Virtual Simulation of Integrated Circuits Combining AP with DE Algorithm

Peipei Hu

Abstract—**As computers develops, virtual simulation technology becomes an important means of integrated circuit design. Therefore, based on the demand for virtual simulation of integrated circuits, a simulation method combining affinity propagation and differential evolution algorithm was proposed. By applying the affinity propagation to circuit fault diagnosis and combining it with differential evolution algorithm, circuit parameters optimization was carried out. These experiments confirm that the fusion of affinity propagation and differential evolution algorithm has a precision of 94.26%, recall of 93.41%, mean F1 of 88.59%, convergence speed of 56.77 seconds, and stability of 93.17%. The affinity propagation performs well in clustering. Especially without pre-defining the classes, it can identify the position and number of class centers automatically. The simulation of integrating affinity propagation and differential evolution algorithm has broad application prospects in virtual simulation of integrated circuits. It can improve simulation effectiveness and performance, providing effective support for circuit design and testing.**

Index Terms—**AP algorithm, differential evolution algorithm, integrated circuits, virtual simulation.**

Original Research Paper
DOI: 10.53314/ELS2529043H

I. Introduction

As technology advances, integrated circuit design faces complex challenges. While meeting performance, power consumption, and area requirements, it is also necessary to consider various issues that may arise during the manufacturing process. Virtual simulation, as an important part of integrated circuit design, can effectively predict and solve these problems [1]. Integrated circuit virtual simulation is an important method for circuit design and testing, which can improve design efficiency and reduce testing costs. However, due to the complexity of integrated circuits and the diversity of faults, traditional simulation methods face certain challenges in terms of efficiency and accuracy [2]. Affinity Propagation (AP) is a commonly used clustering method that can divide the sample set into different categories for circuit fault diagnosis. Differential Evolution (DE) is an optimization algorithm that optimizes circuit parameters to improve circuit performance and stability [3]. The study combines these two algorithms to diagnose circuit faults through AP, and then optimizes circuit parameters through DE to achieve efficient and accurate circuit virtual simulation. The study provides a more efficient and accurate virtual simulation method for integrated circuit design. The research can provide a more efficient and accurate virtual simulation method for IC design, so as to reduce the design cost, shorten the development cycle, and contribute to the development of IC industry. The paper is organized as follows: In the first section, the research background and purpose are proposed, the research content is introduced, and the research status of AP clustering algorithm, differential evolution algorithm and integrated circuit virtual simulation at home and abroad are discussed and analyzed. In the second section, the integrated circuit virtual simulation combining AP clustering algorithm and differential evolution algorithm is proposed. The circuit parameters of differential evolution algorithm are optimized, and the subpopulation is divided by AP clustering algorithm. In the third section, the effectiveness and performance of the algorithm and the virtual simulation circuit are verified by experiments. In the fourth section, the research results are summarized.

Manuscript received on December 29th, 2023. Received in revised form on July 15th and July 25th, 2024. Accepted for publication on July 25th, 2024.

Peipei Hu. Author is with the Office of Academic Affairs, Wuxi Vocational College of Science and Technology, Wuxi, China phone: +86 18961759064; e-mail: 18961759064@163.com.

II. Related Works

In recent years, AP has gradually become a popular clustering algorithm and has been widely applied. DE is an evolutionary optimization algorithm that has received widespread attention. The information fusing method is an important analyzing means. Zhao et al. proposed a multiple fusing method combing AP and other methods. This study confirmed that it had high classifying accuracy, showing good performance [4]. To provide a reasonable solution for improving the monitoring reliability and economic benefits of the entire industrial process, Ma et al. put forward a distributed quality fault detecting framework with AP. These results confirmed the excellent fault detecting performance of this framework [5]. To improve the classical clustering algorithm, Tang et al. proposed a strategy to extend AP in non-spherical clustering by constructing class similarity of objects. These results confirmed that the adaptive spectral affinity propagation algorithm was superior to benchmark algorithms [6]. How to simultaneously locate multiple high-quality equivalent positioning systems and obtain a uniformly distributed probability density function remains a challenging task. Regarding this, Zhou et al. proposed a method for solving multi-modal multi-objective DE. These results confirmed the superiority of this model, which could locate multiple equivalent positioning systems and obtain a uniformly distributed probability density function [7]. To obtain more energy and reduce total costs for offshore wind turbines, Wang et al. proposed a novel accelerated multi-objective DE to globally optimize the device layout. These results confirmed that the optimized hybrid wind farm had a 37.75% increase in power generation compared to pure wind farms, and a 43.65% reduction in wind turbine foundation wave loads [8]. In many classic DE, population size is usually determined by users based on empirical values. And it remains unchanged during the evolution process, which greatly affects the performance of differential evolution. Li et al. proposed a dynamic population reduction DE that combined linear and nonlinear strategy piece-wise functions. These results confirmed that dynamic population reduction DE performed well in overall performance and was significantly superior to other algorithms [9].

In recent years, virtual simulation of integrated circuits have been continuously developing and applied. To improve the linearity of presynaptic current, Cha et al. proposed a neuron circuit with a capacitive cross impedance amplifier integrator. These results confirmed that the proposed neural circuit successfully improved output linearity at a wide range of input current levels [10]. To propose, design, and implement a novel narrowband pass filter based on half mode substrate integrated wave-guide, Mahant et al. proposed using Ansoft high-frequency structure simulator for simulation. These results confirmed that at a center frequency of 11.2 GHz, the return loss was 20.39 dB, the insertion loss was 1.59 dB, and the 3dB fractional bandwidth was 2.58% [11]. Finding a balance between solar thermal gain and solar transmittance is an appropriate strategy for achieving energy efficiency when designing photovoltaic shading systems. Gao et al. proposed using parameterized script modeling to integrate thermal and lighting performance, and utilizing multi-objective optimization to optimize the balance between them. These results confirmed that the integrated framework was feasible and could be extended to the design of other advanced shading systems or building integrated photovoltaics [12]. In the absence of a multiplier circuit, the differential trans-conductance amplifier exhibits two operating modes: incremental and decreasing. Vista et al. proposed a floating memristor model using a single voltage difference trans-conductance amplifier and grounded passive components. These results confirmed the performance of the proposed memristor by examining the characteristics of all possible temperature and process angles through the Cadence Virtuoso layout [13]. When connected to the grid, photovoltaic power generation systems can improve the quality of electricity. Rajagopal et al. proposed an adaptive neural fuzzy inference system based on an improved moth flame optimization algorithm. These results confirmed that the designed controller could maintain the exchange of active and reactive power, regulate the DC bus voltage, grid voltage, and grid current [14]. Using an optimized circuit breaker model can estimate the energy absorption requirements during fault current suppression. Torwelle et al. proposed a universal fault current calculating means with equations. These results confirmed that this method could represent temporary blockages over a large time after a fault occurs, facilitating the reactors and breakers' determination in the power grid [15].

To sum up, the complexity of IC design requires that virtual simulation technology not only has efficient diagnostic capability, but also has accuracy in parameter optimization. Existing research, such as the work of Lee S and Ali Y, has made progress in battery performance simulation, but there is still room for improvement in the accuracy and efficiency of circuit simulation [16]. In addition, Yan Y et al. 's research on field effect transistors has promoted the development of integrated circuits, but has not fully solved the challenges of simulation technology in fault diagnosis and parameter optimization [17]. In view of these shortcomings, a combination of AP clustering algorithm and differential evolution algorithm is proposed to improve the efficiency and accuracy of IC virtual simulation. The innovation of this research lies in the development of a new simulation method by integrating the adaptive clustering ability of AP clustering algorithm and the global optimization characteristics of DE algorithm. This method shows higher precision, recall and F1 mean in circuit fault diagnosis and parameter optimization, and has faster convergence speed and higher stability. The study also explores the combination of multi-objective optimization algorithms and neural network techniques to provide new perspectives and tools for building energy efficient design, and this approach has been validated in multiple climate regions, showing its broad applicability and overall performance improvement.

III. Optimization of Circuit Parameters and Sub-population Partitioning by Integrating AP

In this study, AP is used for clustering analysis of components and sub-circuits. By measuring the similarity between components, they can be divided into different clusters. DE is used to optimize the parameter configuration of this circuit. By searching and optimizing in the parameter space, the optimal circuit configuration can be found to meet design and performance requirements.

A. Circuit Parameters Optimization based on Differential Evolution Algorithm

In the field of integrated circuit design, we are faced with the dual challenges of fault diagnosis and parameter optimization. To address this challenge, the study proposes a novel approach that combines the advantages of differential evolution and adaptive particle swarm clustering. The differential evolution algorithm, known for its global search capability, iterates to find the optimal circuit parameter configuration by simulating the process of natural selection. The adaptive particle swarm clustering algorithm, with its unique self-organizing characteristics, can automatically identify the similarities of components in the circuit and divide them into different categories, so as to assist fault diagnosis. The workflow of this fusion method can be simplified into the following steps: Firstly, the circuit components are classified by the adaptive particle clustering algorithm, and the possible faulty component groups are identified; The differential evolution algorithm is then optimized for the parameters of these groups to improve the overall performance and stability of the circuit. This process not only improves the accuracy of fault detection, but also speeds up the speed of problem solving by optimizing circuit parameters, thus playing an important role in integrated circuit design and testing. In this way, problems can be found and solved at the design stage, reducing manufacturing costs and shortening the research and development cycle. DE is an optimizing algorithm used to solve continuous optimization problems. This algorithm originates from genetic algorithms and introduces differential operations in evolution to achieve efficient exploration and optimization of the search space. DE seeks the optimal solution by simulating the process of biological evolution. It is usually used to handle continuous optimization problems. In virtual simulation of integrated circuits, DE can find circuit parameters' optimal configuration to improve circuit performance or reduce power consumption. By continuously adjusting the parameters of the circuit, such as resistance, capacitance, and inductance, better circuit design and performance optimization can be achieved [18]. Fig. 1 shows the advantages and disadvantages of DE.

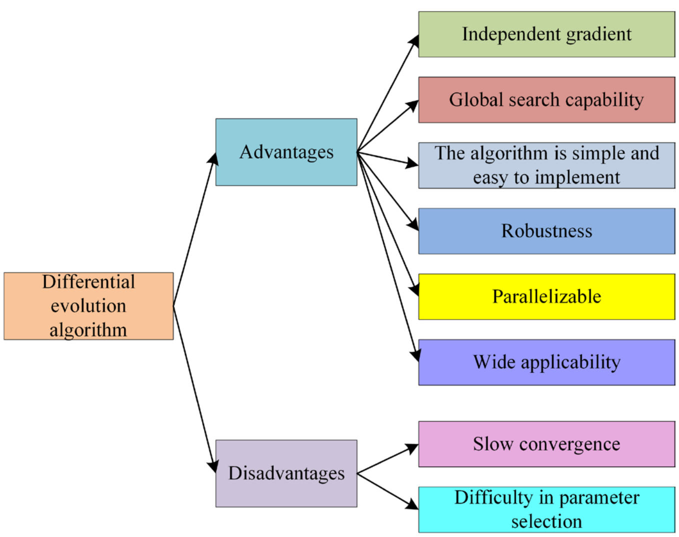


Fig. 1. Advantages and disadvantages of differential evolution algorithm.

DE does not require gradient information for solving problems. This enables DE to be applied to various types of optimization problems, including nonlinear, non-convex, and nonsmooth problems. DE has strong global search ability and can find better solutions in optimization problems of multi-dimensional, multi-modal, and non-convex functions. By randomly generating mutated individuals and global search strategies, it can effectively seek the global optimal solution. The basic principle of DE is relatively simple, easy to understand and implement. It only requires three basic operations: mutation, crossover, and selection. Fig. 2 shows the mutation operation.

DE is relatively insensitive to the selection of initial solutions and population size, and has strong robustness. Therefore, it has good adaptability and application flexibility in practice. DE can be easily parallelized, allowing for the use of multi-core or distributed computing resources to accelerate algorithm execution. The parameter selection in DE has a significant impact on its performance and effectiveness. Choosing appropriate parameter values can improve algorithm's converging speed and quality, but it also increases the difficulty of parameter tuning. DE is used to optimize the parameter configuration of the circuit. By searching and optimizing in the parameter space, the optimal circuit configuration can be found to meet design and performance requirements. DE uses floating-point vectors to encode the population and generate individuals in the population. DE is optimized by selecting two individuals from the parent and performing a difference to obtain the difference vector. Secondly, differential vectors are used to accumulate another individual to obtain a subject. Then the parents and experimental individuals are crossed to produce new offspring. On this basis, by screening the parents and children, offspring that meet the conditions are selected [19]. Equation (1) is the assumed optimization model.

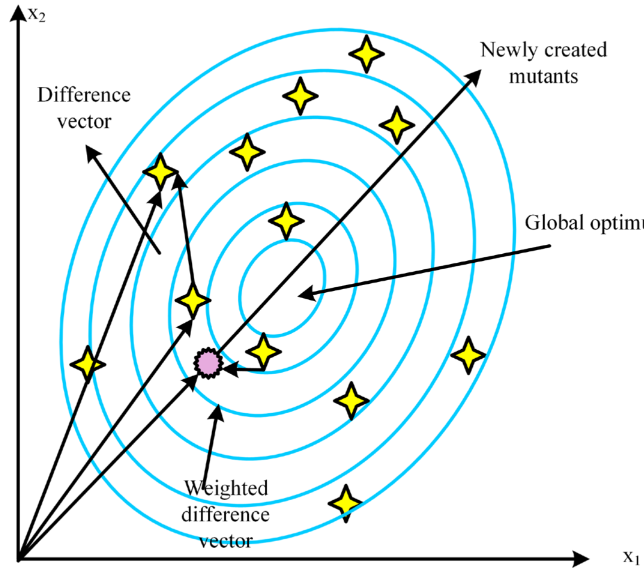


Fig. 2. Mutation operation.

$$\min f(x_1, x_2, ..., x_D), x_j^L \le x_j \le x_j^U, j = 1,2,...,D \tag{1}$$

In equation (1), D is solution space's dimension. x_j^L, x_j^U represent the critical values of the j-th component x_j, respectively. The initialization population is represented by equation (2).

$$x_i(0) \mid x_{j.i}^L \le x_{j,i}(0) \le x_{j.i}^v \tag{2}$$

In equation (2), $i = 1,2,..., NP; j = 1,2,..., D$. $x_i(0)$ stands for the 0th generation's i-th individual. $x_{j,i}(0)$ means the 0th generation individual i's j-th gene. NP stands for the population size. The next step is to randomly generate various groups of individuals, represented by equation (3).

$$x_{j,i}(0) = x_{j,i}^L + rand(0,1) \cdot x_{j,i}^U - x_{j,i}^L \tag{3}$$

In equation (3), $rand(0,1)$ stands for a uniformly distributed random value within [0, 1]. The commonly used differential strategy in DE is represented by equation (4).

$$v_i(g+1) = x_{i1}(g) + F \cdot (x_i(g) - x_{ij}(g)) \tag{4}$$

In equation (4), $i \neq r_1 \neq r_2 \neq r_3$. F stands for a scaling factor. $x_i(g)$ refers to the g-th generation's individual i. The limiting condition for the g-th generation population is represented by equation (5).

$$x_i(g) \mid x_{j.i}^L \leq x_{j,i}(g) \leq x_{j.i}^v \tag{5}$$

The mutated intermediate is represented by equation (6).

$$x_i(g+1) \mid x_{j.i}^L \leq x_{j,i}(g+1) \leq x_{j.i}^v \tag{6}$$

Next, the g-th generation population and its variant intermediates are used for crossover in equation (7).

$$u_{j,i}(g+1) = \begin{cases} v_{j,i}(g+1), rand(0,1) \leq CR, j = j_{rand} \\ x_{j,i}(g) \end{cases} \tag{7}$$

In equation (7), CR stands for the crossover probability. j_{rand} represents a random integer within [1, D]. The selection operation adopts the greedy algorithm in equation (8).

$$x_i(g+1) = \begin{cases} u_i(g+1), f(u_i(g+1)) \leq f(x_i(g)) \\ x_i(g) \end{cases} \tag{8}$$

Fig. 3 shows the basic steps of DE.

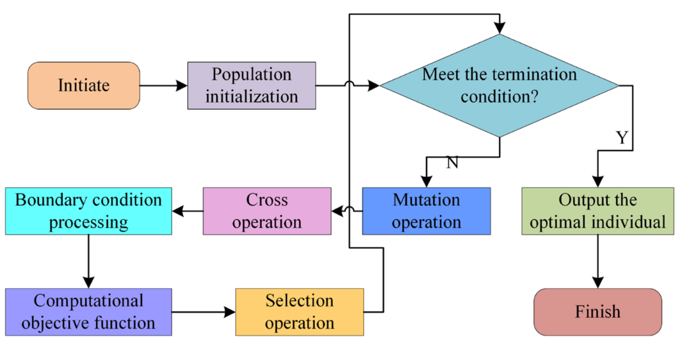


Fig. 3. Basic steps of differential evolution algorithm.

Firstly, the population is initialized, and some individuals are randomly generated to form the initial population. A fitness function can evaluate each individual' fitness values, and some individuals are selected as parent individuals. For each parent individual, mutated individuals are generated through differential operations. Differential operation involves selecting two other individuals from the population and calculating the differential individuals through linear combination. For each mutated individual, crossover operation is performed to generate offspring individuals. A fitness function can evaluate each offspring individual's fitness values, and some individuals are selected as the parents of the next generation. These three operations are repeated until the termination conditions are met. However, DE faces some challenges, such as low search efficiency for high-dimensional problems and the need for appropriate parameter adjustments to improve algorithm performance.

B. Combining Differential Evolution Algorithm with AP for Sub-Population Partitioning

To optimize the performance of DE, researchers have proposed many improvements and variants, such as adaptive DE, multi-objective DE, etc. These improvements and variants can be selected and applied according to the characteristics of specific problems to improve its search efficiency and convergence speed [20]. Research chooses to merge AP to partition the sub-populations of DE. AP is based on the similarity and accessibility of data points. It allocates data points to different clusters through iterative calculation, making the similarity of data points within the same cluster large and the difference of different clusters' data points large. Fig. 4 shows the AP flowchart.

The responsibility and availability in AP are indicators used to measure the degree of similarity and affinity between data points. In AP, each data point calculates its own responsibility and availability based on the similarity and affinity with other data points. Responsibility indicates the suitability to which a data point selects other data points as its pattern, while availability indicates the suitability to which other data points choose that data point as their pattern [21]. Specifically, for a data point i, its responsibility $r(i,k)$ refers to the suitability of data point k for i as its pattern. It is equal to the sum of the

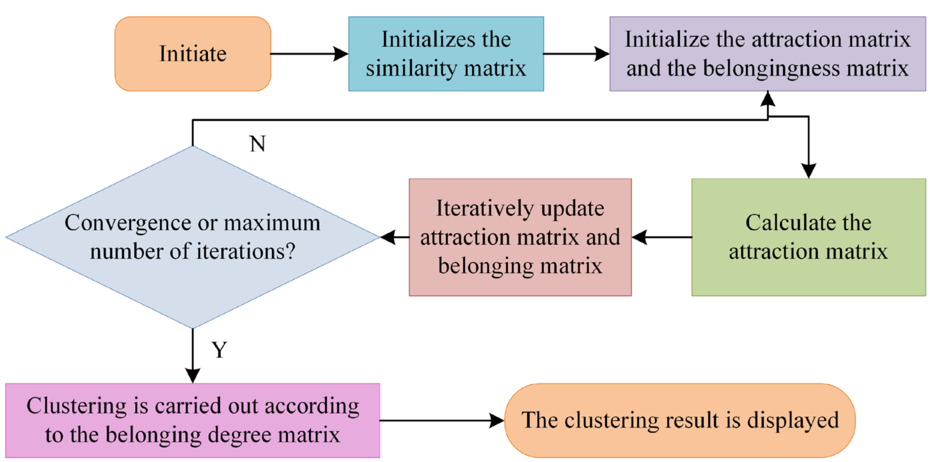


Fig. 4. Flow chart of AP algorithm.

degree to which k chooses i as its attribution and the degree to which k chooses other data points j as its attribution minus the sum of the degree to which j chooses k as its pattern and j chooses other data point l as its attribution in equation (9).

$$r(i,\ k) = s(i,\ k) - max\{a(j,\ k) + s(j,\ k)\} \tag{9}$$

In equation (9), $s(i,k)$ represents i and k's similarity. $a(j,k)$ represents the degree to which j chooses k as its mode availability. Availability $a(i,k)$ indicates the suitability of k's choice of i as its mode. It is equal to the sum of the responsiveness of i selecting k as its mode and i selecting j as its mode minus the maximum sum of the responsiveness of k selecting j as its mode and k selecting other data point l as its mode in equation (10).

$$a(i,k) = min\{0, r(k,k) + sum(max\{0, r(j,k)\})\} \tag{10}$$

In equation (10), sum represents the sum of all data points j. To better utilize the information from the previous iteration while reducing data fluctuations, during the information transmission process, the value from the previous iteration will be multiplied by $1-\lambda$ times and the current value will be multiplied by λ times. The change in responsiveness is represented by equation (11).

$$r(i,k) = (1-\lambda) \times r(i,k) + r(i,k)_{last} \tag{11}$$

Equation (12) represents the change of availability.

$$a(i,k) = (1-\lambda) \times a(i,k) + a(i,k)_{last} \tag{12}$$

By iteratively calculating responsibility and availability, AP divides data points into multiple clusters and selects one data point from each cluster as its pattern. In AP, self-availability is a special case of availability. It represents the degree to which data points choose themselves as their patterns. Self-availability is calculated by subtracting the sum of the responsibilities of data points selecting themselves as their patterns and other data points selecting themselves as their patterns from the sum of the responsibilities of other data points selecting themselves as their patterns, as shown in equation (13).

$$a(i,i) = min\{0, r(i,\ i) + sum(max\{0, r(j,i)\})\} \tag{13}$$

In equation (13), sum represents the sum of all data points j. In AP, self-availability plays a role in self-evaluation, indicating the degree to which the data point is considered suitable for its own cluster. If the self-availability is positive, it indicates that the data point is considered a good pattern and becomes a representative of its cluster. If the self-availability is negative, it indicates that the data point is not suitable as a pattern and will not be selected as a representative of clustering. The combination of self-availability and membership helps AP achieve adaptive clustering. By iteratively calculating responsiveness and belonging, data points will automatically adjust their relationships with each other, forming stable clustering results. Fig. 5 shows the sub-population partitioning based on AP.

The sub-population partitioning method based on AP can be applied to various optimization problems, helping to solve complex and high-dimensional optimizing problems, and improving the efficiency and accuracy of optimization algorithms. Meanwhile, this method can also be combined with other optimization algorithms to further improve optimization performance. Backpropagation Neural Networks (BPNN) are used to optimize and train AP clustering algorithm. By comparing the predicted output of the output layer with the target value, the network achieves fine adjustment of the weight, so as to reduce

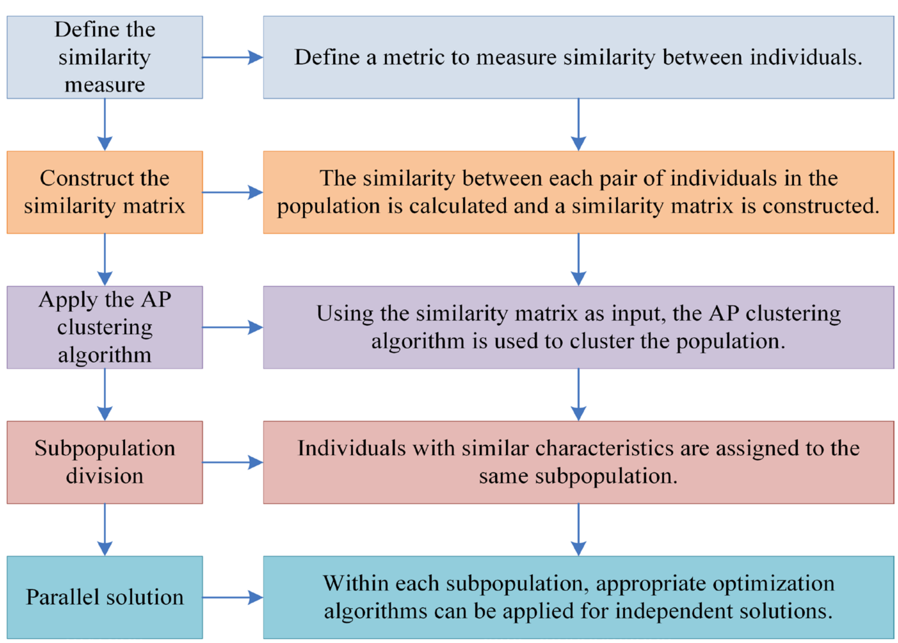


Fig. 5. Sub-population division process.

the prediction error. This process involves several iterations until the network output reaches the desired level of accuracy. The transmission of data in the network first passes through the input layer, then into the hidden layer, and finally generates the prediction result at the output layer. The error is evaluated by comparing the predicted value of the network with the actual observed value, and then using the gradient descent algorithm to optimize the weight and bias of the network. This circular process of forward and backpropagation continues until a preset termination condition is met. The number of nodes in the input layer is directly related to the feature dimension of the input data, while the number of nodes in the hidden layer is the optimal solution obtained through experimental adjustment, which is usually determined by trial and error, as described in formula (14) [22].

$$m = \sqrt{n+l} + a \tag{14}$$

In equation (14), a is the constant from 1 to 10, m is the number of nodes in the hidden layer, l is the number of nodes in the output layer, and n is the number of nodes in the input layer. Through this method, BPNN can learn the complex mapping relationship between input data and output results, and provide an accurate prediction model for building energy efficiency optimization design. In the neural network model of this study, the number of nodes in the output layer is determined according to the number of target variables involved in the research problem. The nonlinear activation function is used in the output layer to enhance the prediction ability of the model. Among the many nonlinear activation functions, hyperbolic functions are widely used because of their ability to handle outputs in the range [-1, 1], which makes them particularly suitable for nonlinear transformations of hidden layers. The specific expression of hyperbolic function is given in formula (15).

$$f(x) = \frac{1}{1+e^{x}} \tag{15}$$

In Formula (15), through this setup, the network can more effectively capture and learn complex nonlinear relationships between input data and output results, thereby improving prediction accuracy and model performance. There is a direct correlation between the sample size used in the experiment and the accuracy of the model's predictions. Increasing the number of samples can improve the response accuracy of the model, but this improvement will gradually become saturated with the increase of the sample size. When the sample size reaches a certain threshold, the accuracy of the model will stabilize at a specific level, and further increasing the sample size will not lead to significant accuracy improvement. At the same time, the expansion of the scale of the network will lead to the increase of the complexity of the mapping relationship. When initializing network weights, there are usually two strategies: one is to give the weights a small enough initial value to avoid saturation of the activation function; The second is to ensure that the weights are initialized to a balanced number of positive and negative values. This approach helps to ensure that the selected model has the best generalization ability and prediction accuracy when dealing with real-world problems. In integrated circuits, AP is widely used in circuit fault diagnosis and testing. Integrated circuits are complex circuit systems composed of various electronic components, which may have faults or defects. Through AP, faults in circuits can be identified and classified. Fig. 6 shows the specific steps.

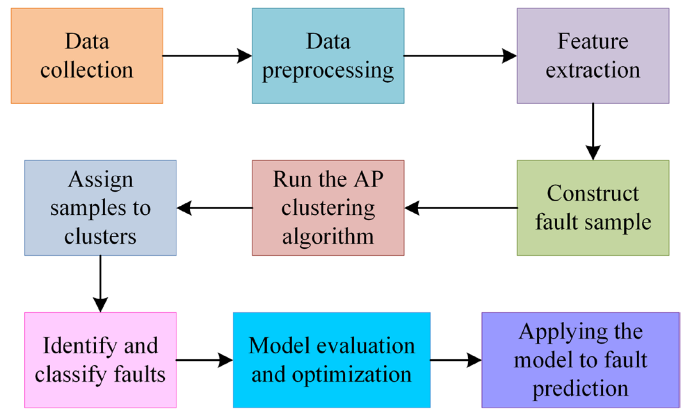


Fig. 6. Fault identification and classification steps in a circuit.

In Fig. 6, in the fault diagnosis of integrated circuits, the AP clustering algorithm is applied through a continuous iterative process, starting from data collection and feature extraction, to build a similarity matrix to quantify the similarity between components. After initializing the algorithm parameters, the damping coefficient controls the propagation of information among the circuit nodes, and the priority affects the number of clusters. In the iterative calculation, the responsiveness and belongingness of each node are updated to assess their suitability as cluster centers. With the progress of the algorithm, the center node of the cluster is gradually determined, and other samples are allocated to the corresponding cluster based on similarity. Finally, the clustering results are verified to ensure accuracy, provide basis for fault analysis, identify fault causes, and propose solutions.

IV. Performance Analysis of Virtual Simulation of Integrated Circuits Integrating AP and Differential Evolution Algorithm

Two 2.5DIC benchmark sets were selected from dies in IWLS '05 and Opencores benchmarks, assuming 20 scan chains per Die and a test frequency f set to 50MHz. In this experiment, the algorithm is verified by using Matlab R2010a software on an Intel computer equipped with 4G memory. For differential evolution algorithm and adaptive particle swarm clustering algorithm, after a series of parameter tuning experiments, the following optimal parameter combinations are determined to ensure the optimal performance of the algorithm on test scheduling problems. The differential evolution algorithm selects a scaling factor F of 0.8, a crossover probability Cr of 0.9, and a population size of 50. For the adaptive particle cluster clustering algorithm, the damping coefficient is determined to be 0.6, and the priority p is 1.5. Such parameter configuration helps the algorithm to automatically identify the cluster center of circuit components without the need to define the number of classes in advance, thus effectively supporting the fault diagnosis process. The study first analyzed the performance of AP and DE, and then conducted virtual simulation and verification of the optimized circuit configuration. By analyzing and comparing simulation results, the performance and reliability of the circuit can be evaluated, and necessary adjustments and improvements can be made.

A. *Performance Analysis of Integrating AP and Differential Evolution Algorithm*

To test the performance of integrating AP and DE, a comparative analysis was conducted between separate DE, separate AP, Ant Colony Algorithm (ACA), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Artificial Neural Network (ANN), Simulated Annealing Algorithm (SAA). These algorithms are all optimization algorithms, but they differ when dealing with optimization problems. The following is a comparative analysis of these algorithms, and Table I is the results of different indicators.

In Table I, the algorithm combination combining DE and AP has the best performance, its accuracy reaches 94.26%, recall rate is 93.41%, F1 value is 88.59%, convergence speed is 56.77 seconds, and stability is 93.17%. This shows that the fusion algorithm has high accuracy and reliability in identifying circuit faults and optimizing parameters. When using DE or AP alone, performance is slightly lower than fusion algorithms, but still better than GA, SAA, ACA, PSO, and ANN. In particular, the convergence speed of the fusion algorithm is the fastest, which means that it can find a solution in a short time, which is especially important for the circuit design and testing process requiring fast iteration and real-time feedback, indicating that the fusion AP clustering algorithm and differential evolution algorithm have higher accuracy and better performance. To further test the performance of fusion AP and DE, the precision, recall, and fitness of fusion AP and DE were compared and analyzed with PSO in Fig. 7.

Fig. 7(a) shows the fusion of AP and DE, with a relatively high precision rate above 0.9, and a recall above 0.8. Fig. 7(b) shows the accuracy and recall of PSO, ranging from 0.7 to 0.8. This indicates that the fusion of AP and DE has higher performance. To verify the clustering effect of AP, a visualization method was used to analyze and compare the clustering results in Fig. 8.

TABLE I
Different Index Results of Six Algorithms

Algorithm index	Precision(%)	Recall(%)	F1 value(%)	Rate of convergence(s)	Stability(%)
DE+AP	94.26	93.41	88.59	56.77	93.17
DE	83.27	81.53	80.26	76.39	88.23
AP	82.73	88.97	83.69	78.52	83.64
GA	73.68	82.38	76.30	70.67	87.31
SAA	86.11	77.25	76.25	39.71	84.15
ACA	69.34	69.58	70.38	48.26	72.64
PSO	81.27	82.34	82.55	52.34	83.55
ANN	78.59	76.53	79.69	47.19	80.42

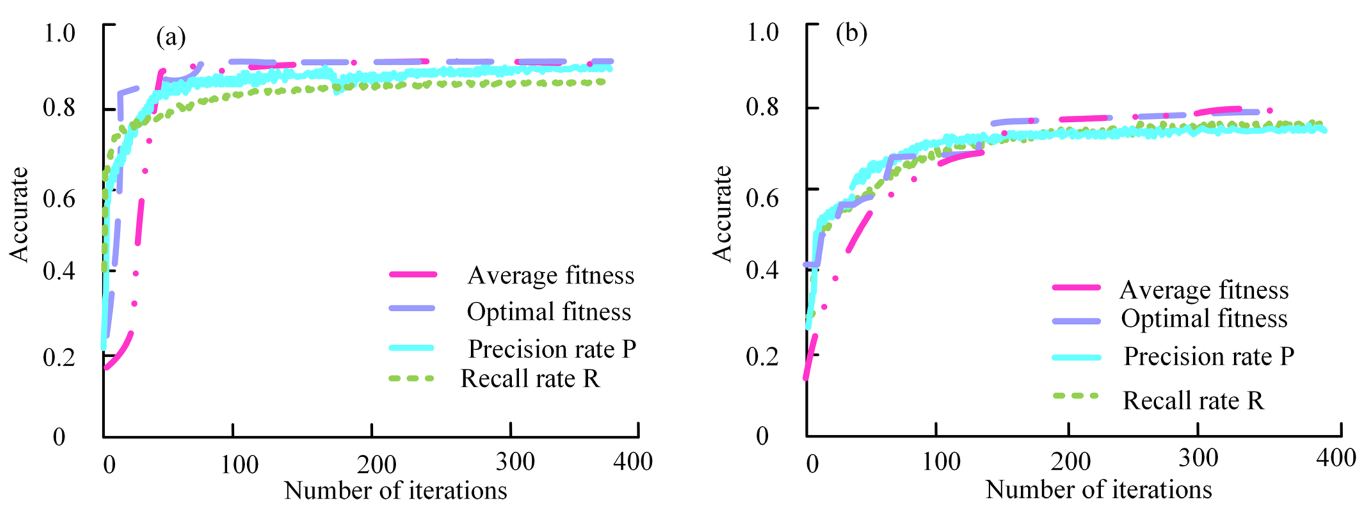


Fig. 7. Comparison of accuracy, recall and fitness of different algorithms: (a) AP+DE; (b) particle swarm optimization algorithm.

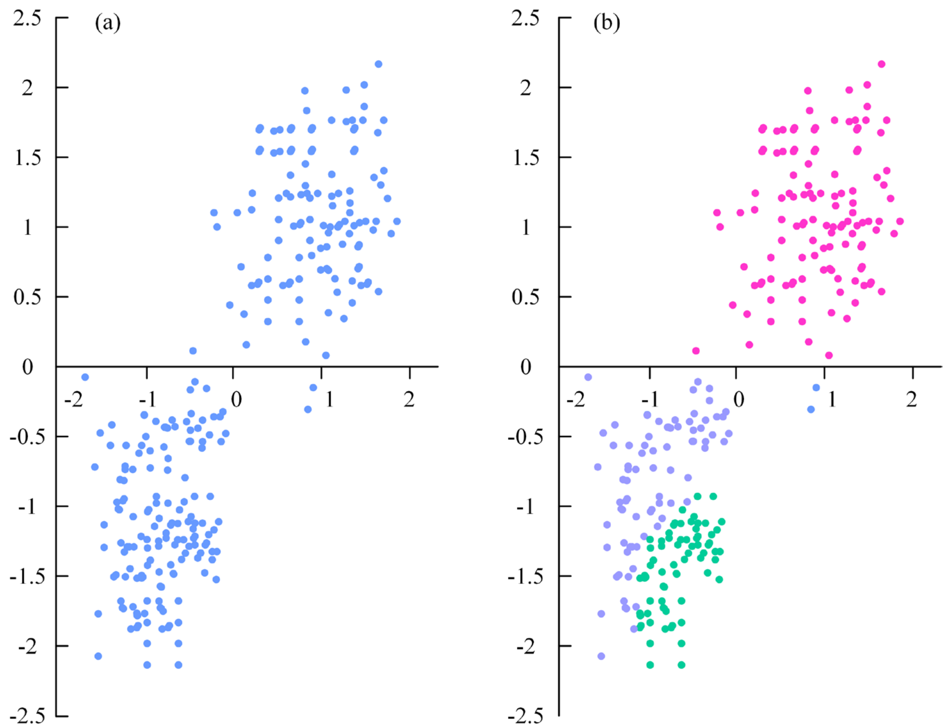
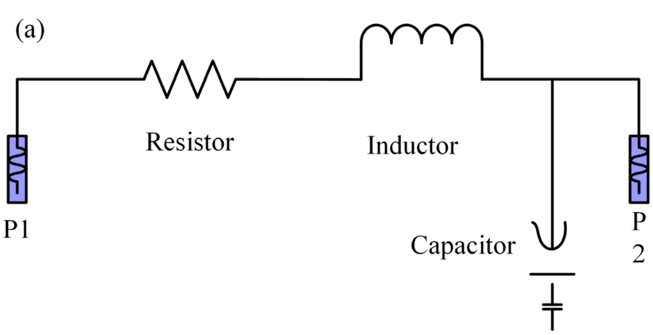


Fig. 8 Clustering effect of AP algorithm: (a) raw data; (b) clustering result.

Fig. 8(a) presents the original integrated circuit data. Fig. 8(b) shows the integrated circuit data after clustering. AP clusters data points on the similarity matrix, with the goal of minimizing the distance between data points and their class representative points. AP performs well in clustering, especially without pre-defining the classes. It can identify the position and number of class centers automatically, maximizing the sum of similarities between all data points and the nearest class representative point.

B. *Virtual Simulation Performance Analysis of Integrated Circuits*

In the developed circuit simulation tool, the method of integrating AP and DE is used. The circuit is built in the circuit simulation tool, with a scanning frequency range of 0.1 GHz-1 GHz and a gradient of 0.1 GHz. The S parameter results are compared in Fig. 9.



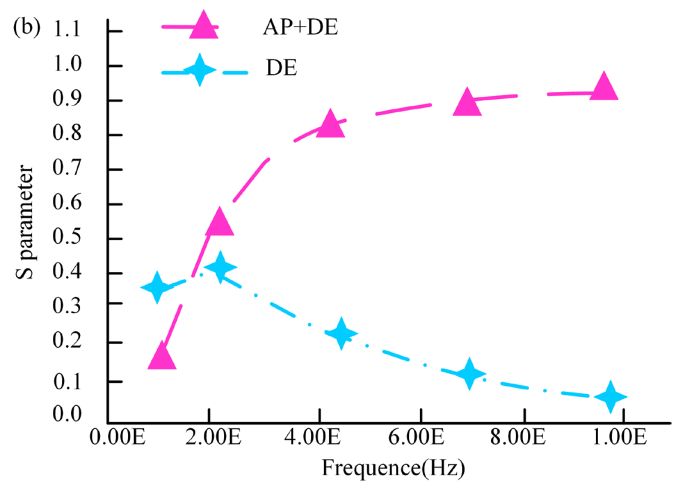


Fig. 9. Integrated circuit simulation diagram and S parameter: (a) integrated circuit virtual simulation diagram; (b) sparameter.

Fig. 9(a) shows the virtual simulation of the integrated circuit, with the scanning frequency set to 0.1 GHz-1 GHz. Fig. 9(b) shows a comparison of the results of fusing AP and DE, as well as individual DE. The points corresponded to each frequency coincide, and the results were consistent, indicating that using the fusion AP and DE to calculate the S parameter was feasible. Further analysis is conducted on the performance of integrating AP and DE in dealing with complex circuit problems, and the accuracy and efficiency of PSO in dealing with complex circuit problems are compared in Fig. 10. In the context of circuit simulation and fault diagnosis, accuracy is a measure of an algorithm's ability to correctly identify faults or optimize parameters. It is determined by calculating the percentage of instances in which the algorithm correctly predicts a failure or finds an effective solution to the total number of instances. Efficiency refers to the speed at which an algorithm solves a problem, including the use of computational resources and the time it takes to solve the problem. In circuit design, efficient algorithms can complete simulation and optimization in a relatively short time, thus speeding up the design cycle.

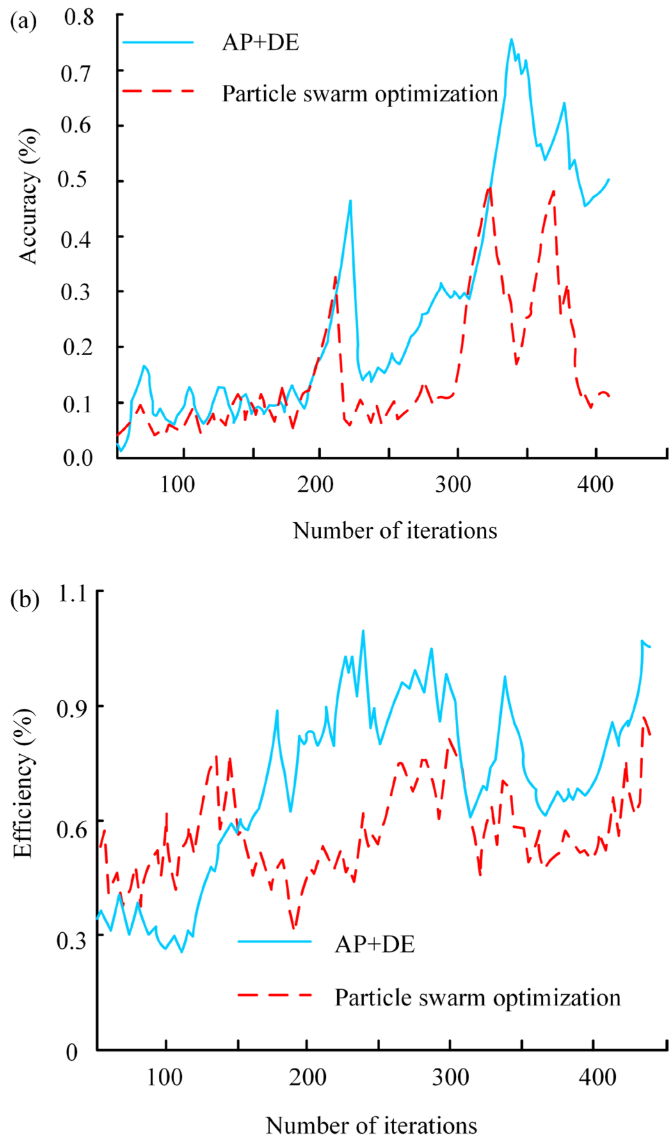


Fig. 10. Accuracy and efficiency in dealing with complex circuit problems: (a) the accuracy; (b) the efficiency.

In Fig. 10, the fusion of AP and DE had higher accuracy and efficiency. After 300 iterations, the accuracy reached 89% and the efficiency reached 100%, indicating that the fusion of AP and DE performed better when dealing with complex integrated circuit problems. To observe the mean features of the sequence before and after change point detection, the following is a graph of the mean features before and after change point detection in Fig. 11.

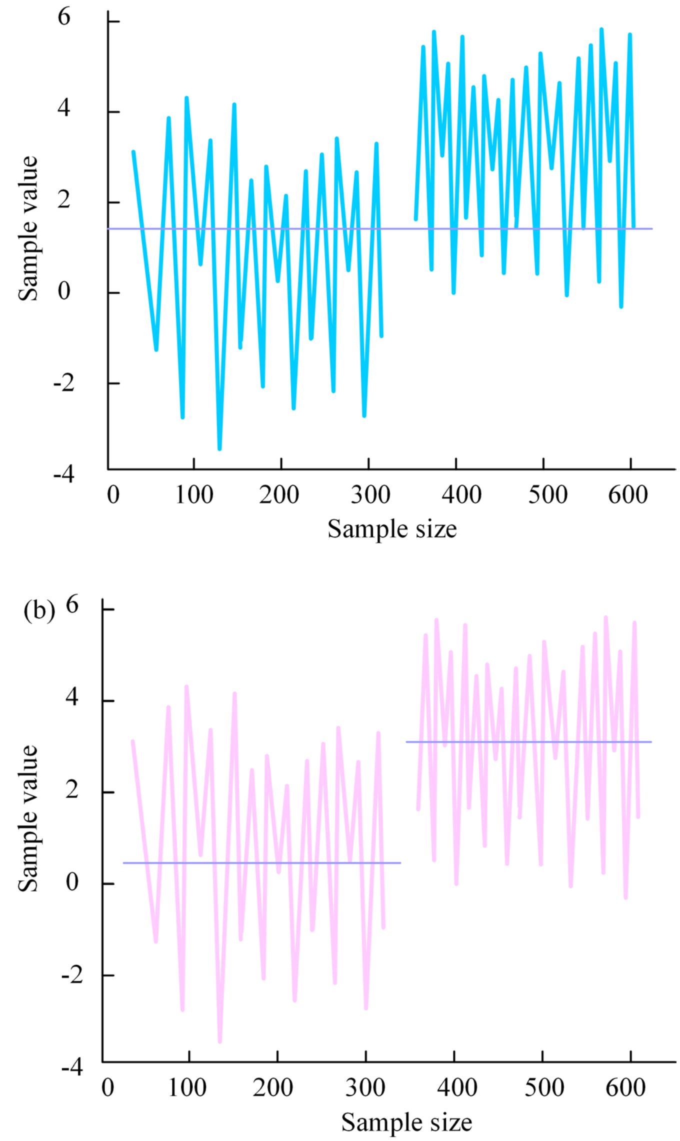


Fig. 11. The mean value feature map before and after change point detection: (a) mean value of sample point features before detection; (b) mean of the features of the sample points after detection.

The straight line in Fig. 11(a) represents the data mean feature of the unchanged point model. The straight line in Fig. 11(b) represents the data mean feature under the change point model, which divided the data into two segments. Each segment characterized the mean of these data, which was more in line with the true data. This indicates that the proposed DE can detect the position of change points and better explain the characteristics of normal distribution means.

V. Conclusion

Virtual simulation provides an effective means for integrated circuit design to simulate circuit operation on a computer. It helps to detect and solve problems early in the design phase, thereby reducing manufacturing costs and shortening research and development cycles. The study adopted a simulation method that integrates AP and DE, aiming to improve the accuracy of circuit fault diagnosis and the efficiency of circuit parameter optimization. These experiments confirm that the fusion method of AP and DE performs well in terms of precision, ultimately stabilizing above 0.9. Its recall remains above 0.8. Specifically, after 300 iterations, the accuracy of this method reaches 89% and the efficiency reaches 100%. Compared to traditional simulation methods, this fusion algorithm exhibits higher accuracy and efficiency in circuit fault diagnosis and parameter optimization. However, this method of integrating AP and DE is complex, involving multiple parameters and operations, resulting in increased computational difficulty and time cost. To improve the efficiency and reliability of virtual simulation of integrated circuits, future research can explore more optimization methods.

References

[1] S. Chander, P. Vijaya, R. Fernandes, A. P. Rodrigues, and R. Maheswari, "Dolphin-political optimized tversky index based feature selection in spark architecture for clustering big data," *Adv. Eng. Softw.*, vol. 176, no. 1, pp. 103331-1-13, Feb. 2023, DOI: 10.1016/j.advengsoft.2022.103331.

[2] G. H. Gao, P. Wang, and H. Wang, "Follow-the-leader motion strategy for multi-section continuum robots based on differential evolution algorithm," *Ind. Robot.*, vol. 48, no. 4, pp. 589-601, Jun. 2021, DOI: 10.1108/IR-01-2021-0001.

[3] X. Zhang and X. Wang, "Hybrid-adaptive differential evolution with decay function applied to transmission network expansion planning with renewable energy resources generation," *IET Gener. Transm. Distrib.*, vol. 16, no. 14, pp. 2829-2839, Oct. 2022, DOI: 10.1049/gtd2.12296.

[4] H. Zhao, J. Mi, and M. Liang, "A multi-granularity information fusion method based on logistic regression model and dempster-shafer evidence theory and its application," *Int. J. Mach. Learn. Cybern.*, vol. 13, no. 10, pp. 3131-3142, Jun. 2022, DOI: 10.1007/s13042-022-01584-w.

[5] L. Ma, M. Wang, J. Dong, and K. Peng, "A novel distributed detection framework for quality-related faults in industrial plant-wide processes," *Neurocomputing*, vol. 492, no. 1, pp. 126-136, Jul. 2022, DOI: 10.1016/j.neucom.2022.04.014.

[6] L. Tang, L. Sun, C. Guo, and Z. Zhang, "Adaptive spectral affinity propagation clustering," *J. Syst. Eng. Electron.*, vol. 33, no. 3, pp. 647-664, Jun. 2022, DOI: 10.23919/JSEE.2022.000060.

[7] T. Zhou, Z. Hu, Q. Su, and W. Xiong, "A clustering differential evolution algorithm with neighborhood-based dual mutation operator for multimodal multiobjective optimization," *Expert Syst. Appl.*, vol. 216, pp. 119438.1-119438.18, Apr. 2023, DOI: 10.1016/j.eswa.2022.119438.

[8] Y. Wang, Z. Liu, and H. Wang, "Proposal and layout optimization of a wind-wave hybrid energy system using GPU-accelerated differential evolution algorithm," *Energ.*, vol. 239, pp. 121850.1-121850.16, Jan. 2022, DOI: 10.1016/j.energy.2021.121850.

[9] K. Li, X. Fu, F. Wang, and H. Jalil, "A dynamic population reduction differential evolution algorithm combining linear and nonlinear strategy piecewise functions," *Concurr. Comput.: Pract. Exp.*, vol. 34, no. 6, pp. 1-14, Dec. 2022, DOI: 10.1002/cpe.6773.

[10] Y. J. Cha, K. H. Park, J. H. Lee, and B. G. Park, "An integrate-and-fire neuron with capacitive trans-impedance amplifier for improving linearity in spiking neural networks," *Solid-State Electron.*, vol. 197, pp. 108435.1-108435.5, Nov. 2022, DOI: 10.1016/j.sse.2022.108435.

[11] K. Mahant, H. K. Mewada, A. V. Patel, A. Vala, and J. Chaudhari, "HMSIW based highly selective filter for radar applications," *Circuit World*, vol. 47, no. 2, pp. 206-212, Jul. 2021, DOI: 10.1108/CW-03-2020-0039.

[12] Q. Gao, Y. Yang, and Q. Wang, "An integrated simulation method for PVSS parametric design using multi-objective optimization," *Front. Archit. Res.*, vol. 11, no. 3, pp. 509-526, Jun. 2022, DOI: 10.1016/j.foar.2021.11.003.

[13] J. Vista and A. Ranjan, "Flux controlled floating memristor employing VDTA: Incremental or decremental operation," *IEEE Trans. Comput.-Aid. Des. Integr. Circuit. Syst.: Publ. IEEE Circ. Syst. Soc.*, vol. 40, no. 2, pp. 364-372, Jun. 2021, DOI: 10.1109/TCAD.2020.2999919.

[14] S. Rajagopal and P. Umapathy, "An investigation of grid-integrated photovoltaic system intended for hybrid moth flame optimization using anfis techniques," *J. Intell. Fuzzy Syst.: Appl. Eng. Technol.*, vol. 42, no. 3, pp. 2505-2519, Feb. 2022, DOI: 10.3233/JIFS-211748.

[15] P. Torwelle, A. Bertinato, B. Raison, T. D. Le, and, M. Petit, "Fault current calculation in MTDC grids considering MMC blocking," *Electr. Power Syst. Res.*, vol. 207, pp. 107662.1-107662.12, Jun. 2022, DOI: 10.1016/j.epsr.2021.107662.

[16] S. Lee and Y. Ali, "An integrated experimental and modeling study of the effect of solid electrolyte interphase formation and Cu dissolution on CuCo2O4-based Li-ion batteries," *Int. J. Energ. Res.*, vol. 46, no. 3, pp. 3017-3033, Oct. 2022, DOI: 10.1002/er.7359.

[17] Y. Yan, Y. Zhao, and Y. Liu, "Recent progress in organic field-effect transistor-based integrated circuits," *J. Polym. Sci.*, vol. 60, no. 3, pp. 311-327, Aug. 2022, DOI: 10.1002/pol.20210457.

[18] C. Van Kien, H. P. H. Anh, and N. N. Son, "Adaptive inverse multilayer fuzzy control for uncertain nonlinear system optimizing with differential evolution algorithm," *Appl. Intell.*, vol. 51, no. 1, pp. 527-548, Aug. 2021, DOI: 10.1007/s10489-020-01819-9.

[19] K. C. Lin and C. C. Liao, "Simulation of slurry residence time during chemical-mechanical polishing using 3-D computational fluid dynamics," *Chem. Eng. Res. Des.*, vol. 191, no. 1, pp. 375-386, Jan. 2023, DOI: 10.1016/j.cherd.2023.01.025.

[20] S. Zhang, G. Li, Y. Zheng, and Y. Liu, "Research on enterprise terminal network planning under the new retail format," *Int. J. Simulat. Process Model.*, vol. 18, no. 3, pp. 200-210, Nov. 2022, DOI: 10.1504/IJSPM.2022.126889.

[21] S. Choudhuri, S. Adeniye, and A. Sen, "Distribution alignment using complement entropy objective and adaptive consensus-based label refinement for partial domain adaptation," *Artif. Intell. Appl.*, vol. 1, no. 1, pp. 43-51, Oct. 2023, DOI: 10.47852/bonviewAIA2202524.

[22] E. Wang, X. Liu, and J. Wan, "An improved filtering algorithm for indoor localization based on DE-PSO-BPNN," *J. Intell. Fuzzy Syst.*, vol. 44, no. 6, pp. 9513-9525, Jun. 2023, DOI: 10.3233/JIFS-230178.