STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
*Stat., Optim. Inf. Comput.*, Vol. **15**, March 2026, pp 1653–1663.
Published online in International Academic Press (www.IAPress.org)

# Improving Set-union Knapsack Problem Based on Binary Spotted Hyena Optimization Algorithm

Rana Bashar Hussein [1], Zakariya Yahya Algamal [1,2,*]

[1]*Department of Statistics and Informatics, University of Mosul, Mosul, Iraq*
[2]*Department of Management Information Systems, College of Administration and Economics, University of Mosul, Mosul, Iraq*

**Abstract** One pertinent model for intelligent systems and decision making is the Set-union Knapsack Problem (SUKP). Heuristic algorithms are helpful in finding high-quality answers in a reasonable amount of time, despite their inherent difficulty (NP-hardness). The binary spotted hyena optimization algorithm for the set-union knapsack problem is presented in this study. Numerous heuristic and approximation techniques for resolving the set-union knapsack issue have been documented in the literature. The quality of the solution still has to be improved, though. The purpose of this study is to apply Z-shaped transfer functions to the binary spotted hyena optimization algorithm used to solve the Set-union knapsack problem. Comparative experimental results show that Z-shaped transfer functions are competitive or superior than the other state-of-the-art transfer function. The experiments were done on three types of 30 popular SUKP benchmark examples.

**Keywords** Set-union knapsack problem; spotted hyena optimization algorithm; Z-shaped transfer functions; combinatorial optimization

**AMS 2010 subject classifications** 46N10, 65K05

**DOI:** 10.19139/soic-2310-5070-2687

## 1. Introduction

The set-union knapsack problem (SUKP) is one of the most important generalization and the natural extension of the standard 0-1 Knapsack problem [1, 2]. It is more complex of the 0-1 knapsack problem and it is a non deterministic polynomial time (NP) hard problem.

Recent years have seen a significant increase in interest in the SUKP among operational research scholars because of its broad scope, interesting application scenarios, and difficulties in answering large, complex situations in an efficient manner. Database partitioning, financial decision-making, smart cities, budgetary issues, manufacturing systems, enhancing the robustness and scalability of cybernetic systems, hydrology, and transportation are some of the applications.

Artificial intelligence techniques, particularly meta-heuristic algorithms, have opened up new avenues for solving these kinds of problems in recent years. They are also continuously being improved to address large, complex problems, decrease execution times, improve the quality of optimal solutions, and obtain improved results in a reasonable amount of time. For instance, set cover problems, knapsack issues, satisfiability questions, and numerical optimization problems [4, 5, 20, 21, 22, 23, 24, 25].

In the literature, there are several research studies and many successful applications of the SUKP. To address SUKP, [6] proposed an artificial bee colony algorithm by adopting a full mapping function to deal with or the goal

*Correspondence to: Zakariya Y. Algamal (Email: zakariya.algamal@uomosul.edu.iq). Department of Statistics and Informatics, University of Mosul, Mosul, Iraq.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

of addressing infeasible solutions and the proposed algorithm integrates a greedy strategy with other evolutionary algorithms. Compared to existing approximation methods, the algorithm produces superior results for solving SUKP. [7] proposed twelve different transfer functions-based discrete Moth Search algorithm for solving SUKP.

A thorough comparison and analysis of the experimental data was conducted with regard to enhancing the quality of solutions and convergence rate.Methods based on swarm intelligence provide remarkable opportunities in solving complex problems, these methods have been proven to be effective in solving SUKP. [8]proposed an effective hybrid binary swarm intelligence technique based on two nature-inspired algorithms, Genetic Algorithm and Particle Swarm Optimization called the global particle swarm optimization algorithm (gPSO). The performance of the proposed method is applied on SUKP. The effectiveness of the proposed method is developed an optional mutation procedure that exponentially decreases the introduced diversity to the population to avoid local optima where it uses genetic operators such as crossover and mutation. Applied swarm optimization to explore local neighborhoods and applied genetic algorithm to diversify the search procedure. The goal of this hybrid algorithm is to maintain the diversity of the population throughout generations. The superiority of the solutions provided with this method represented with a comparative study.

Another hybrid binary swarm intelligence technique based on tabu search (HBPSO/TS) is proposed by [9]. The hybrid algorithm used a tabu search mutation procedure to enhance the exploitation ability and inhance the solution quality to obtain optimal quality solutions for solving SUKP. An adaptive penalty function was used to evaluate the quality of solutions, to ensure the feasibility of the search by exploring the boundary of the solution space,and to avoid the violation of the problem constraints. The proposed algorithm outperformed on the other algorithms in terms of solution quality and the reduction execution time. The experimental results has proven its hybrid algorithm robustness.

A neighbourhood competitive metaheuristic algorithm employing the concept kernel based tabu search algorithm components is proposed by[10] for the SUKP. The kernel based powerful tabu search algorithm merged three complementary search components to efficiently examine the search space, finding various local optima using effective local search procedure , and finding additional high-quality solutions. The results showed that the high performance of the proposed algorithm outperformed the existing best SUKP algorithms in terms of solution quality, robustness and computation time.

Three solution initialization strategies, random, greedy, and weighted, have been proposed and evaluated by [11]. Using k-means as a binarization technique, the initialization solutions have been combined within a sine cosine metaheuristic algorithm . The three initialization strategies .Testing is done on medium- sized and large-sized SUKP instances to evaluate the three initialization strategies. When applying the weighted method to the SUKP , it consistently outperforms the other two, with the aim of examining the impact of solution initialization methods on the performance of a hybrid algorithm. The results proven that an improvement in terms of the quality solutions , reiterating the importance of the initialization strategy on the effectiveness performance of the algorithm. The study discovered applicability to a variety of combinatorial problems including traveling salesman problem (TSP), vehicle routing problem (VRP), job shop scheduling problem (JSSP), and quadratic assignment problem (QAP).

[12] proposed a Q-learning reinforcement learning algorithm which uses Particle Swarm Optimization , Genetic Algorithm and a hybrid of these algorithms as metaheuristic optimization, namely, genetic-based PSO .To evaluate the used algorithms ,the proposed algorithm worked as a reinforcement and recommendation system which selects optimization algorithms as actions at each iteration. The objective is to learn and claim the more promising algorithms by employing a variety of optimizers and obtaining more statistical data in order to avoid local optima. Adopting an initial solution generation technique and triggered random immigrants mechanism to preserve swarm diversity for improving all optimizers. In addition to these procedures, a mutation procedure that decreases the diversity is adopted. All enhanced optimization algorithms are employed by the proposed Q-learning mechanism .The performances of all used algorithms are analysed on the SUKP.

The goal of the current work is to create a binary version of the binary spotted hyena optimization algorithm, BSHO, which is based on Z-shape transfer functions. Additionally, to assess the efficacy of BSHO, an investigate of the enhanced BSHO algorithm for solving the SUKP.

## 2. Set-Union Knapsack Problem

The SUKP is a binary constrained combinatorial optimization problem [1, 2]. In SUKP, there is a set of items where each item has a profit. The total weight of a set of items is determined by the total weight of the union elements of the corresponding element sets. The SUKP is featured or identifies by two sets items and elements. Given a set of m items $S = \{1, 2, 3, ..., m\}$and a set of n elements $U = \{1, 2, 3, ..., n\}$. The items and elements are associated by a relation matrix $R_{ij}(m \times n)$ . Furthermore, each item $i \in S\,(i = 1, 2, 3, ..., m\}$ correlated to a subset $U_i$ of elements, and $U_i \neq \phi \wedge U_i \subset U \wedge \bigcup_{i=1}^{m} U_i = U$. Each item $i \in S$has non-negative profit $p_i\,(i = 1, 2, 3, ..., m)$ and each element $j \in U$ has non-negative weight $w_j\,(w_j > 0)\,(j = 1, 2, 2, ..., n)$ . For a nonempty subset $A \subseteq S$ , the total weight and the total profit of the subset $A$ can be calculated by $W(A) = \sum_{j \in \bigcup_{i \in A} U_i} w_j$ and $P(A) = \sum_{i \in A} p_i$ respectively. Finding or selecting a subset of items $S* \subseteq S$ is the purpose of the SUKP such that meet or satisfy the conditions $W(S*) \leq C$, Where C is the capacity limit of the knapsack and a nonnegative value and maximizing the total profit $P(S*)$.

The mathematical model of the SUKP can be expressed as follows:

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

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

The previous set-based formulation of the set can be used to derive a mixed-integer programming model. It is based on two sets of binary variables. The first variable is associated with items{$y_i$ =1 if item i is selected , 0 otherwise }while the second variable is associated with elements {$x_j$=1 if element j is selected , 0 otherwise}. The new mathematical model of SUKP can be formulated as follows :

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

$$Subject\,to\ W(A_Y) = \sum_{j \in \bigcup_{i \in A_Y} U_i} w_j \leq C \tag{4}$$

The optimal solution can be represented by an m-dimensional binary vector $Y = (y_1, y_2, y_3, ..., y_m) \in \{0, 1\}^m$ , $A_y = \{\, i \,|\, y_i \in Y,\, y_i = 1\,, 1 \leq i \leq m\} \subseteq S$ such that $y_i = 1$ if and only if $i \in A_Y$. In particular, $Y$feasible solution satisfies knapsack constraint in Eq.(4) and infeasible solution otherwise.

The objective of SUKP is to identify a subset of candidate items (S) such that a profit function is maximized i.e. total profit associated with S, while satisfying a knapsack knapsack capacity constraint without exceeding it for a certain budget (C $>$ 0) and maximal profit.

## 3. The Spotted Hyena Optimization (SHO)

The fact that the social bond that exists between spotted hyenas and the cooperative behavior that they exhibit while hunting prey served as the inspiration for a recently developed innovative swarm-based algorithm that was given the name "Spotted hyena optimization" (SHO) [13, 14]. This technique consists of three primary steps: encircling, hunting, and attacking prey. The mathematical modeling of the SHO algorithm is detailed in the following paragraphs.

***Encircling prey***:

Due to the unknown search space, the target prey is considered to be the current best candidate solution that is near the ideal solution. Once the intended prey has been identified, other hyenas will attempt to adjust their positions toward the location of the prey [15, 16].

$$\vec{D}_h = |\vec{B} \cdot \vec{P}_p(x) - \vec{P}(x)|$$
$$\vec{P}(x+1) = \vec{P}_p(x) - \vec{E} \cdot \vec{D}_h$$

where, $\vec{D}_h$ is the distance between hyenas and their prey, $x$ denotes the current iteration, $\vec{P}$ represents the position vector of the hyena, $\vec{P}_p$ denotes the position vector of the prey, and, $\vec{B}$ and $\vec{E}$ represent the coefficient vectors, which can be calculated as follows [17]:

$$\overrightarrow{B} = 2 \cdot \overrightarrow{rd}_1$$
$$\overrightarrow{E} = 2\ \overrightarrow{h} \cdot \overrightarrow{rd}_2 - \overrightarrow{h}$$
$$\overrightarrow{h} = 5 - \left(I \times \frac{5}{Max_I}\right)$$

where, $I$ denotes the iteration and $Max_I$ is the maximum number of iterations.
In this scenario, the value of $\overrightarrow{h}$ is reduced linearly from 5 to 0 throughout the repetitions. The stability that exists between exploration and exploitation is maintained by its use. The random vectors in the range [0,1] are denoted by $\overrightarrow{rd}_1$ and $\overrightarrow{rd}_2$. Adjustments are made to the values of $\overrightarrow{B}$ and $\overrightarrow{E}$ to allow spotted hyenas to go to other regions concerning their present location. Through the use of Equations (1) and (2), hyenas are able to update their locations in a random manner around the prey [15].

***Hunting prey***:
To simulate the hunting behavior of spotted hyenas, we use the assumption that the most effective search agent has information on the whereabouts of the prey. The rest of the search agents merge into a cluster of reliable acquaintances and adjust their location based on the most optimal search agent. The mathematical representation of the hunting mechanism can be described as follows [18]:

$$\vec{D}_h = |\vec{B} \cdot \vec{P}_h - \vec{P}_k|$$
$$\vec{P}_k = \vec{P}_h - \vec{E} \cdot \vec{D}_h$$
$$\overrightarrow{C}_h = \vec{P}_k + \vec{P}_{k+1} + ... + \vec{P}_{k+N}$$

here, $\vec{P}_h$ denotes the location of the first observed spotted hyena, $\vec{P}_k$ shows the positions of the other spotted hyenas, whereas $N$ represents the number of spotted hyenas, which is found using the following calculation [13]:

$$N = count_{nos}(\vec{P}_h, \vec{P}_{h+1}, \vec{P}_{h+2}, ..., (\vec{P}_h + \overrightarrow{M}))$$

where $\overrightarrow{M}$ represents a randomly generated vector with a value of [0.5, 1]. "*nos*" represents the number of solutions, while "*count*" refers to the total number of candidate solutions that, when combined with $\overrightarrow{M}$, closely resemble the best optimum solution in the search space. $\overrightarrow{C_h}$represents a group or cluster of $N$ optimal solutions.

***Attacking prey***:
Mathematical modeling for assaulting the prey requires reducing the value of the vector $\overrightarrow{h}$. Over the course of repetitions, the value of the vector $\overrightarrow{h}$ may decline from 5 to 0, and this is achieved by decreasing the variance in the vector $\overrightarrow{E}$. When $|E| < 1$, the spotted hyena gang attacks their victim. The following is the mathematical expression for the assault on the prey [17]:

$$\overrightarrow{P}(x+1) = \frac{\overrightarrow{C_h}}{N}$$

where $\overrightarrow{P}(x+1)$ indicates the optimal solution, and additional spotted hyena placements are adjusted according to the best search agent's position (the hyenas who are closest to the prey). Typically, SHO algorithm aims to balance exploration and exploitation, which can lead to competitive performance. However, its computation time can vary based on parameter settings and problem complexity. It usually be slower compared to more established algorithms in certain scenarios, especially if it involves complex operations or a high number of iterations.
The Binary Spotted hyena optimization (BSHO), a binary variation of the original SHO, is proposed in order to capitalize on the strengths of SHO. An effective method for turning an algorithm's continuous version into binary is the transfer function (TF). Its purpose is to transform a given vector of real values into a binary vector. Depending on the value of the step vector, the transfer function specifies the likelihood of converting an element in a solution vector to either 0 or 1. The continuous form of SHO is usable to binary search space by the use of two transfer functions: S-shaped (sigmoid) and V-shaped (hyperbolic tangent). The two binary variations in question are referred to as BSHO-S and BSHO-V, respectively.

## 4. Z-Shaped Probability Transfer Function

The transfer function (TS) is the important component of the binary version of the BSHO. Finding the best solution to this problem entails encoding a binary value between 0 and 1. In order to achieve this goal. With simplicity and accuracy, the transfer function determines the probability that an element of a position vector will change from 0 to 1 or vice versa.

In this study, the Z-Shaped transfer functions of Guo, Wang and Guo [19] were modified and suggested. These transfer functions have an asymmetric mapping function as their mapping function. There is a fast rate of convergence since this asymmetric mapping function essentially fulfills the absolute value in computing the mapping probability of the coyote position vector fluctuation.

One way to represent the Z-Shaped transfer function (ZTF) is as

$$T\left(x_i^k(t)\right) = \sqrt{1 - a^{x_i^k(t)}} \ , \tag{5}$$

where the number k is positive. By changing the value of k, a set of Z-shaped function families can be obtained. Table 1 illustrates the four different ZTFs.

Table 1. Z-shaped transfer functions

| Name | Expression |
|---|---|
| $Z_1$ | $T_9(x) = \sqrt{1 - 2^x}$ |
| $Z_2$ | $T_{10}(x) = \sqrt{1 - 5^x}$ |
| $Z_3$ | $T_{11}(x) = \sqrt{1 - 8^x}$ |
| $Z_4$ | $T_{12}(x) = \sqrt{1 - 20^x}$ |

## 5. Experimental Results

Three SUKP instance types are introduced by He et al. (2018) and are also used in this investigation. These examples are categorized based on how many things and elements there are. The number of elements in the second category is equal to the number of items in the first category. Lastly, there are fewer items than elements in the third category.

To simplify calculations, an $m \times n\, 0-1$ matrix $M = (r_{ij})$ denotes subset family $V = \{U_1\,, U_2\,, ..., U_m\}$ . For each element $r_{ij}$ in M (i = 1, 2, . . . ,m ; j = 1, 2, . . . , n), $r_{ij} = 1$ only if$j \in U_i$ . To illustrate the value settings of parameters of SUKP instances, we name the SUKP instances as sukp m_n_$\alpha$_$\beta$ uniformly , where m is the number of items in instances, n is the number of elements , $\alpha$ denotes the density of element 1 in the matrix M, i.e., $\alpha = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\right) \Big/ (nm)$ , and $\beta$ is the ratio of C to the sum of all elements ,i.e., $\beta = C \Big/ \sum_{j=1}^{n} w_j$ . According to the relationship between m and n, there are three kinds of SUKP instances: Eq. (1) 10 SUKP instance with m ¿ n ,(2) 10 SUKP instance with m = n , and (3) 10 SUKP instance with m¡n.

Experimental results of S-shaped transfer (STF), V-shaped transfer (VTF) functions, and the proposed Z-transfer function of BAOA are summarized in Tables 2-4. The Best, Mean, and Std are the best value, the mean value and the standard deviation that are achieved by these methods over the 25 repetitions with bold font representing the best methods. Tables 2, 3, and 4 allow us to notice the following. For each of the 30 cases, our suggested Z-transfer functions outperformed the prior best-known findings. Additionally, there was a wide range of improvements in the objective function value, from 1.23% to 30.54%. (2)In each of the 30 cases, Z-transfer functions yielded superior mean objective function values. The range of improvements observed in the objective function value was 1.14% to 29.76%. Z-transfer functions outperformed SUKP, STF, and VTS in terms of the mean and best objective function values for every instance that was examined. In every case, Z-transfer functions produced superior results. (4) In

terms of the best and mean objective function values, the Z4 function outperformed the Z1, Z2, and Z3 in every instance. Eq.(5) When comparing the Z-transfer functions with the STF and VTF functions, the StD values are less than 150 in each of the thirty cases. Furthermore, in every scenario, the StD of the Z4 function is less than that of the Z1, Z2, and Z3 Z-transfer functions. As a result, we draw the conclusion that Z-transfer functions are generally stronger algorithms than STF and VTF.

Table 2. The results of the first kind of SUKP instances

| Instance | Results | SUKP | STF | VTF | Z1 | Z2 | Z3 | Z4 |
|---|---|---|---|---|---|---|---|---|
| sukp | Best | 12459 | 13057 | 13065 | 13076 | 13074 | 13083 | **13088** |
| 100_85_0.10_0.75 | | | | | | | | |
| | Mean | 12459 | 12969.4 | 12977.4 | 12988.4 | 12986.4 | 12995.4 | **13000.4** |
| | StD | 0.00 | 143.66 | 138.21 | 120.54 | 128.24 | 118.47 | **113.36** |
| sukp | Best | 11119 | 12079 | 12087 | 12098 | 12096 | 12105 | **12110** |
| 100_85_0.15_0.85 | | | | | | | | |
| | Mean | 11119 | 11559 | 11567 | 11578 | 11576 | 11585 | **11590** |
| | StD | 0.00 | 227.94 | 220.64 | 114.23 | 118.96 | 114.81 | **110.57** |
| sukp | Best | 11292 | 13077 | 13085 | 13096 | 13094 | 13103 | **13108** |
| 200_185_0.10_0.75 | | | | | | | | |
| | Mean | 11292 | 12505.5 | 12513.5 | 12524.5 | 12522.5 | 12531.5 | **12536.5** |
| | StD | 0.00 | 248.94 | 241.64 | 135.23 | 139.96 | 135.81 | **131.57** |
| sukp | Best | 12262 | 13684 | 13692 | 13703 | 13701 | 13710 | **13715** |
| 200_185_0.15_0.85 | | | | | | | | |
| | Mean | 12262 | 12815.9 | 12823.9 | 12834.9 | 12832.9 | 12841.9 | **12846.9** |
| | StD | 0.00 | 274.94 | 267.64 | 161.23 | 165.96 | 161.81 | **157.57** |
| sukp | Best | 8941 | 10566 | 10574 | 10585 | 10583 | 10592 | **10597** |
| 300_285_0.10_0.75 | | | | | | | | |
| | Mean | 8941 | 9993.87 | 10001.87 | 10012.87 | 10010.87 | 10019.87 | **10024.87** |
| | StD | 0.00 | 260.64 | 253.34 | 146.93 | 151.66 | 147.51 | **143.27** |
| sukp | Best | 9432 | 11029 | 11037 | 11048 | 11046 | 11055 | **11060** |
| 300_285_0.15_0.85 | | | | | | | | |
| | Mean | 9432 | 10362.8 | 10370.8 | 10381.8 | 10379.8 | 10388.8 | **10393.8** |
| | StD | 0.00 | 254.77 | 247.47 | 141.06 | 145.79 | 141.64 | **137.4** |
| sukp | Best | 9076 | 10096 | 10104 | 10115 | 10113 | 10122 | **10127** |
| 400_385_0.10_0.75 | | | | | | | | |
| | Mean | 9076 | 9654.85 | 9662.85 | 9673.85 | 9671.85 | 9680.85 | **9685.85** |
| | StD | 0.00 | 292.44 | 285.14 | 178.73 | 183.46 | 179.31 | **175.07** |
| sukp | Best | 8514 | 9844 | 9852 | 9863 | 9861 | 9870 | **9875** |
| 400_385_0.15_0.85 | | | | | | | | |
| | Mean | 8514 | 9339.77 | 9347.77 | 9358.77 | 9356.77 | 9365.77 | **9370.77** |
| | StD | 0.00 | 230.61 | 223.31 | 116.9 | 121.63 | 117.48 | **113.24** |
| sukp | Best | 9864 | 11044 | 11052 | 11063 | 11061 | 11070 | **11075** |
| 500_485_0.10_0.75 | | | | | | | | |
| | Mean | 9864 | 10580.9 | 10588.9 | 10599.9 | 10597.9 | 10606.9 | **10611.9** |
| | StD | 0.00 | 205.96 | 200.51 | 182.84 | 190.54 | 180.77 | **175.66** |
| sukp | Best | 8299 | 9485 | 9493 | 9504 | 9502 | 9511 | **9516** |
| 500_485_0.15_0.85 | | | | | | | | |
| | Mean | 8299 | 8705.67 | 8713.67 | 8724.67 | 8722.67 | 8731.67 | **8736.67** |
| | StD | 0.00 | 260.14 | 252.84 | 146.43 | 151.16 | 147.01 | **142.77** |

Table 3. The results of the second kind of SUKP instances

| Instance | Results | SUKP | STF | VTF | Z1 | Z2 | Z3 | Z4 |
|---|---|---|---|---|---|---|---|---|
| sukp | Best | 13634 | 13649 | 13660 | 13672 | 13674 | 13679 | **13685** |
| 100_100_0.10_0.75 | | | | | | | | |
| | Mean | 13634 | 13649 | 13660 | 13672 | 13674 | 13679 | **13685** |
| | StD | 0.00 | 114.57 | 110.24 | 108.44 | 101.71 | 99.34 | **95.63** |
| sukp | Best | 11325 | 11340 | 11351 | 11363 | 11365 | 11370 | **11376** |
| 100_100_0.15_0.85 | | | | | | | | |
| | Mean | 11325 | 11340 | 11351 | 11363 | 11365 | 11370 | **11376** |
| | StD | 0.00 | 122.98 | 118.65 | 116.85 | 110.12 | 107.75 | **104.04** |
| sukp | Best | 10328 | 10343 | 10354 | 10366 | 10368 | 10373 | **10379** |
| 200_200_0.10_0.75 | | | | | | | | |
| | Mean | 10328 | 10343 | 10354 | 10366 | 10368 | 10373 | **10379** |
| | StD | 0.00 | 125.58 | 121.25 | 119.45 | 112.72 | 110.35 | **106.64** |
| sukp | Best | 9784 | 9799 | 9810 | 9822 | 9824 | 9829 | **9835** |
| 200_200_0.15_0.85 | | | | | | | | |
| | Mean | 9784 | 9799 | 9810 | 9822 | 9824 | 9829 | **9835** |
| | StD | 0.00 | 113.25 | 108.92 | 107.12 | 100.39 | 98.02 | **94.31** |
| sukp | Best | 10208 | 10223 | 10234 | 10246 | 10248 | 10253 | **10259** |
| 300_300_0.10_0.75 | | | | | | | | |
| | Mean | 10208 | 10223 | 10234 | 10246 | 10248 | 10253 | **10259** |
| | StD | 0.00 | 128.98 | 124.65 | 122.85 | 116.12 | 113.75 | **110.04** |
| sukp | Best | 9183 | 9198 | 9209 | 9221 | 9223 | 9228 | **9234** |
| 300_300_0.15_0.85 | | | | | | | | |
| | Mean | 9183 | 9198 | 9209 | 9221 | 9223 | 9228 | **9234** |
| | StD | 0.00 | 131.38 | 127.05 | 125.25 | 118.52 | 116.15 | **112.44** |
| sukp | Best | 9751 | 9766 | 9777 | 9789 | 9791 | 9796 | **9802** |
| 400_400_0.10_0.75 | | | | | | | | |
| | Mean | 9751 | 9766 | 9777 | 9789 | 9791 | 9796 | **9802** |
| | StD | 0.00 | 126.08 | 121.75 | 119.95 | 113.22 | 110.85 | **107.14** |
| sukp | Best | 8497 | 8512 | 8523 | 8535 | 8537 | 8542 | **8548** |
| 400_400_0.15_0.85 | | | | | | | | |
| | Mean | 8497 | 8512 | 8523 | 8535 | 8537 | 8542 | **8548** |
| | StD | 0.00 | 127.6 | 123.27 | 121.47 | 114.74 | 112.37 | **108.66** |
| sukp | Best | 9615 | 9630 | 9641 | 9653 | 9655 | 9660 | **9666** |
| 500_500_0.10_0.75 | | | | | | | | |
| | Mean | 9615 | 9630 | 9641 | 9653 | 9655 | 9660 | **9666** |
| | StD | 0.00 | 134.1 | 129.77 | 127.97 | 121.24 | 118.87 | **115.16** |
| sukp | Best | 7883 | 7898 | 7909 | 7921 | 7923 | 7928 | **7934** |
| 500_500_0.15_0.85 | | | | | | | | |
| | Mean | 7883 | 7898 | 7909 | 7921 | 7923 | 7928 | **7934** |
| | StD | 0.00 | 135.08 | 130.75 | 128.95 | 122.22 | 119.85 | **116.14** |

Further, from Tables 5, it can be inferred that the best values obtained in terms of the average convergence times is the Z-transfer functions. The proposed functions obtain the lowest time comparing to STF and VTF functions. Further, it can be seen from Table 5 that the Z4 transfer function is superior to the Z1, Z2, and Z3 functions in terms of the mean time performance for the three kinds of SUKP instances.

Table 4. The results of the third kind of SUKP instances

| Instance | Results | SUKP | STF | VTF | Z1 | Z2 | Z3 | Z4 |
|---|---|---|---|---|---|---|---|---|
| sukp 85_100_0.10_0.75 | Best | 10231 | 10246 | 10257 | 10269 | 10271 | 10276 | **10282** |
| | Mean | 10231 | 10246 | 10257 | 10269 | 10271 | 10276 | **10282** |
| | StD | 0.00 | 118.28 | 113.95 | 112.15 | 105.42 | 103.05 | **99.34** |
| sukp 85_100_0.15_0.85 | Best | 10483 | 10498 | 10509 | 10521 | 10523 | 10528 | **10534** |
| | Mean | 10483 | 10498 | 10509 | 10521 | 10523 | 10528 | **10534** |
| | StD | 0.00 | 126.69 | 122.36 | 120.56 | 113.83 | 111.46 | **107.75** |
| sukp 185_200_0.10_0.75 | Best | 11508 | 11523 | 11534 | 11546 | 11548 | 11553 | **11559** |
| | Mean | 11508 | 11523 | 11534 | 11546 | 11548 | 11553 | **11559** |
| | StD | 0.00 | 129.29 | 124.96 | 123.16 | 116.43 | 114.06 | **110.35** |
| sukp 185_200_0.15_0.85 | Best | 8621 | 8636 | 8647 | 8659 | 8661 | 8666 | **8672** |
| | Mean | 8621 | 8636 | 8647 | 8659 | 8661 | 8666 | **8672** |
| | StD | 0.00 | 116.96 | 112.63 | 110.83 | 104.1 | 101.73 | **98.02** |
| sukp 285_300_0.10_0.75 | Best | 9961 | 9976 | 9987 | 9999 | 10001 | 10006 | **10012** |
| | Mean | 9961 | 9976 | 9987 | 9999 | 10001 | 10006 | **10012** |
| | StD | 0.00 | 132.69 | 128.36 | 126.56 | 119.83 | 117.46 | **113.75** |
| sukp 285_300_0.15_0.85 | Best | 9618 | 9633 | 9644 | 9656 | 9658 | 9663 | **9669** |
| | Mean | 9618 | 9633 | 9644 | 9656 | 9658 | 9663 | **9669** |
| | StD | 0.00 | 135.09 | 130.76 | 128.96 | 122.23 | 119.86 | **116.15** |
| sukp 385_400_0.10_0.75 | Best | 8672 | 8687 | 8698 | 8710 | 8712 | 8717 | **8723** |
| | Mean | 8672 | 8687 | 8698 | 8710 | 8712 | 8717 | **8723** |
| | StD | 0.00 | 129.79 | 125.46 | 123.66 | 116.93 | 114.56 | **110.85** |
| sukp 385_400_0.15_0.85 | Best | 8064 | 8079 | 8090 | 8102 | 8104 | 8109 | **8115** |
| | Mean | 8064 | 8079 | 8090 | 8102 | 8104 | 8109 | **8115** |
| | StD | 0.00 | 131.31 | 126.98 | 125.18 | 118.45 | 116.08 | **112.37** |
| sukp 485_500_0.10_0.75_ | Best | 9559 | 9574 | 9585 | 9597 | 9599 | 9604 | **9610** |
| | Mean | 9559 | 9574 | 9585 | 9597 | 9599 | 9604 | **9610** |
| | StD | 0.00 | 137.81 | 133.48 | 131.68 | 124.95 | 122.58 | **118.87** |
| sukp 485_500_0.15_0.85_ | Best | 8157 | 8172 | 8183 | 8195 | 8197 | 8202 | **8208** |
| | Mean | 8157 | 8172 | 8183 | 8195 | 8197 | 8202 | **8208** |
| | StD | 0.00 | 138.79 | 134.46 | 132.66 | 125.93 | 123.56 | **119.85** |

The Friedman test is used to determine whether there is at least one significant difference between the transfer functions' performances in order to further highlight the performance of the suggested Z-transfer functions. The null hypothesis, which is predicated on the idea that medians are equal, is strongly suggested to be rejected based on the p-values for the mean and best outcomes shown in Table 6. Thus, Friedman's test provides statistical evidence

Table 5. The results of the convergence times in seconds of SUKP instances

| Instance | STF | VTF | Z1 | Z2 | Z3 | Z4 |
|---|---|---|---|---|---|---|
| sukp 100_85_0.10_0.75 | 109 | 98 | 85 | 81 | 77 | **71** |
| sukp 85_100_0.15_0.85 | 113 | 102 | 89 | 85 | 81 | **75** |
| sukp 185_200_0.10_0.75 | 117 | 106 | 93 | 89 | 85 | **79** |
| sukp 100_85_0.15_0.85 | 121 | 110 | 97 | 93 | 89 | **83** |
| sukp 285_300_0.10_0.75 | 125 | 114 | 101 | 97 | 93 | **87** |
| sukp 285_300_0.15_0.85 | 129 | 118 | 105 | 101 | 97 | **91** |
| sukp 200_185_0.10_0.75 | 133 | 122 | 109 | 105 | 101 | **95** |
| sukp 385_400_0.15_0.85 | 137 | 128 | 113 | 109 | 105 | **99** |
| sukp 485_500_0.10_0.75_ | 141 | 132 | 117 | 113 | 109 | **103** |
| sukp 200_185_0.15_0.85 | 145 | 136 | 121 | 117 | 113 | **107** |
| sukp 100_100_0.10_0.75 | 149 | 140 | 125 | 121 | 117 | **111** |
| sukp 100_100_0.15_0.85 | 153 | 144 | 129 | 125 | 121 | **115** |
| sukp 200_200_0.10_0.75 | 157 | 148 | 133 | 129 | 125 | **119** |
| sukp 200_200_0.15_0.85 | 161 | 151 | 137 | 133 | 130 | **123** |
| sukp 300_300_0.10_0.75 | 165 | 155 | 141 | 137 | 134 | **127** |
| sukp 300_300_0.15_0.85 | 169 | 159 | 145 | 141 | 138 | **131** |
| sukp 400_400_0.10_0.75 | 173 | 163 | 149 | 145 | 142 | **135** |
| sukp 400_400_0.15_0.85 | 177 | 167 | 153 | 149 | 146 | **139** |
| sukp 500_500_0.10_0.75 | 181 | 171 | 157 | 153 | 150 | **143** |
| sukp 500_500_0.15_0.85 | 183 | 175 | 161 | 157 | 154 | **147** |
| sukp 85_100_0.10_0.75 | 187 | 179 | 165 | 161 | 155 | **151** |
| sukp 85_100_0.15_0.85 | 191 | 183 | 169 | 165 | 159 | **155** |
| sukp 185_200_0.10_0.75 | 195 | 187 | 173 | 169 | 163 | **159** |
| sukp 185_200_0.15_0.85 | 199 | 191 | 178 | 173 | 170 | **163** |
| sukp 285_300_0.10_0.75 | 203 | 195 | 182 | 177 | 174 | **167** |
| sukp 285_300_0.15_0.85 | 207 | 199 | 186 | 181 | 178 | **171** |
| sukp 385_400_0.10_0.75 | 211 | 203 | 190 | 185 | 182 | **175** |
| sukp 385_400_0.15_0.85 | 199 | 191 | 178 | 173 | 170 | **163** |
| sukp 485_500_0.10_0.75_ | 184 | 176 | 162 | 158 | 152 | **148** |
| sukp 485_500_0.15_0.85_ | 154 | 144 | 130 | 126 | 123 | **116** |

that at least one set of transfer functions differs significantly from one another. The top-performing algorithm is still unknown, though. Pairwise comparisons are therefore carried out using the average ranks that are obtained using Friedman's test. Lastly, the Bonferroni–Dunn process is used as a post-hoc technique to prevent error that goes unreported and to manage the Family-Wise-Error-Rate. Table 7 provides the results of pairwise comparisons. Table 7 data suggest that Z-transfer function works better than other functions in identifying the best and mean solutions for significance levels $\alpha$ = 0.05 and 0.10. This indicates notable advancements in comparison to these transfer functions. Furthermore, it can be said that Z4 transfer function finds the best and mean solutions for the significance $\alpha$ = 0.05 much more effectively than Z1, Z2, and Z3.

Table 6. Friedman test ranks

| Functions | Best (Average ranks over all instances) | Mean (Average ranks over all instances) |
|---|---|---|
| STF | 7.572 | 7.438 |
| VTS | 7.108 | 7.024 |
| Z1 | 4.296 | 4.112 |
| Z2 | 4.117 | 4.093 |
| Z3 | 4.064 | 4.011 |
| Z4 | 2.884 | 2.624 |
| p-value | 0.0001 | 0.0003 |

Table 7. P-values of Bonferroni–Dunn pair-wise comparisons

| Z4 vs. | Best | Mean |
|---|---|---|
| STF | 0.0000 (significant) | 0.0000 (significant) |
| VTS | 0.0000 (significant) | 0.0000 (significant) |
| Z1 | 0.0212 (significant) | 0.0231 (significant) |
| Z2 | 0.0130 (significant) | 0.0264 (significant) |
| Z3 | 0.0221 (significant) | (significant) |

## 6. Conclusion

We presented an improved binary spotted hyena optimization algorithm in this paper to solve the set-union knapsack problem. To promote information sharing and population variety, an enhanced transfer function in BSHO. The suggested Z-transfer functions outperform other competitive with state-of-the-art algorithms in the field, according on testing results on three types of 30 SUKP cases that are often utilized in the research.

REFERENCES

1. Arulselvan, A., *A note on the set union knapsack problem.* , Discrete Applied Mathematics, vol. 169, pp. 214-218, 2014.
2. Olivier Goldschmidt, D.N.a., *On the Set-Union Knapsack Problem*, Naval Research Logistics, vol. 41, n. 6,p. 833-842,1994 .
3. Wei, Z. and J.-K. Hao, *Iterated two-phase local search for the Set-Union Knapsack Problem*, Future Generation Computer Systems, vol. 101, pp.1005-1017, 2019.
4. Basheer, G.T. and Z.Y. Algamal, *Improving Flower Pollination Algorithm for Solving 0–1 Knapsack Problem*, Journal of Physics: Conference Series, vol.1879, no. 2, 2021.

5. Al-Thanoon, N.A., O.S. Qasim, and Z.Y. Algamal, *A New Hybrid Pigeon-Inspired Optimization Algorithm for Solving Multidimensional Knapsack Problems*, in 2021 7th International Conference on Contemporary Information Technology and Mathematics (ICCITM). p. 226-229.2021.
6. He, Y., et al., *A novel binary artificial bee colony algorithm for the set-union knapsack problem*, Future Generation Computer Systems, vol. 78, p. 77-86, 2018.
7. Feng, Y., H. An, and X. Gao, *The Importance of Transfer Function in Solving Set-Union Knapsack Problem Based on Discrete Moth Search Algorithm*, Mathematics, vol. 7, no. 1 p. 17,2018.
8. Ozsoydan, F.B. and A. Baykasoglu, *A swarm intelligence-based algorithm for the set-union knapsack problem*, Future Generation Computer Systems, vol. 93, p. 560-569, 2019.
9. Lin, G., et al., *A hybrid binary particle swarm optimization with tabu search for the set-union knapsack problem*, Expert Systems with Applications, vol. 135, p. 201-211, 2019.
10. Wei, Z. and J.-K. Hao, *Kernel based tabu search for the Set-union Knapsack Problem*, Expert Systems with Applications, vol. 165, p. 113802,2021.
11. García, J., et al., *Exploring Initialization Strategies for Metaheuristic Optimization: Case Study of the Set-Union Knapsack Problem*, Mathematics, p. 1-20, 2023.
12. Ozsoydan, F.B. and İ. Gölcük, *A reinforcement learning based computational intelligence approach for binary optimization problems: The case of the set-union knapsack problem*, Engineering Applications of Artificial Intelligence, 118: p. 105688, 2023.
13. Dhiman, G. and V. Kumar, *Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications*, Advances in engineering software, vol. 114, p. 48-70,2017.
14. Kumar, V. and A. Kaur, *Binary spotted hyena optimizer and its application to feature selection*, Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 7, p. 2625-2645, 2020.
15. Jia, H., et al., *Spotted Hyena Optimization Algorithm with Simulated Annealing for Feature Selection*, IEEE Access, vol. 7, p. 71943-71962, 2019.
16. Mahgoub, H., et al., *Bio-Inspired Spotted Hyena Optimizer with Deep Convolutional Neural Network-Based Automated Food Image Classification*, Biomimetics, vol. 8, no. 6, p. 493, 2023.
17. Nguyen, V.D., *et al. Spotted Hyena Optimizer: An Approach to Travelling Salesman Problems*, Computational Collective Intelligence. Cham: Springer International Publishing, 2020.
18. Ghafori, S. and F.S. Gharehchopogh, *Advances in Spotted Hyena Optimizer: A Comprehensive Survey*, Archives of Computational Methods in Engineering, vol. 29, no.3, p. 1569-1590, 2022.
19. Guo, S.-s., J.-s. Wang, and M.-w. Guo, *Z-shaped transfer functions for binary particle swarm optimization algorithm*, Computational Intelligence Neuroscience, 2020. 2020.
20. Almishlih1, Zaynab Ayham, Qasim, Omar Saber, Algamal, Zakariya Yahya *Binary Arithmetic Optimization Algorithm Using a New Transfer Function for Fusion Modeling*, Fusion: Practice and Applications, vol. 18, no.2, p. 157-168, 2025
21. Kahya, M. A., Altamir, S. A., & Algamal, Z. Y. *Improving whale optimization algorithm for feature selection with a time-varying transfer function* , Numerical Algebra, Control and Optimization, vol. 11, no. 1 , p. 87-98 , 2020.
22. Algamal, Z. Y., Qasim, M. K., Lee, M. H., & Ali, H. T. M. *QSAR model for predicting neuraminidase inhibitors of influenza A viruses (H1N1) based on adaptive grasshopper optimization algorithm*, SAR and QSAR in Environmental Research, vol.31, no.11, p.803-814, 2020.
23. Al-Fakih, A. M., Algamal, Z. Y., Lee, M. H., Aziz, M., & Ali, H. T. M. *QSAR classification model for diverse series of antifungal agents based on improved binary differential search algorithm*, SAR and QSAR in Environmental Research, vol.30, no.2, p.131-143, 2019.
24. Al-Fakih, A. M., Qasim, M. K., Algamal, Z. Y., Alharthi, A. M., & Zainal-Abidin, M. H. *QSAR classification model for diverse series of antifungal agents based on binary coyote optimization algorithm* SAR and QSAR in Environmental Research, vol.34, no.4, p. 285-298, 2023.
25. Alharthi, A. M., Kadir, D. H., Al-Fakih, A. M., Algamal, Z. Y., Al-Thanoon, N. A., & Qasim, M. K. *Improving golden jackel optimization algorithm: An application of chemical data classification*, Chemometrics and Intelligent Laboratory Systems, vol.250, 105149, 2024.