Complex-Valued Crow Search Algorithm for 0–1 KP Problem

Yan Shi[1], Yongquan Zhou[1(✉)], Qifang Luo[1,2], and Huajuan Huang[1,2]

[1] College of Electronic Information, Guangxi University for Nationalities, Nanning 530006, China
yongquanzhou@126.com

[2] Guangxi High School Key Laboratory of Complex System and Computational Intelligence, Nanning 530006, China

Abstract. The crow search algorithm is a novel swarm intelligence optimization algorithm. Aiming at the shortcomings of the crow search algorithm, a complex coding crow search algorithm (CCSA) is proposed to enhance its detection ability. A greedy algorithm is introduced into the algorithm to balance detection and development and enhance the optimization accuracy. And use the Sigmoid function to discretize the CCSA algorithm to solve the 0–1 knapsack problem.

Keywords: Complex-valued · Crow search algorithm (CSA) · Complex-valued crow search algorithm · 0–1 KP

1 Introduction

The crow search algorithm (CSA) [1] is a new swarm intelligence optimization algorithm proposed by Iranian scholar Askarzadeh in 2016. Its principle is to imitate the behavior of crows hiding their food and "stealing" food from each other, and to evolve the behavior of crows "stealing" food from each other into a random optimization process. The crow search algorithm has been widely used in many fields. For example, Laabadi et al. (2020) [2] proposed a Binary CSA (BCSA) for solving the two-dimensional bin packing problem (2D-BPP). Sayed et al. (2019) [3] proposed a Chaotic Crow Search Algorithm (CCSA) based on the integration of chaos in CSA for solving the feature selection problem. Sahoo and Padhy (2019) [4] proposed an Improved Crow Search Algorithm (ICSA) for solving the multiprocessor task scheduling problem. dos Santos Coelho et al. (2018) [5] proposed a modified CSA, named MCSA, for solving the circular antenna array design problems. Gaussian probability distribution function and population diversity information were used to tune and control AP and Flight Length parameters. Mandala and Rao (2019) [6] proposed an improved CSA, called Adaptive Awareness Probability-based CSA (AAP-CSA) for medical data preservation. Fred et al. (2020) [7] proposed a hybrid technique (FCM-CSA) based on the hybridization of CSA with FCM for medical image segmentation. Shekhawat and Saxena (2019) [8] proposed an enhanced CSA called Intelligent Crow Search Algorithm (ICSA) for solving real structural design problems. Rizk-Allah et al. (2020) [9] developed a multi-objective

D.-S. Huang et al. (Eds.): ICIC 2022, LNAI 13395, pp. 860–875, 2022.
https://doi.org/10.1007/978-3-031-13832-4_69

orthogonal opposition-based CSA (M2O-CSA) for solving large-scale multi-objective optimization problems. Yundi Rao et al. (2022) [10] introduced a probability simplified sine cosine algorithm to form a new hybrid algorithm called PSCCSA (Probabilistic Simplified Sine Cosine Crow Search Algorithm). PSCCSA has been used to solve four classic engineering problems (pressure vessel design, speed reducer design, welded beam design and tension/compression spring design problem). Farh et al. (2020) [11] proposed a hybrid method (CSA-PSO) based on the combination of CSA with PSO for solving the optimal power flow problem with Renewable Distributed Generations (RDGs). Li L, Liu Z F, Tseng M L, et al. (2021) [12] the enhanced crow search algorithm optimization-extreme learning machine (ENCSA-ELM) model is proposed to accurately forecast short-term wind power to improve the utilization efficiency of clean energy. Necira A, Naimi D, Salhi A, et al. (2021) [13] an enhanced version of CSA called dynamic crow search algorithm (DCSA) is proposed to overcome the drawbacks of the conventional CSA.

In order to improve the optimization performance of CSA algorithm, this paper proposes a complex-valued encoding crow search algorithm (CCSA) based on the idea of a diploid structure. In order to avoid local optimization, greedy search strategy is introduced in CCSA algorithm and used to solve 0–1 knapsack problem. Finally, abnormal solution is modified. In solving discrete problems like 0–1 knapsack, variables need to be discretized. In this paper, Sigmoid function is used to discretize CCSA algorithm. Finally, the improved algorithm is simulated and compared with other optimization algorithms, and CCSA algorithm achieves good optimization effect.

2 Crow Search Algorithm

Crows are considered the most intelligent birds. They can also remember their food's hiding place and recall their food's hiding place up to several months later [1]. Crows are able to track other crows to find their food's hiding place and steal the food. The tracked crows have a certain probability of sensing that they are being tracked and then fly to an arbitrary location to make sure their food is not found.

When solving the problem, it is assumed that there are N crows randomly distributed in the d-dimensional environment, After (iteration) $iter$ times, the position of crow i in the search space is specified by a vector $x^{i,iter}(i = 1, 2, \ldots, N; iter = 1, 2, \ldots, iter_{max})$ where $x^{i,iter} = [x_1^{i,iter}, x_2^{i,iter}, \ldots x_d^{i,iter}]$ and $iter_{max}$ is the maximum number of iterations. $m^{i,iter}$ represents the best position of the food hidden by crow i during iteration $iter$ times. Assume that during iteration $iter$ times, crow j wants to visit its own hidden location $m^{i,iter}$. During this iteration, crow i decides to follow crow j to approach the location where crow j hides the food. When crow j is unaware that crow i is following itself, this causes that crow i will approach to the location where crow j hides the food. When crow j notices that crow i is following it, crow j will lead crow i to another position in the search space. the new position of crow i is obtained as follows:

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + r_i \times fl^{i,iter} \times \left(m^{j,iter} - x^{i,iter}\right), r_j \geq AP^{j,iter} \\ a\ random\ position, \qquad\qquad otherwise \end{cases} \tag{1}$$

where r_i is a random number with uniform distribution between 0 and 1; $fl^{i,iter}$ denotes the flight length of crow i at iteration $iter$; $AP^{j,iter}$ denotes the awareness probability of crow j at iteration $iter$.

It is seen that if the fitness function value of the new position of a crow is better than the fitness function value of the memorized position, the crow updates its memory by the new position. The update method is as follows:

$$m^{i,iter+1} = \begin{cases} x^{i,iter+1}, f(x^{i,iter+1}) \geq f\left(m^{i,iter}\right) \\ m^{i,iter}, \qquad otherwise \end{cases} \tag{2}$$

3 Complex-Valued Crow Search Algorithm

The principle of complex-valued encoding is to use the idea of a diploid structure to encode individuals, to expand the information capacity of individuals. Complex-valued encoding [14] maps one-dimensional expression space with two-dimensional coding space, and the real and imaginary parts are updated separately. So, the CCSA algorithm greatly enriches the diversity of the population, improves the shortcoming that the CSA algorithm is easy to fall into the local optima, and extends the application range of the CSA algorithm to the complex range. Complex number have two-dimensional properties, so the CCSA algorithm can represent larger spatial dimensions.

3.1 Initialize the Complex-Valued Encoding Population

Based on the definition interval of the problem $[A_k, B_k]$,$k = 1, 2, \ldots, 2M$, generate $2M$ complex modulus and $2M$ phase angle randomly [14].

$$\rho_k \in \left[0, \frac{A_k - B_k}{2}\right], k = 1, 2, \ldots, 2M \tag{3}$$

$$\theta_k \in [-2\pi, 2\pi], k = 1, 2, \ldots, 2M \tag{4}$$

According to the Eq. (5), get $2M$ complex number:

$$X_{Rk} + iX_{Ik} = \rho_k(cos\theta_k + i\, sin\theta_k), k = 1, 2, \ldots, 2M \tag{5}$$

Thus, obtain $2M$ real parts and $2M$ imaginary parts, and the real and imaginary parts are updated according to the following way.

3.2 The Updating Method of CCSA

Update the real parts

$$X_R^{i,iter+1} = \begin{cases} X_R^{i,iter} + r_i \times fl^{i,iter} \times \left(m_R^{j,iter} - X_R^{i,iter}\right), & r_j \geq AP^{j,iter} \\ a\ random\ position, & otherwise \end{cases} \tag{6}$$

Update the imaginary parts

$$X_1^{i,iter+1} = \begin{cases} X_1^{i,iter} + r_i \times fl^{i,iter} \times \left(m_1^{j,iter} - X_1^{i,iter}\right), & r_j \geq AP^{j,iter} \\ a\ random\ position, & otherwise \end{cases} \tag{7}$$

where $m_R^{j,iter}$ and $m_I^{j,iter}$ are positions in the crow j's memory.

3.3 Fitness Function Value Calculation

Due to the complex domain has two parts, a real part and an imaginary part, when calculating the fitness value, it is necessary to convert the complex number into the real number first [14], and then calculate its fitness value. Specific practices are as follows:

(1) Take complex modulus as the value of the real number:

$$\rho_{\mathrm{k}} = \sqrt{\mathrm{X}_{\mathrm{Rk}}^2 + \mathrm{X}_{\mathrm{Ik}}^2}, \mathrm{k} = 1, 2, \ldots, M \tag{8}$$

(2) The sign is determined by phase angle:

$$X_k = \rho_k \, sgn\left(sin\left(\frac{X_{Ik}}{\rho_k}\right)\right) + \frac{A_k + B_k}{2}, \mathrm{k} = 1, 2, \ldots, M \tag{9}$$

where, X_k represent the converted real variables.

4 CCSA Algorithm for Solving 0–1 KP Problem

The knapsack problem is a classical combinatorial optimization problem and is applied to many practical problems. For example, interactive multimedia systems [15], item selection, resource allocation and loading cargo issues, etc. Also, there are several variants of the knapsack problem, one of which is the 0–1 knapsack problem (0–1 KP). The purpose of 0–1 KP is to maximize the total profit of items put into the backpack under the premise of meeting the maximum capacity of the backpack.

The mathematical model of the 0–1 knapsack problem is as follows:

$$Maximize \sum_{i=1}^{n} p_i x_i \tag{10}$$

$$Subject\ to \begin{cases} \sum_{i=1}^{n} w_i x_i \le C \\ x_i \in \{0, 1\}, 1 \le i \le n \end{cases} \tag{11}$$

where x_i is a binary decision variable. If item i is included in the knapsack, then $x_i = 1$; otherwise, $x_i = 0$. p_i is the profit of item i; w_i is the weight of item i; C is the maximum capacity item i.

4.1 Discretization Processing Method

The 0–1 knapsack problem is a discrete problem, and the problem needs to be discretized when solving such problems. In this paper the sigmoid function (Eq. (12), Eq. (13)) is used to transform a continuous space value into a binary one.

Discretization method is as follows:

$$S_k = \frac{1}{1 + e^{-X_k}} \tag{12}$$

$$X_k = \begin{cases} 1, \ Rand < S_k \\ 0, \ otherwise \end{cases} \tag{13}$$

4.2 The Algorithm Flowchart of CPBA

Pseudo code of discrete CCSA
BEGIN
Initialize the bat population: $\rho_k \in \left[0, \frac{A_k - B_k}{2}\right]$ and $\theta_k \in [-2\pi, 2\pi]$
Get the real and imaginary part of the complex [Eq. (5)]
Convert to real variables [Eq. (8) and Eq. (9)]
Discrete X_k [Eq. (12) and Eq. (13)]
For all X_i **do**
Calculate fitness $f(X_i)$ [Eq. (10) and Eq. (11)]
End for
Get the best solution
$iter = 1$
For $iter = 1$: Max_iteration
Update the real part [Eq. (6)]
Update the imaginary part [Eq. (7)]
Convert to real variables [Eq. (8) and Eq. (9)]
Discrete X_k [Eq. (12) and Eq. (13)]
Calculate fitness $p_i = f(X_i)$ [Eq. (10)]
Calculate total weight $W_i = w_i * X_i$ [Eq. (11)]
If $W_i > C$ then
$p_i = 0$
End If
Get the best solution
Update the crow's memory
$iter = iter + 1$
End for
End

5 Simulation Experiments and Results Analysis

To evaluate the performance of the CCSA algorithms, their results have been compared with the state-of-the-art meta-heuristics. Three data sets, including low-dimensional, medium-dimensional, and high-dimensional KP01 instances, have been chosen for comparative studies. All these data sets can be found in Ref. [16]. The optimal values of the proposed CCSA are taken from Table 1, and the optimal values of the parameters of the other optimization algorithms are considered from their original articles. All experimental procedures are implemented using Matlab R2020(b) in a PC with Intel(R) Core(TM) i7-9700 @ 3.0 GHz CPU, and 16 GB DDR4 of RAM under Windows 10 Operating System. It should be noted that the statistical criteria of best, average, worst, and percentage deviation (PDav (%)) were used to evaluate the performance of the proposed model. [17] The percentage deviation criterion indicates how far the result obtained from the

optimizers is from the optimal value of each data set and is calculated using Eq. (14).

$$\text{PDav}(\%) = \frac{\text{Opt} - \text{Avg}}{\text{Opt}} \times 100 \tag{14}$$

Table 1. The parameter setting

Parameters	Value
Max_iteration	30
NPOP	30
Number of runs	10
fl	2
AP	0.1

5.1 Low-Dimensional 0–1 KP

In this section, the proposed CCSA is compared with CSA [1], BMBO [18], NBBA [19], BABC [20], CWDO [21], PSA [22], PSO [23], and BA [24]. The results of the comparative study on this data set are shown in Table 2. From Table 2, some facts can be elicited: The CCSA algorithm is as effective as other algorithms on low-dimensional instances, and can reach the optimal value. According to the average value, it can be judged that the stability of the CCSA algorithm is better than that of the standard CSA algorithm.

Table 2. Comparison of low-dimensional 0–1 KP instance CCSA with other algorithms

Dataset	Dim	Opt	Metric	CCSA	CSA	BMBO	NBBA	BABC	CWDO	PSA	PSO	BA
KP1	10	295	Best	**295**	**295**	**295**	**295**	**295**	**295**	**295**	**295**	**295**
			Avg	295	295	295	295	295	295	295	295	295
			Worst	295	295	295	295	295	295	295	295	295
KP2	20	1024	Best	**1024**	**1024**	**1024**	**1024**	**1024**	**1024**	**1024**	**1024**	**1024**
			Avg	1024	1024	1024	1024	1024	1024	1024	1024	1024
			Worst	1024	1024	1024	1024	1024	1024	1024	1024	1024
KP3	4	35	Best	**35**	**35**	**35**	**35**	**35**	**35**	**35**	**35**	**35**
			Avg	35	35	35	35	35	35	35	35	35
			Worst	35	35	35	35	35	35	35	35	35
KP4	4	23	Best	**23**	**23**	**23**	**23**	**23**	**23**	**23**	**23**	**23**

(*continued*)

Table 2. (*continued*)

Dataset	Dim	Opt	Metric	CCSA	CSA	BMBO	NBBA	BABC	CWDO	PSA	PSO	BA
			Avg	23	22.2	23	23	23	23	23	22.9	22.3
			Worst	23	22	23	23	23	23	23	22	22
KP5	15	481.07	Best	**481.07**	**481.07**	**481.07**	**481.07**	**481.07**	**481.07**	**481.07**	**481.07**	**481.07**
			Avg	481.07	481.07	481.07	481.07	481.07	481.07	481.07	481.07	481.07
			Worst	481.07	481.07	481.07	481.07	481.07	481.07	481.07	481.07	481.07
KP6	10	52	Best	**52**	**52**	**52**	**52**	**52**	**52**	**52**	**52**	**52**
			Avg	52	52	52	52	52	52	52	52	52
			Worst	52	52	52	52	52	52	52	52	52
KP7	7	107	Best	**107**	**107**	**107**	**107**	**107**	**107**	**107**	**107**	**107**
			Avg	107	107	107	107	107	107	107	107	107
			Worst	107	107	107	107	107	107	107	107	107
KP8	23	9767	Best	**9767**	**9767**	**9767**	**9767**	**9767**	**9767**	**9767**	**9767**	**9767**
			Avg	9767	9767	9766.12	9767	9767	9767	9767	9765.6	9767
			Worst	9767	9767	9765	9767	9767	9767	9767	9763	9767
KP9	5	130	Best	**130**	**130**	**130**	**130**	**130**	**130**	**130**	**130**	**130**
			Avg	130	130	130	130	130	130	130	130	130
			Worst	130	130	130	130	130	130	130	130	130
KP10	20	1025	Best	**1025**	**1025**	**1025**	**1025**	**1025**	**1025**	**1025**	**1025**	**1025**
			Avg	1025	1025	1025	1025	1025	1025	1025	1025	1025
			Worst	1025	1025	1025	1025	1025	1025	1025	1025	1025

* The bold numbers are the best obtained values

5.2 Medium-Dimensional 0–1 KP

In this section, the proposed CCSA is compared with CSA [1], NBBA [19], CI [19], B&B [19], PSA [22], PSO [23], and BA [24]. The comparison results are shown in Table 3, where "–" indicates that the cited data is not given. From Table 3 can be concluded that (1) the CCSA algorithm and the PSA algorithm reach the best-known solution, (2) by comparing the optimal value and the average value, the CCSA algorithm except that in the KP_19 instance, the average value and There is a small gap between the optimal solutions, and all the remaining examples reach the best-known solution, which shows that the stability of the CCSA algorithm is also very good. It can be seen from Fig. 1 and Fig. 2 that the CCSA algorithm has certain advantages in terms of convergence speed and convergence accuracy compared with the PSO, BA, and CSA algorithms.

Table 3. Comparison of Medium-dimensional 0–1 KP instance CCSA with other algorithms

Dataset	Dim	Opt	Metric	CCSA	CSA	NBBA	CI	B&B	PSA	PSO	BA
KP11	30	1437	Best	**1437**	**1437**	**1437**	**1437**	**1437**	**1437**	**1437**	**1437**
			Avg	1437	1437	1437	1418	–	1437	1436.1	1437
			Worst	1437	1437	1437	1398	–	1437	1428	1437
KP12	35	1689	Best	**1689**	**1689**	**1689**	**1689**	**1689**	**1689**	**1689**	**1689**
			Avg	1689	1689	1689	1686.5	–	1689	1689	1689
			Worst	1689	1689	1689	1679	–	1689	1689	1689
KP13	40	1821	Best	**1821**	**1821**	**1821**	1816	**1821**	**1821**	**1821**	**1821**
			Avg	1821	1820.2	1821	1807.5	1821	1821	1817.9	1821
			Worst	1821	1817	1821	1791	1821	1821	1810	1821
KP14	45	2033	Best	**2033**	**2033**	**2033**	2020	**2033**	**2033**	**2033**	**2033**
			Avg	2033	2033	2033	2017	–	2033	2025.2	2033
			Worst	2033	2033	2033	2007	–	2033	2016	2033
KP15	50	2440	Best	**2440**	**2440**	**2448**	**2440**	**2440**	**2449**	**2440**	**2440**
			Avg	2440	2439.8	2448	2436.1	–	2449	2437.2	2440
			Worst	2440	2438	2448	2421	–	2449	2427	2440
KP16	55	2651	Best	**2651**	**2651**	2643	2643	2440	**2651**	2640	**2651**
			Avg	2651	2648.8	2642.6	2605	–	2651	2627.9	2651
			Worst	2651	2637	2632	2581	–	2651	2601	2651
KP17	60	2917	Best	**2917**	**2917**	**2917**	**2917**	**2917**	**2917**	2904	**2917**
			Avg	2917	2917	2917	2915	–	2917	2894.3	2917
			Worst	2917	2917	2917	2905	–	2917	2889	2917
KP18	65	2818	Best	**2818**	**2818**	**2818**	2814	**2818**	**2818**	2813	**2818**
			Avg	2818	2816.7	2817.6	2773.6	–	2818	2802.7	2817.6
			Worst	2818	2810	2814	2716	–	2818	2790	2817
KP19	70	3223	Best	**3223**	**3223**	**3223**	3221	**3223**	**3223**	3219	**3223**
			Avg	3223	3220.6	2322.6	3216	–	3223	3212.7	3220.8
			Worst	3223	3220	3219	3211	–	3223	3206	3220
KP20	75	3614	Best	**3614**	**3614**	**3614**	**3614**	**3614**	**3614**	3591	**3614**
			Avg	3614	3612.3	3613.2	3603.8	–	3614	3578.3	3614
			Worst	3614	3597	3605	3591	–	3614	3556	3614

* The bold numbers are the best obtained values

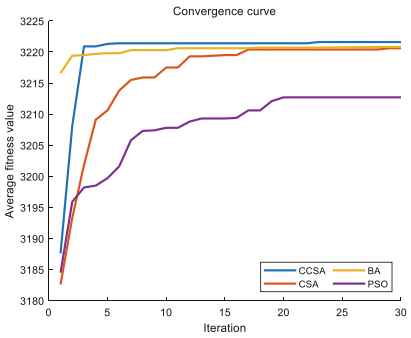

Fig. 1. KP_19 Convergence curve

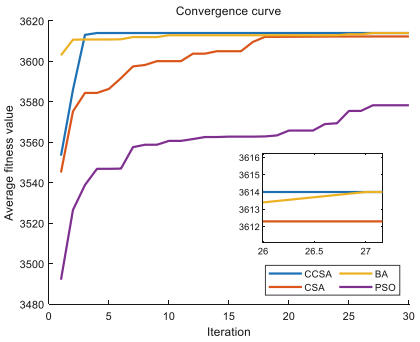

Fig. 2. KP_20 Convergence curve

5.3 High-Dimensional 0–1 KP

In this section, the proposed CCSA is compared with CSA [1], BA [24], PSO [23], EOS2 [25], BB [26], GSA [26], and SA [26]. The comparison results are shown in Table 4, where "–" indicates that the cited data is not given. Table 4 introduces the results obtained by each algorithm on the high-dimensional datasets, in which our proposed algorithm could achieve the optimal solution for 17 out of 21 datasets. Moreover, according to the percentage deviation PDav(%), it can be judged that the stability of the CCSA algorithm is better than other algorithm. Figure 3 shows the convergence curve of high-dimensional KP01. It can be concluded that the convergence speed and convergence accuracy of the proposed CCSA algorithm are better than other algorithms.

Table 4. Comparison of High-dimensional 0–1 KP instance CCSA with other algorithms

Instance	Opt	Metric	CCSA	CSA	BA	PSO	EOS2	BB	GSA	SA
KP1_100	9147	Best	**9147**	**9147**	**9147**	**9147**	**9147**	–	–	–
		Avg	9147	9147	9147	9037.6	9147	8026	2983	9147
		Worst	9147	9147	9147	8842	9147	–	–	–
		PDav	**0**	**0**	**0**	120.744	**0**	12.255	67.388	**0**
KP1_200	11238	Best	**11238**	**11238**	**11238**	**11238**	**11238**	–	–	–
		Avg	11238	11238	11238	11131.1	11238	10438	9865	10163
		Worst	11238	11238	11238	10973	11238	–	–	–
		PDav	**0**	**0**	**0**	103.672	**0**	7.119	12.217	9.566
KP1_500	28857	Best	**28857**	**28857**	**28857**	27407	**28857**	–	–	–
		Avg	28857	28843.2	28854.7	26692	28857	28043	9865	21390
		Worst	28857	28834	28834	26008	28857	–	–	–
		PDav	**0**	0.048	7.273	368.051	**0**	2.821	65.814	25.876
KP1_1000	54503	Best	**54503**	**54503**	54328	50120	**54503**	–	–	–
		Avg	54503	54451.7	54197.4	49209.4	54503	53397	14927	36719
		Worst	54503	54264	54017	48442	54503	–	–	–

(*continued*)

Table 4. (*continued*)

Instance	Opt	Metric	CCSA	CSA	BA	PSO	EOS2	BB	GSA	SA
		PDav	**0**	0.094	92.511	626.496	**0**	2.029	72.613	32.629
KP1_2000	110625	Best	**110625**	110578	108946	97132	110578	–	–	–
		Avg	110587.2	110495.3	108090.5	96214.2	110578	109679	25579	95739
		Worst	110578	110227	107577	95647	110578	–	–	–
		PDav	0.034	0.117	420.771	442.447	**0**	0.813	76.868	13.419
KP1_5000	276457	Best	**276456**	276379	269432	234008	275725	–	–	–
		Avg	276363.7	274908	265037	231920.5	274358	275720	39677	150731
		Worst	276149	272876	261947	230005	273367	–	–	–
		PDav	**0.033**	0.560	2318.496	1206.367	0.759	0.267	85.648	45.478
KP1_10000	563647	Best	**563606**	563298	544254	469465	–	–	–	–
		Avg	563503.1	561898.6	534233.7	464408.3	–	–	–	–
		Worst	562585	556235	525132	461039	–	–	–	–
		PDav	**0.026**	0.310	5.218	17.607	–	–	–	–
KP2_100	1514	Best	**1514**	**1514**	**1514**	1512	**1514**	–	–	–
		Avg	1514	1514	1514	1512	1514	1440	1041	1486
		Worst	1514	1514	1514	1512	1514	–	–	–
		PDav	**0**	**0**	**0**	**0**	**0**	4.888	31.242	1.849
KP2_200	1634	Best	**1634**	**1634**	**1634**	**1634**	**1634**	–	–	–
		Avg	1634	1634	1634	1634	1634	1603	1073	1537
		Worst	1634	1634	1634	1634	1634	–	–	–
		PDav	**0**	**0**	**0**	**0**	**0**	1.897	34.333	5.936
KP2_500	4566	Best	**4566**	4557	**4566**	4497	**4566**	–	–	–
		Avg	4559.2	4556.1	4564.2	4462.4	4564.4	4484	2951	3744
		Worst	4556	4556	4557	4433	4556	–	–	–
		PDav	**0.018**	0.217	3.795	19.608	0.035	1.796	35.370	18.003
KP2_1000	9052	Best	**9052**	9051	9051	8723	**9052**	–	–	–
		Avg	9051.1	9049	9045.8	8668.7	9050.8	9006	5675	6831
		Worst	9051	9046	9036	8628	9047	–	–	–
		PDav	**0.010**	0.033	4.442	29.258	0.013	0.508	37.307	24.536
KP2_2000	18051	Best	**18050**	18047	17999	17164	17698	–	–	–
		Avg	18046.8	18035.3	17933.5	17067.3	17497	17794	11064	12780
		Worst	18046	17985	17890	17006	16875	–	–	–
		PDav	**0.018**	0.087	34.471	53.139	3.069	1.424	38.707	29.201
KP2_5000	44356	Best	**44356**	44352	43943	41705	44305	–	–	–
		Avg	44353.4	44308.5	43690.2	41537.9	44298	44198	25448	29220
		Worst	44353	44218	43469	41395	44291	–	–	–
		PDav	**0.006**	0.107	159.932	90.972	0.131	0.356	42.628	34.124
KP2_10000	90204	Best	**90085**	90059	89164	83678	–	–	–	–
		Avg	89735.1	89493.2	88397.4	83431.9	–	–	–	–

(*continued*)

Table 4. (*continued*)

Instance	Opt	Metric	CCSA	CSA	BA	PSO	EOS2	BB	GSA	SA
		Worst	89377	88453	87922	83083	–	–	–	–
		PDav	**0.520**	0.788	2.003	7.508	–	–	–	–
KP3_100	2397	Best	**2397**	**2397**	**2397**	2396	**2397**	–	–	–
		Avg	2396.9	2396.6	2397	2389.4	2397	2268	1095	2296
		Worst	2396	2396	2397	2375	2397	–	–	–
		PDav	0.004	0.017	**0**	6.753	**0**	5.382	54.318	4.214
KP3_200	2697	Best	**2697**	**2697**	**2697**	**2697**	**2697**	–	–	–
		Avg	2697	2697	2697	2693.6	2697	2542	1095	2594
		Worst	2697	2697	2697	2686	2697	–	–	–
		PDav	**0**	**0**	**0**	3.921	**0**	5.747	59.399	3.819
KP3_500	7117	Best	**7117**	**7117**	**7117**	6886	**7117**	–	–	–
		Avg	7117	7116.7	7117	6776.3	7117	6995	2916	6103
		Worst	7117	7116	7117	6703	7117	–	–	–
		PDav	**0**	0.004	**0**	53.275	**0**	1.714	59.028	14.248
KP3_1000	14390	Best	**14390**	**14390**	14378	13689	**14390**	–	–	–
		Avg	14390	14388.8	14287.5	13427.1	14390	14271	6290	11789
		Worst	14390	14386	14190	13284	14390	–	–	–
		PDav	**0**	0.008	44.388	113.989	**0**	0.827	56.289	18.075
KP3_2000	28919	Best	**28919**	**28919**	28605	26408	**28919**	–	–	–
		Avg	28919	28914.6	28456	26252.6	28919	28726	12312	22482
		Worst	28919	28900	28198	26069	28919	–	–	–
		PDav	**0**	0.015	131.725	133.015	**0**	0.667	57.426	22.259
KP3_5000	72505	Best	**72505**	72498	70504	65281	72205	–	–	–
		Avg	72504.9	72316.6	69970.3	64740.6	71984	72345	30302	53672
		Worst	72504	71980	69504	64195	71705	–	–	–
		PDav	**0**	0.260	280.271	325.025	0.719	0.221	58.207	25.975
KP3_10000	146919	Best	**146919**	146819	143778	129305	–	–	–	–
		Avg	146810	146262	141191.5	129010.4	–	–	–	–
		Worst	146513	145518	139315	128507	–	–	–	–
		PDav	**0.074**	0.447	3.898	12.189	–	–	–	–

* The bold numbers are the best obtained values

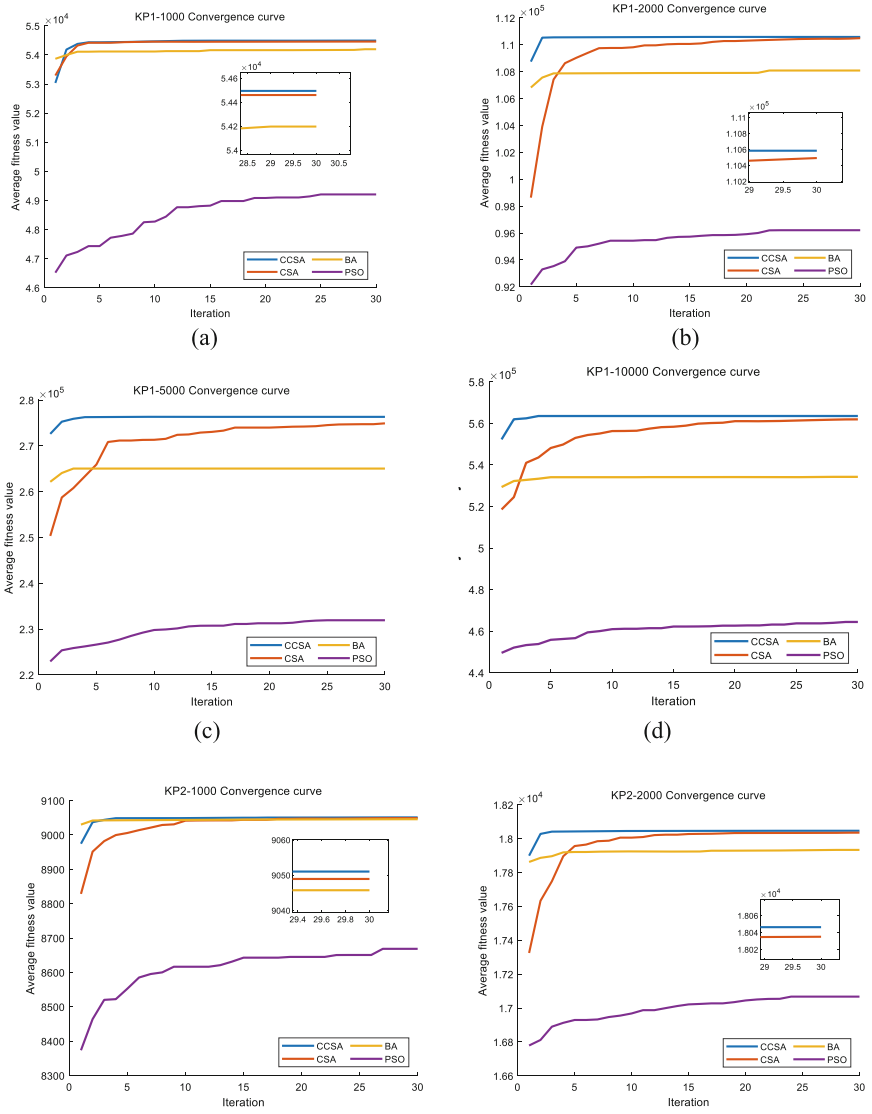

Fig. 3. Convergence curve of higher dimension 0–1 KP

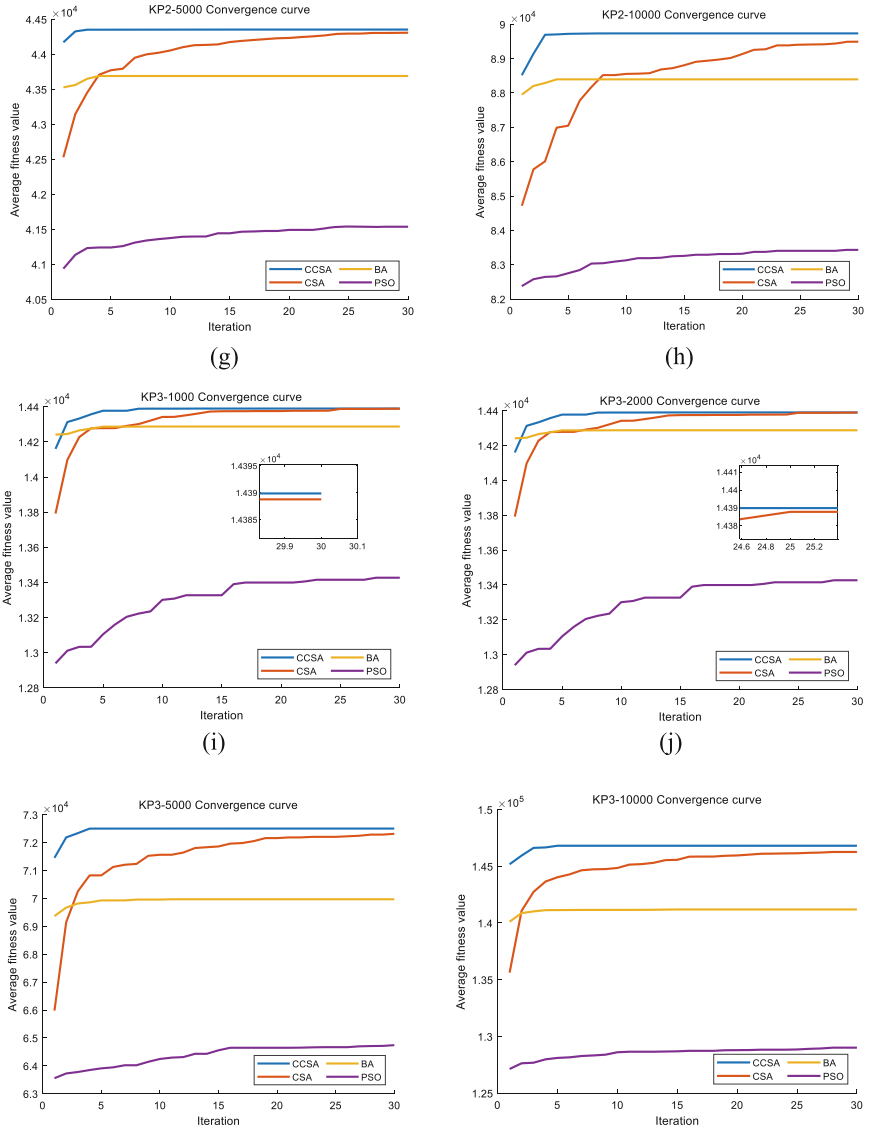

Fig. 3. continued

To sum up, (a) among all the algorithms from the literature, the proposed CCSA algorithm is the most effective optimization algorithm for 0–1 KP; (b) the performance of an algorithm directly depends on the type of high-dimensional KP01 instance, while CCSA achieves the optimal solution in all instances; (c) Compared with the standard CSA algorithm, the CCSA algorithm has greatly improved the convergence accuracy, convergence speed and stability, especially in high-dimensional instances. Finally, Table

5 shows p-values of the Wilcoxon ranksum test on high-dimensional 0–1 KP, among which 43 of the 54 p-values are less than 0.05. It can be concluded that in the high-dimensional test examples, the CCSA algorithm is significantly different from the CSA, BA, and PSO algorithms.

Table 5. p-values of the Wilcoxon ranksum test on high-dimensional 0–1 KP

Instance	CSA	BA	PSO
KP1_100	0.8061	6.5896e−04	1.2022e−08
KP1_200	0.0528	0.0419	3.9877e−12
KP1_500	4.0356e−04	0.3678	0.3678
KP1_1000	9.7445e−05	4.3128e−10	2.0283e−11
KP1_2000	3.3103e−10	2.1196e−11	2.4399e−11
KP1_5000	1.1093e−10	1.4484e−12	1.9801e−11
KP2_100	0.3337	NaN	2.0551e−13
KP2_200	0.3005	0.0815	3.9935e−04
KP2_500	2.1708e−10	2.7201e−11	7.9306e−11
KP2_1000	2.2572e−07	3.8938e−08	2.7739e−11
KP2_2000	1.3561e−09	3.2590e−10	2.5078e−11
KP2_5000	1.2762e−10	2.6477e−12	2.5062e−11
KP3_100	0.0534	1.2247e−12	1.1050e−09
KP3_200	0.0726	0.0419	2.9408e−11
KP3_500	1.3086e−07	0.9528	1.5956e−11
KP3_1000	6.3767e−07	1.7147e−10	2.4808e−11
KP3_2000	6.3767e−07	1.7147e−10	2.4808e−11
KP3_5000	4.8658e−10	1.0759e−11	2.5286e−11

6 Conclusions

The crow search algorithm (CSA) is a new swarm intelligence optimization algorithm proposed in recent years. Aiming at the shortcomings of the crow search algorithm, this paper integrates the idea of complex coding into the crow search algorithm, and proposes a complex-valued crow search algorithm (CCSA). The unique two-dimensional characteristics of complex numbers are used to increase the diversity of the population and improve the optimization performance of the algorithm. And use the CCSA algorithm to solve the 0–1 knapsack problem, and use low, medium and high dimensional examples to conduct simulation experiments. The experimental results show that the theoretical optimal solution can be found in the low and medium dimensions, and the theoretical optimal solution can be found in 16 of the 18 high-dimensional instances. It shows that

the CCSA algorithm proposed in this paper is effective and correct. More experiments or other improvement methods are needed in future work. In the future, you can try to propose other coding methods for crow search algorithms.

Acknowledgment. This work is supported by National Science Foundation of China under Grant No. 62066005, U21A20464.

References

1. Askarzadeh, A.: A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput. Struct. **169**, 1–12 (2016)
2. Laabadi, S., Naimi, M., El Amri, H., Achchab, B.: A binary crow search algorithm for solving two-dimensional bin packing problem with fixed orientation. Procedia Comput. Sci. **167**, 809–818 (2020)
3. Sayed, G.I., Hassanien, A.E., Azar, A.T.: Feature selection via a novel chaotic crow search algorithm. Neural Comput. Appl. **31**(1), 171–188 (2017). https://doi.org/10.1007/s00521-017-2988-6
4. Sahoo, R.M., Padhy, S.K.: Improved crow search optimization for multiprocessor task scheduling: a novel approach. In: Nayak, J., Balas, V.E., Favorskaya, M.N., Choudhury, B.B., Rao, S., Naik, B. (eds.) ARIAM 2019, pp. 1–13. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-30271-9_1
5. dos Santos Coelho, L., Klein, C.E., Mariani, V.C., do Nascimento, C.A.R., Askarzadeh, A.: Electromagnetic optimization based on Gaussian crow search approach. In: 2018 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), pp 1107–1112. IEEE (2018)
6. Mandala, J., Rao, M.C.S.: Privacy preservation of data using crow search with adaptive awareness probability. J. Inf. Secur. Appl. **44**, 157–169 (2019)
7. Lenin Fred, A., Kumar, S.N., Padmanaban, P., Gulyas, B., Ajay Kumar, H.: Fuzzy-crow search optimization for medical image segmentation. In: Oliva, D., Hinojosa, S. (eds.) Applications of Hybrid Metaheuristic Algorithms for Image Processing. SCI, vol. 890, pp. 413–439. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-40977-7_18
8. Shekhawat, S., Saxena, A.: Development and applications of an intelligent crow search algorithm based on opposition based learning. ISA Trans. **99**, 210–230 (2019)
9. Rizk-Allah, R.M., Hassanien, A.E., Slowik, A.: Multi-objective orthogonal opposition-based crow search algorithm for large-scale multi-objective optimization. Neural Comput. Appl. **32**(17), 13715–13746 (2020). https://doi.org/10.1007/s00521-020-04779-w
10. Rao, Y., He, D., Qu, L.: A probabilistic simplified sine cosine crow search algorithm for global optimization problems. Eng. Comput., 1–19 (2022). https://doi.org/10.1007/s00366-021-01578-2
11. Farh, H.M., Al-Shaalan, A.M., Eltamaly, A.M., Al-Shammaa, A.A.: A novel crow search algorithm autodrive PSO for optimal allocation and sizing of renewable distributed generation. IEEE Access **8**, 2780–27820 (2020)
12. Li, L.L., Liu, Z.F., Tseng, M.L., et al.: Using enhanced crow search algorithm optimization-extreme learning machine model to forecast short-term wind power. Expert Syst. Appl. **184**, 115579 (2021)
13. Necira, A., Naimi, D., Salhi, A., Salhi, S., Menani, S.: Dynamic crow search algorithm based on adaptive parameters for large-scale global optimization. Evol. Intell., 1–17 (2021). https://doi.org/10.1007/s12065-021-00628-4

14. Zhou, Y., Li, L., Ma, M.: A complex-valued encoding bat algorithm for solving 0–1 knapsack problem. Neural Process. Lett. **44**(2), 407–430 (2015). https://doi.org/10.1007/s11063-015-9465-y
15. Abdel-Basset, M., El-Shahat, D., Faris, H., Mirjalili, S.: A binary multi-verse optimizer for 0–1 multidimensional knapsack problems with application in interactive multimedia systems. Comput. Ind. Eng. **132**, 187–206 (2019)
16. Moradi, N., Kayvanfar, V., Rafiee, M.: An efficient population-based simulated annealing algorithm for 0–1 knapsack problem. Eng. Comput. **38**, 2771–2790 (2021). https://doi.org/10.1007/s00366-020-01240-3
17. Abdollahzadeh, B., Barshandeh, S., Javadi, H., Epicoco, N.: An enhanced binary slime mould algorithm for solving the 0–1 knapsack problem. Eng. Comput., 1–22 (2021). https://doi.org/10.1007/s00366-021-01470-z
18. Feng, Y., Wang, G.-G., Deb, S., Lu, M., Zhao, X.-J.: Solving 0–1 knapsack problem by a novel binary monarch butterfly optimization. Neural Comput. Appl. **28**(7), 1619–1634 (2015). https://doi.org/10.1007/s00521-015-2135-1
19. Rizk-Allah, R.M., Hassanien, A.E.: New binary bat algorithm for solving 0–1 knapsack problem. Complex Intell. Syst. **4**(1), 31–53 (2017). https://doi.org/10.1007/s40747-017-0050-z
20. Cao, J., Yin, B., Lu, X., Kang, Y., Chen, X.: A modified artificial bee colony approach for the 0–1 knapsack problem. Appl. Intell. **48**, 1582–1595 (2018)
21. Zhou, Y., Bao, Z., Luo, Q., Zhang, S.: A complex-valued encoding wind driven optimization for the 0–1 knapsack problem. Appl. Intell. **46**, 684–702 (2017)
22. Moradi, N., Kayvanfar, V., Rafiee, M.: An efficient population-based simulated annealing algorithm for 0–1 knapsack problem (2021)
23. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948. IEEE (1995)
24. Yang, X.-S.: A new metaheuristic bat-inspired algorithm. In: González, J.R., Pelta, D.A., Cruz, C., Terrazas, G., Krasnogor, N. (eds.) NICSO 2010. SCI, vol. 284, pp. 65–74. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12538-6_6
25. Abdel-Basset, M., Mohamed, R., Mirjalili, S.: A binary equilibrium optimization algorithm for 0–1 knapsack problems. Comput. Ind. Eng. **151**(3), 106946 (2020)
26. Ezugwu, A.E., et al.: A Comparative study of meta-heuristic optimization algorithms for 0–1 knapsack problem: some initial results. IEEE Access **7**, 43979–44001 (2019)