EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

Enhancing AI-Inspired Analog Circuit Design: Optimizing Component Sizes with the Firefly Algorithm and Binary Firefly Algorithm

Trang Hoang[1,*]

[1]Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City (VNU-HCM), 268 Ly Thuong Kiet, District 10, Ho Chi Minh City, Vietnam

Abstract

This paper explores the use of the Firefly Algorithm (FA) and its binary variant (BFA) in optimizing analog circuit component sizing, specifically as a case study for a two-stage operational amplifier (op-amp) designed with a 65nm CMOS process. Recognizing the limitations of traditional optimization approaches in handling complex analog design requirements, this study implements both FA and BFA to enhance convergence speed and accuracy within multi-dimensional search spaces. The Python-Spectre framework in this paper facilitates automatic, iterative simulation and data collection, driving the optimization process. Through extensive benchmarking, the BFA outperformed traditional FA, balancing exploration and exploitation while achieving superior design outcomes across key parameters such as voltage gain, phase margin, and unity-gain bandwidth. Comparative analysis with existing optimization methods, including Particle Swarm Optimization (PSO) and Genetic Algorithm (GA), underscores the efficiency and accuracy of BFA in optimizing circuit metrics, particularly in power-constrained environments. This study demonstrates the potential of swarm intelligence in advancing automatic analog design and establishes a foundation for future enhancements in analog circuit automation.

Keywords: Firefly Algorithm, Binary Firefly Algorithm, simulation-based optimization method, two-stage op-amp

Received on 16 11 2024; accepted on 03 01 2025; published on 08 01 2025

doi:10.4108/eetinis.v12i2.7859

1. Introduction

The rapid advancements in Very-Large-Scale Integration (VLSI) technology have paved the way for the full integration of analog, digital, and mixed-signal circuits on a single chip. This high level of integration has led to the development of numerous Electronic Design Automation (EDA) tools designed to optimize the use of these technologies and minimize time-to-market. However, despite significant progress in digital design automation, the automation of analog design re-mains considerably challenging due to the inherent complexity of analog circuits [1, 2]. The automation of analog design is essential as it ensures optimal solutions for critical design criteria such as gain, power, bandwidth, and area. The analog design process typically involves three main optimization directions: circuit structure selection, parameter sizing, and layout optimization [2–4].

Analog circuit component sizing is a monotonous, time-consuming, and iterative process. This complexity is due to the large number of parameters and the often-unpredictable interactions between them. For complex circuits with extensive search spaces, finding optimal parameter values is one of the most labor-intensive tasks for designers, posing a significant challenge in the design process. Efficient and accurate sizing of analog circuit components is crucial for achieving high-performance designs and is a cornerstone of automatic optimization efforts. Automatic optimization methods for analog circuit component sizing can be broadly classified into two categories: equation-based and simulation-based approaches [4]. The equation-based approach involves deriving polynomial

*Corresponding author. Email: hoangtrang@hcmut.edu.vn

or monomial equations that represent performance parameters. While this method offers faster execution times and higher assurance of global optimization, it is labor-intensive and requires substantial effort from designers, especially for complex circuits [5]. Additionally, the simplifications and approximations needed for these equations can lead to significant deviations from actual values, thus com-promising model accuracy [4, 6]. In contrast, the simulation-based approach utilizes real-time simulation data to optimize objective functions and design constraints, providing a more accurate and general exploration of the search space [4]. The application of optimization algorithms in analog circuit design not only simplifies the component sizing process but also ensures adherence to initial design constraints. In general, many metaheuristic optimization algorithms have been widely applied to address successfully complex and multi-objective optimization problems and a variety of engineering problems. However, in the field of analog circuit design, there has been limited research and published work on optimization algorithms. The few algorithms that have been explored in this domain include Particle Swarm Optimization (PSO) [7–9], Genetic Algorithm (GA) [10–12]. PSO, a swarm intelligence-based algorithm, has demonstrated effective-ness in optimizing CMOS circuit parameters, enhancing performance while reducing design time [7–9]. Similarly, the Genetic Algorithm (GA), which simulates natural selection processes, has been successfully employed to explore optimal configurations in analog circuit design, particularly in problems that require balancing conflicting objectives such as distortion and power consumption [10, 11].

The key research question is whether, after the development of new metaheuristic algorithms, there remains a need to create even newer algorithms specifically for analog circuit design or not. The no free lunch (NFL) theorem [13] addresses this question by stating that the success of an algorithm in solving certain optimization problems does not guarantee similar performance in other problems.

Motivated by this theorem, the author of this study aim to develop a Firefly Algorithm (FA)-inspired automated optimization framework, to develop FA's modified version in details, and to verify these versions' efficacy for analog circuit design.

Among existing optimization algorithms, the FA, a notable swarm intelligence method, has shown remarkable efficiency in balancing exploration and exploitation in the solution space, making it particularly effective for optimizing design variables. The FA inspired by the flashing behavior of fireflies for communication and attraction, has been successfully applied to a variety of engineering and optimization problems. In particular, FA has demonstrated its efficacy in complex optimization tasks such as feature selection, image processing, and wireless sensor network optimization. However, in analog circuit design, FA-inspired optimization method has not been demonstrated its efficiency. This study specifically focuses on applying the FA to optimize analog circuit parameters using the figure of merit (FoM), with a case study of a two-stage operational amplifier (op-amp) circuit.

Since the traditional FA might have slow converging speed, this research also aims to develop a strategy to improve FA's exploration as well as exploitation, thereby enhancing its overall convergence. The mentioned novel method em-ploys binary sequence representation for FA's population, called the Binary FA (BFA), in order to manipulate the algorithm's population update mechanism at a higher level of detail and complexity.

The structure of this paper is organized as follows. Section 2 introduces the concept of the FA and its implementation with two approaches of traditional FA and BFA. Section 3 details the optimization process, including the proposed Python-Spectre model for optimization with FoM as the objective function and the satisfaction of design constraints. Section 4 applies the theory to a case study of a two-stage op-amp circuit. Section 5 presents simulation results and discussions, followed by the conclusions in Section 6.

2. Firefly algorithm

The FA, proposed by Xin-She Yang [14, 15], is a notable swarm intelligence algorithm inspired by the bio-luminescent behavior of fireflies. It is designed to solve global optimization problems by simulating the way fireflies attract each other through their light intensity. The algorithm operates on three ideal assumptions [14]:

- Fireflies are attracted to each other regardless of gender because they are a unisex species.
- Attraction is proportional to their brightness, with dimmer fireflies being attracted to brighter ones. However, attraction decreases as the distance between two fireflies increases. The brightest firefly moves randomly.
- The brightness of a firefly is influenced or determined by the value of the objective function.

To implement the FA effectively, two critical aspects must be considered, namely the variation in light intensity and the formulation of attractiveness. These factors allow for the customization of the algorithm to suit specific problems. In the standard FA, a firefly's light intensity, which represents potential solutions, is proportional to the fitness function value. We observe that fireflies communicate through attractiveness, allowing them to explore the search space via the objective function. This method is more efficient than a random search using a Gaussian distribution because each firefly in the swarm explores the search space by considering the results obtained by others, while still applying their own random movements.

2.1. Traditional FA (FA)

Since the light intensity at a certain distance r from the light source follows the inverse square law, the attractiveness of a firefly decreases with its distance from another firefly. This means that the light intensity I decreases as the distance r increases according to $I \propto 1/r^2$. Additionally, the atmosphere to absorb light becomes weaker as the distance increases. These combined factors limit the visibility of most fireflies to a restricted distance, which implies that effective communication only occurs among individuals in proximity, leading to strong interactions among them.
The brightness of a firefly is typically proportional to the objective function of the problem f_{obj}, which can be expressed as [15]

$$I_i \propto f_{obj}(x_i) \tag{1}$$

where I_i, x_i denote the brightness and the position corresponding to the i^{th} firefly of the population. The light intensity $I(r)$ varies according to the inverse square law as

$$I(r) = \frac{I_S}{r^2} \tag{2}$$

where I_S is the light intensity at the source, and r is the distance from the considered point to the light source.
In an environment with a fixed light absorption coefficient γ, the light intensity I varies with distance r according to the formula

$$I(r) = I_0 e^{-\gamma r} \tag{3}$$

To avoid the undefined singularity at $r = 0$ in the expression I_s/r^2, the light intensity can be approximately calculated in the form of a Gaussian as [15]

$$I(r) = I_0 e^{-\gamma r^2} \tag{4}$$

Additionally, the attractiveness of a firefly is perceived by other fireflies. A firefly with lower brightness will be attracted to the one with higher brightness, and each firefly at different positions will have its own unique attractiveness value β. Since this attractiveness also varies depending on distance, light intensity I and attractiveness β are somewhat synonymous. While intensity I is considered an absolute measure of light emitted by a firefly, attractiveness β is a relative measure of light needing to be seen and is evaluated by other fireflies. Therefore, the attractiveness of a firefly is proportional to the light intensity emitted by it and is defined as [14, 16]

$$\beta = \beta_0 e^{-\gamma r^2} \tag{5}$$

where β_0 is the attractiveness of the firefly at $r = 0$. In equation (5), the distance r between the i^{th} firefly at position X_i and the j^{th} firefly at position X_j is calculated as [16]

$$r_{ij} = \|X_i - X_j\| = \sqrt{\sum_{k=1}^{d}(x_k^i - x_k^j)^2} \tag{6}$$

where $\|.\|$ is the symbol for distance in Descartes coordinate, d is the dimension of the search space, x_k^i, x_k^j are the respective coordinate of the k^{th} dimension of the i^{th}, j^{th} firefly.
The movement of the i^{th} individual when attracted by a brighter j^{th} individual is defined as [14]

$$X_i^{t+1} = X_i^{t+1} + \beta_0 e^{-\gamma r_{ij}^2}(X_j^t - X_i^t) + \alpha \epsilon_i^t \tag{7}$$

where $X_i^(t+1)$ is the position of the i^{th} firefly after updating its position, X_i^t, X_j^t are the initial position of the i^{th}, j^{th} firefly, α represents the coefficient affecting the random motion of fireflies, ϵ_i^t represents a random number following a normal distribution.
The movement of a firefly in equation (7) consists of three parts: the first part is the current position of the i^{th} firefly, the second part on the right-hand side represents the attraction of another firefly that is more attractive, and the last part represents the random movement with the coefficient ϵ_i^t.
The algorithm compares the attractiveness of the new firefly position with the old one. Firefly positions can be sequentially updated by comparing and updating each pair of fireflies after each iteration. If the new position generated is more attractive, the firefly will move to the new position; otherwise, it will remain at its current position.
Local search algorithms often face the risk of getting trapped at a local optimum while the global optimum lies outside the search scope. To address this issue, randomness is often introduced into an algorithm to allow it to jump out of such positions. Random components can take various forms, such as simple randomization by sampling randomly within the search space or by taking random jumps. The movement expression of a firefly in equation (8) individual reflects a random step biased towards brighter firefly individuals. In the case where the firefly individual has the highest brightness intensity ($\beta_0 = 0$), the expression simplifies to a random movement step as

$$X_i^{t+1} = X_i^{t+1} + \alpha \epsilon_i^t. \tag{8}$$

In FA, fireflies interact with each other through their attractiveness, allowing them to explore the search space via the objective function. This method is more effective than random search following a normal distribution because each individual in the swarm explores their space based on the results obtained by other individuals, while still applying their own random movement steps.

According to the theory of FA, a firefly tends to be attracted to other fireflies with higher attractiveness. Therefore, this algorithm has a noticeable advantage, namely automatic congregation. Fireflies may divide the population and automatically congregate into smaller groups because the attraction of nearby individuals is stronger than those at a distance; thus, one can expect each group of individuals to concentrate around a local optimum. This second advantage makes it particularly suitable for multi-objective global optimization problems. The combination of individual movement and position updates based on brightness helps the algorithm balance exploration and exploitation.

The pseudocode for traditional FA used in this study is given in Algorithm 1 as follows.

Algorithm 1: Firefly Algorithm (FA)

Initialize an n-individual population, each firefly has the representation $X^n(t) = (x_1^n, x_2^n, ..., x_d^n)$ where $n = 1, 2, \ldots, N$ and d is the dimension or number of variables of each individual, x_m^n are decimal values ($m = 1, 2, ..., d$);
Define the light absorption coefficient γ, the number of generations g_{max};
Define the objective function $f_{obj}(X)$ and calculate the fitness values $f_{obj}(X^n)$ of each firefly;
$t \leftarrow 0$;
while $(t < g_{max})$ **do**
  for $i = 1 : N$ **do**
    for $j = 1 : N$ **do**
      if $f_{obj}(X^i)$ *is more optimal than* $f_{obj}(X^j)$ **then**
        Move X^j closer to X^i based on equation (7);
      Move the best firefly X^* randomly based on (8);
      Calculate the new fitness values of each firefly;
  $t \leftarrow t + 1$
return X^*;

2.2. Binary FA (BFA)

Since the traditional FA might not overcome the obstacle related to slow converging speed, this study aims to develop a strategy to improve FA's exploration as well as exploitation, thereby enhancing its overall convergence. The mentioned novel method employs binary sequence representation for FA's population, called the Binary FA (BFA), in order to manipulate the algorithm's population update mechanism at a higher level of detail and complexity.

2.3. Representation

Similar to the traditional FA, the population consists of N fireflies. However, individual fireflies X^n (solutions) are represented by d design variables, each represented by a p-bit binary string. $X^n(t) = (x_1^n, x_2^n, \ldots, x_d^n)$ where $n = 1, 2, \ldots, N$. The bit string $(\vec{x_m^n}) \in 0, 1^p$ represents a design variable with $m = 1, 2, \ldots, d$. Each design variable's corresponding real value is decoded from the binary bit string as follows:

$$value = LB + \frac{decimal}{2^m} \times (UB - LB) \tag{9}$$

where $decimal$ is the unsigned integer decoded from the binary bit string, UB and LB are the upper and lower bounds of the corresponding real-valued design variable.

2.4. Distance between two fireflies

The distance between any two fireflies i and j is determined by the Hamming distance, which is the number of differing elements between their permutations. The attractiveness of firefly X_j to firefly X_i is given by the formula

$$\beta(r_{ij}) = \beta_0 e^{-\gamma r_{ij}^2} \tag{10}$$

where r_{ij} is the Hamming distance between fireflies X_i and X_j. The theoretical value of the light absorption coefficient γ is $\gamma \in [0, \infty]$.

2.5. Movement range of fireflies

$$x_i = (1 - \beta(r_{ij}))x_i + \beta(r_{ij})x_j \tag{11}$$

$$x_i = x_i + \alpha(\epsilon - 0.5) \tag{12}$$

Any firefly moves in the search space in two steps. In the first step, $\beta(r_{ij})$ determines the movement of firefly X_i towards firefly X_j as given by equation (7). In the next step, α determines the random movement of firefly X_i as given by equation (8).

The first step is called the β-step. From the equation, it is evident that x_i will equal x_j with a probability given by $\beta(r_{ij})$; and x_i will remain unchanged with a probability given by $(1 - \beta(r_{ij}))$. The β-step procedure is illustrated in Algorithm 2. The next α-step represents the change in bit value for a specific design binary variable. When comparing two fireflies X_i and X_j, α

represents the probability that a specific design variable of X_i will change. The number of variables (n_{var}) change their bit values depending on the Hamming distance (r_{ij}). Since the objective here is to minimize the distance between firefly X_i and X_j, given that X_j is brighter than X_i, then X_i moves towards X_j. The coefficient $n_{var} = \alpha \times r_{ij}$ and α must be small; otherwise, the Hamming distance will increase between X_i and X_j instead of decreasing.
Once the number of variables (n_{var}) that need to change is determined, we randomly select the variables to change from D_{var}. We select the number of bit (nB) that will change in bit value, which depends on the Hamming distance between a pair of design variables r_{ij}^k, where $k \in D_{var}$. Similarly, we have $nB = \alpha \times r_{ij}^k$. The α-step procedure is illustrated in Algorithm 3.

Algorithm 2: FA's β-step

Evaluate the objective function of firefly X_i and X_j, $f_{obj}(X_i)$ and $f_{obj}(X_j)$;
if $f_{obj}(X^j)$*is more optimal than*$f_{obj}(X^i)$ **then**
  Calculate the Hamming's distance $r_{ij} = d(X_i, X_j)$;
  Calculate the attractiveness using equation (3);
for $k = 1 : length(X_i)$ **do**
  if $x_{ik} = x_{jk}$ **then**
    $x(new)_{ik} = x_{jk}$;
  else
    Generate a random number $r \in (0, 1)$;
    if $r < \beta(r_{ij})$ **then**
      $x(new)_{ik} = x_{jk}$;
    else
      $x(new)_{ik} = x_{ik}$;
  Move the best firefly X^* randomly based on (8);
  Calculate the new fitness values of each firefly;
return $X_i(new)$;

3. Python-Spectre framework

To implement an optimization process driven by real-time circuit simulation, it is essential to establish an environment that integrates the optimization algorithm with circuit simulation seamlessly.
In this study, Python is selected for implementing the optimization algorithm due to its extensive library ecosystem. The analog circuit design parameters are simulated using the Spectre simulator from the Cadence Virtuoso tool. Spectre's Ocean scripting language, which supports SKILL programming syntax,

Algorithm 3: FA's α-step

Evaluate the objective function of firefly X_i and X_j, $f_{obj}(X_i)$ and $f_{obj}(X_j)$;
if $f_{obj}(X^j)$ *is more optimal than* $f_{obj}(X^i)$ **then**
  Calculate the Hamming's distance $r_{ij} = d(X_i, X_j)$;
  Calculate $n_{var} = \alpha \times r_{ij}$;
  Generate randomly n_{var} pairs in D_{var}, where D_{var} is the set of different pairs of design variables between X_i and X_j;
for k *in* (n_{var} *pairs*) **do**
  Calculate the Hamming's distance $r_{ij}^k = d(x_i^k, x_j^k)$;
  Calculate $nB = \alpha \times r_{ij}^k$;
  Generate randomly nB bit-pairs in D_bit, where D_bit is the set of difference bit-pairs between bit-pairs k;
  for h *in* (nB *bit pairs*) **do**
    Flip bit of X_i;
return $X_i(new)$;

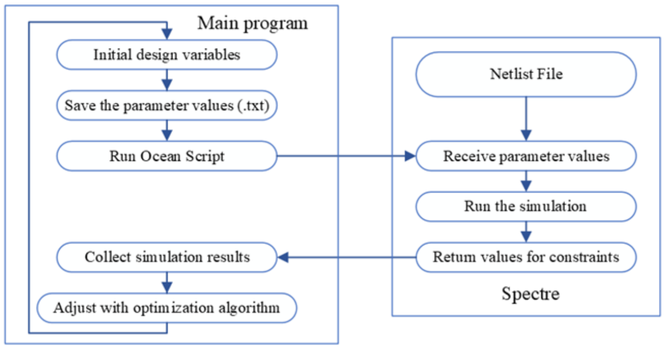

Figure 1. Python-Spectre framework.

organizes simulation results into a format compatible with optimization algorithms.
The interaction between Spectre and Python is as follows: Python generates values for design variables and passes them to Spectre via Ocean scripts. These scripts automate the circuit simulations using the provided values, and the results are then returned to Python for further processing by the optimization algorithm. This iterative process continues until the termination condition for the optimization is met. The interaction between Python and Spectre is illustrated in Figure 1.

4. General problem of analog IC sizing and case study of two-stage op-amp

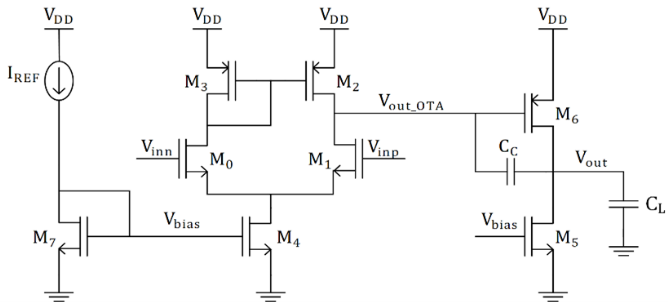

Figure 2. Two-stage op-amp.

4.1. General problem

The general problem of analog circuit component sizing is defined as follows

$$\begin{aligned} &\text{Optimize } F(X), X = (x_1, x_2, \ldots, x_d) \\ &\text{Subject to}: P_k \leq S_k, k = 1, ..., k \\ &LB_m \leq x_m \leq UB_m, m = 1, ..., N \end{aligned} \tag{13}$$

where $F(X)$ is the objective function (or cost function) of the problem, $F(\vec{x})$ represented as a vector of a objectives $f_1(X), f_2(X), ..., f_a(X)$ to be minimized or maximized depending on the problem. For $m = 1$ case, it is a single-objective optimization problem, and for $m > 1$ case, it is a multi-objective optimization problem. The constraint set is established by the circuit's technical specifications $\vec{S}$ to delineate acceptable performance conditions. Performance conditions here can include initial circuit setup constraints, MOSFET operation in saturation region, or performance parameter constraints such as voltage gain, bandwidth, etc. X is a multi-dimensional vector of optimization variables, each dimension bounded within a value range between minimum LB_n and maximum UB_n limits. A solution X is considered acceptable when all problem constraints are satisfied. This way, the optimization method's response will be an acceptable solution X that creates an optimal value for the function $f(X)$ with $f(X)$ representing the solution's performance or quality.

4.2. Case study: Two-stage op-amp

As shown in Figure 2, the two-stage op-amp with Miller compensation capacitor consists of two stages: a differential amplifier (OTA) and a common-source (CS) amplifier stage. In this figure, suppose the current running through the OTA stage is I_{REF} and the current running through the CS stage is $k \times I_{REF}$. Then, $k \times W_5/L_5 = W_4/L_4 = W_7/L_7 (L_5 = L_4 = L_7)$ (where W_i, L_i are the width and length of MOSFET M_i and k is a constant). Moreover, to ensure the symmetry of the OTA stage, $W_0/L_0 = W_1/L_1, W_2/L_2 = W_3/L_3$. To ensure normal operation of the op-amp, all MOSFETs must operate in saturation mode. Additionally, a $30mV$ margin for V_{GS} and V_{OV} voltages must be ensured for potential applications.

In the analog circuit sizing optimization problem, the FoM is often used as a quantitative measure to evaluate the system's performance, functioning similarly to the objective function. The FoM function for comparing the performance of the op-amp is given by [17, 18]:

$$\text{FoM} = \frac{\text{UGB} \times C_L}{I_{total}}, \tag{14}$$

where UGB is the unity gain-bandwidth product, C_L is the load capacitance at the output node and I_{total} is the total current of the op-amp.

Although equation (14) evaluates the efficiency of the op-amp based on its unity gain-bandwidth product as well as drivability of the load capacitance per unit current, the phase margin of the op-amp is overlooked. An op-amp with larger value for the FoM of equation (14) is considered with better quality; however, this deteriorates the op-amp's phase margin and hence stability. In other words, considering equation (14), a large FoM value is no longer meaningful if the phase margin is too small [19], indicating the unsuitability of the above-mentioned objective function.

Taking into account the role of the op-amp's phase margin, the enhanced version of the FoM, which is applied in this research, is expressed as [19]

$$\text{FoM} = \frac{\text{UGB} \times C_L}{I_{total}} \times \frac{tan(\text{PM})}{tan(PM_{REF})}, \tag{15}$$

where PM_{REF} is the reference phase margin of the op-amp and chosen with the standard value of 60° [20]. Regarding the two-stage Miller-compensated op-amp, its design is executed in the TSMC 65nm process. The op-amp's setup condition includes $V_{DD} = 1.2V, I_{REF} = 20\mu A, C_L = 1\mu F$. The input common-mode voltage for both V_{inn}, V_{inp} is $V_{in_{CM}} = 650mV$.

In order to ensure the copying ratio of the current mirror block, we need $W_5/L_5 = k \times W_4/L_4 = k \times W_7/L_7$ (where $L_5 = L_4 = L_7$). Therefore, we declare three optimization variables: W_5, W_{47}, L_{457}. Moreover, to verify the symmetry of the OTA stage, $W_0 L_0 = W_1 L_1, W_2/L_2 = W_3/L_3$, indicating four additional variables: $W_{01}, L_{01}, W_{23}, L_{23}$. Similarly, we also need W_6, L_6, C_C as our optimization variables. In total, there are ten variables for our optimization process, namely $W_{01}, L_{01}, W_{23}, L_{23}, W_5, W_{47}, L_{457}, W_6, L_6, C_C$. The specifications for the op-amp are as follows. First of all, every MOSFETs should operate in the saturation region with their margin of $30mV$ ensured for both the gate-source voltage V_{GS} and the overdrive voltage. For the sake of convenience, the mentioned condition is labeled $cond_1$. When $cond_1$ satisfies, the standards for the op-amp's performance metrics, or $cond_2$, are expressed as

$$\begin{aligned} &A_V > 50\ dB, UGB > 50\ MHz, PM > 60°, \\ &CMRR > 50\ dB, Power < 250\ \mu W, SR > 50\ V/\mu s. \end{aligned} \tag{16}$$

It should be noticed that $cond_2$ satisfies only when all design specifications of the op-amp above are met.
As a higher value of the FoM is preferable and $cond_1$ is prioritized over $cond_2$, the FoM formula should be added with the cases of $cond_1$ and $cond_2$, which can be rewritten as:

$$FoM = \begin{cases} -1, & \text{if } cond_1 = 0 \\ 0, & \text{if } cond_1 = 1, cond_2 = 0 \\ \frac{UGB \times C_L}{I_{total}} \times \frac{tan(PM)}{tan(60°)}, & \text{otherwise} \end{cases} \tag{17}$$

where the value of 0 and 1 is equivalent to that each condition passes or fails. For the cases when both $cond_1$ and $cond_2$ are not satisfied, the values of -1 and 0 are chosen for the FoM with the purpose of excluding the equivalent potential solutions. Moreover, the order of -1 and 0 corresponds to the priority of $cond_1$ and $cond_2$. So as to ensure the saturation condition to the greatest possible extent, based on experience, the bounds for optimization variables declared above are given by

$$\begin{aligned} &W_{01} \in [0.85\mu m, 4\mu m], L_{01} \in [0.23\mu m, 0.4\mu m], \\ &W_{23} \in [0.7\mu m, 1\mu m], L_{23} \in [0.06\mu m, 0.4\mu m], \\ &W_{47} \in [2\mu m, 2.8\mu m], W_5 \in [18\mu m, 23\mu m], \\ &L_{457} \in [0.1\mu m, 1\mu m], W_6 \in [16\mu m, 22\mu m], \\ &L_6 \in [0.25\mu m, 0.5\mu m], C_C \in [0.3pF, 1pF]. \end{aligned} \tag{18}$$

In conclusion, our optimization problem can be summarized as

$$\begin{aligned} &maximize\ \text{FoM}(W_{01}, L_{01}, W_{23}, L_{23}, W_5, W_{47}, L_{457}, \\ &\qquad W_6, L_6, C_C) \\ &subject\ to\ V_{DD} = 1.2\ V,\ V_{in_{CM}} = 650\ mV, \\ &\qquad I_{REF} = 20\ \mu A,\ C_L = 1\ pF, \\ &\qquad Equations\ (16), (18). \end{aligned} \tag{19}$$

Table 1. Post-optimization technical specifications of the two algorithm versions

Design parameters	Requirement	FA	BFA
$A_V(dB)$	> 50	50.04	50.8
UGB (MHz)	> 50	71.5	67.02
PM (°)	> 60	66.2	69.28
CMRR (dB)	> 50	55.75	53.48
Power (μW)	< 250	183.2	204
SR ($V/\mu s$)	> 50	62.5	59.98
PSRR(+) (dB)	< 50	53.38	52.84
PSRR(-) (dB)	> 120	134.87	134.8

Table 2. Post-optimization design variables of the two algorithm versions

Variables	FA	BFA
$(W/L)_{01}$	$2.29\mu m/0.3\mu m$	$1.82\mu m/0.28\mu m$
$(W/L)_{23}$	$1\mu m/0.11\mu m$	$0.99\mu m/0.12\mu m$
$(W/L)_4$	$2.74\mu m/0.53\mu m$	$2.38\mu m/0.64\mu m$
$(W/L)_5$	$18.09\mu m/0.53\mu m$	$18.26\mu m/0.64\mu m$
$(W/L)_6$	$19.63\mu m/0.25\mu m$	$20.82\mu m/0.25\mu m$
$(W/L)_7$	$2.74\mu m/0.53\mu m$	$2.38\mu m/0.64\mu m$
C_C	$0.3pF$	$0.31pF$
FoM	0.6133	0.6021

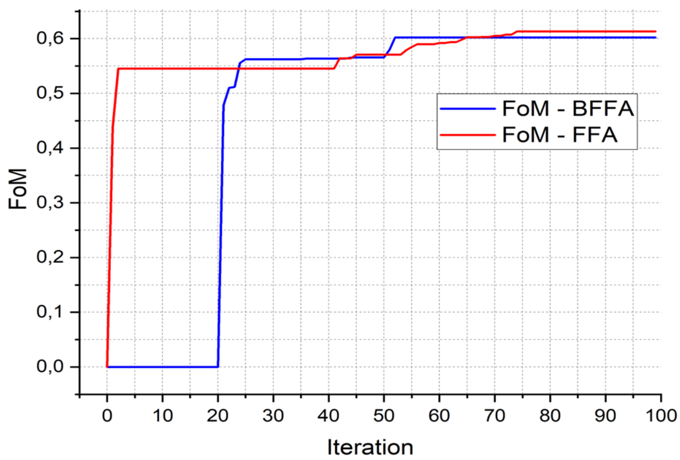

Figure 3. FoM value over 100 generations of two FA versions.

5. Results and discussion

This section demonstrates the effectiveness of the algorithm through simulation data. Simulations were performed on Cadence Virtuoso using a 65nm process technology. Each data was collected after the algorithm executed 100 iterations with specific parameters summarized in Table 1 and Table 2. The optimization program was executed with the initial population randomly selected from the given constraint ranges as per expression (16) for both versions of the FA. The specific algorithm parameters γ and α_0 were chosen in pairs for each program run, with $\gamma \in (0, 1)$ and $\alpha_0 \in (0, 1)$. Table 2 presents the best results of the two algorithm versions. The best execution result for FA yielded an FoM value of 0.6133 with configuration parameters $\gamma = 0.5$ and $\alpha_0 = 0.75$. For BFA, the best FoM result is 0.6021 with configuration parameters $\gamma = 0.1$ and $\alpha_0 = 0.75$, and the best solution's bit-flip probability was $r_{mutation} = 0.1$.

The FoM values over 100 iterations for both FA versions are illustrated in Figure 3 above. These are the best results corresponding to Table 2. Although the values for op-amp design variables may unpredictably vary across iterations, their corresponding objective function values always follow a monotonic increasing trend as shown in Figure 3. Table 3 presents a performance comparison of the two-stage op-amp with a Miller compensation capacitor developed in this study against other notable studies [7, 8, 10, 11]. The author identified only these four previous works that studied the same two-stage op-amp configuration with a Miller compensation capacitor. Consequently, Table 3 provides a comparative analysis specifically with these four studies to ensure relevance and accuracy in benchmarking performance. This focused comparison highlights the advancements made in this study relative to established designs.

The results demonstrate that both versions of the FA achieve superior FoM values, as well as higher voltage gain (A_V), phase margin (PM), and common-mode rejection ratio ($CMRR$) compared to [8]. Specifically, the A_V for FA and BFA are more than double that of [8] (50.04 dB and 50.8 dB vs. 21.6 dB), which indicates a higher current in the circuit and consequently, increased power consumption. Furthermore, the larger compensation and load capacitors (C_C and C_L) in FA and BFA results in improved unity-gain bandwidth (UGB) and slew rate (SR) values, as these parameters are inversely proportional to capacitance.

In comparison with studies [7, 10, 11], the UGB and PM parameters of FA and BFA are significantly higher, particularly the UGB, which is approximately three times better than [18], over sixty times better than [10], and substantially higher than [11]. Although the A_V gain in FA and BFA is about 10-30 dB lower, likely due to the larger MOSFET size range and higher supply voltage required for the 180nm process compared to the 65nm process, the overall FoM value remains competitive. Notably, the FoM value in [20] is higher, attributed to their use of a C_L capacitor three times larger than those used in FA versions.

Overall, FA and BFA outperform the benchmarks [7, 8, 10, 11] across most parameter values, with their FoM values being particularly noteworthy. This highlights the effectiveness of the optimization approach despite the increased complexity of working with the 65nm process compared to the 180nm process.

Examining the general performance of the algorithms, the superior results of the FA versions are evident. When compared to the Genetic Algorithm (GA), the FA excels in local search capability due to its mechanism of comparing neighboring fireflies to identify better solutions. The attraction property ensures that fireflies influence each other more when they are closer, resulting in automatic partitioning of the population into smaller groups. This leads to faster and more efficient convergence than GA. For the Particle Swarm Optimization (PSO) algorithm, PSO can be seen as a special case of the FA when the parameter $\gamma = 0$. Thus, FA not only inherits the advantages of PSO but also offers a more comprehensive and effective exploration of the search space.

In this study, traditional FA and BFA are also compared in Fig. 3 and Table 3. As illustrated in Algorithm 2 and Algorithm 3 regarding the beta-step and alpha-step, BFA employs these two dynamic parameters at binary bit level. This means that for a specific optimization variable, which is represented by a sequence of binary bits, the alpha and beta parameters of BFA affect every component of the variable. This might create in-depth and necessary changes to the optimization variables in greater detail compared to only one value for alpha and beta for each variable in the decimal representation. As a result, BFA should achieve better convergence speed compared to its FA counterpart.

6. Conclusion

This study demonstrates the effective use of the Firefly Algorithm (FA) in optimizing analog circuit design, focusing on a two-stage operational amplifier (op-amp) with a 65nm CMOS process. By developing two versions of FA—traditional and Binary Firefly Algorithm (BFA)—and optimizing them with the FoM objective function, a robust tool emerges for determining optimal component sizes under technical constraints. The results highlight FA's strong performance in balancing exploration and exploitation, with the BFA showing enhanced convergence due to its binary-level control over optimization variables. The BFA achieves faster, more precise results, outperforming benchmarks in key parameters like voltage gain, phase margin, unity-gain bandwidth, and common-mode rejection ratio. These findings underscore the potential of FA-based approaches for automating analog design, minimizing component sizing time, and achieving high-performance outcomes. This work illustrates the promise of swarm intelligence in advancing analog design automation.

Acknowledgements

The author acknowledges the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City (VNU-HCM) for this study.

References

[1] Lberni, A., Marktani, M.A., Ahaitouf, A. and Ahaitouf, A. (2024) Analog circuit sizing based on evolutionary algorithms and deep learning. *Expert Systems with Applications* **237**: 121480. doi:10.1016/j.eswa.2023.121480.

[2] Girardi, A., De-Oliveira, T., Ghissoni, S., Aguirre, P.C. and Compassi-Severo, L. (2022) A comprehensive review on automation-based sizing techniques for analog ic

Table 3. Comparison between this study and other studies

Parameter	PSO [7]	GA [10]	GA [11]	PSO [8]	FA (this study)	BFA (this study)
CMOS process (nm)	180	180	180	65	65	65
V_{DD} (V)	1.8	2.5	1.5	1.1	1.2	1.2
C_L (pF)	1	1	3	0.2	1	1
A_V (dB)	59.19	87	82.4	21.6	50.04	50.8
UGB (MHz)	20.03	1.11	9.77	169.7	71.5	67.02
PM (°)	63.53	64	60	62.4	66.2	69.28
CMRR (dB)	67.08	N.A	N.A	35.5	55.75	53.48
Power (μW)	184	51	52	89	183.2	204
SR ($V/\mu s$)	18.35	2.19	5.07	288	62.5	59.98
FoM (HzF/A)	0.3935	0.064	0.8455	0.1432	0.613	0.602

design. *Journal of Integrated Circuits and Systems* **17**: 1–14. doi:10.29292/jics.v17i3.642.

[3] Liu, B., Wang, Y., Yu, Z., Liu, L., Li, M., Wang, Z., Lu, J. *et al.* (2009) Analog circuit optimization system based on hybrid evolutionary algorithms. *Integration* **42**(2): 137–148. doi:10.1016/j.vlsi.2008.04.003.

[4] Nguyen, H.T. and Hoang, T. (2023) A novel framework of genetic algorithm and spectre to optimize delay and power consumption in designing dynamic comparators. *Electronics* **12**(16). doi:10.3390/electronics12163392.

[5] Lahoz-Beltra, R. (2016) Quantum genetic algorithms for computer scientists. *Computers* **5**(4).

[6] Liu, Z., Li, X., Jiang, J. and Wang, S. (2016) A novel improved quantum genetic algorithm for robot coalition problem. In *2016 IEEE International Conference on Information and Automation (ICIA)*: 2061–2064. doi:10.1109/ICInfA.2016.7832159.

[7] Prajapati, P.P. and Shah, M.V. (2015) Two stage cmos operational amplifier design using particle swarm optimization algorithm. In *2015 IEEE UP Section Conference on Electrical Computer and Electronics (UPCON)*: 1–5. doi:10.1109/UPCON.2015.7456700.

[8] Rashid, R. and Nambath, N. (2022) Area optimisation of two stage miller compensated op-amp in 65 nm using hybrid pso. *IEEE Transactions on Circuits and Systems II: Express Briefs* **69**(1): 199–203. doi:10.1109/TCSII.2021.3089937.

[9] Hoang, T., Quoc, T.N., Zhang, L. and Duong, T.Q. (2023) Novel methods for improved particle swarm optimization in designing the bandgap reference circuit. *IEEE Access* **11**: 139964–139978. doi:10.1109/ACCESS.2023.3341492.

[10] Barra, S., Dendouga, A., Kouda, S. and Bouguechal, N.E. (2012) Multi-objective genetic algorithm optimization of cmos operational amplifiers. In *2012 24th International Conference on Microelectronics (ICM)*: 1–4. doi:10.1109/ICM.2012.6471451.

[11] Jafari, A., Zekri, M., Sadri, S. and Mallahzadeh, A. (2010) Design of analog integrated circuits by using genetic algorithm. In *2010 Second International Conference on Computer Engineering and Applications*, **1**: 578–581. doi:10.1109/ICCEA.2010.118.

[12] Hoang, T., Thang, N. and Phan-Tran-Minh, H. (2024) Novel ga-ocean framework for automatically designing the charge-pump circuit. *IEEJ Transactions on Electrical and Electronic Engineering* **19**. doi:10.1002/tee.24129.

[13] Wolpert, D. and Macready, W. (1997) No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* **1**(1): 67–82. doi:10.1109/4235.585893.

[14] Fister, I., Fister, I., Yang, X.S. and Brest, J. (2013) A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation* **13**: 34–46. doi:https://doi.org/10.1016/j.swevo.2013.06.001.

[15] Yang, X.S. (2010) *Nature-Inspired Metaheuristic Algorithms* (Luniver Press).

[16] Yang, X. (2018) *Optimization Techniques and Applications with Examples* (Wiley).

[17] Li, Y., Wang, Y., Li, Y., Zhou, R. and Lin, Z. (2020) An artificial neural network assisted optimization system for analog design space exploration. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **39**(10): 2640–2653. doi:10.1109/TCAD.2019.2961322.

[18] Budak, A., Bhansali, P., Liu, B., Sun, N., Pan, D. and Kashyap, C. (2021) Dnn-opt: An rl inspired optimization for analog circuit sizing using deep neural networks. In *58th Design Automation Conference (DAC 2021)*. doi:10.48550/arXiv.2110.00211.

[19] Yoshizawa, H. (2015) An improved figure-of-merit equation for op-amp evaluation. *IEICE Electron. Express* **12**: 20150533.

[20] Razavi, B. (2017) *Design of Analog CMOS Integrated Circuits* (New York: McGraw-Hill), 2nd ed.