Self-adaptive Dragonfly Optimizer with Random Forest and Kernel Principal Component Analysis for Complex Optimization Problems

Ziqi Wang
College of Computer Science
Beijing University of Technology
Beijing, China
ziqi_wang@emails.bjut.edu.cn

Boqi Yu
College of Computer Science
Beijing University of Technology
Beijing, China
yuboqi1101@163.com

Hexiong Yang
College of Computer Science
Beijing University of Technology
Beijing, China
yanghexiong34@163.com

Yanan Liu
College of Computer Science
Beijing University of Technology
Beijing, China
yanan_liu1211@163.com

Jiahui Zhai
College of Computer Science
Beijing University of Technology
Beijing, China
zhaijiahui@emails.bjut.edu.cn

Jing Bi
College of Computer Science
Beijing University of Technology
Beijing, China
bijing@bjut.edu.cn

***Abstract*—Optimization algorithms (OAs) frequently address complex industrial automation and system scheduling design problems. However, when handling high-dimensional complex problems, these algorithms often struggle to explore the entire solution space due to their reliance on extensive computational resources and significant time costs. Additionally, they are prone to becoming trapped in local optima when solving high-dimensional problems. To address these challenges, this work proposes a novel OA named Self-adaptive Dragonfly Optimizer with random Forest and kernel-Principal component analysis (SDOFP). The Self-adaptive Dragonfly Optimizer (SDO) is designed to dynamically adjust parameters for adaptive exploration of the decision space. Kernel-Principal Component Analysis (K-PCA) is a dimensionality reduction method, turning the high-dimensional search space into a lower-dimension one. It can effectively guide the population toward the global optimum. Random Forest (RF) is utilized as a surrogate model to balance training time and prediction accuracy, conserving computational resources and enhancing overall performance. The performance of SDOFP is evaluated by comparing it with several typical peers across eight high-dimensional benchmark functions. Further validation is conducted by applying SDOFP to a real-world flexible job shop scheduling problem, demonstrating its practical applicability.**

***Index Terms*—Optimization algorithms, surrogate model, principal component analysis, high-dimensional optimization.**

I. INTRODUCTION

In contemporary engineering applications, optimization algorithms (OAs) are extensively used in design optimization, resource allocation, and scheduling. They are crucial in data mining, big data analytics, information security, and cloud computing [1]. Many practical problems involve a large number of decision variables, categorizing them as high-dimensional problems [2]. The emergence of high-dimensional problems poses challenges to traditional OAs because they cannot fully explore decision spaces. This challenge is particularly pronounced in engineering scheduling and financial investment, where high-dimensional problems necessitate costly and time-consuming function evaluations [3], [4].

Various research approaches are proposed to tackle practical high-dimensional problems, broadly categorized into three main types [5]. Firstly, integrating surrogate models within OAs plays a pivotal role [6]. Surrogate models simplify complex objective functions, significantly reducing computational costs and time [7], [8]. However, constructing and maintaining intricate surrogate models demands substantial computational resources and time investment. Additionally, selecting appropriate surrogate models and mitigating the risk of overfitting are important because inaccurate surrogate models directly impact optimization results.

The second category of methods involves dimensionality reduction. Through feature selection and feature extraction, dimensionality reduction helps reduce the number of decision variables, simplifying models' computational and interpretative processes [9], [10]. However, selecting appropriate dimensionality reduction methods is crucial, as it may lead to the loss of important data information during the reduction process, especially when dealing with large-scale datasets. The third category of methods focuses on improving OAs. By incorporating additional optimization strategies and adaptive parameter adjustments, the algorithm's search efficiency and convergence speed are improved [11], [12].

Based on the above analysis, this work proposes a novel optimization algorithm based on Self-adaptive Dragonfly Optimizer (SDO), integrating Random Forest (RF) and

This work was supported by the National Natural Science Foundation of China under Grants 62473014 and 62173013, the Beijing Natural Science Foundation under Grants L233005 and 4232049, and in part by Beijing University of Technology and Huawei "Intelligent Foundation" Industry Education Integration Collaborative Education Base Project (No. 40310790202101).

2024 China Automation Congress (CAC) | 979-8-3503-6860-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/CAC63892.2024.10865278

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.

Kernel-Principal Component Analysis (K-PCA) to address high-dimensional problems, named Self-adaptive Dragonfly Optimizer with random Forest and kernel-PCA (SDOFP). It is specifically designed for high-dimensional optimization tasks, utilizing RF and K-PCA as surrogate models and dimensionality reduction tools, respectively [13]. SDOFP dynamically adjusts parameters during algorithm iterations to find the global optimal and balance exploration and exploitation abilities, effectively exploring solution spaces. The integration of RF as a surrogate model learns complex nonlinear relationships within the data, balancing prediction accuracy and training time to reduce the computational costs of SDOFP. Additionally, K-PCA effectively reduces the dimensionality of high-dimensional space, further enhancing the efficiency and reliability of SDOFP in handling large-scale complex problems. Finally, a novel framework is proposed to integrate dimensionality reduction and surrogate models to tackle high-dimensional problems. Experimental validations on various unimodal and multimodal high-dimensional benchmark functions and applications in flexible workshop scheduling problems demonstrate the superior performance of SDOFP compared to its typical peers.

II. Proposed Framework

A. *Self-adaptive Dragonfly Optimizer*

SDO is proposed as a base optimizer for SDOFP. In SDOFP, gaussian process models (GPM) are employed for initializing the positions of dragonflies:

$$X=\{x_i\}_{i=1}^{n}, \quad x_i \sim \mathcal{U}(l_b, u_b) \tag{1}$$

$$Y=\{y_i\}_{i=1}^{n}, \quad y_i=f(x_i) \tag{2}$$

where n is the population size, x_i is the sample i. X represents n samples generated within a given range, and each sample has d dimensions. l_b and u_b denote the minimum and maximum values within the range, respectively [14]. Y denotes the objective function values computed for each sample point and forms the training set. A radial basis function (RBF) kernel defines GPM with a small noise level q. The generated training set is used to train GPM.

In each iteration, the position of each dragonfly is computed. Each dragonfly considers interactions with other dragonflies. To guide dragonflies move toward the global optimum while maintaining population diversity, multiple attractiveness functions ($\beta_1, \beta_2, \beta_3$) are designed to control the mutual attraction among dragonflies, *i.e.*,

$$\beta_1: \quad k=(f_1, f_2, \beta_0, \gamma)=\beta_0 \cdot e^{-\gamma \cdot d(f_1,f_2)^2} \tag{3}$$

$$\beta_2: \quad k=(f_1, f_2, \beta_0, \gamma)=\beta_0 \cdot e^{-2\gamma \cdot d(f_1,f_2)^2} \tag{4}$$

$$\beta_3: \quad k=(f_1, f_2, \beta_0, \gamma)=\beta_0 \cdot e^{-0.5\gamma \cdot d(f_1,f_2)^2} \tag{5}$$

where k denotes the attractiveness value. β_0 is the initial attractiveness coefficient and γ is the attractiveness decay coefficient. f_1 and f_2 are the positions of two dragonflies, and $d(f_1, f_2)$ is the distance between them, calculated by Euclidean distance.

β_1 is the standard attractiveness function, where attractiveness decays exponentially with the square of the distance. β_2 is the fast-decaying attractiveness function, reducing mutual attraction between dragonflies at longer distances. β_3 is the slow-decaying attractiveness function, leading to stronger mutual attraction between dragonflies even at longer distances.

After selecting the attractiveness function for each iteration, the positions of the dragonflies are updated. Dragonfly f_1 updates its position after being attracted by dragonfly f_2, and the new positions of dragonflies x_m^w are updated as:

$$x_m^w=f_1+k \cdot (f_2-f_1)+\alpha(r-0.5) \tag{6}$$

$$x_m^w \notin t_l \tag{7}$$

where α is the step size, r is a random value within [0,1], and $\alpha(r-0.5)$ introduces randomness to simulate the random flight behavior of dragonflies in nature. t_l is a list that contains positions to which the dragonflies are forbidden to move. It helps prevent the algorithm from getting trapped in local optima or revisiting already explored positions during the optimization process.

To enhance population diversity, SDO introduces the adaptive differential evolution (ADE) algorithm for mutation. This process simulates the mutation mechanisms in biological evolution to explore the solution space, guided by fitness evaluation strategies, thereby improving global search capabilities. It introduces adaptivity by dynamically adjusting the differential mutation factor (F) and crossover probability (CR) [15], enhancing optimization stability to prevent the algorithm from getting trapped in local optima. Specifically, three individuals from the current population are randomly selected as the base individuals for the mutation operation. These three individuals are P_a, P_b, and P_c, and $P_a, P_b, P_c \neq t_s$. t_s represents the index of the target individual. Next, a mutant individual v_p is generated using differential mutation, *i.e.*,

$$v_p=P_a+F \cdot (P_b-P_c) \tag{8}$$

To ensure v_p remains within the search space, truncation (clip) is applied:

$$v_p=\text{clip}(v_p, -3, 3) \quad p=1, 2, \ldots, d \tag{9}$$

Finally, CR is adopted to perform crossover between the mutant individual v_p and the target individual x_p, generating a trial individual u_p, *i.e.*,

$$u_p=\begin{cases} v_p & \text{if } l<CR \\ x_p & \text{otherwise} \end{cases} \tag{10}$$

where l is a random value in [0,1]. (10) states that for each dimension p, if a random number is less than CR, u_p takes the value from the mutant vector v_p; otherwise, u_p takes the value from the target individual x_p at that dimension.

In addition, Grey Wolf Optimizer (GWO) [16] is introduced into SDO to enhance the population diversity and the algorithm's global search capability. It is particularly effective

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.

for complex high-dimensional optimization problems. The population is updated as:

$$x_m^w = \frac{3X_{\alpha,m} + X_{\beta,m} + X_{\delta,m}}{3} \tag{11}$$

where $X_{\alpha,m}, X_{\beta,m}, X_{\delta,m}$ are positions of the three best individuals in the dimension m, respectively. α, β, δ are the dynamically adjusted parameters. They are updated in each generation based on the iteration count.

B. Kernel Principal Component Analysis

K-PCA is an extension of traditional PCA that leverages kernel methods for nonlinear dimensionality reduction [17], [18]. It is particularly advantageous for handling complex, high-dimensional datasets in optimization problems. By projecting the original high-dimensional data into a lower-dimensional space, K-PCA captures essential structures and patterns within the data. It is adopted to reduce the dimensionality of the input data before training the surrogate model to reduce its training time. Specifically, after the initial phase in the optimization process, K-PCA is used to compress the dimensionality of all individuals generated by OA before training the surrogate model. It does not require separate training because K-PCA is an unsupervised method. Thus, it enhances the algorithm's efficiency. In that case, surrogate models can be trained in a lower-dimensional space to accelerate their training speed.

C. Random Forest

RF is adopted as the surrogate model to enhance prediction accuracy and stability by combining predictions from multiple decision trees [19]–[21]. It achieves model diversity by independently constructing multiple decision trees on different subsets of the training data. Each tree randomly selects a subset of features for node splitting, generating a collection of trees with diverse structures that mitigate the risk of overfitting compared to a single model. During training, the model sequentially splits the input space by selecting features and splitting points to minimize the variance of the target variable until the predefined stopping criteria are met. This splitting process enables each tree to capture complex relationships between input variables and target values. In the prediction phase, each decision tree in RF independently predicts input data and the final prediction is the average of all tree predictions. By integrating results from multiple models, RF reliably approximates complex equations, reducing bias and variance.

RF excels at handling high-dimensional data. In SDOFP, this capability is crucial as it adapts quickly to optimization process changes and effectively manages the complexity of high-dimensional data. These characteristics make RF an ideal surrogate model, effectively approximating complex optimization problems.

D. SDO with RF and K-PCA

At the initial stage of SDOFP, the population is first initialized via GPM. Subsequently, it utilizes multiple attractiveness functions for iterative optimization, continuously adjusting the positions and brightness of dragonflies. The introduction of ADE enhances population diversity, effectively exploring and expanding the problem's solution space. GWO's hunting strategy enhances algorithm diversity and global search capability.

Then, SDOFP continuously collects data samples from each generation to train RF. Once predefined conditions are met, these data samples undergo processing in the K-PCA, which selects the most significant features in the decision space. This reduces data dimensionality and complexity, enabling RF to learn and accurately fit complex optimization patterns. This preprocessing alleviates the computational burden of subsequent optimization iterations and significantly reduces the training cost of the surrogate model itself. This ensures sufficient and representative data volume when transferring dimensionally reduced data to RF training. Thus, RF can learn and generalize intrinsic optimization strategies effectively. In subsequent optimization iterations, SDOFP adopts a surrogate-assisted computation approach, It integrates K-PCA with RF to enhance the algorithm's computational efficiency and global search capability. This provides a feasible approach to solving high-dimensional optimization problems. Fig. 1 shows the framework of SDOFP, and Algorithm 1 shows its pseudocode.

III. Experimental Results and Discussion

A. Benchmark Functions and Comparative Experiments

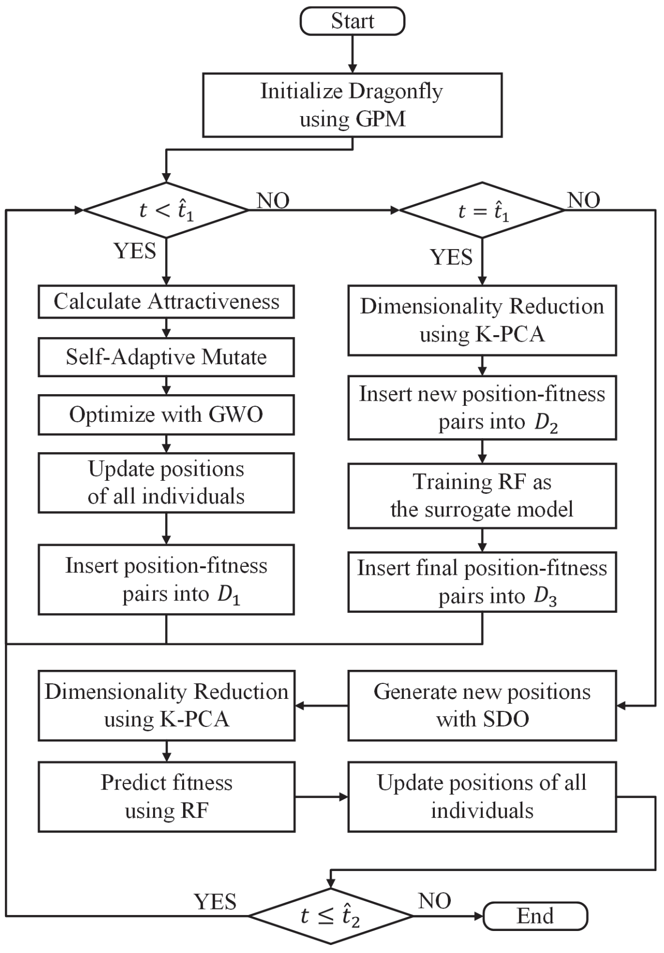
SDOFP is compared with four typical OAs, including ant colony algorithm (ACA) [22], original dragonfly algorithm (ODA), artificial bee colony algorithm (ABCA) [23], [24], and particle swarm optimization (PSO) [25]. In addition, six different benchmark functions are selected, including three unimodal and three multimodal functions. The details of benchmark functions are shown in Table I. Each benchmark function is executed independently 20 times, and the optimal solution's average value and standard deviation are recorded.

For SDOFP and the other four OAs, the population size is set to 100, maximum iterations are set to 1000, and the dimension of each individual is set to 100. For SDOFP, α is set to 0.5, β_0 and γ are both set to 1.0. For RF, 100 decision trees are used with a maximum depth of 10, controlling tree growth depth. Minimum samples per leaf are set to 2 and 1, respectively, with a sample ratio of 0.8 per tree. For K-PCA, 50 principal components are selected after dimensionality reduction. All these algorithms are implemented on a computer with an 11th Gen Intel(R) Core(TM) i7CPU 11800H operating at 2.30 GHz and 16 GB RAM.

B. Experimental Results

Table II presents the statistical results of benchmark functions after 1000 iterations, and Fig. 2 shows the corresponding convergence curves. It is shown in Fig. 2 that SDOFP achieves convergence in fewer iterations among all five algorithms. During the search process, RF assists SDOFP in exploring the solution space more effectively, improving convergence speed

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.



Start
Initialize Dragonfly using GPM
$t < \hat{t}_1$
NO
YES
$t = \hat{t}_1$
NO
YES
Calculate Attractiveness
Self-Adaptive Mutate
Optimize with GWO
Update positions of all individuals
Insert position-fitness pairs into D_1
Dimensionality Reduction using K-PCA
Insert new position-fitness pairs into D_2
Training RF as the surrogate model
Insert final position-fitness pairs into D_3
Dimensionality Reduction using K-PCA
Generate new positions with SDO
Predict fitness using RF
Update positions of all individuals
YES
$t \leq \hat{t}_2$
NO
End

Fig. 1. Framework of SDOFP

Algorithm 1 SDOFP

Input: Maximum iterations of SDO for K-PCA ($\hat{t}_1$), maximum number of iterations in SDOFP ($\hat{t}_2$), dataset before dimensionality reduction by K-PCA (D_1), dataset to train RF (D_2), dataset to evaluate RF performance (D_3)
Output: x_{best}, f_{best}

```
1:  Initialize $X$, $D_1$=∅, $D_2$=∅, and $D_3$=∅
2:  while $t<\hat{t}_1$ do
3:      $X' = \mathbf{SDO}(X)$
4:      Calculate the fitness value $f(X')$ of each individual in $X'$
5:      $D_1 = D_1 \cup (f(X'), X')$
6:      Select $X'$ as $X$ for the next generation
7:      $t = t + 1$
8:  end while
9:  if $t = \hat{t}_1$ then
10:     $D_2 = D_2 \cup (\mathbf{K\text{-}PCA}(D_1(X)), D_1(f(X)))$
11:     Initialize an empty list of decision trees $t_l$
12:     Initialize number of trees $n$
13:     for $i = 1$ to $n$ do
14:         Sample a bootstrap dataset $D_{boot}$ from $D_2$
15:         Train a decision tree $T_i$ on $D_{boot}$ using a Gini-index
16:         Add $T_i$ to $t_l$
17:     end for
18:     Output the decision trees $t_l$ and a trained RF model
19: end if
20: while $t \leq \hat{t}_2$ do
21:     Generate new positions $X$
22:     $X_{K-PCA} = \mathbf{K\text{-}PCA}(X)$
23:     $X' = \mathbf{SDO}(X_{K-PCA})$
24:     Calculate the fitness value $f(X')$ of each individual in $X'$
25:     $D_3 = D_3 \cup (f(X'), X')$
26:     $X_{RF} = \mathbf{RF}(X_{K-PCA})$
27:     Predict the fitness value $f(X'_{RF})$ of each individual in $X_{RF}$
28:     Select $X'$ as $X$ for the next generation
29:     $t = t + 1$
30:     Update $f_{best}$ and $x_{best}$
31: end while
32: Return $f_{best}$ and $x_{best}$
```

TABLE I
BENCHMARK FUNCTIONS

Functions	Range
$\text{F1}(x) = \sum_{i=1}^{n} x_i^2$	[-100,100]
$\text{F2}(x) = \sum_{i=1}^{n} (\lvert x_i \rvert) + \prod_{i=1}^{n} (\lvert x_i \rvert)$	[-10,10]
$\text{F3}(x) = \sum_{i=1}^{n} \left(1000^{\frac{i-1}{n-1}} \right) x_i^2$	[-100,100]
$\text{F4}(x) = = \sum_{i=1}^{n} \left(x_i^2 - 10 \cos(2\pi x_i) + 10 \right)$	[-5.12,5.12]
$\text{F5}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	[-600,600]
$\text{F6}(x) = -20 \cdot \exp\left(-0.2 \cdot \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	[-32,32]

and quality [26]. K-PCA significantly improves data separability through dimensionality reduction, enhancing the performance of SDOFP. SDOFP comprehensively improves the capability and efficiency in solving complex high-dimensional optimization problems [27].

For unimodal problems, Figs. 2(a)-(c) show the F1, F2, and F3 results after 1000 iterations. They demonstrate that SDOFP achieves the fastest convergence speed and optimal convergence performance. For multimodal problems, F4, F5, and F6 results are shown in Figs. 2(d)-(f). They indicate that SDOFP finds better solutions than its peers, exhibiting stronger search capabilities and convergence effects than the other four OAs. Table II shows that SDOFP achieves superior average results across six benchmark functions. Moreover, SDOFP exhibits minimal standard deviations, indicating stable performance and robustness.

C. Real-world Flexible Job Shop Scheduling Problem

SDOFP is further applied to address a practical Flexible Job Shop Scheduling Problem (FJSP) [28] to demonstrate its performance. This problem is crucial in production scheduling because FJSP allows flexible processing routes. Each operation can be processed on multiple machines with varying capabilities, resulting in different processing times. The completion time is the optimization objective function [29]. To enhance the performance of SDOFP, the first half of the population is allocated to represent different machine assignment schemes for operations. In addition, the second half describes the sequence of operations. We evaluate the algorithm's effectiveness in solving FJSP through 1000 iterations. SDOFP is compared with ABCA, ACA, PSO, and ODA on this problem. Fig. 3

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.

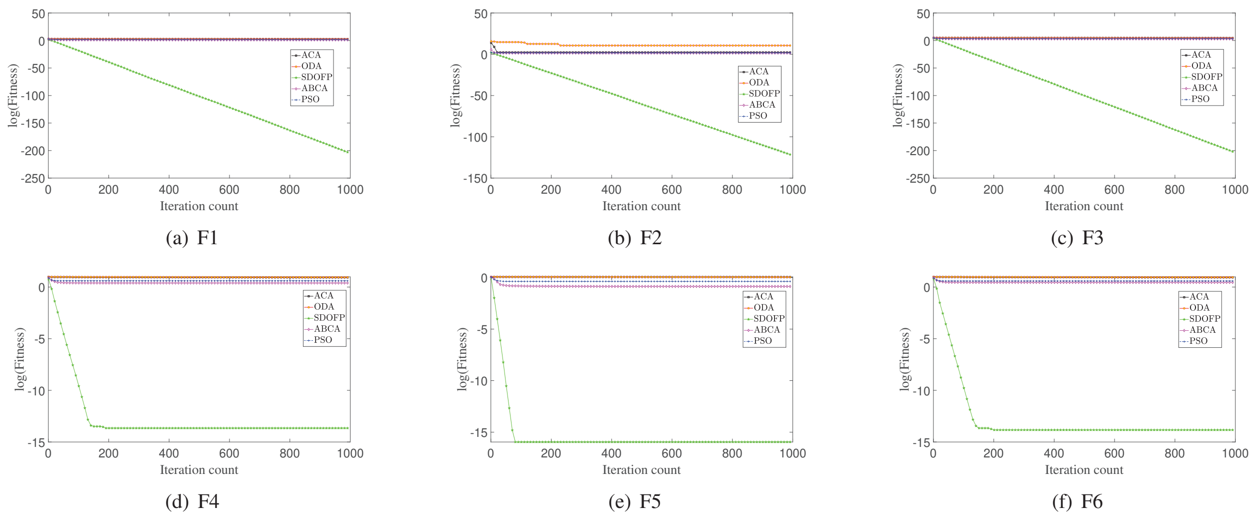


Fig. 2. Results of benchmark functions

TABLE II
RESULTS OF BENCHMARK FUNCTIONS

Functions	Algorithms	Mean	Std
F1	**SDOFP**	**1.1876×10^{-204}**	4.2173×10^{-205}
	PSO	$3.5781\times10^{+01}$	$6.0200\times10^{+00}$
	ABCA	$1.2966\times10^{+01}$	$2.5239\times10^{+00}$
	ACA	$4.6151\times10^{+02}$	$1.5970\times10^{+01}$
	ODA	$3.8160\times10^{+02}$	$7.2558\times10^{+01}$
F2	**SDOFP**	**1.7589×10^{-123}**	6.0788×10^{-124}
	PSO	$4.8301\times10^{+01}$	$4.2144\times10^{+00}$
	ABCA	$2.7114\times10^{+01}$	$2.5304\times10^{+00}$
	ACA	$1.8234\times10^{+02}$	$3.6345\times10^{+00}$
	ODA	$3.8210\times10^{+08}$	$5.4171\times10^{+07}$
F3	**SDOFP**	**1.9006×10^{-204}**	5.4692×10^{-205}
	PSO	$2.7374\times10^{+03}$	$5.4945\times10^{+02}$
	ABCA	$1.0308\times10^{+03}$	$1.8204\times10^{+02}$
	ACA	$3.4573\times10^{+04}$	$2.9206\times10^{+03}$
	ODA	$2.6643\times10^{+04}$	$5.1884\times10^{+03}$
F4	**SDOFP**	**1.4655×10^{-14}**	2.7687×10^{-15}
	PSO	**$4.0384\times10^{+00}$**	2.1122×10^{-01}
	ABCA	$2.8429\times10^{+00}$	1.2419×10^{-01}
	ACA	**$8.6660\times10^{+00}$**	1.4606×10^{-01}
	ODA	$8.7542\times10^{+00}$	1.0619×10^{-01}
F5	**SDOFP**	**2.7756×10^{-16}**	5.5511×10^{-17}
	PSO	4.5854×10^{-01}	9.5290×10^{-02}
	ABCA	1.9060×10^{-01}	3.5874×10^{-02}
	ACA	$1.1151\times10^{+00}$	4.9169×10^{-03}
	ODA	$1.0963\times10^{+00}$	1.0113×10^{-01}
F6	**SDOFP**	**1.8208×10^{-14}**	3.5527×10^{-15}
	PSO	$4.0421\times10^{+00}$	1.8547×10^{-01}
	ABCA	$2.8481\times10^{+00}$	1.3287×10^{-01}
	ACA	$8.6461\times10^{+00}$	1.6632×10^{-01}
	ODA	$8.7554\times10^{+00}$	1.1134×10^{-01}

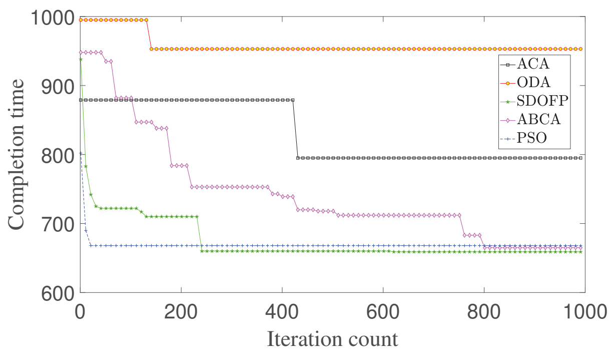


Fig. 3. Result of real-world problem

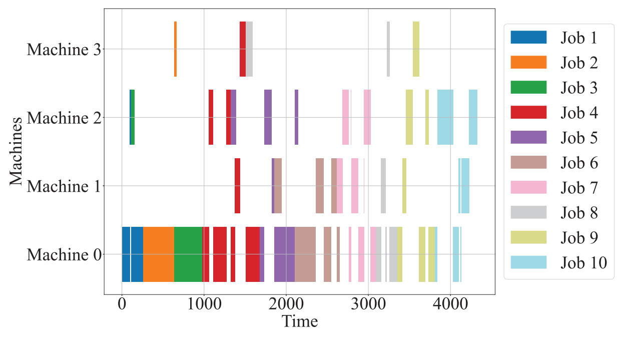


Fig. 4. Gantt of real-world problem

shows the iteration curve of each algorithm. It is shown that as the iteration progresses, the completion time of SDOFP gradually decreases, reaching the optimal value. Compared to other OAs, SDOFP achieves the shortest completion time, indicating its effectiveness. A Gantt chart is presented to illustrate the optimization results. It is shown in Fig. 4 that multiple tasks are scheduled on four machines, with different colors representing different tasks. Tasks on each machine are arranged chronologically to ensure that each machine executes only one task at each time point while minimizing idle time between tasks to maximize overall efficiency. As a result, it is proved that SDOFP can be applied to real-world problems.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.

IV. CONCLUSIONS

In modern industrial engineering, high-dimensional problems have extensive applicability and significance in production line optimization, resource allocation, and scheduling design. However, solving high-dimensional problems is constrained by the vast solution space and the high computational cost of function evaluations. Traditional optimization algorithms often cannot fully explore all possible solutions, leading to a trap in local optima. This study proposes a Self-adaptive Dragonfly Optimizer with random Forest and kernel-PCA (SDOFP). First, a self-adaptive dragonfly optimizer (SDO) is designed, and kernel-principal component analysis (K-PCA) is introduced for data dimensionality reduction. Meanwhile, random forest (RF) is introduced as a surrogate model to accelerate the optimization process and reduce computational costs. It can better balance exploration and exploitation in high-dimensional search spaces. SDOFP is compared with four state-of-the-art algorithms on six benchmark functions. Experimental results show that SDOFP achieves the best search performance. In addition, SDOFP is applied to a real-world job shop scheduling problem. The results show that SDOFP can generate solutions superior to its peers. Future work will explore other advanced surrogate models to enhance its performance further.

REFERENCES

[1] J. Bi, Z. Wang, H. Yuan, J. Zhang and M. Zhou, "Cost-Minimized Computation Offloading and User Association in Hybrid Cloud and Edge Computing," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16672–16683, May 2024.

[2] J. Bi, Z. Wang, H. Yuan, J. Zhang, M. Zhou, "Self-adaptive Teaching-learning-based Optimizer with Improved RBF and Sparse Autoencoder for High-dimensional Problems," *Information Sciences*, vol. 630, pp. 463–481, Jun. 2023.

[3] H. Yuan, J. Bi, Z. Wang, J. Yang, and Jia Zhang, "Partial and Cost-minimized Computation Offloading in Hybrid Edge and Cloud Systems," *Expert Systems with Applications*, vol. 250, pp. 1–13, Sept. 2024.

[4] Y. Hou, Y. Wu, H. Han, "Multiobjective Differential Evolution Algorithm Balancing Multiple Stakeholders for Low-Carbon Order Scheduling in E-Waste Recycling," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 6, pp. 1912–1925, Jan, 2023.

[5] M. Ming, A. Trivedi, R. Wang , D. Srinivasen, T. Zhang, "A dual-population-based evolutionary algorithm for constrained multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 739–753, Mar. 2021.

[6] Z. Akhtar, K. Rajawat, R. Wang , "Zeroth and first order stochastic Frank-Wolfe algorithms for constrained optimization.," *IEEE Transactions on Signal Processing*, vol. 70, pp. 2119–2135, Mar. 2022.

[7] R. Liu, Z. Liang, Z. Wang, W. Li, "Indoor visible light positioning based on improved whale optimization method with min-max algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–10, Jan. 2023.

[8] Q. Liu, X. Zhao, G. Wang "A clustering ensemble method for cell type detection by multiobjective particle optimization," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 1, pp. 1–14, Dec. 2021.

[9] F. Tonin, Q. Tao, P. Patrinos, J. A. Suykens, "Deep Kernel Principal Component Analysis for multi-level feature learning," *Neural Networks*, vol. 170, pp. 578–595, Feb. 2024.

[10] X. Yao, Q. Zhao, D. Gong and S. Zhu, "Solution of Large-scale Many-objective Optimization Problems Based on Dimension Reduction and Solving Knowledge Guided Evolutionary Algorithm," *IEEE Transactions on Evolutionary Computation*, pp. 1–15, Sept. 2021.

[11] L. Dey, A. Mukhopadhyay , "Compact Genetic Algorithm-Based Feature Selection for Sequence-Based Prediction of Dengue–Human Protein Interactions," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 19, no. 4, pp. 2137–2248, Mar. 2021.

[12] L. Liu, Z. Zhang, G. Chen, H. Zhang "Resource management of heterogeneous cellular networks with hybrid energy supplies: A multi-objective optimization approach," *IEEE Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4392–4405, Feb. 2021.

[13] P. Agrawal, T. Ganesh, A. W. Mohamed, "A novel binary gaining–sharing knowledge-based optimization algorithm for feature selection," *Neural Computing and Applications*, vol. 33, no. 11, pp. 5989–6008, Jun. 2021.

[14] C. Zhang, M. Dong, B. Liang, "Ultra-low-complexity algorithms with structurally optimal multi-group multicast beamforming in large-scale systems," *IEEE Transactions on Signal Processing*, vol. 71, pp. 1626–1641, Apr. 2023.

[15] J. Sun, X. Liu, T. Bäck, Z. Xu, "Learning adaptive differential evolution algorithm from optimization experiences by policy gradient," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 666–680, Feb. 2021.

[16] A. Routray, R. K. Singh, R. Mahanty, "Harmonic reduction in hybrid cascaded multilevel inverter using modified grey wolf optimization," *IEEE Transactions on Industry Applications*, vol. 56, no. 2, pp. 1827–1838, Dec. 2019.

[17] Erichson, N. Benjamin, Zheng, "Sparse principal component analysis via variable projection." *SIAM Journal on Applied Mathematics*, vol. 80, no. 2, pp. 977–1002, Oct. 2020.

[18] Abba, S. I,Pham "Emerging evolutionary algorithm integrated with kernel principal component analysis for modeling the performance of a water treatment plant." *Journal of Water Process Engineering*, vol. 33, pp. 101081, Feb. 2020.

[19] M. H. Sales, S. De Bruin, C. Souza, M. Herold, "Land use and land cover area estimates from class membership probability of a random forest classification." *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, Jun. 2021.

[20] Z. Wu, F. Shi, "Mapping forest canopy height at large scales using ICESat-2 and Landsat: An ecological zoning random forest approach." *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–16, Dec. 2022.

[21] D. Liu, Z. Fan "Random forest regression evaluation model of regional flood disaster resilience based on the whale optimization algorithm." *Journal of Cleaner Production*, vol. 250, pp. 119468–119475, Mar. 2020.

[22] S. E. Comert, H. R. Yazgan, "A new approach based on hybrid ant colony optimization-artificial bee colony algorithm for multi-objective electric vehicle routing problems," *Engineering Applications of Artificial Intelligence*, vol. 123, pp. 106375, Aug. 2023.

[23] C. Wang, P. Shang, P. Shen, "An improved artificial bee colony algorithm based on Bayesian estimation," *Complex & Intelligent Systems*, vol. 8, no. 6, pp. 4971–4991, Dec. 2022.

[24] F. Zhao, Z. Wang, L. Wang, "A reinforcement learning driven artificial bee colony algorithm for distributed heterogeneous no-wait flowshop scheduling problem with sequence-dependent setup times," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 4, pp. 2305–2320, Nov. 2022.

[25] Y. Hu, Y. Zhang, D. Gong, "Multiobjective particle swarm optimization for feature selection with fuzzy cost," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 874–888, Sept. 2020.

[26] C. Picard, J. Schiffmann, "Realistic constrained multiobjective optimization benchmark problems from design," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 2, pp. 234–246, Aug. 2020.

[27] S. Liu, Q. Lin, K. C. Wong, Q. Li, K. C. Tan, "Evolutionary large-scale multiobjective optimization: Benchmarks and algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 401–415, Jul. 2021.

[28] S. Luo, L. Zhang, Y. Fan, "Real-time scheduling for dynamic partial-no-wait multiobjective flexible job shop by deep reinforcement learning.", *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3020–3038, Aug. 2021.

[29] F. Zhang, Y. Mei, S. Nguyen, M. Zhang, "Multitask multiobjective genetic programming for automated scheduling heuristic learning in dynamic flexible job-shop scheduling." *IEEE Transactions on Cybernetics*, vol. 53, no. 7, pp. 4473-4486, Aug. 2022.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on July 22,2025 at 03:55:46 UTC from IEEE Xplore. Restrictions apply.