+09(1.13E+09) −	8.81E+09(6.41E+09) −	4.51E+03(4.86E+03) +
f_{02}	3.06E+19(1.04E+20)	1.69E+50(6.16E+50) −	1.89E+24(9.97E+24) −	1.22E+41(6.44E+41) −	1.41E+32(6.85E+32) −	9.59E+39(4.11E+40) −	3.45E+13(2.08E+14) +
f_{03}	4.11E+04(7.41E+03)	9.42E+04(9.06E+03) −	8.42E+04(1.57E+04) −	2.36E+05(6.60E+04) −	5.03E+04(1.09E+04) −	1.38E+05(5.44E+04) −	7.35E+04(2.24E+04) −
f_{04}	1.03E+02(2.43E+01)	2.00E+03(4.56E+02) −	1.50E+02(1.36E+01) −	1.58E+03(8.18E+02) −	2.29E+02(1.34E+02) −	5.15E+02(4.24E+02) −	1.07E+02(2.89E+01) =
f_{05}	1.69E+02(3.63E+01)	2.39E+02(2.74E+01) −	1.44E+02(1.63E+01) +	3.01E+02(3.50E+01) −	1.07E+02(3.61E+01) +	1.83E+02(4.96E+01) =	2.27E+02(4.64E+01) −
f_{06}	9.23E+00(4.10E+00)	6.00E+01(3.27E+00) −	4.33E+00(1.54E+00) +	6.27E+01(1.04E+01) −	8.70E+00(3.48E+00) =	3.35E+01(1.30E+01) −	3.11E+01(1.00E+01) −
f_{07}	2.80E+02(3.32E+01)	3.52E+02(5.29E+01) −	1.89E+02(1.78E+01) +	6.78E+02(1.04E+02) −	1.93E+02(4.94E+01) +	3.74E+02(1.63E+02) −	2.39E+02(6.46E+01) +
f_{08}	1.56E+02(3.62E+01)	1.64E+02(1.96E+01) =	1.45E+02(1.62E+01) =	2.89E+02(3.38E+01) −	8.94E+01(1.96E+01) +	2.09E+02(5.42E+01) −	2.04E+02(4.99E+01) −
f_{09}	1.39E+03(8.87E+02)	3.58E+03(3.86E+02) −	9.73E+02(5.63E+02) +	8.66E+03(3.13E+03) −	1.26E+03(8.71E+02) =	5.51E+03(1.73E+03) −	2.18E+03(1.28E+03) −
f_{10}	5.35E+03(8.47E+02)	4.18E+03(4.93E+02) +	4.82E+03(2.53E+02) +	6.67E+03(5.05E+02) −	3.74E+03(1.47E+03) +	4.53E+03(7.90E+02) +	4.05E+03(5.78E+02) +
f_{11}	1.26E+02(4.30E+01)	5.99E+03(1.31E+03) −	2.35E+02(6.65E+01) −	9.60E+03(5.26E+03) −	7.35E+02(7.58E+02) −	3.05E+03(4.49E+03) −	1.30E+02(4.43E+01) =
f_{12}	7.11E+05(7.03E+05)	1.12E+09(6.44E+08) −	5.71E+06(3.08E+06) −	4.28E+08(4.92E+08) −	8.48E+07(8.96E+07) −	1.31E+08(2.24E+08) −	8.40E+05(7.82E+05) =
f_{13}	1.94E+04(2.14E+04)	1.19E+07(4.11E+07) −	8.48E+05(9.45E+05) −	2.71E+07(4.18E+07) −	3.13E+07(7.80E+07) −	8.79E+06(2.32E+07) −	1.36E+04(1.43E+04) =
f_{14}	2.02E+04(2.56E+04)	1.41E+06(4.70E+05) −	3.45E+04(2.63E+04) −	1.39E+06(1.57E+06) −	4.40E+05(5.45E+05) −	2.12E+05(2.43E+05) −	8.05E+04(7.48E+04) −
f_{15}	9.31E+03(1.05E+04)	1.36E+04(3.38E+03) −	1.30E+05(1.73E+05) −	2.45E+07(4.89E+07) −	2.86E+06(1.02E+07) −	4.96E+04(5.78E+04) −	8.57E+03(9.45E+03) =
f_{16}	1.04E+03(3.93E+02)	2.43E+03(4.23E+02) −	1.32E+03(1.42E+02) −	2.05E+03(4.13E+02) −	9.45E+02(2.78E+02) =	1.50E+03(3.77E+02) −	1.23E+03(3.29E+02) −
f_{17}	3.26E+02(1.39E+02)	1.24E+03(2.34E+02) −	4.46E+02(1.12E+02) −	1.15E+03(3.32E+02) −	3.41E+02(1.38E+02) =	7.74E+02(2.17E+02) −	6.85E+02(2.49E+02) −
f_{18}	3.01E+05(3.15E+05)	3.03E+06(3.04E+06) −	4.06E+05(3.38E+05) −	1.58E+07(1.23E+07) −	1.49E+06(1.82E+06) −	3.74E+06(6.32E+06) −	4.18E+05(4.58E+05) =
f_{19}	8.25E+03(1.07E+04)	7.94E+05(6.22E+05) −	9.59E+04(1.07E+05) −	2.56E+07(3.67E+07) −	1.86E+06(4.56E+06) −	1.05E+07(3.73E+07) −	1.09E+04(1.25E+04) =
f_{20}	3.18E+02(1.39E+02)	1.08E+03(2.21E+02) −	5.21E+02(1.17E+02) −	8.44E+02(1.83E+02) −	4.30E+02(1.42E+02) −	6.88E+02(2.52E+02) −	6.78E+02(2.12E+02) −
f_{21}	3.37E+02(5.23E+01)	5.62E+02(4.57E+01) −	3.34E+02(3.55E+01) =	5.20E+02(3.61E+01) −	2.98E+02(2.78E+01) +	3.85E+02(4.16E+01) −	4.20E+02(5.34E+01) −
f_{22}	1.02E+02(2.24E+00)	5.22E+03(5.42E+02) −	1.17E+03(1.66E+03) −	5.88E+03(1.68E+03) −	2.72E+03(1.83E+03) −	4.15E+03(1.39E+03) −	3.73E+03(1.89E+03) −
f_{23}	5.01E+02(4.43E+01)	1.60E+03(2.22E+02) −	5.01E+02(1.64E+01) =	8.51E+02(8.84E+01) −	4.69E+02(4.59E+01) +	5.24E+02(3.41E+01) −	6.80E+02(6.96E+01) −
f_{24}	5.75E+02(4.46E+01)	1.47E+03(1.28E+02) −	5.91E+02(2.08E+01) =	8.06E+02(6.22E+01) −	5.50E+02(6.03E+01) +	5.71E+02(3.82E+01) =	7.70E+02(1.30E+02) −
f_{25}	4.04E+02(1.64E+01)	6.77E+02(5.29E+01) −	4.27E+02(1.50E+01) −	1.00E+03(2.69E+02) −	4.83E+02(4.25E+01) −	7.73E+02(4.35E+02) −	4.01E+02(1.81E+01) =
f_{26}	1.55E+03(1.22E+03)	6.14E+03(5.39E+02) −	2.40E+03(5.17E+02) −	6.01E+03(7.85E+02) −	2.13E+03(3.92E+02) =	3.07E+03(5.11E+02) −	3.77E+03(1.80E+03) −
f_{27}	5.40E+02(1.60E+01)	2.50E+03(3.78E+02) −	5.43E+02(8.00E+00) =	7.76E+02(7.11E+01) −	5.53E+02(2.31E+01) −	5.46E+02(2.37E+01) =	6.47E+02(6.78E+01) −
f_{28}	4.43E+02(2.11E+01)	1.39E+03(1.86E+02) −	5.51E+02(2.12E+01) −	1.43E+03(3.29E+02) −	6.47E+02(1.42E+02) −	1.31E+03(8.77E+02) −	4.32E+02(2.53E+01) +
f_{29}	9.00E+02(1.88E+02)	3.34E+03(3.76E+02) −	9.40E+02(1.17E+02) =	2.21E+03(5.35E+02) −	9.78E+02(1.82E+02) −	1.27E+03(2.79E+02) −	1.12E+03(2.78E+02) −
f_{30}	1.26E+04(6.82E+03)	8.78E+06(4.57E+06) −	1.60E+05(1.06E+05) −	1.30E+08(1.58E+08) −	1.11E+07(8.45E+06) −	4.94E+05(1.21E+06) −	1.02E+04(7.21E+03) +
+/= /−		1/1/28	5/6/19	0/0/30	7/5/18	1/3/26	6/8/16

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	1.79E+10(3.29E+09) −	1.01E+08(5.04E+07) −	7.68E+10(1.14E+10) −	5.11E+03(5.89E+03) +	7.85E+09(2.26E+09) −	4.82E+10(8.26E+09) −
f_{02}	1.31E+39(5.07E+39) −	5.24E+26(2.57E+27) −	2.75E+56(1.54E+57) −	2.94E+20(9.24E+20) −	6.33E+36(4.46E+37) −	1.33E+55(4.40E+55) −
f_{03}	6.76E+04(1.45E+04) −	2.50E+04(7.50E+03) +	2.21E+06(6.30E+06) −	3.73E+04(1.09E+04) +	6.65E+04(7.95E+03) −	7.62E+04(8.70E+03) −
f_{04}	2.03E+03(5.89E+02) −	2.10E+02(5.99E+01) −	2.84E+04(6.42E+03) −	1.18E+02(3.65E+01) =	7.45E+02(4.76E+02) −	1.83E+04(2.95E+03) −
f_{05}	3.14E+02(2.57E+01) −	1.73E+02(3.10E+01) =	5.40E+02(3.46E+01) −	1.64E+02(5.57E+01) =	2.37E+02(3.76E+01) −	4.12E+02(2.46E+01) −
f_{06}	6.03E+01(4.74E+00) −	4.34E+01(8.76E+00) −	1.12E+02(8.90E+00) −	4.84E+01(1.34E+01) −	5.06E+01(5.62E+00) −	9.03E+01(4.54E+00) −
f_{07}	5.05E+02(5.71E+01) −	2.73E+02(7.04E+01) =	1.04E+03(1.16E+02) −	2.03E+02(5.86E+01) +	4.24E+02(8.39E+01) −	6.95E+02(3.65E+01) −
f_{08}	2.78E+02(1.90E+01) −	1.28E+02(2.42E+01) +	4.45E+02(3.03E+01) −	1.47E+02(4.24E+01) =	1.72E+02(2.47E+01) −	3.38E+02(1.80E+01) −
f_{09}	6.67E+03(1.38E+03) −	2.11E+03(7.68E+02) −	1.65E+04(1.87E+03) −	4.15E+03(1.34E+03) −	4.20E+03(8.32E+02) −	1.02E+04(1.13E+03) −
f_{10}	7.63E+03(3.36E+02) −	3.93E+03(7.05E+02) +	8.91E+03(4.72E+02) −	4.15E+03(7.64E+02) +	4.48E+03(5.45E+02) +	8.13E+03(3.12E+02) −
f_{11}	2.01E+03(6.86E+02) −	3.09E+02(6.93E+01) −	2.70E+04(1.25E+04) −	2.33E+02(7.66E+01) −	2.19E+03(1.08E+03) −	6.81E+03(1.94E+03) −
f_{12}	2.20E+09(6.80E+08) −	5.22E+07(4.10E+07) −	2.16E+10(4.54E+09) −	2.02E+07(1.72E+07) −	3.95E+08(3.10E+08) −	1.10E+10(3.15E+09) −
f_{13}	7.77E+08(3.35E+08) −	6.19E+04(3.52E+04) −	2.27E+10(9.13E+09) −	1.06E+05(5.63E+04) −	1.24E+08(2.06E+08) −	8.38E+09(4.63E+09) −
f_{14}	5.43E+05(4.27E+05) −	6.52E+03(1.18E+04) +	2.53E+07(1.96E+07) −	5.28E+04(4.74E+04) −	1.46E+06(9.83E+05) −	3.10E+06(2.88E+06) −
f_{15}	4.54E+07(3.56E+07) −	1.57E+04(8.28E+03) −	2.70E+09(1.27E+09) −	5.31E+04(4.24E+04) −	1.63E+05(5.59E+05) −	3.40E+08(3.51E+08) −
f_{16}	2.41E+03(2.37E+02) −	1.35E+03(2.79E+02) −	6.83E+03(1.91E+03) −	1.24E+03(3.68E+02) −	1.65E+03(4.36E+02) −	5.06E+03(1.21E+03) −
f_{17}	1.05E+03(1.79E+02) −	5.66E+02(2.05E+02) −	1.44E+04(1.44E+04) −	5.66E+02(1.90E+02) −	7.63E+02(2.55E+02) −	6.11E+03(6.86E+03) −
f_{18}	1.05E+07(5.96E+06) −	1.08E+05(1.21E+05) +	3.83E+08(3.29E+08) −	1.60E+06(1.83E+06) −	6.78E+06(8.00E+06) −	4.63E+07(6.90E+07) −
f_{19}	7.41E+07(3.96E+07) −	3.23E+05(4.35E+05) −	3.25E+09(1.40E+09) −	3.78E+06(2.38E+06) −	4.68E+05(7.59E+05) −	3.61E+08(3.18E+08) −
f_{20}	8.51E+02(1.34E+02) −	4.76E+02(1.40E+02) −	1.47E+03(1.58E+02) −	6.09E+02(1.97E+02) −	7.51E+02(2.26E+02) −	1.06E+03(1.52E+02) −
f_{21}	4.93E+02(2.72E+01) −	3.48E+02(4.07E+01) =	7.87E+02(5.69E+01) −	3.43E+02(4.99E+01) =	4.11E+02(3.75E+01) −	6.05E+02(7.36E+01) −
f_{22}	7.36E+03(1.70E+03) −	2.15E+02(7.12E+01) −	8.49E+03(5.62E+02) −	2.40E+03(2.34E+03) =	3.06E+03(1.91E+03) −	4.19E+03(1.18E+03) −
f_{23}	7.53E+02(4.26E+01) −	7.26E+02(6.70E+01) −	1.68E+03(2.57E+02) −	4.88E+02(3.95E+01) =	7.24E+02(6.74E+01) −	1.28E+03(1.22E+02) −
f_{24}	8.20E+02(3.46E+01) −	7.44E+02(7.84E+01) −	1.96E+03(2.83E+02) −	5.35E+02(3.41E+01) +	8.65E+02(9.95E+01) −	1.65E+03(2.00E+02) −
f_{25}	9.67E+02(2.28E+02) −	4.80E+02(2.87E+01) −	5.73E+03(1.37E+03) −	4.28E+02(2.62E+01) −	6.68E+02(9.09E+01) −	3.14E+03(4.51E+02) −
f_{26}	4.91E+03(3.94E+02) −	2.53E+03(1.83E+03) −	1.17E+04(1.58E+03) −	2.05E+03(1.26E+03) −	4.37E+03(1.20E+03) −	9.07E+03(6.54E+02) −
f_{27}	8.04E+02(6.28E+01) −	8.58E+02(1.41E+02) −	2.92E+03(5.79E+02) −	5.59E+02(2.40E+01) −	7.13E+02(7.90E+01) −	1.69E+03(3.13E+02) −
f_{28}	1.43E+03(2.51E+02) −	5.69E+02(4.93E+01) −	6.92E+03(9.76E+02) −	4.76E+02(4.36E+01) −	9.20E+02(1.69E+02) −	4.90E+03(4.79E+02) −
f_{29}	2.10E+03(2.79E+02) −	1.61E+03(2.98E+02) −	1.61E+04(1.45E+04) −	1.32E+03(2.37E+02) −	1.56E+03(3.29E+02) −	7.14E+03(3.18E+03) −
f_{30}	1.64E+08(7.05E+07) −	3.35E+06(2.81E+06) −	3.40E+09(1.52E+09) −	1.01E+07(7.26E+06) −	8.93E+06(9.08E+06) −	1.16E+09(7.26E+08) −
+/= /−	0/0/30	5/3/22	0/0/30	5/6/19	1/0/29	0/0/30

Fig. 4 provides the convergence profiles of DSOS-1 and the 13 competing algorithms on 30, 50, and 100-dimensional f_1, f_6, f_{22} and f_{30} functions. DSOS-1, LSA, SSA, BSA, and CSA algorithms provide a high-speed convergence on the 30-dimensional f_1 function (see Fig. 4(a)). LSA and DSOS-1 are superior to competing algorithms and have similar convergence rates. From Fig. 4(b, c), when the dimension of the same function is 50 and 100, DSOS-1 converges much faster than others. The convergence curves of the algorithms on the 30-dimensional f_6 multimodal test function are shown in Fig. 4(d). The f_6 function has many local optima near the global optimum. DSOS-1 and GWO algorithms show high exploration ability in the 30-dimensional f_6 function. MFO, SSA, BSA, LSA, and CSA have converged to local optimums, however, MS, SCA, GSA, HSS, BOA, and CGSA convergence prematurely on the f_6 function.

As can be seen from Fig. 4(e, f), similar results are obtained for the 50 and 100-dimensions of the f_6 function. Fig. 4(g, h, i) shows the convergence curves of the algorithms for the three dimensions of the hybrid f_{22} function. It is seen that all the competitors, except DSOS-1, are trapped in local optima at different levels on the hybrid function f_{22}. These results indicate that DSOS-1 applies a good balance between exploration and exploitation phases to find the global optimum.

Fig. 4(j, k, l) shows the convergence curves of the thirteen algorithms for the composition function f_{30}. MS provides high-speed convergence in all three dimensions of the f_{30} function.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 16
Comparison of mean error and standard deviation in objective function value for 50 dimension of CEC2017 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	1.24E+06(5.58E+06)	1.80E+10(3.10E+09) −	5.14E+07(2.18E+07) −	5.90E+10(8.50E+09) −	8.38E+09(3.58E+09) −	3.34E+10(1.77E+10) −	3.30E+04(6.48E+04) +
f_{02}	8.94E+43(6.35E+44)	1.76E+79(7.01E+79) −	1.03E+42(7.35E+42) =	2.28E+78(8.40E+78) −	3.97E+56(1.98E+57) −	2.13E+78(1.46E+79) −	1.82E+27(1.30E+28) +
f_{03}	1.16E+05(1.79E+04)	1.88E+05(1.17E+04) −	1.85E+05(2.89E+04) −	3.92E+05(9.29E+04) −	1.09E+05(2.10E+04) =	3.23E+05(8.14E+04) −	1.68E+05(3.31E+04) −
f_{04}	1.81E+02(5.36E+01)	3.29E+03(8.97E+02) −	2.53E+02(3.30E+01) −	7.26E+03(2.46E+03) −	6.75E+02(3.64E+02) −	3.40E+03(3.14E+03) −	1.87E+02(5.97E+01) =
f_{05}	3.01E+02(6.12E+01)	3.41E+02(2.52E+01) −	2.77E+02(2.03E+01) =	5.92E+02(6.01E+01) −	2.21E+02(3.61E+01) +	4.64E+02(8.46E+01) −	4.62E+02(8.79E+01) −
f_{06}	2.19E+01(7.53E+00)	6.42E+01(2.80E+00) −	3.48E+00(1.03E+00) +	8.19E+01(8.25E+00) −	1.73E+01(4.39E+00) +	5.41E+01(1.21E+01) −	4.92E+01(9.56E+00) −
f_{07}	5.38E+02(7.11E+01)	6.97E+02(7.75E+01) −	3.57E+02(2.45E+01) +	1.50E+03(2.06E+02) −	3.88E+02(6.19E+01) +	1.20E+03(4.08E+02) −	5.32E+02(1.23E+02) =
f_{08}	3.17E+02(6.56E+01)	3.51E+02(2.85E+01) −	2.77E+02(2.45E+01) +	5.80E+02(6.11E+01) −	2.28E+02(4.73E+01) +	4.39E+02(8.30E+01) −	4.45E+02(7.06E+01) −
f_{09}	9.49E+03(3.02E+03)	1.08E+04(8.86E+02) −	3.66E+03(1.61E+03) +	2.73E+04(7.10E+03) −	7.36E+03(4.33E+03) +	1.84E+04(5.43E+03) −	8.02E+03(3.76E+03) +
f_{10}	8.88E+03(1.44E+03)	7.70E+03(7.27E+02) +	8.37E+03(4.31E+02) +	1.25E+04(5.67E+02) −	6.52E+03(1.73E+03) +	7.72E+03(8.82E+02) +	7.34E+03(8.67E+02) +
f_{11}	2.76E+02(9.44E+01)	1.68E+04(2.59E+03) −	1.35E+03(6.46E+02) −	1.89E+04(7.09E+03) −	3.98E+03(1.95E+03) −	1.01E+04(9.40E+03) −	2.26E+02(7.18E+01) +
f_{12}	5.39E+06(3.82E+06)	3.73E+09(1.86E+09) −	1.98E+07(7.63E+06) −	5.14E+09(3.02E+09) −	6.98E+08(7.17E+08) −	3.69E+09(3.53E+09) −	4.05E+06(2.25E+06) =
f_{13}	1.09E+04(9.99E+03)	5.06E+07(1.07E+08) −	1.26E+05(1.28E+05) −	1.47E+09(1.54E+09) −	1.68E+08(1.54E+08) −	7.99E+08(1.49E+09) −	7.18E+03(6.61E+03) =
f_{14}	1.23E+05(1.09E+05)	8.92E+06(7.10E+06) −	3.59E+05(2.66E+05) −	9.04E+06(7.42E+06) −	1.04E+06(1.16E+06) −	1.32E+06(1.40E+06) −	2.43E+05(1.72E+05) −
f_{15}	1.14E+04(7.86E+03)	8.04E+07(2.21E+08) −	2.91E+04(3.65E+04) −	3.78E+08(4.52E+08) −	7.76E+06(1.38E+07) −	8.35E+07(4.04E+08) −	6.60E+03(6.61E+03) +
f_{16}	1.68E+03(5.54E+02)	2.60E+03(4.90E+02) −	2.22E+03(2.14E+02) −	3.77E+03(6.65E+02) −	1.54E+03(4.04E+02) =	2.70E+03(6.04E+02) −	2.23E+03(4.48E+02) −
f_{17}	1.36E+03(3.22E+02)	2.02E+03(3.58E+02) −	1.58E+03(2.17E+02) −	3.69E+03(1.87E+03) −	1.22E+03(3.43E+02) +	2.18E+03(4.34E+02) −	1.73E+03(3.87E+02) −
f_{18}	1.03E+06(1.10E+06)	6.88E+06(6.51E+06) −	2.31E+06(1.08E+06) −	5.09E+07(3.31E+07) −	5.70E+06(8.78E+06) −	1.38E+07(2.62E+07) −	1.79E+06(1.49E+06) −
f_{19}	1.63E+04(1.54E+04)	3.27E+05(1.58E+05) −	1.90E+04(7.46E+03) −	1.62E+08(2.52E+08) −	5.85E+06(1.04E+07) −	7.59E+06(2.54E+07) −	1.42E+04(1.09E+04) =
f_{20}	1.08E+03(3.27E+02)	1.65E+03(3.39E+02) −	1.40E+03(1.73E+02) −	1.99E+03(2.56E+02) −	1.02E+03(4.65E+02) =	1.44E+03(2.82E+02) −	1.37E+03(3.66E+02) −
f_{21}	4.38E+02(6.74E+01)	7.66E+02(5.22E+01) −	4.73E+02(2.21E+01) −	8.62E+02(5.32E+01) −	4.14E+02(4.28E+01) =	6.32E+02(6.20E+01) −	6.68E+02(8.22E+01) −
f_{22}	5.47E+03(4.73E+03)	9.57E+03(8.16E+02) −	8.98E+03(8.68E+02) −	1.31E+04(6.44E+02) −	7.65E+03(2.30E+03) =	8.22E+03(9.50E+02) =	8.12E+03(9.21E+02) =
f_{23}	6.86E+02(6.91E+01)	2.51E+03(2.09E+02) −	7.26E+02(2.18E+01) −	1.59E+03(1.48E+02) −	6.84E+02(5.92E+01) =	8.44E+02(6.24E+01) −	1.23E+03(1.70E+02) −
f_{24}	7.96E+02(8.13E+01)	2.07E+03(1.33E+02) −	8.36E+02(2.82E+01) −	1.33E+03(9.75E+01) −	7.79E+02(1.05E+02) =	8.15E+02(4.84E+01) =	1.39E+03(1.70E+02) −
f_{25}	6.23E+02(3.40E+01)	2.13E+03(2.78E+02) −	6.85E+02(3.34E+01) −	4.50E+03(1.41E+03) −	1.09E+03(2.57E+02) −	2.92E+03(2.35E+03) −	5.95E+02(3.77E+01) +
f_{26}	3.60E+03(2.62E+03)	9.86E+03(6.46E+02) −	4.14E+03(3.25E+02) =	1.19E+04(1.27E+03) −	4.08E+03(8.48E+02) =	5.56E+03(7.04E+02) −	8.89E+03(1.89E+03) −
f_{27}	8.56E+02(1.20E+02)	5.72E+03(6.36E+02) −	8.30E+02(4.04E+01) =	1.81E+03(3.62E+02) −	9.02E+02(1.07E+02) −	8.73E+02(1.10E+02) =	1.53E+03(2.91E+02) −
f_{28}	5.86E+02(4.31E+01)	2.94E+03(3.02E+02) −	9.10E+02(8.82E+01) −	4.03E+03(8.39E+02) −	1.53E+03(3.95E+02) −	5.22E+03(1.33E+03) −	5.46E+02(4.09E+01) +
f_{29}	1.45E+03(3.10E+02)	8.23E+03(5.22E+03) −	1.36E+03(1.62E+02) =	9.38E+03(4.57E+03) −	1.71E+03(2.91E+02) −	2.35E+03(5.47E+02) −	1.90E+03(3.86E+02) −
f_{30}	1.19E+06(6.15E+05)	2.24E+08(3.72E+07) −	2.57E+06(4.36E+05) −	5.85E+08(5.65E+08) −	1.24E+08(3.96E+07) −	1.23E+08(2.59E+08) −	1.52E+06(4.21E+05) −
+/= /−		1/0/29	5/5/20	0/0/30	7/8/15	1/3/26	8/6/16

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	5.51E+10(6.54E+09) −	3.10E+08(1.20E+08) −	1.38E+11(9.69E+09) −	5.87E+03(8.08E+03) +	1.82E+10(4.04E+09) −	9.88E+10(8.59E+09) −
f_{02}	8.66E+70(4.68E+71) −	1.56E+47(1.03E+48) −	3.47E+92(1.84E+93) −	4.43E+41(2.21E+42) =	2.70E+65(1.92E+66) −	4.36E+92(1.91E+93) −
f_{03}	1.55E+05(1.75E+04) −	6.99E+04(1.32E+04) +	4.81E+08(1.77E+09) −	1.17E+05(3.23E+04) =	1.53E+05(1.57E+04) −	2.98E+05(1.16E+05) −
f_{04}	9.22E+03(1.91E+03) −	4.18E+02(8.25E+01) −	5.54E+04(6.39E+03) −	2.19E+02(4.18E+01) −	2.64E+03(8.44E+02) −	3.65E+04(4.18E+03) −
f_{05}	5.99E+02(3.03E+01) −	3.04E+02(3.68E+01) =	8.21E+02(4.55E+01) −	3.24E+02(6.62E+01) =	3.80E+02(3.50E+01) −	6.70E+02(2.86E+01) −
f_{06}	7.93E+01(6.40E+00) −	5.49E+01(6.99E+00) −	1.23E+02(5.53E+00) −	5.89E+01(8.42E+00) −	5.35E+01(5.99E+00) −	1.01E+02(5.83E+00) −
f_{07}	1.06E+03(9.27E+01) −	6.30E+02(1.22E+02) −	1.82E+03(1.15E+02) −	4.59E+02(1.13E+02) +	7.89E+02(9.22E+01) −	1.28E+03(4.43E+01) −
f_{08}	6.12E+02(3.06E+01) −	3.16E+02(4.86E+01) =	8.57E+02(4.03E+01) −	3.52E+02(8.16E+01) −	3.99E+02(3.04E+01) −	7.00E+02(2.48E+01) −
f_{09}	2.71E+04(4.36E+03) −	9.34E+03(2.27E+03) =	5.27E+04(4.09E+03) −	1.30E+04(2.60E+03) −	1.31E+04(1.38E+03) −	3.69E+04(2.47E+03) −
f_{10}	1.42E+04(4.01E+02) −	6.87E+03(8.90E+02) +	1.56E+04(5.25E+02) −	6.94E+03(9.16E+02) +	7.92E+03(8.31E+02) +	1.45E+04(3.99E+02) −
f_{11}	8.49E+03(1.99E+03) −	9.05E+02(2.27E+02) −	4.46E+04(1.92E+04) −	4.58E+02(1.14E+02) −	6.54E+03(1.75E+03) −	2.29E+04(2.03E+03) −
f_{12}	1.59E+10(3.70E+09) −	2.53E+08(1.68E+08) −	1.13E+11(2.07E+10) −	8.55E+07(4.91E+07) −	4.22E+09(2.33E+09) −	6.90E+10(1.37E+10) −
f_{13}	5.21E+09(2.46E+09) −	9.53E+04(6.85E+04) −	6.53E+10(1.55E+10) −	1.75E+05(1.22E+05) −	9.79E+08(1.31E+09) −	3.55E+10(9.98E+09) −
f_{14}	4.88E+06(2.79E+06) −	1.16E+05(1.19E+05) =	2.44E+08(1.15E+08) −	2.50E+05(2.12E+05) −	6.03E+06(5.93E+06) −	8.29E+07(5.85E+07) −
f_{15}	7.60E+08(3.88E+08) −	3.01E+04(1.43E+04) −	1.71E+10(4.77E+09) −	6.39E+04(4.19E+04) −	1.28E+08(1.60E+08) −	6.19E+09(2.75E+09) −
f_{16}	4.37E+03(3.95E+02) −	2.17E+03(4.77E+02) −	1.23E+04(2.31E+03) −	2.16E+03(4.88E+02) −	2.61E+03(4.88E+02) −	8.67E+03(1.49E+03) −
f_{17}	3.03E+03(3.57E+02) −	1.61E+03(3.28E+02) −	2.33E+05(2.80E+05) −	1.68E+03(3.76E+02) −	1.77E+03(4.39E+02) −	1.18E+04(5.71E+03) −
f_{18}	3.35E+07(1.52E+07) −	9.54E+05(9.59E+05) =	4.95E+08(2.25E+08) −	2.34E+06(1.98E+06) −	1.88E+07(1.70E+07) −	1.19E+08(5.43E+07) −
f_{19}	4.52E+08(1.60E+08) −	6.92E+05(5.72E+05) −	8.35E+09(2.85E+09) −	4.70E+06(3.36E+06) −	2.30E+07(4.14E+07) −	3.12E+09(1.44E+09) −
f_{20}	2.11E+03(1.90E+02) −	1.11E+03(2.57E+02) =	2.75E+03(2.44E+02) −	1.29E+03(3.79E+02) −	1.41E+03(2.95E+02) −	2.27E+03(1.98E+02) −
f_{21}	8.23E+02(4.25E+01) −	5.36E+02(5.75E+01) −	1.38E+03(9.22E+01) −	5.07E+02(6.68E+01) −	6.13E+02(4.75E+01) −	1.11E+03(6.20E+01) −
f_{22}	1.44E+04(4.56E+02) −	6.85E+03(3.12E+03) =	1.64E+04(4.99E+02) −	7.35E+03(2.06E+03) =	8.88E+03(8.71E+02) −	1.47E+04(9.24E+02) −
f_{23}	1.33E+03(7.11E+01) −	1.31E+03(2.06E+02) −	2.93E+03(3.44E+02) −	7.36E+02(8.17E+01) −	1.15E+03(1.36E+02) −	2.29E+03(2.21E+02) −
f_{24}	1.40E+03(6.91E+01) −	1.26E+03(1.74E+02) −	3.42E+03(4.05E+02) −	7.60E+02(6.61E+01) +	1.41E+03(1.53E+02) −	2.96E+03(2.98E+02) −
f_{25}	5.30E+03(1.01E+03) −	8.13E+02(8.32E+01) −	1.86E+04(2.52E+03) −	5.88E+02(2.96E+01) +	2.04E+03(4.29E+02) −	1.30E+04(9.58E+02) −
f_{26}	1.05E+04(6.94E+02) −	6.02E+03(3.12E+03) −	1.89E+04(1.59E+03) −	2.91E+03(2.51E+03) +	8.17E+03(1.23E+03) −	1.48E+04(5.79E+02) −
f_{27}	2.03E+03(2.23E+02) −	2.20E+03(4.72E+02) −	6.07E+03(9.14E+02) −	8.66E+02(1.26E+02) =	1.45E+03(2.19E+02) −	3.88E+03(7.59E+02) −
f_{28}	5.03E+03(7.02E+02) −	1.04E+03(1.42E+02) −	1.48E+04(2.52E+03) −	5.54E+02(3.72E+01) +	2.47E+03(3.77E+02) −	1.08E+04(1.10E+03) −
f_{29}	5.49E+03(9.33E+02) −	3.29E+03(8.00E+02) −	9.89E+05(1.29E+06) −	2.51E+03(5.63E+02) −	2.66E+03(4.71E+02) −	1.88E+05(1.36E+05) −
f_{30}	9.39E+08(3.15E+08) −	9.99E+07(3.43E+07) −	1.43E+10(4.04E+09) −	9.49E+07(2.99E+07) −	1.07E+08(8.20E+07) −	6.34E+09(2.66E+09) −
+/= /−	0/0/30	2/7/21	0/0/30	7/5/18	1/0/29	0/0/30

In addition, although not as fast as MS, the DSOS-1, GWO and BOA algorithms converge to the global optimum in all three dimensions.

As a result, our proposed DSOS-1 algorithm shows high stability in four different problem types. The FDB method provides the SOS algorithm to obtain better solutions and faster convergence.

In order to consolidate the convergence analysis, the performance of the 13 competing algorithms on the four problems selected from the CEC 2017 benchmark suite is investigated. These functions are f_3, f_9, f_{20} and f_{29}. The properties and types of functions are as follows [61]. f_3: Shifted and Rotated Sum of Different Power Function, unimodal, non-separable. f_9: Shifted and Rotated Levy Function, multimodal, non-separable, where the local optima's number is huge. f_{20}: Hybrid Function 6 (N = 6, contains six functions), f_{29}: Composition Function 9 (N = 3), multimodal, non-separable, asymmetrical, different properties around different local optima, and different properties for different variable subcomponents.

The convergence curves of the four algorithms used for problems and 30, 50 and 100-dimensions are shown in Fig. 5. When the curves are quickly reviewed, similarities to the convergence behaviors given in Fig. 4 are noteworthy. From Fig. 5, DSOS-1 is the most successful convergent algorithm in the four problem types and in all dimensions. This is the strongest indicator that DSOS-1 can balance global search ability and local search ability in the optimization process. MFO, BSA, SCA, and CGSA algorithms

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 17
Comparison of mean error and standard deviation in objective function value for 100 dimension of CEC2017 test problems.

F_n	DSOS-1	GSA	BSA	HSS	GWO	MFO	LSA
f_{01}	3.14E+08(5.86E+08)	4.87E+10(6.64E+09) −	1.19E+08(4.08E+07) =	1.54E+11(1.84E+10) −	4.14E+10(8.90E+09) −	1.34E+11(5.14E+10) −	1.82E+06(2.36E+06) +
f_{02}	1.81E+126(1.30E+127)	1.97E+168(6.55E+04) −	8.24E+83(5.58E+84) +	1.62E+175(6.55E+04) −	1.51E+137(9.43E+97) −	1.96E+158(6.55E+04) −	2.61E+89(1.86E+90) +
f_{03}	2.81E+05(1.86E+04)	3.68E+05(1.78E+04) −	4.23E+05(3.25E+04) −	8.73E+05(1.35E+05) −	2.71E+05(2.31E+04) +	8.38E+05(1.79E+05) −	5.15E+05(6.68E+04) −
f_{04}	5.99E+02(1.23E+02)	8.76E+03(3.12E+03) −	5.56E+02(4.71E+01) =	2.64E+04(6.37E+03) −	3.62E+03(1.10E+03) −	2.59E+04(1.39E+04) −	3.83E+02(6.22E+01) +
f_{05}	7.56E+02(7.84E+01)	8.22E+02(3.90E+01) −	6.94E+02(5.37E+01) +	1.46E+03(9.04E+01) −	6.49E+02(6.32E+01) +	1.28E+03(1.67E+02) −	1.20E+03(1.53E+02) −
f_{06}	4.57E+01(5.61E+00)	6.29E+01(1.98E+00) −	4.07E+00(9.92E−01) +	9.99E+01(7.61E+00) −	3.55E+01(4.61E+00) +	7.30E+01(6.92E+00) −	6.50E+01(6.38E+00) −
f_{07}	1.52E+03(2.14E+02)	1.93E+03(1.40E+02) −	9.53E+02(6.15E+01) +	3.78E+03(3.11E+02) −	1.24E+03(1.16E+02) +	3.97E+03(8.36E+02) −	1.59E+03(4.05E+02) =
f_{08}	7.29E+02(8.74E+01)	9.91E+02(5.17E+01) −	6.93E+02(4.54E+01) +	1.61E+03(1.06E+02) −	6.30E+02(5.23E+01) +	1.31E+03(1.56E+02) −	1.28E+03(1.61E+02) −
f_{09}	4.74E+04(5.29E+03)	2.15E+04(9.63E+02) +	2.11E+04(5.78E+03) +	7.50E+04(1.08E+04) −	3.06E+04(1.21E+04) +	4.60E+04(5.68E+03) =	2.58E+04(9.64E+03) +
f_{10}	1.68E+04(1.96E+03)	1.56E+04(1.05E+03) +	1.96E+04(5.07E+02) −	2.87E+04(8.79E+02) −	1.47E+04(1.23E+03) +	1.69E+04(1.84E+03) =	1.55E+04(1.22E+03) +
f_{11}	1.14E+04(4.17E+03)	1.40E+05(1.49E+04) −	3.21E+04(7.37E+03) −	3.60E+05(1.12E+05) −	5.68E+04(1.41E+04) −	1.47E+05(8.71E+04) −	1.97E+04(1.03E+04) −
f_{12}	4.84E+07(4.37E+07)	1.06E+10(5.69E+09) −	9.23E+07(2.29E+07) −	3.13E+10(8.67E+09) −	7.06E+09(3.41E+09) −	3.76E+10(1.99E+10) −	2.09E+07(1.41E+07) +
f_{13}	7.51E+03(5.73E+03)	3.15E+07(1.78E+08) −	3.44E+03(1.39E+03) +	4.14E+09(2.44E+09) −	8.57E+08(8.78E+08) −	5.21E+09(4.12E+09) −	6.89E+03(6.34E+03) =
f_{14}	6.93E+05(3.54E+05)	5.38E+06(1.36E+06) −	3.02E+06(1.03E+06) −	3.75E+07(1.70E+07) −	6.76E+06(4.04E+06) −	9.98E+06(1.01E+07) −	1.92E+06(1.01E+06) −
f_{15}	3.26E+03(5.10E+03)	2.01E+04(3.89E+03) −	9.52E+02(5.32E+02) +	3.27E+09(1.41E+09) −	1.15E+08(1.41E+08) −	1.40E+09(1.58E+09) −	2.88E+03(2.99E+03) =
f_{16}	3.89E+03(7.70E+02)	7.29E+03(1.41E+03) −	5.20E+03(4.14E+02) −	1.16E+04(1.25E+03) −	4.40E+03(1.01E+03) −	6.57E+03(7.88E+02) −	4.62E+03(6.92E+02) −
f_{17}	3.57E+03(6.58E+02)	4.31E+03(8.39E+02) −	3.64E+03(2.72E+02) =	4.53E+05(7.18E+05) −	3.25E+03(6.70E+02) +	7.98E+03(5.27E+03) −	4.01E+03(5.42E+02) −
f_{18}	1.47E+06(7.57E+05)	2.45E+06(8.96E+05) −	2.68E+06(1.16E+06) −	6.93E+07(4.45E+07) −	6.32E+06(4.02E+06) −	1.49E+07(1.40E+07) −	2.77E+06(1.22E+06) −
f_{19}	5.16E+03(7.11E+03)	9.15E+05(7.29E+05) −	1.22E+03(1.38E+03) +	8.51E+08(6.12E+08) −	1.30E+08(2.03E+08) −	6.69E+08(7.94E+08) −	4.20E+03(4.47E+03) =
f_{20}	3.38E+03(7.16E+02)	4.19E+03(6.02E+02) −	3.90E+03(2.77E+02) −	5.43E+03(2.91E+02) −	2.96E+03(8.32E+02) +	3.90E+03(7.04E+02) −	3.74E+03(5.20E+02) −
f_{21}	7.88E+02(9.93E+01)	1.82E+03(1.66E+02) −	8.69E+02(4.39E+01) −	2.00E+03(1.37E+02) −	8.53E+02(9.43E+01) −	1.57E+03(1.59E+02) −	1.63E+03(1.70E+02) −
f_{22}	1.68E+04(6.55E+03)	1.90E+04(1.11E+03) =	2.09E+04(5.82E+02) −	2.96E+04(9.44E+02) −	1.74E+04(3.84E+03) +	1.81E+04(1.56E+03) =	1.71E+04(1.31E+03) +
f_{23}	1.11E+03(9.46E+01)	5.04E+03(3.57E+02) −	1.05E+03(3.73E+01) +	2.82E+03(2.36E+02) −	1.20E+03(6.94E+01) −	1.48E+03(1.05E+02) −	2.35E+03(3.59E+02) −
f_{24}	1.66E+03(1.19E+02)	5.90E+03(9.10E+02) −	1.59E+03(4.45E+01) +	3.66E+03(2.70E+02) −	1.76E+03(1.10E+02) −	1.97E+03(1.56E+02) −	2.99E+03(2.83E+02) −
f_{25}	1.14E+03(9.39E+01)	4.06E+03(8.25E+02) −	1.23E+03(5.10E+01) −	1.40E+04(2.34E+03) −	3.60E+03(8.17E+02) −	1.05E+04(5.24E+03) −	9.39E+02(7.36E+01) +
f_{26}	1.55E+04(6.05E+03)	2.56E+04(1.65E+03) −	1.07E+04(5.66E+02) +	3.55E+04(3.16E+03) −	1.22E+04(1.47E+03) +	1.57E+04(1.72E+03) =	2.48E+04(2.56E+03) −
f_{27}	1.16E+03(1.70E+02)	1.07E+04(6.42E+02) −	9.58E+02(3.07E+01) +	3.58E+03(5.97E+02) −	1.31E+03(1.51E+02) −	1.25E+03(2.28E+02) =	2.04E+03(5.62E+02) −
f_{28}	9.57E+02(9.78E+01)	7.34E+03(9.43E+02) −	1.66E+03(1.63E+02) −	1.45E+04(1.71E+03) −	5.37E+03(9.51E+02) −	1.65E+04(2.16E+03) −	7.60E+02(6.04E+01) +
f_{29}	3.96E+03(5.19E+02)	9.38E+03(1.50E+03) −	4.14E+03(3.53E+02) =	8.32E+04(1.18E+05) −	5.21E+03(6.76E+02) −	8.27E+03(3.99E+03) −	4.68E+03(6.78E+02) −
f_{30}	3.26E+04(3.83E+04)	7.57E+08(8.51E+08) −	6.07E+05(6.11E+05) −	4.03E+09(2.06E+09) −	7.89E+08(7.80E+08) −	2.33E+09(1.44E+09) −	2.30E+05(2.03E+05) −
+/= /−		2/1/27	13/4/13	0/0/30	11/0/19	0/5/25	9/4/17

F_n	SCA	CSA	CGSA	SSA	MS	BOA
f_{01}	1.82E+11(1.29E+10) −	1.07E+09(2.54E+08) −	3.16E+11(1.87E+10) −	1.33E+04(1.37E+04) +	5.32E+10(7.86E+09) −	2.48E+11(1.21E+10) −
f_{02}	5.21E+158(6.55E+04) −	1.86E+121(1.32E+122) −	3.50E+194(6.55E+04) −	4.62E+104(3.29E+105) +	7.35E+143(4.61E+144) −	1.18E+200(6.55E+04) −
f_{03}	3.99E+05(4.07E+04) −	1.92E+05(2.17E+04) +	2.47E+09(7.85E+09) −	3.28E+05(7.24E+04) −	3.17E+05(1.24E+04) −	3.89E+05(7.28E+04) −
f_{04}	3.48E+04(4.41E+03) −	1.14E+03(2.67E+02) −	1.57E+05(2.25E+04) −	3.75E+02(5.50E+01) +	5.89E+03(1.26E+03) −	1.05E+05(9.10E+03) −
f_{05}	1.47E+03(4.68E+01) −	7.92E+02(5.36E+01) −	1.83E+03(5.80E+01) −	8.15E+02(1.12E+02) −	8.96E+02(5.17E+01) −	1.57E+03(3.38E+01) −
f_{06}	9.70E+01(5.19E+00) −	6.61E+01(3.59E+00) −	1.25E+02(3.68E+00) −	6.60E+01(5.84E+00) −	5.51E+01(3.51E+00) −	1.09E+02(3.06E+00) −
f_{07}	3.01E+03(1.55E+02) −	1.97E+03(2.83E+02) −	4.04E+03(2.11E+02) −	1.31E+03(3.00E+02) +	2.06E+03(1.46E+02) −	3.16E+03(7.95E+01) −
f_{08}	1.52E+03(6.51E+01) −	9.04E+02(7.77E+01) −	2.03E+03(5.31E+01) −	8.59E+02(1.23E+02) −	9.82E+02(7.21E+01) −	1.73E+03(4.95E+01) −
f_{09}	7.75E+04(8.02E+03) −	2.47E+04(2.62E+03) +	1.00E+05(6.33E+03) −	2.77E+04(3.24E+03) +	2.65E+04(1.60E+03) +	7.98E+04(4.00E+03) −
f_{10}	3.13E+04(5.89E+02) −	1.49E+04(1.41E+03) +	3.33E+04(8.49E+02) −	1.45E+04(1.49E+03) +	1.70E+04(1.31E+03) =	3.17E+04(6.81E+02) −
f_{11}	1.08E+05(1.69E+04) −	1.63E+04(4.49E+03) −	4.82E+07(2.40E+08) −	3.65E+03(7.95E+02) +	7.23E+04(1.45E+04) −	2.91E+05(9.69E+04) −
f_{12}	7.41E+10(9.10E+09) −	9.31E+08(3.23E+08) −	2.48E+11(2.36E+10) −	4.10E+08(1.85E+08) −	1.39E+10(4.40E+09) −	1.76E+11(2.06E+10) −
f_{13}	1.17E+10(2.04E+09) −	5.72E+04(1.88E+04) −	6.12E+10(5.72E+09) −	8.46E+04(3.46E+04) −	1.96E+09(1.18E+09) −	3.98E+10(5.91E+09) −
f_{14}	3.75E+07(1.44E+07) −	6.78E+05(3.53E+05) =	3.74E+08(1.69E+08) −	1.37E+06(7.52E+05) −	1.22E+07(3.94E+06) −	9.27E+07(4.39E+07) −
f_{15}	3.63E+09(1.11E+09) −	4.18E+04(1.95E+04) −	3.45E+10(5.46E+09) −	7.19E+04(3.24E+04) −	5.44E+08(6.77E+08) −	2.06E+10(4.72E+09) −
f_{16}	1.17E+04(6.75E+02) −	6.24E+03(9.11E+02) −	3.02E+04(3.97E+03) −	5.05E+03(8.57E+02) −	6.40E+03(8.08E+02) −	2.25E+04(2.01E+03) −
f_{17}	1.62E+04(6.92E+03) −	4.19E+03(7.13E+02) −	4.81E+07(3.04E+07) −	3.86E+03(5.13E+02) −	9.46E+03(4.25E+03) −	7.00E+06(5.69E+06) −
f_{18}	6.21E+07(2.52E+07) −	7.42E+05(3.39E+05) +	6.52E+08(2.74E+08) −	2.68E+06(1.53E+06) −	1.15E+07(4.70E+06) −	1.90E+08(1.09E+08) −
f_{19}	3.39E+09(8.01E+08) −	8.26E+06(5.14E+06) −	3.54E+10(4.97E+09) −	1.08E+07(5.00E+06) −	4.42E+08(3.82E+08) −	1.98E+10(4.89E+09) −
f_{20}	5.58E+03(3.30E+02) −	3.18E+03(4.73E+02) =	6.66E+03(3.51E+02) −	3.24E+03(4.60E+02) =	3.68E+03(5.44E+02) −	5.87E+03(2.08E+02) −
f_{21}	1.94E+03(8.75E+01) −	1.42E+03(1.58E+02) −	3.32E+03(2.29E+02) −	1.02E+03(1.37E+02) −	1.32E+03(1.18E+02) −	2.60E+03(1.57E+02) −
f_{22}	3.24E+04(5.17E+02) −	1.83E+04(1.43E+03) =	3.43E+04(7.29E+02) −	1.62E+04(1.57E+03) +	1.94E+04(1.27E+03) −	3.27E+04(7.79E+02) −
f_{23}	2.72E+03(1.03E+02) −	2.96E+03(3.78E+02) −	6.06E+03(5.77E+02) −	1.23E+03(1.32E+02) −	1.87E+03(1.78E+02) −	4.02E+03(2.37E+02) −
f_{24}	4.45E+03(2.44E+02) −	4.45E+03(5.59E+02) −	1.17E+04(1.23E+03) −	1.70E+03(1.71E+02) =	2.77E+03(2.33E+02) −	9.24E+03(1.33E+03) −
f_{25}	1.56E+04(2.03E+03) −	1.61E+03(1.82E+02) −	3.76E+04(3.74E+03) −	9.44E+02(6.81E+01) +	4.57E+03(6.69E+02) −	2.63E+04(1.49E+03) −
f_{26}	3.38E+04(2.06E+03) −	2.14E+04(6.31E+03) −	6.52E+04(4.84E+03) −	1.13E+04(3.69E+03) +	2.33E+04(3.59E+03) −	5.32E+04(1.86E+03) −
f_{27}	4.81E+03(4.82E+02) −	3.48E+03(8.40E+02) −	1.46E+04(1.84E+03) −	1.11E+03(1.15E+02) =	2.10E+03(2.77E+02) −	1.12E+04(1.19E+03) −
f_{28}	1.93E+04(1.75E+03) −	1.82E+03(4.47E+02) −	4.12E+04(3.47E+03) −	7.24E+02(4.22E+01) +	7.24E+03(1.04E+03) −	3.34E+04(1.82E+03) −
f_{29}	1.82E+04(4.87E+03) −	8.63E+03(1.12E+03) −	4.85E+06(3.31E+06) −	6.34E+03(7.72E+02) −	7.36E+03(1.92E+03) −	6.22E+05(4.62E+05) −
f_{30}	8.45E+09(1.35E+09) −	2.07E+08(1.37E+08) −	5.57E+10(8.05E+09) −	9.89E+07(3.53E+07) −	1.68E+09(8.63E+08) −	3.49E+10(6.09E+09) −
+/= /−	0/0/30	4/3/23	0/0/30	11/3/16	1/1/28	0/0/30

converge prematurely in the unimodal problem f_3. This is an indication that these algorithms are not capable of searching for a sensitive local neighborhood (see Fig. 5 a, b, and c). From Fig. 5, we can also see that DSOS-1, SSA, MS, and GWO achieve significantly better convergence performance on f_3.

As can be seen from Fig. 5(d, e and f), the BOA, HSS, and SSA algorithms are easily trapped in the local optima at different levels on the multimodal function f_9. We can also see that DSOS-1, LSA, CSA, GWO, and BSA perform significantly better exploration on f_9. Fig. 5(g, h, and i) provides the convergence profiles of DSOS-1 and the other algorithms on hybrid function f_{20}.

As shown in the figures, the convergence curve of DSOS-1 drops rapidly in all dimensions and converges to the global optimum. Among the other twelve algorithms, CSA is the most successful on f_{20}. In the same function CGSA, BOA, HSS, and SCA algorithms tend to trap in local optima and unbalanced exploitation. On the composition function f_{29}, DSOS-1 obtains better results and convergent speed for all of the function dimensions. The curves of some algorithms are not seen in the figures. This is because the error values for those algorithms are very high. Therefore, the error curve of the CGSA for the f_{29} function cannot be seen in Fig. 5(g, h, and i). Convergence analysis confirmed that the FDB-based SOS (DSOS-1) algorithm has the ability to perform better solutions and faster convergence in most of the benchmark functions including unimodal, multimodal, low/middle / high-dimensional, hybrid, and composition functions.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Table 18
Statistical comparison results of algorithms.

vs. DSOS-1		CLASSIC			CEC 2014			CEC 2017		
		D=30	D=50	D=100	D=30	D=50	D=100	D=30	D=50	D=100
GSA	+ (better)	4	4	4	6	4	3	1	1	2
	= (similar)	2	1	1	0	0	2	1	0	1
	− (worse)	24	25	25	24	26	25	28	29	27
BSA	+ (better)	6	6	6	13	11	13	5	5	13
	= (similar)	0	0	0	2	5	4	6	5	4
	− (worse)	24	24	24	15	14	13	19	20	13
HSS	+ (better)	0	0	0	0	0	0	0	0	0
	= (similar)	0	0	0	1	0	0	0	0	0
	− (worse)	30	30	30	29	30	30	30	30	30
GWO	+ (better)	2	2	2	6	7	5	7	7	11
	= (similar)	1	4	3	4	3	0	5	8	0
	− (worse)	27	24	25	20	20	25	18	15	19
MFO	+ (better)	3	4	3	4	5	4	1	1	0
	= (similar)	1	0	2	1	1	1	3	3	5
	− (worse)	26	26	25	25	24	25	26	26	25
LSA	+ (better)	1	2	2	9	11	8	6	8	9
	= (similar)	4	1	1	6	4	5	8	6	4
	− (worse)	25	27	27	15	15	17	16	16	17
SCA	+ (better)	0	0	0	0	1	0	0	0	0
	= (similar)	0	0	0	0	0	1	0	0	0
	− (worse)	30	30	30	30	29	29	30	30	30
CSA	+ (better)	1	1	0	4	3	8	5	2	4
	= (similar)	1	1	3	9	10	3	3	7	3
	− (worse)	28	28	27	17	17	19	22	21	23
CGSA	+ (better)	1	0	0	0	0	0	0	0	0
	= (similar)	0	0	1	0	0	0	0	0	0
	− (worse)	29	30	29	30	30	30	30	30	30
SSA	+ (better)	3	2	2	7	8	9	5	7	11
	= (similar)	0	0	1	4	9	7	6	5	3
	− (worse)	27	28	27	19	13	14	19	18	16
MS	+ (better)	5	6	6	7	7	7	1	1	1
	= (similar)	0	0	2	3	1	2	0	0	1
	− (worse)	25	24	22	20	22	21	29	29	28
BOA	+ (better)	0	0	1	3	3	3	0	0	0
	= (similar)	0	0	1	2	3	3	0	0	0
	− (worse)	30	30	28	25	24	24	30	30	30

3.3.3. Algorithm complexity

Experimental studies were conducted to test the computational complexities of the algorithms based on the IEEE CEC 2014 [61] definition document. The time complexities of DSOS-1 and the other algorithms for 30, 50, and 100-dimensional functions are shown in Table 19. The results of time complexity show that the most complex algorithms are GSA, CGSA, MS, LSA, and HSS. Alternatively, the least time complexity is observed with CSA, BOA, and BSA. The time complexities of DSOS-1, GWO, MFO, SCA, and SSA are similar.

Fig. 6 is plotted to more visible present the computational complexity. The average computation time of DSOS-1 in dimensions 30, 50 and 100 is approximately 224. Especially in multimodal problems, the most successful algorithms after DSOS1 are BSA and GWO. The average of the computational complexities of these two algorithms is 48 and 147, respectively. The computational complexity of the DSOS-1 algorithm is lower than the average compared to those of competing algorithms.

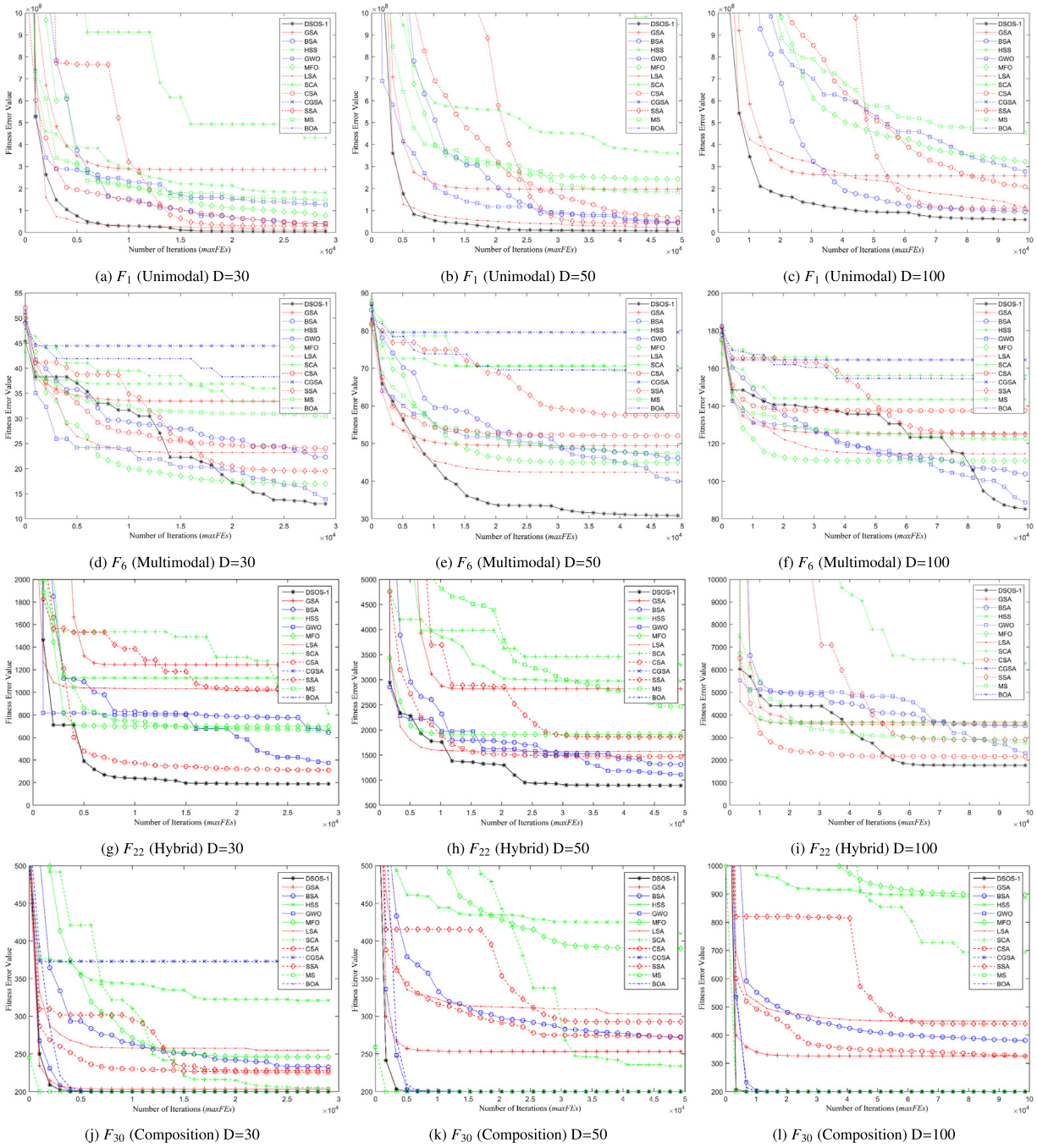
Table 19
Algorithm complexity.

	D = 30	D = 50	D = 100
T0	2,16E−02	2,16E−02	2,16E−02
T1	8,20E−01	1,36E+00	4,66E+00
DSOS-1	1,97E+02	1,99E+02	2,52E+02
GSA	5,52E+02	8,64E+02	1,56E+03
BSA	1,74E+01	4,29E+01	8,49E+01
HSS	2,53E+02	2,85E+02	3,64E+02
GWO	9,50E+01	1,26E+02	2,21E+02
MFO	8,02E+01	1,00E+02	1,60E+02
LSA	4,15E+02	5,59E+02	8,96E+02
SCA	8,09E+01	1,04E+02	1,71E+02
CSA	−1,23E+01	−7,21E+00	8,00E−01
CGSA	6,03E+02	8,94E+02	1,69E+03
SSA	9,78E+01	1,07E+02	1,98E+02
MS	8,13E+02	1,25E+03	2,43E+03
BOA	5,58E+01	6,08E+01	7,71E+01

4. Conclusion and future work

In this study, a new selection method called FDB, which provides effective guidance on meta-heuristic search algorithms in the search process, has been proposed. The FDB selection method takes into account the fitness values of the solution candidates as well as their distance to the best solution in the current population. A comprehensive experimental study has been conducted to test and verify the extent to which the proposed method provided the intended outcomes. For this purpose, the FDB selection method has first applied to the SOS process, which is a successful and up to date MHS algorithm. Search performances of both the SOS algorithm and FDB-based SOS variations have been tested and compared. The test process was carried out in 30, 50 and 100 dimensions of 30 classical benchmark problems that are used most frequently in the literature. For each test function, 51 independent runs were implemented. Furthermore, the results of the Wilcoxon statistical test at 5% significance difference was reported. The convergence analysis of FDB-based

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.



(a) F_1 (Unimodal) D=30 (b) F_1 (Unimodal) D=50 (c) F_1 (Unimodal) D=100

(d) F_6 (Multimodal) D=30 (e) F_6 (Multimodal) D=50 (f) F_6 (Multimodal) D=100

(g) F_{22} (Hybrid) D=30 (h) F_{22} (Hybrid) D=50 (i) F_{22} (Hybrid) D=100

(j) F_{30} (Composition) D=30 (k) F_{30} (Composition) D=50 (l) F_{30} (Composition) D=100

Fig. 4. Convergence curves on multi-dimensional (30-50-100) CEC2014 test functions.

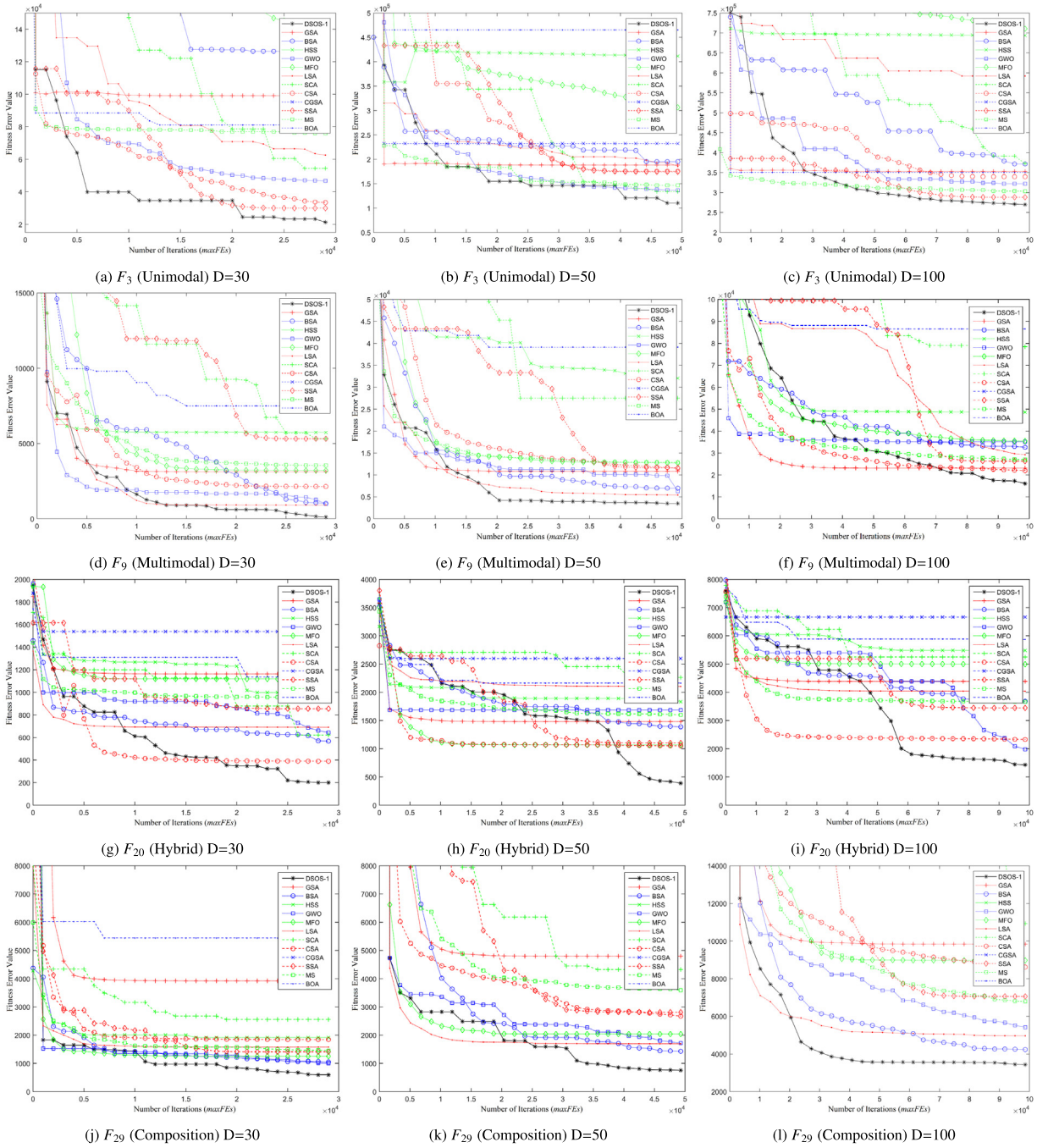
SOS variants and the SOS algorithm for unimodal and multimodal was investigated.

The experimental results obtained in the testing process show that the FDB selection method improves both the exploitation and exploration ability of the SOS algorithm. Moreover, all FDB-based SOS variants are able to establish a clear superiority over the base SOS algorithm. This result indicates that the FDB selection method meets the intended outcomes and shows its stability and success.

Following the test studies, verification studies were carried out. In the validation process, a comprehensive experimental study was conducted between the FDB-based SOS algorithm and the current and most powerful 13 algorithms developed in the last 10 years with. In these studies, 90 benchmark problems including unimodal, multimodal, hybrid and composition types were used. Benchmark functions were designed in 30, 50 and 100 dimensions, and 51 independent runs were applied. In this respect, validation studies of the FDB method are one of the most comprehensive experiments in the literature. The results obtained from the experimental studies show that the FDB-based SOS algorithm has consistently discovered better solutions in all problem types and dimensions, is resistant to local solution traps and provides strong convergence.

The FDB selection method has the potential to be applied to thousands of studies. In future studies, it is expected that the FDB selection method will be applied to other MHS algorithms. In addition, studies should be carried out on the development of FDB-based search operators.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.



(a) F_3 (Unimodal) D=30 (b) F_3 (Unimodal) D=50 (c) F_3 (Unimodal) D=100

(d) F_9 (Multimodal) D=30 (e) F_9 (Multimodal) D=50 (f) F_9 (Multimodal) D=100

(g) F_{20} (Hybrid) D=30 (h) F_{20} (Hybrid) D=50 (i) F_{20} (Hybrid) D=100

(j) F_{29} (Composition) D=30 (k) F_{29} (Composition) D=50 (l) F_{29} (Composition) D=100

Fig. 5. Convergence curves on multi-dimensional (30-50-100) CEC2017 test functions.

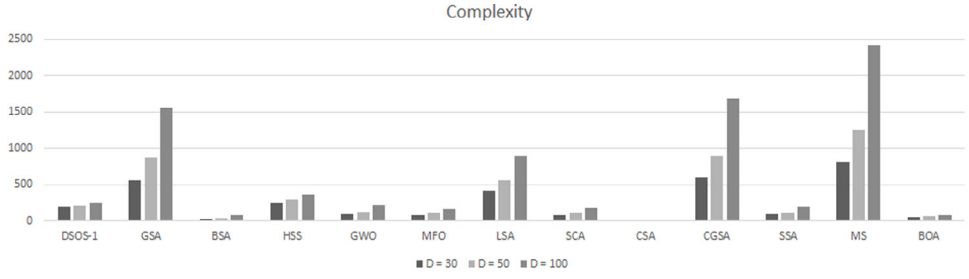


Fig. 6. Algorithm complexity for dimension 30, 50 and 100.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

Appendix

FDB-SOS algorithm code can be downloaded from the link below:

https://www.mathworks.com/matlabcentral/fileexchange/72311-fdb-sos

References

[1] A.P. Piotrowski, J.J. Napiorkowski, Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure? Swarm Evol. Comput. 43 (2018) 88–108.
[2] A.W. Mohamed, A.A. Hadi, K.M. Jambi, Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization, Swarm Evol. Comput. (2018) 100455, http://dx.doi.org/10.1016/j.swevo.2018.10.006.
[3] N. Di Cesare, M. Domaszewski, A new hybrid topology optimization method based on I-PR-PSO and ESO. Application to continuum structural mechanics, Comput. Struct. 212 (2019) 311–326.
[4] D.T. Le, D.-K. Bui, T.D. Ngo, Q.-H. Nguyen, H. Nguyen-Xuan, A novel hybrid method combining electromagnetism-like mechanism and firefly algorithms for constrained design optimization of discrete truss structures, Comput. Struct. 212 (2019) 20–42.
[5] 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, Inform. Sci. 422 (2018) 122–143.
[6] S. Torabi, F. Safi-Esfahani, Improved raven roosting optimization algorithm (IRRO), Swarm Evol. Comput. 40 (2018) 144–154.
[7] B. Jana, S. Mitra, S. Acharyya, Repository and mutation based particle swarm optimization (RMPSO): A new PSO variant applied to reconstruction of gene regulatory network, Appl. Soft Comput. 74 (2019) 330–355.
[8] A. Zhang, G. Sun, J. Ren, X. Li, Z. Wang, X. Jia, A dynamic neighborhood learning-based gravitational search algorithm, IEEE Trans. Cybern. 48 (1) (2016) 436–447.
[9] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2014) 191–204.
[10] S. Salcedo-Sanz, Modern meta-heuristics based on nonlinear physics processes: A review of models and design procedures, Phys. Rep. 655 (2016) 1–70.
[11] 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, Adv. Eng. Softw. 114 (2017) 163–191.
[12] L. Wang, B. Yang, J. Orchard, Particle swarm optimization using dynamic tournament topology, Appl. Soft Comput. 48 (2016) 584–596.
[13] Q. Qin, S. Cheng, Q. Zhang, L. Li, Y. Shi, Particle swarm optimization with interswarm interactive learning strategy, IEEE Trans. Cybern. 46 (10) (2015) 2238–2251.
[14] L.T. Al-Bahrani, J.C. Patra, A novel orthogonal PSO algorithm based on orthogonal diagonalization, Swarm Evol. Comput. 40 (2018) 1–23.
[15] W. Gao, Z. Wei, Y. Luo, J. Cao, Artificial bee colony algorithm based on Parzen window method, Appl. Soft Comput. 74 (2019) 679–692.
[16] M.Z. Ali, N.H. Awad, R.G. Reynolds, P.N. Suganthan, A balanced fuzzy cultural algorithm with a modified levy flight search for real parameter optimization, Inform. Sci. 447 (2018) 12–35.
[17] X. Han, Q. Liu, H. Wang, L. Wang, Novel fruit fly optimization algorithm with trend search and co-evolution, Knowl.-Based Syst. 141 (2018) 1–17.
[18] S. Gupta, K. Deep, Improved sine cosine algorithm with crossover scheme for global optimization, Knowl.-Based Syst. 165 (2019) 374–406.
[19] W.-f. Gao, S.-y. Liu, L.-l. Huang, A novel artificial bee colony algorithm based on modified search equation and orthogonal learning, IEEE Trans. Cybern. 43 (3) (2013) 1011–1024.
[20] F. Harfouchi, H. Habbi, C. Ozturk, D. Karaboga, Modified multiple search cooperative foraging strategy for improved artificial bee colony optimization with robustness analysis, Soft Comput. 22 (19) (2018) 6371–6394.
[21] Q. Huang, K. Zhang, J. Song, Y. Zhang, J. Shi, Adaptive differential evolution with a Lagrange interpolation argument algorithm, Inform. Sci. 472 (2019) 180–202.
[22] S. Elsayed, R. Sarker, C.A. Coello, Sequence-based deterministic initialization for evolutionary algorithms, IEEE Trans. Cybern. 47 (9) (2016) 2911–2923.
[23] G. Sun, P. Ma, J. Ren, A. Zhang, X. Jia, A stability constrained adaptive alpha for gravitational search algorithm, Knowl.-Based Syst. 139 (2018) 200–213.
[24] W. Long, J. Jiao, X. Liang, M. Tang, An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization, Eng. Appl. Artif. Intell. 68 (2018) 63–80.
[25] X. Cai, X.-z. Gao, Y. Xue, Improved bat algorithm with optimal forage strategy and random disturbance strategy, Int. J. Bio-Inspired Comput. 8 (4) (2016) 205–214.
[26] Z. Cui, M. Zhang, H. Wang, X. Cai, W. Zhang, A hybrid many-objective cuckoo search algorithm, Soft Comput. 23 (21) (2019) 10681–10697.
[27] M. Črepinšek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey, ACM Comput. Surv. 45 (3) (2013) 35.
[28] K. Ng, C. Lee, F.T. Chan, Y. Lv, Review on meta-heuristics approaches for airside operation research, Appl. Soft Comput. 66 (2018) 104–133.
[29] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm Evol. Comput. 44 (2019) 148–175.
[30] D. Tian, Z. Shi, MPSO: Modified particle swarm optimization and its applications, Swarm Evol. Comput. 41 (2018) 49–68.
[31] A. Cheraghalipour, M. Hajiaghaei-Keshteli, M.M. Paydar, Tree Growth Algorithm (TGA): A novel approach for solving optimization problems, Eng. Appl. Artif. Intell. 72 (2018) 393–414.
[32] A. Sadollah, H. Sayyaadi, D.G. Yoo, H.M. Lee, J.H. Kim, Mine blast harmony search: a new hybrid optimization method for improving exploration and exploitation capabilities, Appl. Soft Comput. 68 (2018) 548–564.
[33] L. Wu, Q. Liu, X. Tian, J. Zhang, W. Xiao, A new improved fruit fly optimization algorithm IAFOA and its application to solve engineering optimization problems, Knowl.-Based Syst. 144 (2018) 153–173.
[34] A.P. Piotrowski, J.J. Napiorkowski, P.M. Rowinski, How novel is the "novel" black hole optimization approach? Inform. Sci. 267 (2014) 191–200.
[35] M.-Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112.
[36] H. Nenavath, R.K. Jatoth, Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking, Appl. Soft Comput. 62 (2018) 1019–1043.
[37] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, MIT Press, Cambridge, MA, USA, 1992.
[38] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Ieee, 1995, pp. 39–43, http://dx.doi.org/10.1109/MHS.1995.494215.
[39] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. 179 (13) (2009) 2232–2248.
[40] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.
[41] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471.
[42] A. Kaveh, V. Mahdavi, Colliding bodies optimization: a novel meta-heuristic method, Comput. Struct. 139 (2014) 18–27.
[43] A. Kaveh, A. Dadras, A novel meta-heuristic optimization algorithm: thermal exchange optimization, Adv. Eng. Softw. 110 (2017) 69–84.
[44] S. Mirjalili, A.H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, Appl. Soft Comput. 53 (2017) 407–419.
[45] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249.
[46] G.-G. Wang, Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems, Memet. Comput. 10 (2) (2018) 151–164.
[47] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, Comput. Struct. 169 (2016) 1–12.
[48] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, Soft Comput. 23 (3) (2019) 715–734.
[49] P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems, Appl. Math. Comput. 219 (15) (2013) 8121–8144.
[50] P.R. Singh, M.A. Elaziz, S. Xiong, Modified spider monkey optimization based on Nelder–Mead method for global optimization, Expert Syst. Appl. 110 (2018) 264–289.
[51] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (1) (2013) 17–35.
[52] H. Shareef, A.A. Ibrahim, A.H. Mutlag, Lightning search algorithm, Appl. Soft Comput. 36 (2015) 315–333.
[53] C. Caraveo, F. Valdez, O. Castillo, A new optimization meta-heuristic algorithm based on self-defense mechanism of the plants with three reproduction operators, Soft Comput. 22 (15) (2018) 4907–4920.
[54] G. Dhiman, V. Kumar, Emperor penguin optimizer: A bio-inspired algorithm for engineering problems, Knowl.-Based Syst. 159 (2018) 20–50.
[55] A. Lopez-Rincon, A. Tonda, M. Elati, O. Schwander, B. Piwowarski, P. Gallinari, Evolutionary optimization of convolutional neural networks for cancer miRNA biomarkers classification, Appl. Soft Comput. 65 (2018) 91–100.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.

[56] M. Tian, X. Gao, Differential evolution with neighborhood-based adaptive evolution mechanism for numerical optimization, Inform. Sci. 478 (2019) 422–448.

[57] S. Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133.

[58] P. Bujok, J. Tvrdík, R. Poláková, Comparison of nature-inspired population-based algorithms on continuous optimisation problems, Swarm Evol. Comput. (2019) 100490, http://dx.doi.org/10.1016/j.swevo.2019.01.006.

[59] S. Das, A. Abraham, U.K. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE Trans. Evol. Comput. 13 (3) (2009) 526–553.

[60] R.L. Rardin, R. Uzsoy, Experimental evaluation of heuristic optimization algorithms: A tutorial, J. Heuristics 7 (3) (2001) 261–304.

[61] J. Liang, B. Qu, P. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Vol. 635, Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Nanyang Technological University, Singapore, 2013.

[62] N. Awad, M. Ali, J. Liang, B. Qu, P. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Technical Report, 2016.

[63] H. Karami, M.J. Sanjari, G.B. Gharehpetian, Hyper-spherical search (HSS) algorithm: a novel meta-heuristic algorithm to optimize nonlinear functions, Neural Comput. Appl. 25 (6) (2014) 1455–1465.

Please cite this article as: H.T. Kahraman, S. Aras and E. Gedikli, Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems (2019) 105169, https://doi.org/10.1016/j.knosys.2019.105169.