Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **9**, Number **1**, February **2024**

Parallel Binary Tumbleweed Algorithm and Its Application for Intrusion Detection Systems

Ruo-Bin Wang

School of Information Science and Technology
North China University of Technology
Beijing 100144, China
robin945@163.com

Fang-Dong Geng

School of Information Science and Technology
North China University of Technology
Beijing 100144, China
gfd21@qq.com

Lin Xu*

STEM
University of South Australia
Adelaide 5095
callyxu1005@gmail.com

*Corresponding author: Lin Xu

Received September 15, 2023, revised November 2, 2023, accepted December 27, 2023.

Abstract. *The tumbleweed algorithm(TA) is a newly proposed population-based meta-heuristic algorithm that cannot be directly used to solve binary optimization problems. In this paper, we propose a parallel binary tumbleweed algorithm(PBTA) for solving the feature selection problem of intrusion detection systems using a novel time-varying transfer function. In addition, a parallel mechanism and two communication strategies are introduced to enhance the diversity of the population to achieve better algorithm performance. The proposed PBTA is verified on the CEC2013 benchmark and in comparison with some popular algorithms, and the results demonstrate the superiority of PBTA. Eventually, PBTA is employed to solve the problem of feature selection for intrusion detection system, and the experimental results prove that PBTA is effective in finding the optimal feature subset which improves the classification quality while minimizing the amount of features employed.*

Keywords: tumbleweed algorithm; parallel binary; intrusion detection system

1. **Introduction.** In the presence of increasingly complex optimization problems in the real world, the search for efficient novel optimization approaches has become a focus of attention. Generally, statistical methods are employed to solve real-world application problems in various fields [1]. Unfortunately, these mathematical methods encounter great challenges in addressing complex optimization problems, like multimodal problems. In consequence, nature-inspired meta-heuristic algorithms have been designed to solve various combinatorial optimization problems [2]. Although they cannot find the optimal

solution of the problem every time, they continue to be capable of finding near-optimal or optimal solutions within limited resources [3].

Meta-heuristic algorithms can be categorized into two categories: evolution-based as well as swarm intelligence-based. Evolution-based algorithms mimic the process of evolution in nature, including concepts such as heredity and mutation. Popular evolutionary algorithms include genetic algorithms [4][5] and differential evolutionary algorithms [6][7]. In recent years, some novel evolutionary algorithms have been widely applied in a variety of real-world applications. For example, Hu et al. proposed a deep reinforcement learning-assisted co-evolutionary differential evolution, which can effectively use reinforcement learning to help evolutionary algorithms solve constrained optimization problems [8]. Nazeri et al. proposed a centrality-based genetic algorithm by introducing novel genetic operators and chromosome representations to solve the graph burning problem [9]. Kang et al. optimize the hyperparameters of gaussian process regression using non-inertial particle swarm optimization with elite mutation and apply it to uncertain measurement and prediction of non-smooth time series [10]. Chen et al. proposed a genetic algorithm combined with simulated annealing to provide an alternative approach for solving the time-varying network problem [11]. Shaik et al. proposed gaussian mutation–spider monkey optimization model, which obtained high accuracy in solving scene classification problems [12]. Swarm intelligence-based algorithms achieve the solution or optimization of a problem by simulating the interaction and cooperation between individuals in a population. These algorithms have inspiration from the behavior of groups in nature, such as ant colonies [13][14], bird swarm [15][16][17], and fish schools [18][19]. A number of meta-heuristic algorithms based on swarm intelligence have been proposed extensively for the solution of complex optimization problems [20]. For instance, Dhiman et al. designed a seagull optimization algorithm based on the migration and attack behaviors of seagulls, which is applicable for solving challenging large-scale constrained problems [21]. Khishe et al. proposed a chimp optimization algorithm, which simulates individual intelligence in group hunting, and it provides a new option for solving high-dimensional optimization problems [22]. Pan et al. proposed the gannet optimization algorithm inspired by the feeding behavior of gannet, which is a good choice for solving engineering optimization problems [23].

Tumbleweed algorithm is a newly proposed swarm intelligence algorithm, which can effectively solve the logistics center location problem [24]. Yet, the algorithm is incapable of solving discrete and binary optimization problems, such as the intrusion detection system(IDS) feature selection problem. While the binary tumbleweed optimization algorithm has been proposed to solve binary optimization problems, the algorithm does not have competitive performance in solving the intrusion detection system feature selection problem [25].

In this paper, we propose a parallel binary tumbleweed algorithm (PBTA) that employs a new time-varying transfer function, and two parallel strategies are introduced to enhance the diversification of the population to achieve better computational efficiency of the algorithm. A summary of the main contributions of our work can be presented as follows:

- A parallel binary tumbleweed algorithm is proposed which employs a novel time-varying transfer function, a parallel mechanism and two communication strategies are employed to achieve better performance of the algorithm.
- We tested the performance of PBTA on CEC2013 benchmark functions and compared it with popular algorithms.
- PBTA is applied to solve the problem of feature selection for intrusion detection system.

The remainder of this paper is presented as shown below: section 2 describes the original tumbleweed algorithm, section 3 introduces the proposed parallel binary tumbleweed algorithm, section 4 evaluates the performance of the algorithm on the CEC2013 test function, PBTA is used to solve the feature selection problem for intrusion detection systems in section 5, and section 6 summarizes the work in this paper.

2. **Tumbleweed Algorithm.** The Tumbleweed Algorithm (TA) is a novel swarm intelligence optimization algorithm inspired by the seedling growth and seed reproduction of tumbleweeds [24][26]. In TA, N tumbleweed seedlings are denoted by $X_i (i = 1, 2, \ldots, N)$ and their fitness is denoted as $fitness(X_i)$. Only one of the most vigorously growing of the N seedlings was capable of growing into a complete tumbleweed plant, which is denoted by X_{gbest}, and spread its seeds. During the seedling growth stage, the growth of seedlings is always influenced by two parts of environmental factors. One part is the effect of the strong seedling on the seedling, denoted by $F1$, and the other part is influenced by factors including light and soil, denoted by $F2$. The expression formulae for $F1$ and $F2$ are shown in Equation 1 and Equation 2.

$$F1 = r_1 * \left(X_{\text{gbest}} - X^i_{\text{old}} \right) \tag{1}$$

$$F2 = \frac{r_2}{\frac{fitness(X^i_{old})}{sum(fitness(X_{old}))} + 1} \tag{2}$$

where r_1 is an influence coefficient that decreases linearly from t to 0 with iterations of the algorithm, as shown in Equation 3. And r_2 stands for the factor of environmental impact, which takes values from 0 to 0.1.

$$r_1 = t * (1 - \frac{iter}{Max\_iter}) \tag{3}$$

where t takes the value 2, $iter$ stands for the iteration in progress, and Max_iter indicates the iteration limit. Lastly, the position update equation for the seedling growing phase appears in Equation 4.

$$X^i_{new} = X^i_{old} + F1 + F2 \tag{4}$$

At the phase of propagation of the seed, only a most vigorous seedling grow into a complete tumbleweed and spread the seeds, with position updated as shown in Equation 5.

$$X^i_{new} = X_{gbest} + V_i * \frac{iter}{Max\_iter} \tag{5}$$

where V_i denotes a random value from $[lb, ub]$, lb and ub denote the upper and lower boundaries of the search space, respectively. The pseudo-code of TA is shown in Algorithm 1.

3. **Parallel Binary Tumbleweed Algorithm.** In the continuous version of TA, tumbleweeds update their positions in the continuous search space. In the binary version of TA, however, the position can only take values of 0 or 1. Many discrete and binary optimization problems can be solved using these two values, such as the knapsack problem and the feature selection problem.

Algorithm 1 Pseudo-code of the TA

input: N: size of population; dim: dimensions of the problem; $f(x)$: evaluation function; grc: growth cycle; Max_iter: maximum iterations
output: X_{best} and $f(X_{best})$.
Initialize the position of the solution X_i in the population randomly;
While ($iter < Max\_iter$) do
  Evaluate the fitness score for all individuals;
  Update X_{best} and $f(X_{best})$;
  if $mod(Max\_iter, grc) < grc/2$ do
    compute $F1$ and $F2$ by Equation 1 and Equation 2;
    update r_1 by Equation 3;
    update X_{new} by Equation 4;
  else
    update X_{new} by Equation 5;
  end if
  Boundary value processing;
end while

3.1. **Transfer function.** Transfer functions play an important role in the conversion of a continuous algorithm to a binary algorithm and they have a significant impact on the performance of binary algorithms [27]. The family of commonly used S-type transfer functions is presented in Table 1, and the curves are plotted in Figure 1. As can be seen from Figure 1, the S1 function have a high probability of positional change, which makes it easy for the algorithm to get rid of local traps. However, this still prevents the algorithm from converging to an optimal value. Consequently, we propose a novel time-varying transfer function to prevent the algorithm from converging prematurely, as shown in Equation 6, and the function curve is shown in Figure 2. As can be seen in Figure 2, the proposed transfer function changes with the iteration of the algorithm. In the early stage of the algorithm, the switching rate of 0 and 1 is large, which is conducive to the algorithm's large-scale search; in the late stage of the algorithm, the switching rate of 0 and 1 slows down, which is conducive to the local optimization search.

Table 1. The family of S-type transfer functions.

Name	Transfer function
S1	$T = \frac{1}{1+e^{-2x}}$
S2	$T = \frac{1}{1+e^{-x}}$
S3	$T = \frac{1}{1+e^{-\frac{x}{2}}}$
S4	$T = \frac{1}{1+e^{-\frac{x}{3}}}$

$$S(x) = \frac{1}{1 + e^{-\frac{Max\_iter}{iter}(x-0.5)}} \tag{6}$$

3.2. **Parallel mechanism.** A parallel binary tumbleweed algorithm(PBTA) is proposed for further improving the performance of binary algorithms. In PBTA, a population of size N is equally divided into 3 groups. Each subpopulation operates independently and exchanges through two communication strategies. The first communication strategy is

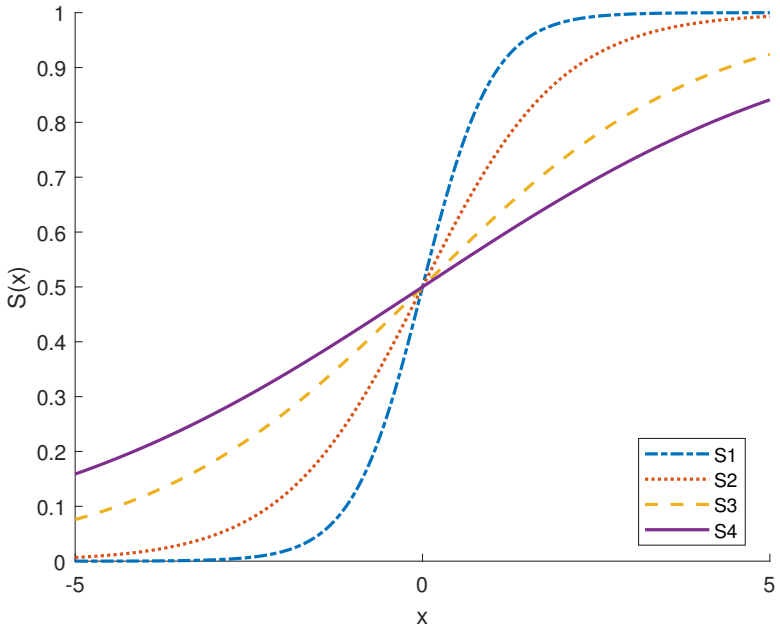

FIGURE 1. The curves of S-type transfer function family.

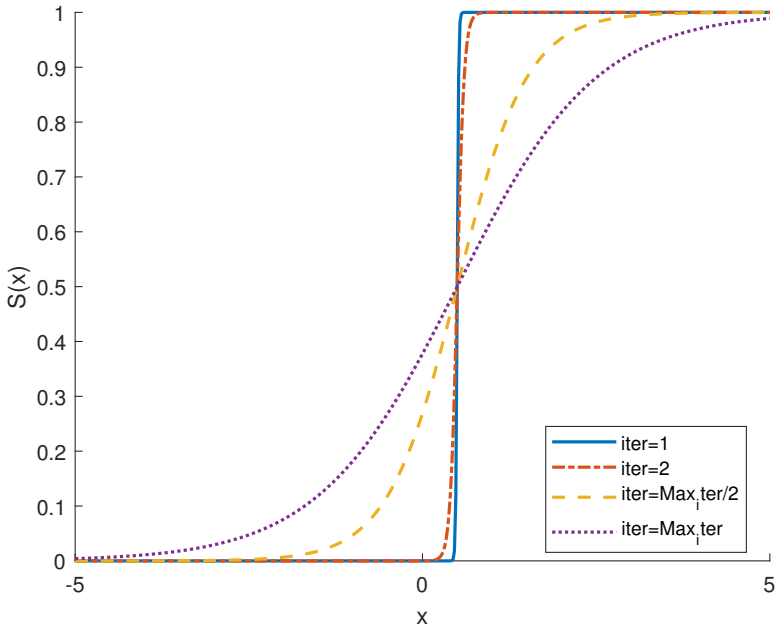

FIGURE 2. The curve of the proposed time-varying transfer function.

represented as randomly selecting two subpopulations every M generations to exchange half of the individuals. The second communication strategy is represented as every $2M$ generations, the two worst values of each subpopulation are replaced by the best values of the other two subpopulations.The two communication strategies are shown in Figure 3. It can be seen that the worst individuals in the three subpopulations are gradually eliminated by the two strategies, which increases the diversity of the population and

improves the capability of the algorithm to escape from localized traps. The pseudo-code of PBTA is shown in Algorithm 2.

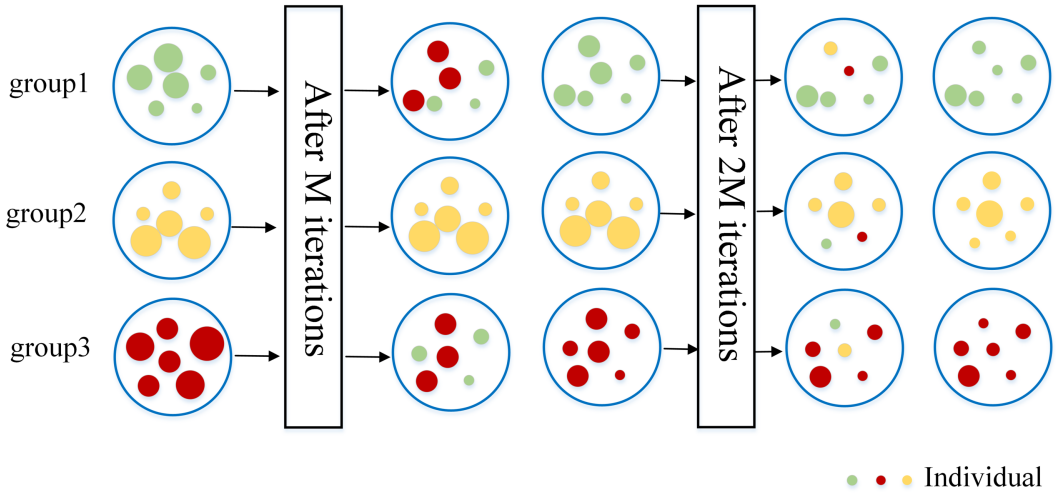

FIGURE 3. The proposed two parallel strategies.

Algorithm 2 Pseudo-code of the PBTA

input: N: size of population; dim: dimensions of the problem; $f(x)$: evaluation function; grc: growth cycle; Max_iter: maximum iterations
M: iteration frequency for group communication
output: X_{best} and $f(X_{best})$.
Initialize the position of the solution X_i with 0 or 1;
While ($iter < Max\_iter$) do
  Using transfer function to update X_{new};
  Calculate the fitness value for all individuals;
  Update X_{best} and $f(X_{best})$;
  if $mod(Max\_iter, grc) < grc/2$ do
    compute $F1$ and $F2$ by Equation 1 and Equation 2;
    update r_1 by Equation 3;
    update X_{new} by Equation 4;
  else
    update X_{new} by Equation 5;
  end if
  Boundary value processing;
  if mod($iter$,M)=0 do
    Execute the first communication strategy;
  elif mod($iter$,$2M$)=0 do
    Execute the second communication strategy;
  end if
end while

4. **Statistical Results of the Experiments.** The capability of PBTA on the CEC2013 benchmark function is verified in this section. These benchmark functions include unimodal functions(f1-f5), multimodal functions (f6-f20), and composition functions (f21-f28). Firstly, we verify the effect of different transfer functions on the binary version

of the tumbleweed algorithm(BTA), and secondly, we compare the PBTA with other popular algorithms such as binary particle swarm optimization (BPSO)[28], binary bat algorithm (BBA)[29], binary grey wolf optimizer (BGWO)[30], binary fish migration algorithm (BFMO)[31], and the binary version of the hybrid grey wolf optimization and particle swarm optimization (BGWOPSO)[32]. Some algorithm parameters are shown in Table 2. All algorithms run independently 30 times.

Table 2. Parameter setting of the algorithms.

Algorithm	Parameters
PBTA	N=60, Max_iter=500, t=2
BBA	N=60, Max_iter=500, A=0.25, r=0.1
BPSO	N=60, Max_iter=500, $c1$=$c2$=1.5, V=1, $Wmax$=0.9, $Wmin$=0.4
BGWO	N=60, Max_iter=500
BFMO	N=60, Max_iter=500
BGWOPSO	N=60, Max_iter=500, W=[0.5,1], v=[0,0.3], $c1$=$c2$=0.5

4.1. **Comparison of different transfer functions.** Table 3 and Table 4 shows the statistical results of the BTA family of algorithms, where PBTA uses the proposed transfer function with parallel mechanism, BTA uses the proposed transfer function without parallel mechanism, and BTA_S1, BTA_S2, BTA_S3, and BTA_S4 employ the transfer functions S1, S2, S3, and S4, respectively, without parallel mechanism. The *Average rank* is the result of the Friedman test and the *Overall rank* means the final ranking. From Table 3, we can conclude that compared to BTA_S1, BTA_S2, BTA_S3 and BTA_S4, BTA performs more superior on all functions. Table 4 shows the best stabilization of BTA performance. These demonstrate the capability of the proposed transfer function to improve the efficiency of the algorithm. Furthermore, it can be seen that PTA is stable and the performance of PTA is superior to BTA across all functions. This proves that the proposed parallel mechanism improves the performance of the algorithm without losing the stability of the algorithm.

4.2. **Comparison results between PBTA and other popular algorithms.** Results from some classic meta-heuristics algorithms are compared with results of PBTA to see how PBTA performs. These algorithms include BPSO, BBA, BGWO, BFMO, and BGWOPSO. Table 5 and Table 6 demonstrate the statistical results of 30 runs (mean and standard deviation).

The results indicates that the PBTA performs better than the BPSO, BFMO, and BGWO on unimodal functions, while exhibiting the same level of performance as the BBA and BGWOPSO. It proves the good convergence capability of the PBTA. And PBTA performs worse than BGWOPSO only on F9, F15, and F16 in multimodal functions, but outperforms all other algorithms on all functions, which demonstrates the superior performance of PBTA in escaping from local optima. The composition functions can be employed to test the comprehensive capability of an algorithm, and PBTA outperforms BBA, BPSO, and BGWO on all composition functions. In addition, the Friedman average ranking results show that PBTA has a competitive performance and outperforms the other five algorithms overall. The foregoing discussion demonstrates that PBTA achieves fast convergence and excellent local optima escaping capability while also demonstrating competitive performance in resolving binary optimization problems.

TABLE 3. The mean of 30 experiments for the BTA family of algorithms.

Function	PBTA	BTA	BTA_S1	BTA_S2	BTA_S3	BTA_S4
F1	6.775E+04	6.775E+04	6.787E+04	6.792E+04	6.797E+04	6.800E+04
F2	7.334E+09	7.334E+09	7.362E+09	7.377E+09	7.386E+09	7.395E+09
F3	8.883E+22	8.883E+22	9.355E+22	9.435E+22	9.573E+22	9.610E+22
F4	6.838E+04	6.844E+04	6.859E+04	6.857E+04	6.861E+04	6.859E+04
F5	1.024E+05	1.024E+05	1.024E+05	1.024E+05	1.024E+05	1.024E+05
F6	2.474E+04	2.474E+04	2.483E+04	2.485E+04	2.487E+04	2.488E+04
F7	2.632E+08	2.655E+08	2.649E+08	2.638E+08	2.666E+08	2.658E+08
F8	-6.790E+02	-6.790E+02	-6.789E+02	-6.790E+02	-6.789E+02	-6.790E+02
F9	-5.427E+02	-5.429E+02	-5.426E+02	-5.425E+02	-5.424E+02	-5.423E+02
F10	1.471E+04	1.471E+04	1.472E+04	1.474E+04	1.475E+04	1.476E+04
F11	8.353E+02	8.353E+02	8.405E+02	8.429E+02	8.466E+02	8.468E+02
F12	8.790E+02	8.790E+02	8.854E+02	8.896E+02	8.931E+02	8.945E+02
F13	9.803E+02	9.803E+02	9.814E+02	9.832E+02	9.863E+02	9.865E+02
F14	1.162E+04	1.162E+04	1.168E+04	1.174E+04	1.179E+04	1.180E+04
F15	1.160E+04	1.159E+04	1.167E+04	1.172E+04	1.180E+04	1.182E+04
F16	2.029E+02	2.030E+02	2.034E+02	2.035E+02	2.035E+02	2.036E+02
F17	1.396E+03	1.396E+03	1.399E+03	1.402E+03	1.406E+03	1.409E+03
F18	1.369E+03	1.362E+03	1.368E+03	1.376E+03	1.375E+03	1.376E+03
F19	1.964E+06	1.964E+06	1.965E+06	1.965E+06	1.965E+06	1.965E+06
F20	6.150E+02	6.150E+02	6.150E+02	6.150E+02	6.150E+02	6.150E+02
F21	3.439E+03	3.439E+03	3.439E+03	3.440E+03	3.442E+03	3.442E+03
F22	1.241E+04	1.241E+04	1.244E+04	1.248E+04	1.252E+04	1.253E+04
F23	1.254E+04	1.254E+04	1.260E+04	1.265E+04	1.268E+04	1.269E+04
F24	2.051E+03	2.051E+03	2.056E+03	2.059E+03	2.063E+03	2.063E+03
F25	1.635E+03	1.635E+03	1.638E+03	1.639E+03	1.639E+03	1.639E+03
F26	5.144E+03	5.144E+03	5.226E+03	5.241E+03	5.266E+03	5.265E+03
F27	4.699E+03	4.698E+03	4.705E+03	4.708E+03	4.712E+03	4.713E+03
F28	1.058E+04	1.063E+04	1.074E+04	1.074E+04	1.076E+04	1.077E+04
Average rank	1.6071	1.7500	3.0893	3.8393	5.0536	5.6607
Overall rank	1	2	3	4	5	6

5. **Application for IDS Feature Selection.** The effectiveness of intrusion detection systems is negatively impacted by a number of characteristics, so we actively work to identify features with a positive impact on performance [33]. In this section, PBTA is applied to solve the intrusion detection system feature selection problem, where PBTA is used to search for the best subset of features and K-nearest neighbor (KNN) is employed as a classifier.

5.1. **Dataset description and data pre-processing.** The NSL-KDD dataset is a subset of KDD99 that addresses some of the shortcomings of the original dataset [34]. It consists of nine different kinds of cyberattacks that are categorized into training and testing sets, KDD_{train} and KDD_{test}, they contain 125973 instances and 22544 instances with 41 features respectively. All attacks are categorized as normal or abnormal. Since different features may be measured differently, the data must be normalized to eliminate

TABLE 4. The standard deviation of 30 experiments for the BTA family of algorithms.

Function	PBTA	BTA	BTA_S1	BTA_S2	BTA_S3	BTA_S4
F1	5.920E-11	5.920E-11	5.086E+01	5.101E+01	5.539E+01	5.006E+01
F2	2.910E-06	2.910E-06	1.295E+07	1.434E+07	1.463E+07	9.390E+06
F3	5.119E+07	5.119E+07	1.366E+21	1.364E+21	1.439E+21	1.592E+21
F4	7.374E+01	6.695E+01	7.436E+01	8.404E+01	1.058E+02	1.157E+02
F5	0.000E+00	0.000E+00	1.247E-02	3.186E-02	3.981E-02	4.904E-02
F6	1.850E-11	1.850E-11	3.026E+01	3.226E+01	3.480E+01	3.171E+01
F7	5.285E+06	6.389E+06	5.254E+06	5.347E+06	6.421E+06	5.554E+06
F8	5.643E-02	5.339E-02	4.284E-02	7.384E-02	6.117E-02	4.317E-02
F9	2.141E-01	1.574E-01	1.440E-01	1.712E-01	1.378E-01	1.030E-01
F10	7.400E-12	7.400E-12	7.075E+00	1.049E+01	1.329E+01	1.242E+01
F11	0.000E+00	0.000E+00	1.885E+00	2.613E+00	2.783E+00	2.249E+00
F12	0.000E+00	0.000E+00	2.534E+00	3.485E+00	3.556E+00	4.081E+00
F13	1.033E+00	8.834E-01	9.379E-01	1.682E+00	2.209E+00	2.816E+00
F14	5.550E-12	5.550E-12	2.602E+01	3.864E+01	4.221E+01	4.252E+01
F15	2.696E+01	5.550E-12	4.424E+01	4.260E+01	4.030E+01	3.739E+01
F16	4.828E-01	3.599E-01	3.679E-01	4.482E-01	4.618E-01	4.214E-01
F17	0.000E+00	0.000E+00	1.441E+00	2.604E+00	3.309E+00	3.244E+00
F18	1.424E+01	1.241E+01	1.380E+01	1.091E+01	1.103E+01	8.726E+00
F19	9.472E-10	9.472E-10	1.200E+02	1.418E+02	3.724E+02	3.703E+02
F20	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
F21	1.388E-12	1.388E-12	2.005E-01	5.013E-01	9.071E-01	9.705E-01
F22	1.850E-12	1.850E-12	1.879E+01	2.150E+01	2.485E+01	3.740E+01
F23	2.580E+00	7.400E-12	3.916E+01	3.168E+01	3.290E+01	3.559E+01
F24	9.250E-13	9.250E-13	1.711E+00	2.700E+00	2.422E+00	2.284E+00
F25	8.532E-01	7.030E-01	5.253E-01	8.027E-01	1.115E+00	8.938E-01
F26	9.250E-13	9.250E-13	2.679E+01	2.432E+01	2.442E+01	2.809E+01
F27	1.787E+00	1.363E+00	3.608E+00	3.517E+00	3.399E+00	3.187E+00
F28	8.483E+01	5.873E+01	7.178E+01	6.195E+01	3.550E+01	6.785E+01

effects. The min–max normalization method is employed to normalize the data, as shown in Equation 7.

$$\tilde{d_{ij}} = \frac{d_{ij} - min(d_i)}{max(d_i) + min(d_j)} \tag{7}$$

where $min(d_i)$ and $max(d_i)$ are the minimum and maximum values of the ith feature d_i, and $\tilde{d_{ij}}$ is the normalized value.

Subsequently, we process the three symbolic features(protocol type, service, and flag) using the one-hot-encoding method. Eventually, the three symbolic features are represented by binary values, and from the initial three symbolic dimensions we obtain 84 binary-valued characteristics.

5.2. **Fitness function.** The results of the experiment are evaluated by a combination of the cross-validation error and the number of features selected. The fitness function used in our work is given in Equation 8.

TABLE 5. The mean of 30 experiments comparing PBTA with other classical algorithms.

Function	PBTA	BBA	BPSO	BGWO	BFMO	BGWOPSO
F1	6.775E+04	6.775E+04	6.798E+04	6.789E+04	6.815E+04	6.775E+04
F2	7.334E+09	7.334E+09	7.386E+09	7.365E+09	7.417E+09	7.334E+09
F3	8.883E+22	8.883E+22	9.557E+22	9.346E+22	9.954E+22	8.883E+22
F4	6.838E+04	6.824E+04	6.863E+04	6.888E+04	6.887E+04	6.841E+04
F5	1.024E+05	1.024E+05	1.024E+05	1.024E+05	1.024E+05	1.024E+05
F6	2.474E+04	2.474E+04	2.487E+04	2.483E+04	2.496E+04	2.474E+04
F7	2.632E+08	2.657E+08	2.649E+08	2.815E+08	2.717E+08	2.681E+08
F8	-6.790E+02	-6.789E+02	-6.789E+02	-6.788E+02	-6.789E+02	-6.788E+02
F9	-5.427E+02	-5.428E+02	-5.423E+02	-5.423E+02	-5.420E+02	-5.429E+02
F10	1.471E+04	1.471E+04	1.476E+04	1.474E+04	1.480E+04	1.471E+04
F11	8.353E+02	8.353E+02	8.464E+02	8.433E+02	8.545E+02	8.353E+02
F12	8.790E+02	8.790E+02	8.929E+02	8.885E+02	9.043E+02	8.790E+02
F13	9.803E+02	9.992E+02	9.859E+02	9.950E+02	9.983E+02	9.954E+02
F14	1.162E+04	1.162E+04	1.180E+04	1.174E+04	1.200E+04	1.162E+04
F15	1.160E+04	1.160E+04	1.179E+04	1.170E+04	1.193E+04	1.160E+04
F16	2.029E+02	2.030E+02	2.035E+02	2.038E+02	2.040E+02	2.027E+02
F17	1.396E+03	1.396E+03	1.409E+03	1.402E+03	1.424E+03	1.396E+03
F18	1.369E+03	1.370E+03	1.378E+03	1.395E+03	1.375E+03	1.372E+03
F19	1.964E+06	1.964E+06	1.965E+06	1.967E+06	1.966E+06	1.964E+06
F20	6.150E+02	6.150E+02	6.150E+02	6.150E+02	6.150E+02	6.150E+02
F21	3.439E+03	3.439E+03	3.443E+03	3.440E+03	3.447E+03	3.439E+03
F22	1.241E+04	1.241E+04	1.252E+04	1.249E+04	1.264E+04	1.241E+04
F23	1.254E+04	1.254E+04	1.268E+04	1.264E+04	1.274E+04	1.254E+04
F24	2.051E+03	2.051E+03	2.062E+03	2.059E+03	2.070E+03	2.051E+03
F25	1.635E+03	1.634E+03	1.638E+03	1.639E+03	1.640E+03	1.636E+03
F26	5.144E+03	5.144E+03	5.244E+03	5.233E+03	5.276E+03	5.144E+03
F27	4.699E+03	4.698E+03	4.713E+03	4.706E+03	4.719E+03	4.698E+03
F28	1.058E+04	1.056E+04	1.075E+04	1.076E+04	1.086E+04	1.054E+04
Average rank	2.0893	2.2143	4.3750	4.5179	5.5536	2.2500
Overall rank	1	2	4	5	6	3

$$fitness = c * error + (1 - c) * \frac{|se|}{|al|} \tag{8}$$

where c is a weighting factor that takes the value of 0.99 [35], $error$ is the classification error of cross-validation, $1 - error$ means the accuracy of classification, se denotes the selected features, and al represents all the features.

5.3. **Discussion of the numerical results.** We compared PBTA with other feature selection algorithms: the BPSO and the BGWO. The limit on the number of iterations was uniformly set to 50 and the size of the population to 30.

Table 7 shows the results of the comparison between PBTA and BPSO, BGWO, and shown in Figure 4 are the fitness convergence curves of the three algorithms. From Table 7, it can be seen that among the three algorithms, PBTA not only obtains the best

TABLE 6. The standard deviation of 30 experiments comparing PBTA with other classical algorithms.

Function	PBTA	BBA	BPSO	BGWO	BFMO	BGWOPSO
F1	5.920E-11	5.920E-11	4.932E+01	9.026E+01	4.848E+01	5.920E-11
F2	2.910E-06	2.910E-06	1.539E+07	2.162E+07	1.319E+07	2.910E-06
F3	5.119E+07	5.119E+07	1.357E+21	2.607E+21	2.114E+21	5.119E+07
F4	7.374E+01	5.499E+01	9.253E+01	1.692E+02	1.297E+02	1.085E+02
F5	0.000E+00	0.000E+00	4.441E-02	2.503E+00	6.952E-01	0.000E+00
F6	1.850E-11	1.850E-11	4.118E+01	5.116E+01	2.963E+01	1.850E-11
F7	5.285E+06	8.140E+06	4.725E+06	1.274E+07	7.186E+06	1.338E+07
F8	5.643E-02	5.424E-02	4.649E-02	5.164E-02	5.389E-02	6.877E-02
F9	2.141E-01	1.909E-01	1.519E-01	2.327E-01	1.728E-01	2.226E-01
F10	7.400E-12	7.400E-12	1.048E+01	1.441E+01	1.701E+01	7.400E-12
F11	0.000E+00	0.000E+00	3.113E+00	5.229E+00	3.499E+00	0.000E+00
F12	0.000E+00	0.000E+00	3.292E+00	7.714E+00	4.211E+00	0.000E+00
F13	1.033E+00	2.717E+01	4.009E+00	1.655E+01	6.500E+00	2.550E+01
F14	5.550E-12	5.550E-12	3.969E+01	7.994E+01	6.622E+01	5.550E-12
F15	2.696E+01	2.682E+01	3.955E+01	7.888E+01	7.873E+01	2.626E+01
F16	4.828E-01	8.426E-01	3.723E-01	8.554E-01	3.811E-01	7.249E-01
F17	0.000E+00	0.000E+00	2.417E+00	4.190E+00	4.629E+00	0.000E+00
F18	1.424E+01	1.312E+01	8.028E+00	1.386E+01	8.108E+00	1.538E+01
F19	9.472E-10	9.472E-10	3.167E+02	2.715E+03	7.103E+02	9.472E-10
F20	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
F21	1.388E-12	1.388E-12	9.477E-01	1.122E+00	1.756E+00	1.388E-12
F22	1.850E-12	1.850E-12	2.978E+01	3.421E+01	3.814E+01	1.850E-12
F23	2.580E+00	7.400E-12	3.581E+01	6.890E+01	2.209E+01	7.400E-12
F24	9.250E-13	9.250E-13	2.519E+00	4.957E+00	2.389E+00	9.250E-13
F25	8.532E-01	5.222E-01	9.203E-01	1.514E+00	1.010E+00	5.861E-01
F26	9.250E-13	9.250E-13	1.927E+01	6.366E+01	2.619E+01	9.250E-13
F27	1.787E+00	1.270E+00	3.011E+00	5.668E+00	4.720E+00	1.003E+00
F28	8.483E+01	6.830E+01	5.211E+01	1.227E+02	6.927E+01	8.595E+01

fitness value, but also has the smallest number of selected features. Nevertheless, the classification error of the PBTA is obviously not the most favorable, which is due to the fact that we take fitness as the main optimization objective, while fitness takes into account both the classification error and the number of features selected.

TABLE 7. Comparison between PBTA and other algorithms under NSL-KDD dataset.

Algorithm	Fitness	Feature number	Accuracy
PBTA	0.0093	53	0.9949
BPSO	0.0101	58	0.9946
BGWO	0.0101	76	0.9961

6. **Conclusions.** A parallel binary tumbleweed algorithm(PBTA) is proposed in this paper and we introduce a novel transfer function to improve the capability of the algorithm

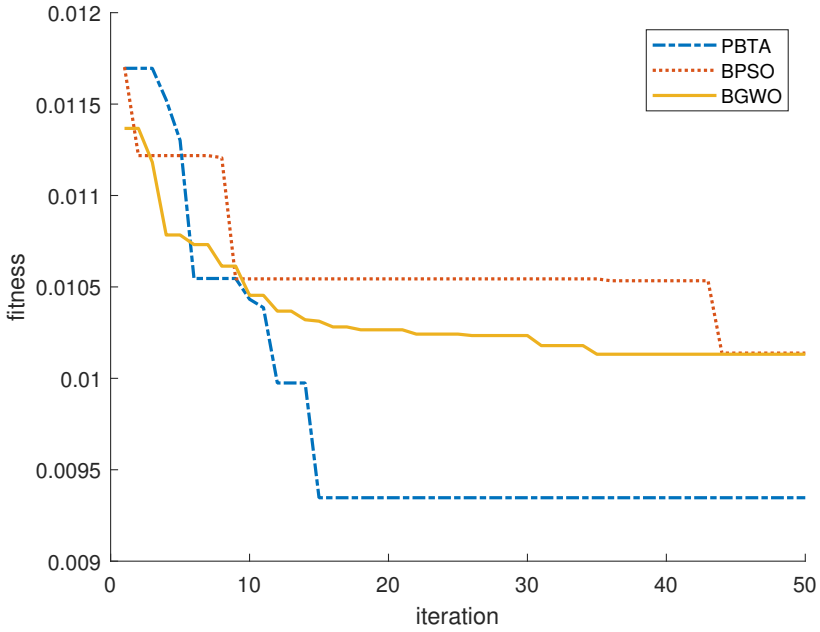

FIGURE 4. The convergence curve of the fitness value.

to avoid the local traps. Meanwhile, our proposed parallel communication strategies help the algorithm to enhance the population diversities and improve the efficiency of the algorithm. PBTA is evaluated against other well-known algorithms on CEC2013 benchmark, and the obtained data show the favorable performance of PBTA. Ultimately, PBTA was applied to address the feature selection problem for intrusion detection systems. The experimental results show that PBTA realizes the minimization of the selected features while considering the accuracy.

REFERENCES

[1] S. H. R. Pasandideh, S. T. A. Niaki, and A. Gharaei, "Optimization of a multiproduct economic production quantity problem with stochastic constraints using sequential quadratic programming," *Knowledge-Based Systems*, vol. 84, pp. 98–107, 2015.

[2] K. Hussain, M. N. Mohd Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artificial Intelligence Review*, vol. 52, pp. 2191–2233, 2019.

[3] A. Tzanetos and G. Dounias, "Nature inspired optimization algorithms or simply variations of metaheuristics?" *Artificial Intelligence Review*, vol. 54, pp. 1841–1862, 2021.

[4] K.-S. Tang, K.-F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 22–37, 1996.

[5] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994.

[6] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2010.

[7] K. R. Opara and J. Arabas, "Differential evolution: A survey of theoretical analyses," *Swarm and Evolutionary Computation*, vol. 44, pp. 546–558, 2019.

[8] Z. Hu, W. Gong, W. Pedrycz, and Y. Li, "Deep reinforcement learning assisted co-evolutionary differential evolution for constrained optimization," *Swarm and Evolutionary Computation*, vol. 83, p. 101387, 2023.

[9] M. Nazeri, A. Mollahosseini, and I. Izadi, "A centrality based genetic algorithm for the graph burning problem," *Applied Soft Computing*, vol. 144, p. 110493, 2023.

[10] L. Kang, R.-S. Chen, N. Xiong, Y.-C. Chen, Y.-X. Hu, and C.-M. Chen, "Selecting hyper-parameters of gaussian process regression based on non-inertial particle swarm optimization in internet of things," *IEEE Access*, vol. 7, pp. 59 504–59 513, 2019.
[11] C.-M. Chen, S. Lv, J. Ning, and J. M.-T. Wu, "A genetic algorithm for the waitable time-varying multi-depot green vehicle routing problem," *Symmetry*, vol. 15, no. 1, p. 124, 2023.
[12] A. L. H. P. Shaik, M. K. Manoharan, A. K. Pani, R. R. Avala, and C.-M. Chen, "Gaussian mutation–spider monkey optimization (gm-smo) model for remote sensing scene classification," *Remote Sensing*, vol. 14, no. 24, p. 6279, 2022.
[13] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.
[14] X. Zhou, W. Gui, A. A. Heidari, Z. Cai, G. Liang, and H. Chen, "Random following ant colony optimization: Continuous and binary variants for global optimization and feature selection," *Applied Soft Computing*, vol. 144, p. 110513, 2023.
[15] K.-L. Du, M. Swamy, K.-L. Du, and M. Swamy, "Particle swarm optimization," *Search and Optimization by Metaheuristics: Techniques and Algorithms Inspired by Nature*, pp. 153–173, 2016.
[16] Z. Zhang, C. Huang, K. Dong, and H. Huang, "Birds foraging search: a novel population-based algorithm for global optimization," *Memetic Computing*, vol. 11, pp. 221–250, 2019.
[17] E. Varol Altay and B. Alatas, "Bird swarm algorithms with chaotic mapping," *Artificial Intelligence Review*, vol. 53, pp. 1373–1414, 2020.
[18] A. B. Serapião, G. S. Corrêa, F. B. Gonçalves, and V. O. Carvalho, "Combining k-means and k-harmonic with fish school search algorithm for data clustering task on graphics processing units," *Applied Soft Computing*, vol. 41, pp. 290–304, 2016.
[19] Y. Xing, X. Wang, and Q. Shen, "Test case prioritization based on artificial fish school algorithm," *Computer Communications*, vol. 180, pp. 295–302, 2021.
[20] T.-Y. Wu, H. Li, and S.-C. Chu, "Cppe: An improved phasmatodea population evolution algorithm with chaotic maps," *Mathematics*, vol. 11, no. 9, p. 1977, 2023.
[21] G. Dhiman and V. Kumar, "Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems," *Knowledge-Based Systems*, vol. 165, pp. 169–196, 2019.
[22] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, p. 113338, 2020.
[23] J.-S. Pan, L.-G. Zhang, R.-B. Wang, V. Snášel, and S.-C. Chu, "Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems," *Mathematics and Computers in Simulation*, vol. 202, pp. 343–373, 2022.
[24] Q.-Y. Yang, S.-C. Chu, A. Liang, and J.-S. Pan, "Tumbleweed algorithm and its application for solving location problem of logistics distribution center," in *Genetic and Evolutionary Computing: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computing, October 21-23, 2021, Jilin, China 14*. Springer, 2022, pp. 641–652.
[25] X. Yuan, J.-S. Pan, S.-C. Chu, and V. Snášel, "Binary tumbleweed algorithm for application of feature selection," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. Springer, 2022, pp. 13–20.
[26] T.-Y. Wu, A. Shao, and J.-S. Pan, "Ctoa: Toward a chaotic-based tumbleweed optimization algorithm," *Mathematics*, vol. 11, no. 10, p. 2339, 2023.
[27] S. Mirjalili and A. Lewis, "S-shaped versus v-shaped transfer functions for binary particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 9, pp. 1–14, 2013.
[28] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 5. IEEE, 1997, pp. 4104–4108.
[29] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications*, vol. 25, pp. 663–681, 2014.
[30] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, 2016.
[31] J.-S. Pan, P. Hu, and S.-C. Chu, "Binary fish migration optimization for solving unit commitment," *Energy*, vol. 226, p. 120329, 2021.
[32] S. Sundaramurthy and P. Jayavel, "A hybrid grey wolf optimization and particle swarm optimization with c4. 5 approach for prediction of rheumatoid arthritis," *Applied Soft Computing*, vol. 94, p. 106500, 2020.

[33] M. Qaraad, S. Amjad, N. K. Hussein, S. Mirjalili, and M. A. Elhosseini, "An innovative time-varying particle swarm-based salp algorithm for intrusion detection system and large-scale global optimization problems," *Artificial Intelligence Review*, vol. 56, no. 8, pp. 8325–8392, 2023.

[34] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. IEEE, 2009, pp. 1–6.

[35] P. Hu, J.-S. Pan, and S.-C. Chu, "Improved binary grey wolf optimizer and its application for feature selection," *Knowledge-Based Systems*, vol. 195, p. 105746, 2020.