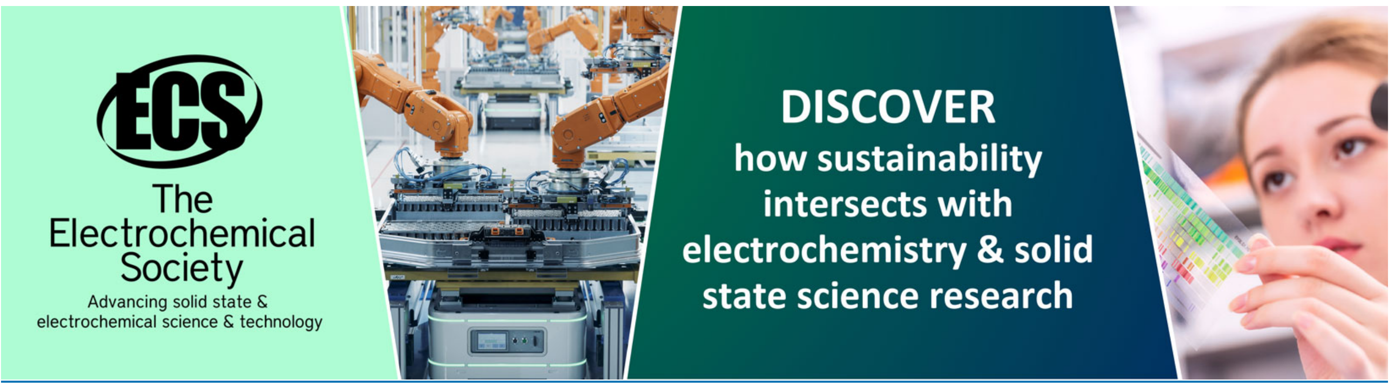
Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

Research on the hyper-heuristic of Sub-domain Elimination Strategies based on Firefly Algorithm

To cite this article: Mingquan Sun *et al* 2021 *J. Phys.: Conf. Ser.* **1966** 012024



Research on the hyper-heuristic of Sub-domain Elimination Strategies based on Firefly Algorithm

Mingquan Sun [1, a], Bangsheng Xing [1, b*], Daolong Yang [1, c].

[1]School of Mechatronic Engineering, Jiangsu Normal University, Xuzhou, Jiangsu, 221116, China

[a]email: sunmingquanc@qq.com, [c]email: yangdl@jsnu.edu.cn

[*b] Corresponding author's e-mail: 6019990415@jsnu.edu.cn

Abstract. In this study, a hyper-heuristic named Sub-domain Elimination Strategies based on Firefly Algorithm (SESFA) is proposed. First, a typical hyper-heuristic is usually using the high-level strategy selection or the combination of the low-level heuristics to obtain a new hyper-heuristic, each round of optimization process is carried out in the whole problem domain. However, SESFA evaluates the problem domain through the feedback information of the meta-heuristic at the lower level, eliminating the poor performance areas, and adjusting the underlying heuristic or adjusting the algorithm parameters to improve the overall optimization performance. Second, the problem domain segmentation function in SESFA can reduce the complexity of the objective function within a single sub-domain, which is conducive to improving the optimization efficiency of the underlying heuristic. Further, the problem domain segmentation function in SESFA also makes there is no direct correlation between different sub-domains, so different underlying heuristics can be adopted in different sub-domains, which is beneficial to the realization of parallel computing. Comparing SESFA with Firefly Algorithms with five standard test functions, the results show that SESFA has advantages in precision, stability and success rate.

1. Introduction

With the development of computer technology, a large number of meta-heuristics based on swarm intelligence simulating natural characteristics have been proposed successively in the field of optimization algorithm. Although this kind of algorithms is widely used to solve kinds of optimization problems, a single algorithm is often designed to solve a certain kind of problems. In general, when the algorithm is applied to different cases, or even different types of problem in the same case, the solution performance may vary greatly[1]. According to the No Free Lunch (NFL) theory[2], the single meta-heuristic is still insufficient in the application research of generality[3]. In addition, meta-heuristics contain complex random behaviors and are difficult to have a general framework, which makes the algorithms have some limitations in mathematical analysis such as complexity, convergence and computing power. The application of meta-heuristics requires users to have professional knowledge and algorithm skills[4]. All the above factors restrict the extension and application of meta-heuristics.

Based on these, the concept of hyper-heuristic is proposed[5]. A typical feature of this algorithm is the logical separation between the heuristic methodology and the problem domain at the High-Level Strategies (HLS), as shown in 1. In other words, hyper-heuristics focus on shielding applied domain knowledge from high-level methodology. The HLS provides a logical framework to manage one or more heuristics as its Lower-Level Heuristic (LLH) and obtain new heuristics by means of dynamic parameter control or selection and combination. This HLS makes the hyper-heuristics no longer restricted by the NFL theory[6-9] and has higher universality. When the problem domain is changed, the original HLS can be applied to the new problem domain only by changing the corresponding LLH, problem description and evaluation function in the new problem domain.

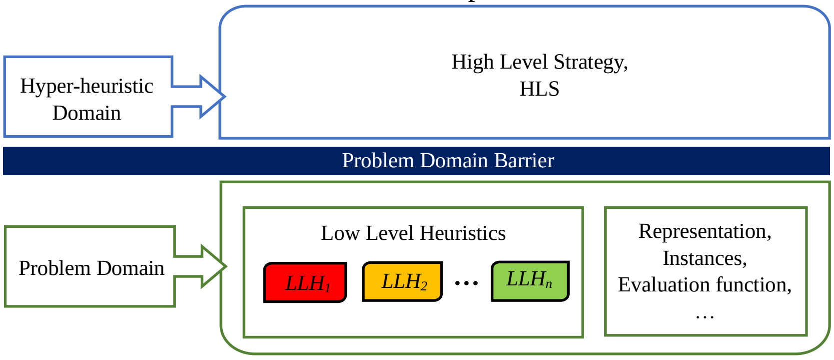


Figure 1. Typical hyper-heuristic framework

Since it was proposed, the Hyper-Heuristic Algorithm has been applied in the combinatorial optimization field, such as timetabling problems[10-13], scheduling problems[14-17], bin-packing problems[18-19] etc.

In order to overcome the local optimal solution and the poor stability of the general meta-heuristics in the large search domain, this paper proposes a hyper-heuristic which integrates problem domain partition, allocation, evaluation and elimination with multi-point search algorithm: Sub-domain Elimination Strategies based on Firefly Algorithm, SESFA.

2. Material and Methods

2.1. Low-level heuristic

In 2008, Xin-She Yang proposed the Firefly Algorithm(FA)[20], which originates from the simplification and simulation of firefly group behavior and is a high-level heuristic. The basic principle of Firefly Algorithm is: regard all points in space as fireflies, and utilize the characteristics of fireflies with high luminescence intensity to attract fireflies with low luminescence intensity. In the process of moving from the dimmer firefly to the brighter firefly, the position iteration is completed to find the brightest position, that is, the optimization process is completed.

2.2. The mathematical description of the optimization problem

The essence of optimal design for a problem is to solve the maximum/minimum value of the function describing the problem in its problem domain. After defining the design variables, constraints and objective functions, the mathematical model of the optimization problem can be expressed as:

Solve for $\boldsymbol{x}$ to satisfy

$$\min_{x \in R} f(\vec{x}) \tag{1}$$

In the above equation, R is the feasible domain of the optimization problem, also known as the problem domain.

For the objective function $f(\boldsymbol{x})$ defined on the d dimensional space, the design variable $\boldsymbol{x}$ can be expressed as:

$$\begin{aligned} & x \in (x_1, x_2, \ldots, x_d) \\ & Lb_m \le x_m \le Ub_m \\ & m = 1, 2, \ldots, d \end{aligned} \tag{2}$$

Where x_m is the component of $\boldsymbol{x}$ in m dimension, the upper and lower bounds it can take are Ub_m and

Lb_m respectively.

In order to reduce the complexity of the problem and facilitate the identification of sub-domains, the problem domain is divided into k segments of equal length in each dimension, so that a total of k^d sub-domains with the same dimension, shape and size can be generated, and the length of the segments divided in the m dimension is denoted as L_m. Then, each sub-domain is coded and identified. The coding rules adopt the segment number s_m of corresponding problem domain dimension counting from the lower boundary of the problem domain, and take these sequence numbers as the elements of vectors to form the identity vector ***sub*** of each sub-domain according to the order of dimension. The count of the segment number starts with "1", that is, the first segment starting from the lower bound of the search is 1. Thus, one identity vector of a sub-domain can be expressed as:

$$\begin{aligned} & sub(s_1, s_2, ..., s_d) \\ & s = 1, 2, ..., k \\ & m = 1, 2, ..., d \end{aligned} \tag{3}$$

In the above equation, s_m is the serial number of the m dimensional segment in the problem domain.

Then the value range of the sub-domains in m dimension is:

$$\left[(s_m - 1)\cdot L_m + Lb_m, s_m \cdot L_m + Lb_m\right] \quad m = 1, 2, ..., d \tag{4}$$

As each sub-domain is an independent search unit, design variables should always be limited within the scope of the sub-domain during the search of the LLH. Therefore, when FA search in the sub-domain, firefly j, attracted by firefly i, moves towards it and updates its position. Its position updating formula should be adjusted from the original

$$\vec{x}_j(t+1) = \vec{x}_j(t) + \vec{\Delta}_j(t) \tag{5}$$

to:

$$\vec{x}_j(t+1) = \begin{cases} (s-1)\cdot L + Lb, \ \vec{x}_j(t) + \vec{\Delta}_j(t) < (s-1)\cdot L + Lb \\ \vec{x}_j(t) + \vec{\Delta}_j(t), \quad (s-1)\cdot L + Lb \le \vec{x}_j(t) + \vec{\Delta}_j(t) \le s\cdot L + Lb \\ s\cdot L + Lb, \qquad \vec{x}_j(t) + \vec{\Delta}_j(t) > s\cdot L + Lb \end{cases} \tag{6}$$

In the above formula,

$$\vec{\Delta}_j(t) = \beta_{ij}(r_{ij})\cdot\left(\vec{x}_i(t) - \vec{x}_j(t)\right) + \alpha\vec{\varepsilon}_j \tag{7}$$

Among them,

$$\beta_{ij}(r_{ij}) = \beta_0 \cdot e^{-\gamma r_{ij}^2} \tag{8}$$

Represents the attraction of firefly i relative to j[10],

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{m=1}^{d}(x_{i,m} - x_{j,m})^2} \tag{9}$$

Represents the Descartes distance between firefly i and j[10].

2.3. The basic steps of High-level strategy

Based on the general description above of the optimization problem, the basic steps for SESFA are given below:

(a) Set the number n of initial fireflies in the sub-domain and the maximum number *maxGen* of iterations;

(b) Initialize the positions of n fireflies randomly in the sub-domain:

$$u0_{j,m}^{sub(s_1,s_2,...,s_d)}(t) = (s_m - 1)\cdot L_m + L_m \cdot \text{rand} + Lb_m \tag{10}$$

The above formula represents the position component of firefly j in m dimension when ***sub***($s_1, s_2, ..., s_d$) is in the t generation.

(c) Call Firefly Algorithm in LLH to calculate the optimal value of ***sub***($s_1, s_2, ..., s_d$):

$$f_{\min}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)} = \min_{1,2,\ldots,n} f\left(u0_j^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}(t)\right) \tag{11}$$

(d) Evaluate according to the optimal feedback results of each sub-domain:

$$sortedf = \text{sort}\left(f_{\min}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}\right) \tag{12}$$

(e) Record the best values in the current ranking:

$$f_{tempbest} = sortedf\,(\text{last}) \tag{13}$$

(f) The subdomain corresponding to the poorly ranked optimal solution is taken as the inferior solution space, that is, the subdomain which is unlikely to have the global optimal solution is eliminated.

(g) When the termination condition is satisfied (For example, the number of remaining sub-domains is less than or equal to the set value), the loop is stopped.

(h) Otherwise, adjust the parameters of the FA in LLH, repeat steps (b) to (g), the sub-domains that have not been eliminated are evaluated and eliminated in the next round;

(i) Update the parameters of FA in LLH, perform further search on the remaining problem domains, repeat steps (b) and (c):

$$u0_{j,m}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}(t) = (s_m - 1)\cdot L_m + L_m \cdot \text{rand} + Lb_m \tag{14}$$

$$f_{\min}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)} = \min_{1,2,\ldots,z} f\left(u0_j^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}(t)\right) \tag{15}$$

(j) In order to ensure the quality of the solution, the optimal value $f_{best\min}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}$ obtained is compared with the historical optimal value $f_{tempbest}$ recorded in, and the Minimum value is taken as the global optimal value $f_{bestoverall}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}$, and the design variable $\vec{x}_{bestoverall}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}$ in the sub-domain corresponding to this optimal value is also obtained:

$$\vec{x}_{bestoverall}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)} = arc\left(f_{bestoverall}^{\overrightarrow{sub}(s_1,s_2,\ldots,s_d)}\right) \tag{16}$$

2.4. Flowchart of SESFA

SESFA requires three major phases of initialization, computation, and evaluation. The basic steps of SESFA are shown in Figure 2.

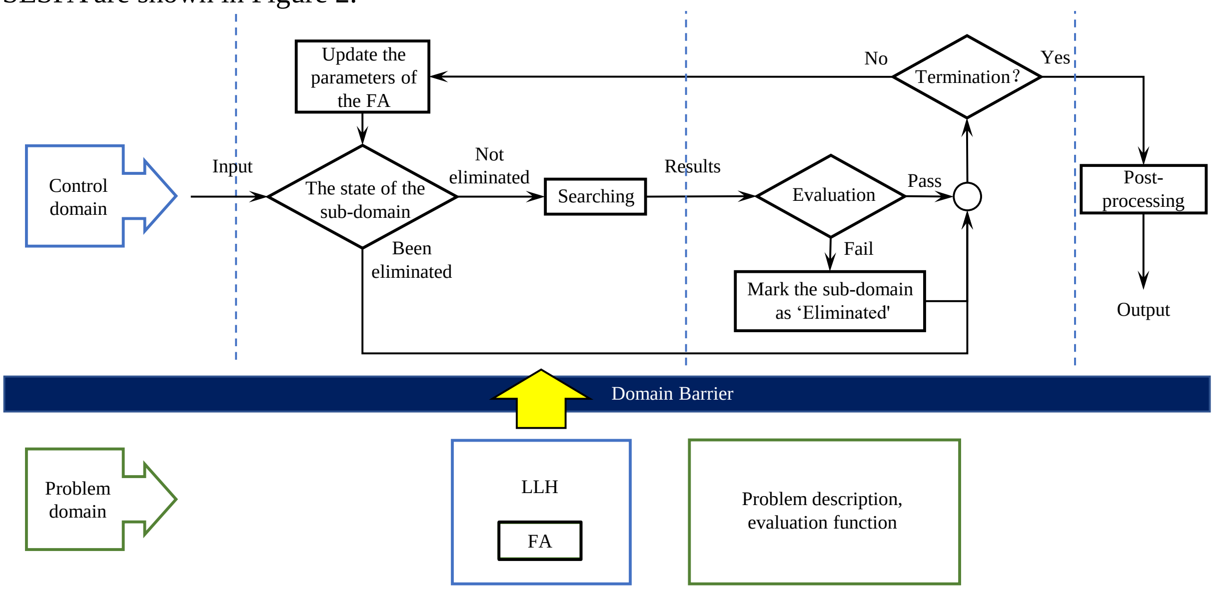
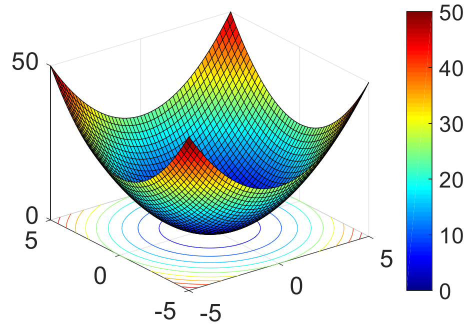


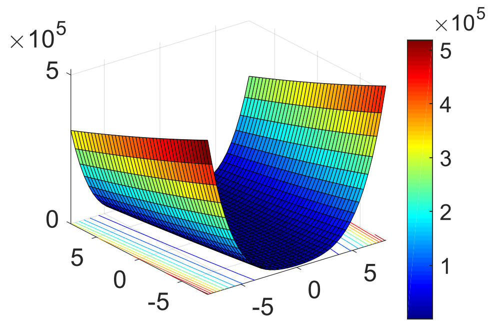
Figure 2. Flowchart of SESFA

2.5. Comparative experiments
In order to evaluate the performance of the SESFA, the comparison group performed optimization experiments on several typical test functions using two kinds of FA with different parameters.

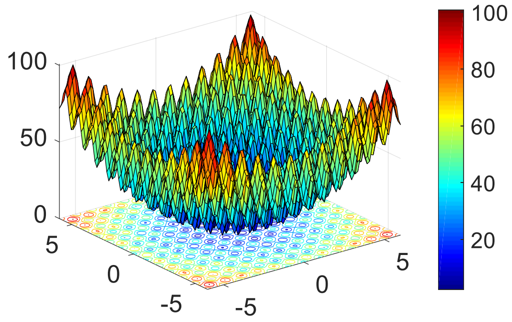
Five test functions, Sphere, Rosenbrock, Rastrigin, Ackley and Griewank, were selected from the test functions[21]. The three-dimensional diagram of the test functions is shown in Figure 3. In order to comprehensively evaluate the effect of the size of the problem domain on the algorithm performance, the search domains were divided into [-5.12, 5.12] and [-100, 100], the optimization performance of SESFA and two different parameters of Firefly Algorithms were compared.



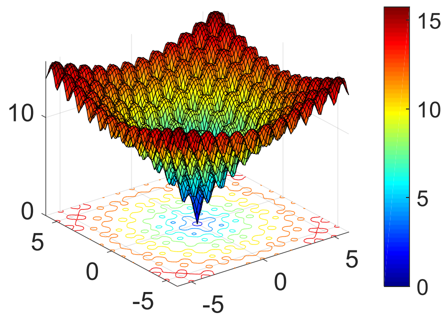
(a). Sphere



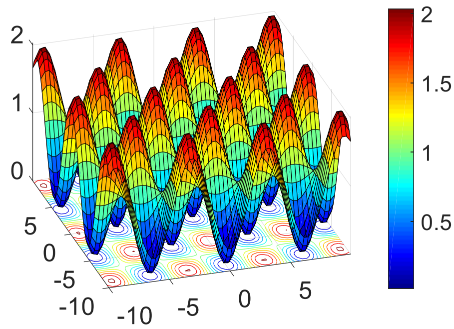
(b). Rosenbrock



(c). Rastrigin



(d). Ackley



(e). Griewank

Figure 3. The three-dimensional diagram of the test functions

In order to avoid errors caused by contingency, each test function is run independently for 30 times. The optimal value, the worst value, the mean value and the standard deviation of the global minimum value of the objective function are counted and the success rate is calculated. The best experimental data for each evaluation indicator are marked in bold.

In addition, in order to effectively distinguish the Firefly Algorithms with different parameters and simplify the algorithm names, the Firefly Algorithms with two different parameters are denoted as FA_1 and FA_2 respectively.

In the Firefly Algorithm, the random term coefficient α, the original attraction β_0, the minimum attraction β_{min} and the light absorption intensity coefficient γ are all set as constant numbers in the algorithm, which need not be set separately. The number of fireflies *n* and the maximum number of iterations *maxGen* were set according to the experimental requirements. The specific parameter setting values are shown in Table 1.

Table 1. The initial value of each algorithm parameter in the experiment

The algorithm name	Number of fireflies	Maximum iteration number	Random term coefficient	Original attraction	Minimum attraction	Light absorption intensity coefficient	The dimension of the problem domain
	n	$maxGen$	α	β_0	β_{min}	γ	d
SESFA	15	2000	0.5	1.0	0.2	1.0	5
FA_1	15	2000	0.5	1.0	0.2	1.0	5
FA_2	30	2000	0.5	1.0	0.2	1.0	5

3. Results and Discussion

3.1. The results of the SESFA, FA_1 and FA_2 in search domain [-5.12,5.12]

Table 2. The results of the SESFA, FA_1 and FA_2 in the search domain [-5.12,5.12]

Test functions	Algorithm name	Evaluation indicators				
		Best value	Worst value	Mean value	Std deviation	Success rate
Sphere	SESFA	**7.24640E-11**	**2.54137E-09**	**1.24157E-09**	**5.99509E-10**	**100.00%**
	FA_1	4.35586E-09	5.44567E-08	2.72709E-08	1.19603E-08	**100.00%**
	FA_2	3.09644E-09	5.66697E-08	2.34110E-08	1.13522E-08	**100.00%**
Rosenbrock	SESFA	5.28572E-03	**1.08063E-01**	**4.93701E-02**	**2.19928E-02**	**100.00%**
	FA_1	3.30919E-02	5.43600E+00	2.99860E-01	9.72670E-01	93.33%
	FA_2	**2.98785E-03**	1.60143E-01	8.30204E-02	3.74109E-02	**100.00%**
Rastrigin	SESFA	**1.10320E-07**	**1.98992E+00**	**4.97480E-01**	**5.59895E-01**	**53.33%**
	FA_1	5.82811E-06	5.96975E+00	2.48740E+00	1.51439E+00	3.33%
	FA_2	2.61151E-06	3.97984E+00	1.82410E+00	9.97719E-01	6.67%
Ackley	SESFA	**1.81012E-05**	**9.16514E-05**	**6.55053E-05**	**1.76158E-05**	**100.00%**
	FA_1	1.38741E-04	4.21858E-04	2.61545E-04	7.14466E-05	**100.00%**
	FA_2	1.20786E-04	3.65389E-04	2.56152E-04	5.25297E-05	**100.00%**
Griewank	SESFA	**8.72381E-11**	**4.66060E-10**	**2.95598E-10**	**1.06922E-10**	**100.00%**
	FA_1	1.27998E-09	1.99257E-01	1.85441E-02	4.58350E-02	46.67%
	FA_2	1.58728E-09	2.86502E-02	3.52250E-03	8.40488E-03	80.00%

It can be seen from Table 2 that, in small search range, the three algorithms all have a 100% optimization success rate for Sphere function, indicating that the three algorithms all have a good convergence ability. However, all evaluation indicators of SESFA are superior to the comparison algorithms, reflecting the critical role of the evaluation and elimination strategy in the optimization process of the HLS.

For Rosenbrock function, among the optimization results of the three algorithms, FA_2 achieved the best optimal value and 100% optimization success rate, that's because Rosenbrock is a non-convex sick function, but it's still a unimodal function. In the Firefly Algorithm, any two fireflies can communicate with each other in the optimization process. Obviously, the Firefly Algorithm with higher population density will have a higher probability to find the global optimal value. The SESFA performs best on all assessment measures other than its optimal value, and its optimal value is not significantly different from the optimal value of FA_2, it shows that SESFA inherits the excellent performance of FA, thanks to the coordination of HLS, the stability and success rate of optimization are higher than that of single Firefly Algorithm with the same population density.

For the Rastrigin function, SESFA performed best on all measures of evaluation, with a much higher success rate than the other two algorithms, this is due to the sub-domain division strategy, which can effectively avoid falling into the local optimal value, thus having a higher success rate of searching for the global optimal value. It should be noted that FA_2 presents a higher success rate of searching than FA_1, because the individual density of fireflies in FA_2 is twice as high as that in FA_1. This indicates that increasing population density in multi-peak function optimization will improve the probability of finding the global optimal value.

For Ackley function, the three algorithms also have 100% optimization success rate, because although Ackley is a multi-peak function, there is a large gap between its global optimal value and its suboptimal value, In Figure 3(d), a sharp peak (or trough) appears near the global optimal point. Due to the information exchange among fireflies in the FA, it can effectively avoid falling into the local optimal value in the search process, so that the fireflies can stably focus on the global optimal value within a limited number of iterations. The SESFA results are superior to those of the other two algorithms, reflecting the improved precision and stability of SESFA in addition to the excellent performance of the Firefly Algorithm.

For Griewank function, it can be seen from Figure 3(e) that there are a large number of suboptimal values near the global optimal value of this function, which are very close to the global optimal value. These suboptimal values will cause great interference to algorithm optimization, causing algorithms to easily fall into the local optimal value. The SESFA performs optimally across all the evaluation metrics and is significantly superior to the other two algorithms in terms of stability, which benefited from the problem domain division strategy, which ensures good stability and success rate by evaluating the sub-domain and eliminating the inferior region and gradually limiting the search region to the sub-domain where the global optimal value is located.

3.2. The results of the SESFA, FA_1 and FA_2 in search domain [-100,100]

Table 3. The results of the SESFA, FA_1 and FA_2 in the search domain [-100,100]

Test functions	Algorithm name	Evaluation indicators				
		Best value	Worst value	Mean value	Std deviation	Success rate
Sphere	SESFA	**4.23535E-08**	**3.32320E-07**	**1.69712E-07**	**7.17763E-08**	**100.00%**
	FA_1	2.01298E-06	1.99438E-05	1.05121E-05	4.29718E-06	**100.00%**
	FA_2	2.10427E-06	1.71744E-05	9.49358E-06	4.17781E-06	**100.00%**
Rosenbrock	SESFA	**1.52446E-02**	**9.69293E-02**	**6.08760E-02**	**2.23178E-02**	**100.00%**
	FA_1	9.68933E-02	3.09327E+02	2.32269E+01	5.61866E+01	43.33%
	FA_2	6.65758E-02	3.54057E+02	3.35239E+01	7.97018E+01	53.33%
Rastrigin	SESFA	**2.87264E-05**	**2.98490E+00**	**1.02819E+00**	**7.48234E-01**	**23.33%**
	FA_1	9.96109E-01	5.97256E+00	3.05303E+00	1.40539E+00	0.00%
	FA_2	2.13706E-03	4.97566E+00	2.15793E+00	1.20697E+00	3.33%
Ackley	SESFA	**3.95086E-04**	**1.28464E-03**	**7.16719E-04**	**1.76665E-04**	**100.00%**
	FA_1	2.44638E-03	2.00001E+01	4.00432E+00	7.99780E+00	46.67%
	FA_2	1.90636E-03	2.00000E+01	6.71822E-01	3.58915E+00	96.67%
Griewank	SESFA	**6.44620E-08**	**2.71008E-02**	**1.12535E-02**	**5.67769E-03**	**6.67%**
	FA_1	3.68262E-06	2.95604E-01	5.69959E-02	5.67685E-02	3.33%
	FA_2	1.61157E-06	1.01027E-01	3.30136E-02	2.02452E-02	3.33%

As shown in Table 3, for the larger problem domain, The SESFA delivers the best results for all the evaluation metrics across the five test functions.

For unimodal functions like Sphere, all three algorithms still have a 100% success rate, this is because in the Firefly Algorithm, each individual has the ability to communicate with other individuals. As long as one firefly searches for the optimal value, it will attract other fireflies through information exchange and then search for the global optimal value. SESFA's evaluation and elimination strategy does not have a significant advantage over a single meta-heuristic algorithm in the optimization of such a simple unimodal function, but it does have a slight advantage over the single meta-heuristic algorithm in terms of solution accuracy and stability.

For Rosenbrock function, it belongs to the extremely difficult non-convex ill-condition function with minimum value due to the interaction between variables in the Rosenbrock function. However, the SESFA achieves a 100% success rate, which is much higher than the comparison group algorithms. This is because the evaluation and elimination strategy can continuously narrow the scope of problem domain through evaluation in the optimization process, and then effectively improve the success rate of the underlying heuristic to find the global optimal value. Furthermore, SESFA's HLS also plays a key role in solving stability, which is evident in the data under the standard deviation index of Rosenbrock function in Table 3.

For Rastrigin function, both algorithms in the comparison group had a very low success rate, with FA_1 having a 0% success rate in 30 independent trials and SESFA having a 23.33% optimization success rate, this is due to the fact that the problem domain partition strategy can decompose the complex multi-peak function into several relatively simple multi-peak functions, which is more conducive to the optimization performance of the underlying heuristic. Thanks to the evaluation and elimination strategy of HLS in SESFA, the algorithm achieves the best global optimal value, which is significantly better than the corresponding index of the comparison algorithms, reflecting the excellent global optimization precision of the algorithm.

For Ackley function, it can be seen that in the statistics of 30 independent trials, the worst value of SESFA solution is much better than that of the comparison algorithms under the premise that the optimal values solved by the three algorithms are not significantly different, making SESFA solution much more stable than that of the comparison algorithms. The SESFA's problem domain evaluation and elimination strategy can effectively improve the algorithm's stability in the optimization of larger problem domain. In terms of convergence, the population density of LLH's Firefly Algorithm in SESFA is only half that of FA_2, but the success rate of SESFA is slightly higher than FA_2. This advantage stems from the evaluation and elimination strategy of the HLS of SESFA mentioned in the previous paragraph, which takes full advantage of the communication capabilities of FA between fireflies, enabling SESFA to converge to the global optimal value in 30 separate trials.

For Griewank function, the optimization success rate of the three algorithms is low. One reason is that there are a large number of suboptimal values extremely close to the global optimal value around the function, which easily makes the algorithms fall into the local optimal value, leading to the failure of optimization. Another reason is that with the expansion of the search range, in order to accurately find the global optimal value, the number of fireflies or the maximum number of iterations should be increased, so as to fully search the whole problem domain. In addition, SESFA benefits from the HLS's evaluation and elimination strategy for the problem domain, enabling SESFA to continuously narrow the algorithm's search scope and take full advantage of the optimization capabilities of the Firefly Algorithm, resulting in higher solution stability compared to the comparison group algorithms.

3.3. Overall performance

Overview of the data in Table 2 and Table 3, with the expansion of the problem domain, the optimization performance of the three algorithms all reduced to varying degrees. However, SESFA has the lowest level of decline, which means that SESFA performs best in large problem domain searches, especially in search success rates, as shown in Figure 4.

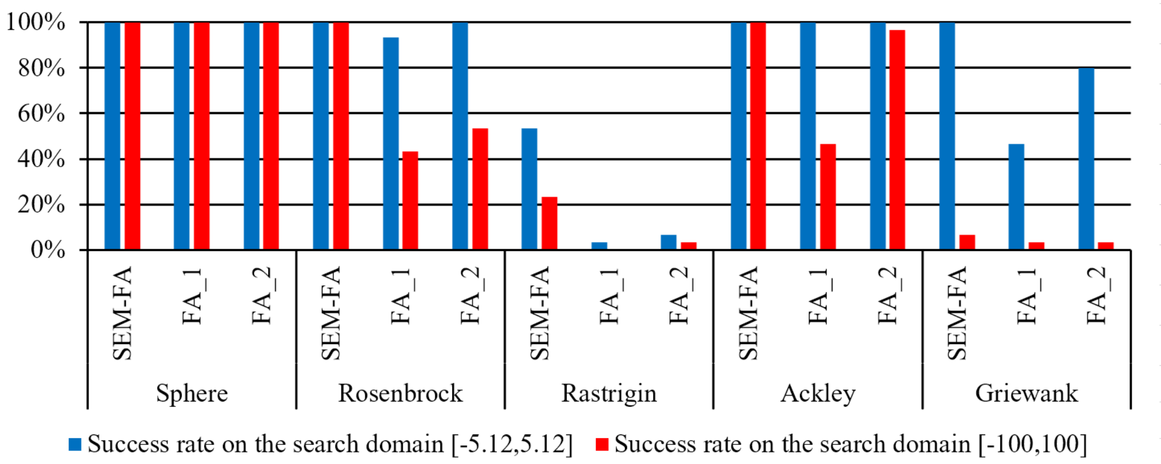


Figure 4. The success rate of algorithm optimization in different problem domains

3.4 Limitations

Although the SESFA performs better than the comparison algorithms in terms of performance. However, it should be pointed out that, due to the existence of sub-domain division strategy, with the increase of solving domain dimension, the number of sub-domains after division will increase exponentially, and this will bring huge computation and unacceptable solution time, namely the dimensional disaster mentioned in literature[22]. Therefore, the SESFA is not suitable for solving high-dimensional data at present, which limits the application scope of the algorithm. In the next step, Non-negative Matrix Factorization (NMF)[23] will be taken as the next research direction. It is hoped that the algorithm can shorten the solving time and obtain a wider application range without significantly reducing the solving precision.

4. Conclusions

Analysis of the simulation results shows that SESFA has advantages in search precision, search stability, and solution success over a small search range. Compared with the other two algorithms, the advantages are more obvious in large search range. This indicates that SESFA is an effective optimization algorithm for complex functions with large search space and multiple peaks.

Acknowledgments

Xuzhou Science and Technology Plan Project (Ref.: KC20202).

Postgraduate Research and Practice Innovation Program of Jiangsu Normal University (Ref.: 2019XKT164).

References

[1] Serpell M. C., Smith J. E. (2010) Self-adaptation of mutation operator and probability for permutation representations in genetic algorithms. Evolutionary Computation, 18(3): 491-514.
[2] Wolpert D. H., Macready W. G. (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1): 67-82.
[3] Wang L., Zheng D. Z. (2000) Progress in Meta-Heuristic algorithms. Control and Decision, 15(3): 257-262. Chinese.
[4] Xie Y., Hou Y. E., Chen X. P., et al. (2017) Review of research progress of hyper-heuristic algorithms. Computer Engineering and Applications, 53(14): 1-8.
[5] Cowling P., Kendall G., Soubeiga E. (2000) A hyperheuristic approach to scheduling a sales summit. In: Proceedings of the International Conference on the Practice and Theory of Automated Timetabling. pp. 176-190.

[6] Poli R., Graff M. (2009) There is a free lunch for hyper-heuristics. In: Genetic Programming. pp. 195-207.
[7] Marshall J. A., Hinton T. G. (2010) Beyond no free lunch: realistic algorithm for arbitrary problem classes. In: WCCI 2010 IEEE World Congress on Computational Intelligence, Barcelona, Spain, July, 18-23: 1319-1324.
[8] Schumacher C., Vose M., Whitley D. (2001) The no free lunch and problem description length. In: Genetic and Evolutionary Computation Conference, GECCO, 565-570.
[9] Spall J. C., Hill S. D., Stark D. R. (2006) Theoretical framework for comparing several stochastic optimization algorithms. In: Probabilistic and Randomized Methods for Design Under Uncertainty, Springer, London, 99-117.
[10] Ahmadi S., Barone R., Cheng P., et al. (2003) Perturbation based variable neighborhood search in heuristic space for examination timetabling problem. In: Proceedings of Multi-disciplinary International Scheduling: Theory and Applications (MISTA 2003), pp. 13-16.
[11] Burke E., Dror M., Petrovic S., et al. (2005) Hybrid graph heuristics within a hyper- heuristic approach to exam timetabling problems. Operations Research/Computer Science Interfaces, 29: 79-91.
[12] Pillay N., Banzhaf W. (2009) A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. European Journal of Operational Research, 197(2): 482-491.
[13] Sabar N. R., Ayob M., Qu R., et al. (2011) A graph coloring constructive hyper heuristic for examination timetabling problems. Applied Intelligence, 37(1): 1-11.
[14] Dimopoulos C., Zalzala A. M. (2001) Investigating the use of genetic programming for a classic one-machine scheduling problem. Advances in Engineering Software, 32(6): 489-498.
[15] Cowling P., Kendall G., Han L. (2002) An investigation of a hyper-heuristic genetic algorithm applied to a trainer scheduling problem. In: Proceedings of the Evolutionary Computation, pp. 1185-1190.
[16] Ho N. B., Tay J. (2005) Evolving dispatching rules for solving the flexible job-shop problem. In: Proceeding of the IEEE Congress on Evolutionary Computation, pp. 2848-2855.
[17] Vazquez-Rodriguez J., Petrovic S. (2010) A new dispatching rule based genetic algorithm for the multi-objective job shop problem. Journal of Heuristics, 16(6): 771-793.
[18] Ross P., Schulenburg S., Marín-Blázquez J. G., et al. (2002) Hyper-heuristics: learning to combine simple heuristics in bin-packing problems. In: Proceedings of the GECCO, pp. 942-948.
[19] Cuesta-Canada A., Garrido L., Terashima-Marin H. (2005) Building hyper-heuristics through ant colony optimization for the 2D bin packing problem. In: Knowledge-based Intelligent Information and Engineering Systems, pp. 907-907.
[20] Yang X. S. (2008) Nature-Inspired Metaheuristic Algorithms. Luniver Press Publishing, Frome.
[21] Bao L., Zeng J. C. (2009) Comparison and analysis of the selection mechanism in the artificial bee colony algorithm. In: 9th International Conference on Hybrid Intelligent Systems. Shenyang China: IEEE Computer Society. pp. 411-416.
[22] Duda R. O., Hart P. E., Stork D. G. (2001) Pattern Classification. 2nd ed. Wiley Publishing, New York.
[23] Lee D. D., Seung H. S. (1999) Learning the parts of objects with nonnegative matrix factorization. Nature, 401: 788-791.