IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 21, 2020, accepted March 12, 2020, date of publication March 17, 2020, date of current version March 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2981535

Automatic Structure Generation and Parameter Optimization for CMOS Voltage Reference Circuit

JINTAO LI[1], YANHAN ZENG[1], (Member, IEEE), JIAQI WANG[2], JINRUI LIAO[2], JINGCI YANG[1], AND HONG-ZHOU TAN[2], (Senior Member, IEEE)

[1]School of Physics and Electronic Engineering, Guangzhou University, Guangzhou 510006, China
[2]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China

Corresponding author: Yanhan Zeng (yanhanzeng@gzhu.edu.cn)

This work was supported in part by the Science and Technology Project of Guangzhou under Grant 201804010464, in part by the National Natural Science Foundation of China under Grant 61704037, in part by the Natural Science Foundation of Guangdong Province, China, under Grant 2017A030310655, and in part by the Foundation for the Public Welfare Research and Capacity Building in the Science and Technology Project of Guangdong Province under Grant 2016A010101006.

ABSTRACT An automatic design system for the voltage reference circuit is presented in this paper. The circuit coding method based on tracking coding is improved and the instruction set is expanded, thus the circuit structure with MOS transistors can be automatically generated. The differential evolution method is used to optimize the reference circuit. To improve the convergence speed and optimization effect, a new constrained solution and a fast non-dominated differential evolution algorithm based on weights are proposed. At the same time, HashMap is used as the cache to reduce the optimization time. Based on the proposed automatic structure generation and parameter optimization, two CMOS voltage reference circuits are automatically implemented in a 0.18 μm standard CMOS process. Simulation results show that the line sensitivity, temperature coefficient, power and chip area are improved by 85.7%, 50%,92.7% and 59.5%, respectively, compared with the artificial solution.

INDEX TERMS Automatic design for analog integrated circuits, CMOS voltage reference, differential evolution, trail coding, Pareto comparison.

I. INTRODUCTION

Analog circuits play an indispensable role in electronic system design because the world is analog in essence. Actually even digital design need analog modules so as to connect with the outside circuits. Circuit design, especially for the difficult analog integrated circuit (IC) design such as voltage reference circuit with complex structures and parameters evolution, should consider many factors such as power consumption, bandwidth, stability and manufacturing area at the same time. The circuit designer find the optimal performance by precise calculation and parameters adjustment, which consumes time and efforts and the automatic design would help the designer to find the optimum point easily. Thus the automatic design for analog IC is needed.

Differ from the mature electronic design automatization (EDA) technology in the digital IC, there are only a few reported work on the parameters optimization in some analog IC such as bandgap voltage reference, error amplifier and traditional LDO [3]–[8]. Besides, the traditional circuit simulation software, such as Analog Design Environment (ADE) GXL provided by Cadence [1] and WiCkeD provided by MunEDA [2], also have the function of circuit parameter optimization. Moreover, no automatic structure designs for the analog IC are proposed so far. Besides, problems do exist in these approaches, such as: (1) The performances constraint of analog circuits are complex and difficult to solve. Neglecting the constraint causes slow optimization speed, over-fitting or under-fitting, and even the failure of practical optimization [3], [4]; (2) The same weight is given to each performance index, resulting in over or under optimization [4], [7]; (3) The chip area has not yet been considered [4]–[6], [8].

In this paper, the reference circuit is optimized from two aspects: circuit structure and circuit parameters. In terms of circuit structure, the circuit coding method is improved and its instruction set is expanded, so that the circuit structure with MOS transistors can be automatically generated by tracking coding. For the optimization of circuit parameters, in which the constraints are simplified by OOR function, and a fast non-dominated sorting algorithm based on weight

The associate editor coordinating the review of this manuscript and approving it for publication was Dušan Grujić.

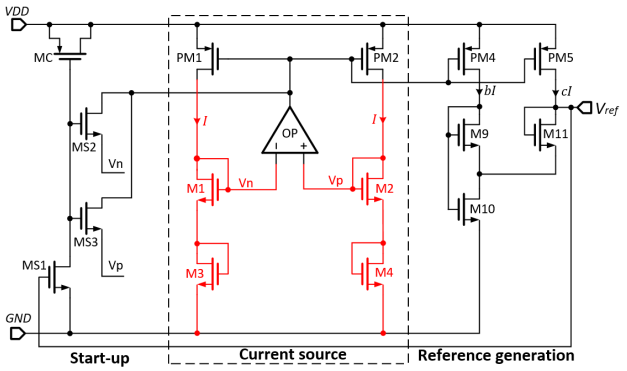


FIGURE 1. The CMOS voltage reference circuit without special devices in [9].

ranking is proposed. Furthermore, parallel computing is used in the optimization system to add caching and compression information, which reduces the time-consuming of the optimization algorithm.

The organizational structure of this paper is as follows. Firstly, the automatic design system and the optimization algorithm are described in the section II and section III. Then, we discuss the simulation results in Section IV. Finally, the conclusion is given in the section V.

II. PROPOSED AUTOMATIC DESIGN SYSTEM

A. OPERATING SCHEME

The circuit of a CMOS voltage reference is illustrated in Fig.1 [9]. It consists of a start-up circuit, a current source that generates a current I almost independent of supply voltage, and an output circuit that generates the reference voltage. Due to the multiple loops, voltage reference needs a start-up circuit (formed by MS1 MS4) to settle at the proper operating point and ensure that the stable state can always be achieved. A differential-input amplifier is used to keep M1 and M2 with a same gate voltage and improve the performance of power supply rejection ratio (PSRR) and line sensitivity (LS). Eq. (1) and Eq. (2) show the expressions of the bias current and the reference voltage.

$$I = \frac{1}{2}\left(\eta^2 \frac{\sqrt{K_1 K_2}}{\sqrt{K_1}-\sqrt{K_2}} \ln\frac{K_4}{K_3}\right)^2 \mu C_{ox} V_T^2. \tag{1}$$

$$V_{ref} = V_{TH}(T_0) - \kappa(T - T_0) + \eta V_T \ln\left[\frac{1}{2}\left(\eta^2 \frac{\sqrt{K_1 K_2}}{\sqrt{K_1}-\sqrt{K_2}} \ln\frac{K_4}{K_3}\right)^2 \frac{(1+\beta)K_5}{\beta(\eta-1)K_6 K_7}\right]. \tag{2}$$

where K is the W/L ratio of the transistor, μ is the carrier mobility, C_{ox} is the gate oxide capacitance per unit area, η is the sub-threshold slope factor, $V_T (= k_B T/q)$ is the thermal voltage, k_B is the Boltzmann constant, T is the absolute temperature, and q is the electron charge. V_{TH} and V_{GS} are the threshold and gate-source voltage of the MOS transistor, respectively. κ is the temperature coefficient (TC) of V_{TH}. Setting $\partial V_{REF}/\partial T = 0$, an output reference voltage independent of temperature can be obtained without the use of

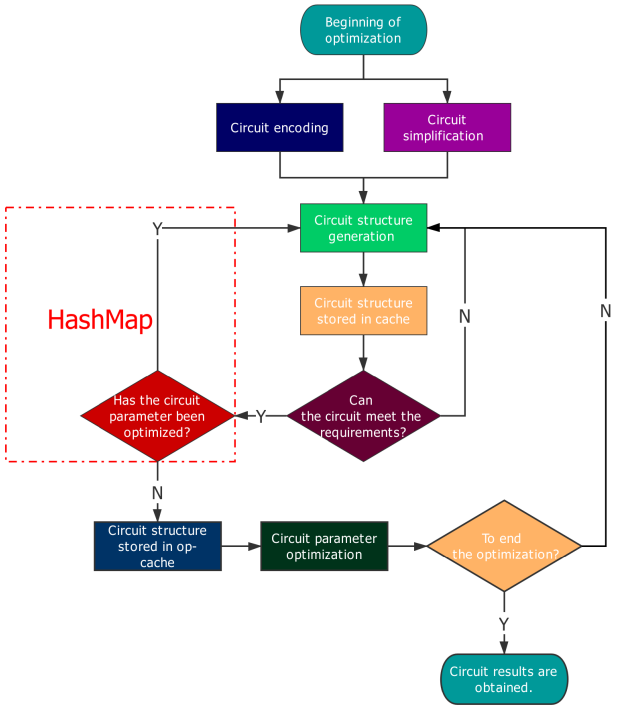


FIGURE 2. The work-flow chart.

resistors and any other special devices such as thick gate oxides MOS transistors with higher threshold voltage in the case of:

$$\frac{1}{2}\left(\eta^2 \frac{\sqrt{K_1 K_2}}{\sqrt{K_1}-\sqrt{K_2}} \ln\frac{K_4}{K_3}\right)^2 \frac{(1+\beta)K_5}{\beta(\eta-1)K_6 K_7} = \exp\frac{\kappa q}{\eta k_B} \tag{3}$$

And PSRR is obtained as:

$$PSRR \approx \frac{\frac{1}{g_{m6}} + \frac{1}{g_{m7}}}{\frac{1}{g_{m6}} + \frac{1}{g_{m7}} + r_{o11} + A_{op} g_{mll} r_{o11}\left(\frac{1}{g_{m2}} + \frac{1}{g_{m4}} - \frac{1}{g_{m1}} - \frac{1}{g_{m3}}\right)} \tag{4}$$

From the above expressions, it is difficult to design the optimal parameters by artificial work and get the best performances of TC, LS and PSRR with power and time constrains, because of the complex expressions to be calculated, the lots of variables and the many trade-offs that appear intrinsically. Not to mention the irregular circuit structure. In this paper, an automatic design system is proposed to not only optimize the parameter but also design the new structure of the voltage reference circuit based on the different evolutionary algorithm. The evolutionary system is divided into two parts: circuit structures generation and parameter optimization. Circuit structures generation is used to create lots of new circuit structures, of which meet the performance target set are selected to enter into parameter optimization [11]. The selected circuit structure is fixed during the parameter optimization.

The work-flow chart is shown in Fig. 2. Firstly, the circuit is coded and simplified, then the circuit structure is generated and stored in the cache. Finally, the circuit parameters are

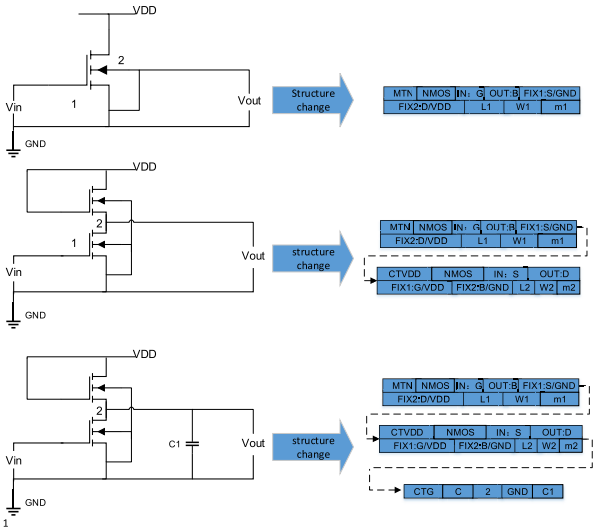


FIGURE 3. The examples of circuit encoding.

optimized and the optimized structure is stored in the op-cache to get the final circuit results. It is worth mentioning that if the circuit has been optimized, whether it is available or not, the system will skip this step and go back to generate a new circuit structure.

B. CIRCUIT CODING

1) DEVICE CODING

The critical problem is to encode the four-ports device, MOS transistor. In this paper, combined with trail encoding, new definition and model are brought out.

The unified model for a four-ports device [12] has one input terminal and three output terminals. In order to simplify the evolutionary model, the four-ports device is converted to two ports by fixing two of the ports randomly. When a four ports device such as MOS transistor is added to the existing circuit, the input and output terminals are randomly selected, and the fixed points and the respective connections are also randomly implemented. By these means the four-ports device is encoded by the trail encoding.

2) CODING INSTRUCTION

The instruction set for the two-ports and three-ports devices specified by the original trail encoding is incapable of satisfying some instructions of the MOS transistor, such as the connecting to a bias current or power supply. Thus the original instruction set should be expanded by adding cast-to-VSS (CTVSS), cast-to-VDD (CTVDD), cast-to-bias (CTB) and cast-to-self (CTS). At this point, nine instructions of MTN, CTP, CTG, CTI, CTO, CTVSS, CTVDD, CTB and CTS are implemented, where MTN represents adding a new device and creating a new node, CTP means adding a new device and connecting the output terminal of this device with the previous node, as shown in Table.1.

The examples of circuit encoding are shown in Fig. 3. By the proposed device model and coding instructions,

TABLE 1. The coding of a four-ports device.

Encoded information fragments	Meaning	Containing instruction sets
INS	Instruction Information	None
Tpye	Components and Component Types	MTN;CTP;CTG
IN	Device Input Node Selection	CTI;CTS
OUT	Device Output Node Selection	CTO;CTS
FIX	Connection information of fixed nodes	CTVSS;CTVDD CTB
L	Channel length of MOS transistor	None
W	Channel Width of MOS transistor	None
m	Number of MOS transistors in parallel	None

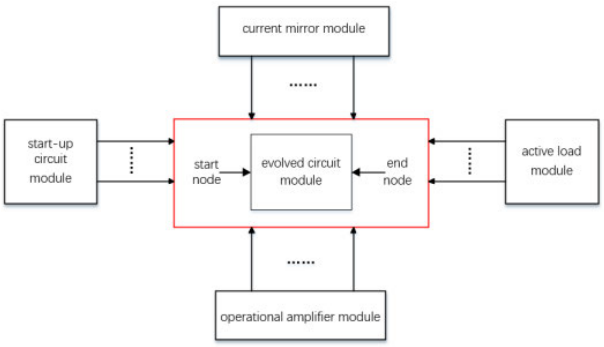


FIGURE 4. The initial model of circuit.

the MOS transistors are efficiently coded and the circuit coding is further obtained.

C. CIRCUIT SIMPLIFICATION

The CMOS reference voltage with temperature compensation can be fundamentally generated by the linear superposition of threshold voltage and thermal voltage, which is implemented by a current source I and a diode-connected MOSFET working in saturation region or the output circuit working in subthreshold region as shown in Fig 1. And the bias current I should satisfy the following expression [9]:

$$I = \alpha \mu C_{ox} V_T^2. \tag{5}$$

where α is a coefficient. How to get a current with the same form as Eq. (5) is the major challenge for the CMOS voltage reference design. The current mirror, the start-up circuit, the output circuit and the amplifier all play a relatively constant role [13], which can be regarded as the public modules of the voltage reference. Keeping these public modules fixed in advance and externally exposing their interfaces would significantly reduce the search space of the evolutionary algorithm. With this strategy, the initial circuit model to be evolved as illustrated in Fig. 4.

It should be noted that since the initial circuit changes, the instruction set further increases, including four CTx which means adding a new device and connecting the output terminal of this device with the current mirror, the start-up circuit, the output circuit and the amplifier, respectively.

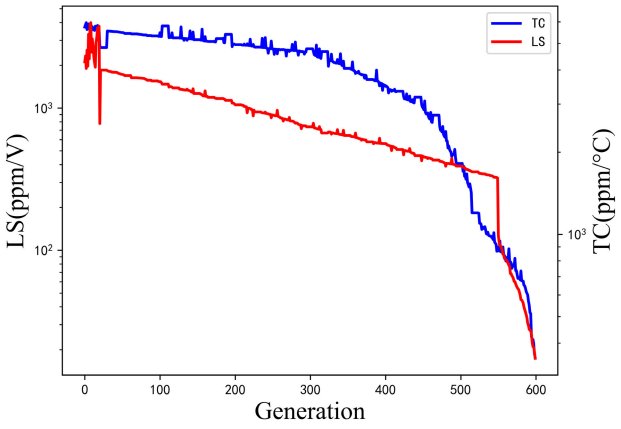


FIGURE 5. The optimization processes of TC and LS.

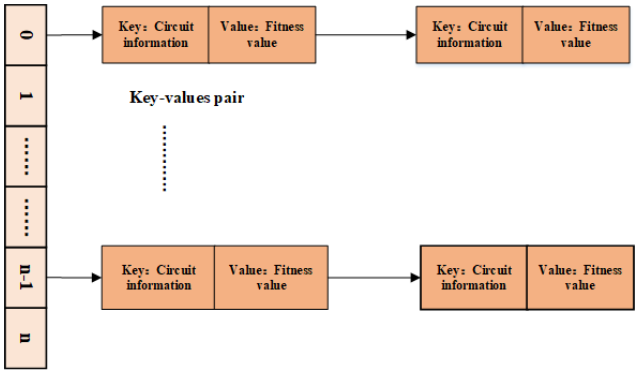


FIGURE 6. Key-value pairs of HashMap in cache.

D. STRUCTURE SCREENING

The optimization algorithm can not guarantee efficiently searching the structure and parameters at the same time, because of the different preferences caused by the different genetic operators. In general, the efficient optimization of one part leads to inefficiency of the other part. To ensure the optimization efficiency of both parts, the 'first-screening, post-tuning' is proposed to separate these two parts.

'First-screening' refers to the preliminary screening of the circuit structures. In the preprocessing stage, the evolutionary algorithm focuses on finding the circuit structure that satisfies the certain constraints and weakens the optimization of the transistor parameters. The moderately loose screening criteria gets the suitable circuits in a short time. 'Post-tuning' stands for the parameters optimization of the selected circuit individuals. In the tuning stage, the structure is fixed, and only the transistor parameters are used as optimization variables by the improved differential evolution algorithm.

At the screening stage, the optimization processes of LS and TC are shown in the Fig. 5. Both the optimized LS and TC are in a relatively loose range.

E. HashMap

It is necessary to calculate the fitness value of each individual both in the structure and parameter evolutionary algorithm, and the fitness value must be simulated by SPICE simulator. However, one simulation takes about 0.5s and the same individual in different generation uses multiple times, which wastes the time. In this paper, HashMap is proposed as the cache, in which the time complexity of both adding and searching element [15] are $O(1)$. As shown in Fig. 6, the key-value pairs of HashMap store circuit information and fitness values, respectively. When an individual needs to calculate the fitness value, it firstly searches in the cache. If the individual simulated before, it gets its fitness value in the cache directly instead of simulating again. For Op-cache, HashMap is used to detect whether the circuit structure has been optimized. If it has been optimized, the circuit structure will not be optimized repeatedly to save time.

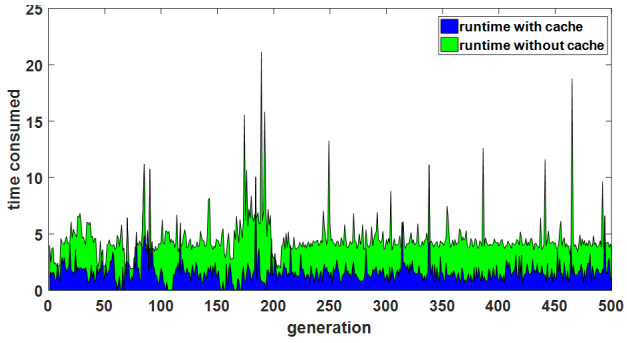


FIGURE 7. Time comparison.

Fig. 7 shows the time comparison in the optimization process. One generation consumes an average of 1.69s with cache while consumes about 4.49s without cache. The evolutionary process can save about 62.3% time by the HashMap.

III. OPTIMIZATION ALGORITHM

Taking the above measurements, the automatic design system is built and has the ability to generate the circuit structure and optimize the circuit parameter. The differential evolution algorithm is used as the optimization algorithm and the flow chart is shown in Fig. 8, which is divided into three parts.

1) Initialize the population, generate circuit parameters randomly and then simplify the circuit constraints through OOR function.
2) Sort the generated circuits by using the improved fast non-dominated sorting algorithm based on weight ranking.
3) Use the improved differential evolution algorithm to optimize circuit parameters by selecting operation, crossover operation and mutation operation.

It is worth mentioning that the NSGA-II algorithm is used in the circuit structure optimization and the specific flow is the same as parameter optimization. OOR function and weight-based fast non-dominated sorting algorithm are also used.

A. CONSTRAINT SIMPLIFICATION

The traditional optimization algorithm for the analog circuit, which is a multi-objective optimization, is low-efficiency due to the performance compromise. For example, it should get the best performances of TC, LS and PSRR with power and

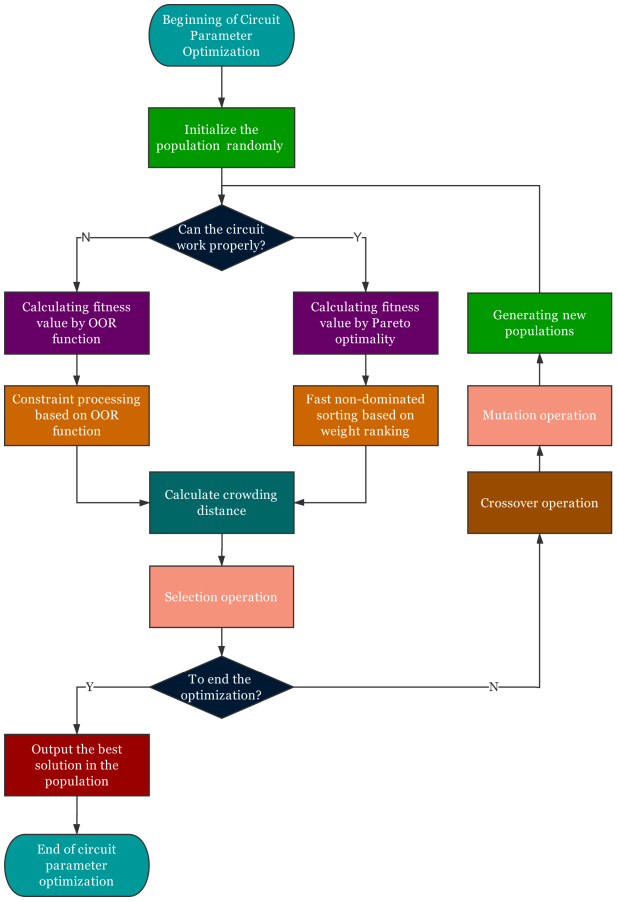


FIGURE 8. The flow chart of proposed differential evolution optimization.

time constrains for the voltage reference circuit. An multi-objective optimization is composed of multiple objective functions, related equations and inequality constraints. It can be mathematically described by:

$$\begin{aligned} min \quad & \{f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})\}, \boldsymbol{x} = \{x_{0,n}, x_{1,n}, x_{2,n}, \ldots, x_{M,n}\}^T \\ s.t. \quad & g_i(\boldsymbol{x}) \geq 0 \\ & h_i(\boldsymbol{x}) = 0 \end{aligned} \tag{6}$$

where $f_i(\boldsymbol{x}), \{i = 1, 2, 3, \ldots, m\}$ are the objective functions, $g_i(\boldsymbol{x})$ and $h_i(\boldsymbol{x})$ are inequality and equality constraint functions, respectively. $X = \{\boldsymbol{x} | \boldsymbol{x} \in R^n, g_i(x) \geq 0, h_j(x) = 0, i = 1, 2, \ldots, p, j = 1, 2, \ldots, q\}$ are called the feasible region of the above formula. The inequality constraints can be modelled by OOR*(out-of-range)* function and denoted as $u_i(x)$ [17]–[19]. Solving constraints takes precedence over solution optimization and OOR function is improved to solve constraints in proposed optimization algorithm. The OOR functions only return positive values and are proportional to constraint violations. Thus Eq.(6) is rewritten as:

$$\begin{aligned} min \quad & \{f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})\} \\ s.t. \quad & u_i(\boldsymbol{x}) = 0 \end{aligned} \tag{7}$$

It is the feasible solution if $\boldsymbol{x}$ satisfies the constraints. For the infeasible solutions, the constraint violation value

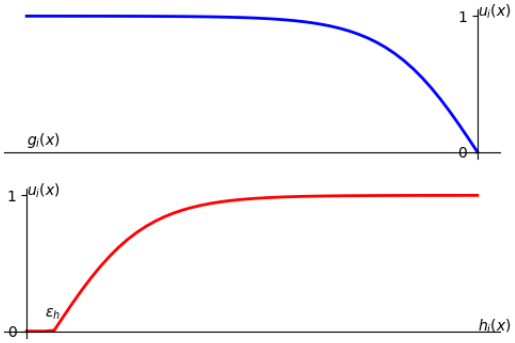


FIGURE 9. OOR function $u_i(x)$ corresponding to equality constraint $h_i(x)$ and inequality constraint $g_i(x)$.

quantitatively describes the degree violating the constraints. It is easy to convert $g_i(\boldsymbol{x}), h_i(\boldsymbol{x})$ to $u_i(\boldsymbol{x})$ by ramp function. However, considering the actual circuit situation, logical regression is used instead of ramp function, which compresses the constraint range to $(0, 1)$, and then multiplies $u_i(\boldsymbol{x})$ by penalty value N_{punish} to calculate the true constraint violation degree. Although the value of N_{punish} does not affect the results, it is not recommended to be too small.

$$u_i(\boldsymbol{x}) = \begin{cases} 0 & \textbf{if} \quad g_i(\boldsymbol{x}) \geq 0 \\ \dfrac{1 - e^{g_i(\boldsymbol{x})}}{1 + e^{g_i(\boldsymbol{x})}} & otherwise \end{cases} \tag{8}$$

$$u_i(\boldsymbol{x}) = \begin{cases} 0 & \textbf{if} \quad |h_i(\boldsymbol{x})| \leq \epsilon_h \\ \dfrac{1 - e^{-|h_i(\boldsymbol{x})| + \epsilon_h}}{1 + e^{-|h_i(\boldsymbol{x})| + \epsilon_h}} & otherwise \end{cases} \tag{9}$$

In Eq.(9) ϵ_h is a small but not very small number. It is necessary to search for ϵ_h in small gaps near equal constraints. Compared with the fixed penalty value, OOR function guides the solution set from infeasible to feasible solution better.

B. MULTI-OBJECTIVE OPTIMIZATION

For the voltage reference circuit, when the output voltage is within a certain value, the smaller TC, LS and current mean better performances. The fitness value is a combination of such indicators. The SPICE simulator is used to obtain the fitness and circuit optimization focuses on comprehensive performance. There are two issues: (1) The solution must firstly ensure that the circuit can work normally; (2) Different weights should be given to each optimization index to avoid the over-fitting or under-fitting.

The proposed algorithm calculates the fitness $f(x_{i,n}^g)$ based on Pareto optimum domain [23] and adds fast non-dominated sorting based on weight ranking to achieve multi-objective optimization. In the solution space, m individuals are randomly generated, each of which is uniform and n-dimensional. The solution should not only converge to the approximate Pareto optimum domain, but also be uniformly and sparsely distributed in the Pareto optimum domain. For the m objective components $f_i(\boldsymbol{x})$ and two decision variables $x_{a,n}$ and $x_{b,n}$, $f_i(x_{a,n})$ is better than $f_i(x_{b,n})$ in the following conditions [22]:

1) $f_i(x_{a,n})$ is feasible while $f_i(x_{b,n})$ is infeasible;

Algorithm 1 Select Solution

Require: Individuals in a population $\boldsymbol{x}$
Ensure: $L_{win}(\boldsymbol{x}), L_{loss}(\boldsymbol{x})$
$N_{const}(\boldsymbol{x}), f(\boldsymbol{x}) \leftarrow 0$
$L_{win}, L_{loss} = [\,]$
for $x_{a,n}$ in $\boldsymbol{x}$ **do**
  for OOR constraint i in $x_{a,n}$ **do**
    if $u_i(x_{a,n}) > 0$ **then**
      $N_{const}(x_{a,n}) + +$
      $f(x_{a,n}) - = N_c + u_i(x_{a,n}) \times N_{weight}[i] \times N_{punish}$
    end if
  end for
  if $N_{const}(x_{a,n}) > 0$ **then**
    $L_{loss}.append(x_{a,n})$
  else
    $L_{win}.append(x_{a,n})$
  end if
end for
$Sort\_reverse(L_{loss}(\boldsymbol{x}))$ by $f(\boldsymbol{x})$
return $L_{win}(\boldsymbol{x}), L_{loss}(\boldsymbol{x})$

2) Both $f_i(x_{a.n})$ and $f_i(x_{b,n})$ are not feasible, but $u_i(x_{a.n}) < u_i(x_{b.n})$;
3) Both $f_i(x_{a.n})$ and $f_i(x_{b.n})$ are feasible, but $\{\forall i \in \{1, 2, \ldots, n\}, f_i(x_{a.n}) \geq f_i(x_{b.n})\} \wedge \{\exists j \in \{1, 2, \ldots, n\}, f_j(x_{a.n}) \geq f_j(x_{b.n})\}$.

The proposed algorithm changes the selection method of the feasible solution, and incorporate the calculation of fitness value and OOR function, which are discussed in details as follows.

1) SELECT SOLUTION

First of all, OOR function $u_i(\boldsymbol{x})$ should be equal to 0 to ensure proper working of the circuit. Then Pareto comparison is built among the individual parameters and finally sorted by the degree of dominance or the fitness value. For the record, the whole population are passed rather than two individuals in the select-solution.

As shown in Algorithm 1, i represents the selected constraints, which dose not enter the non-dominated sorted algorithm if the individual fails to resolve the constraints. Besides, $N_{const}(\boldsymbol{x})$ is the number of constraints violated by an individual. $N_{weight}(\boldsymbol{x})$ is a list, which stores the indicator weights. N_c is the coefficient to distinguish the number of constraints violated, which needs to be larger than the sum of weight coefficients ($N_c \geq \sum_0^i N_{weight}$ and $N_c \geq 1$).

2) NON-DOMINATED SORTING

The evaluation of circuit depends on some important indicators, thus the weight ranking is used in the fast non-dominated algorithm [23], [24]. By this way, it can preferentially optimize the more important indicators, which reduces over/under-fitting and the evolutionary time. More importantly, it retains the circuit parameters that meet the requirement, so as to obtain a better circuit performance.

Algorithm 2 Non-dominatedSorting

Require: $L_{win}(\boldsymbol{x})$
Ensure: $\boldsymbol{P}, f(\boldsymbol{x})$
$\boldsymbol{x} \leftarrow L_{win}(\boldsymbol{x})$
$f(\boldsymbol{x}) \leftarrow 0$
$\boldsymbol{P} = [\,]$
for $x_{p,n}$ in $\boldsymbol{x}$ **do**
  $Sp = [\,]$
  $Np = 0$
  for $x_{q,n}$ in $\boldsymbol{x}$ **do**
    if $x_{p,n} > x_{q,n}$ **then**
      $f(x_{p,n}) + = N_{weight}$
      $Sp.append(x_{q,n})$
    else if $x_{p,n} < x_{q,n}$ **then**
      $Np + = 1$
    end if
  end for
  if $Np == 0$ **then**
    $x_{q,n}\_rank = 1$
    $P_1.append(x_{p,n})$
  end if
end for
$\boldsymbol{P}.append(P_1)$
$i = 1$
while $P[i] \neq \emptyset$ **do**
  $i + = 1$
  $P_i = [\,]$
  for $x_{q,n}$ in Sp **do**
    $Nq - = 1$
    if $nq == 0$ **then**
      $x_{q,n}\_rank = i$
      $P_i.append(x_{q,n})$
    end if
  end for
  $\boldsymbol{P}.append(P_i)$
end while
return $\boldsymbol{P}, f(\boldsymbol{x})$

As shown in Algorithm 2, the number of individuals occupied is multiplied by exponential weights to increase the fitness value and get a clear direction of evolution. In the actual process, compared with that of the same weight, it is better to firstly increase the weight of LS and then TC. Besides, the fitness value is multiplied by the weight ranking. Once one individual outperforms another, the fitness value increases and the increment equals to the weight.

3) CROWDING DISTANCE

In order to rank the individuals which have the same ranking, the crowding distance between individuals is measured by a simple method, as given by Eq. (10).

$$Cd(x_{i,n}^g) = \frac{f(x_{i-1,n}^g) - f(x_{i+1,n}^g)}{f_{max} - f_{min}} \tag{10}$$

Algorithm 3 Crowding Distance

Require: $P(\boldsymbol{x}), f(\boldsymbol{x})$
Ensure: $\boldsymbol{Cd}$
$\boldsymbol{Cd} \leftarrow 0$
for $P_{i,m}(\boldsymbol{x})$ in $P(\boldsymbol{x})$ **do**
  if $len(P_{i,m}(\boldsymbol{x})) == 1$ **then**
    continue
  else
    $K_{i,m}(\boldsymbol{x}) \leftarrow Sort\_reverse(P_{i,m}(\boldsymbol{x}))$ by $f_i(\boldsymbol{x})$
    $Cd(K_{i,0}), Cd(K_{i,m}) \leftarrow$ **INF**
    for $x_{a,n}$ in $K_{i,m}(\boldsymbol{x})$, *except first and last* **do**
      $$Cd(K_i(x_{a,n})) = \frac{f(K_i(x_{a-1,n})) - f(K_i(x_{a+1,n}))}{f_{max}(K_i) - f_{min}(K_i)}$$
    end for
  end if
end for
return $\boldsymbol{Cd}$

where $f(x_{i,n}^g)$ is the fitness value of individual, $f(x_{i-1,n}^g)$ and $f(x_{i+1,n}^g)$ are the fitness values of individuals ranked above and below $x_{i,n}^g$, respectively. Besides, f_{min} and f_{max} are the fitness of the worst and the best individuals in the current population, respectively. The algorithm gives preference to the individual with large crowding distance [25], thus the results is more evenly distributed in the target space to improve the diversity of population.

C. THE IMPROVEMENTS OF THE DIFFERENTIAL EVOLUTION ALGORITHM

The improvements include the adaptive parameters of mutation operator F and crossover operator C_r, and the selection operation, which is more suited to the weight-based fast non-dominated sorting algorithm and also accelerate the convergence speed.

1) MUTATION OPERATION

In the $g-th$ evolutionary algebra, five individuals $x_{k_1,n}^g, x_{k_2,n}^g, x_{k_3,n}^g, x_{k_4,n}^g, x_{k_5,n}^g, (k_1 \neq k_2 \neq k_3 \neq k_4 \neq k_5)$ are randomly selected from the population. The mutated individuals is given by:

$$v_{i,n}^{g+1} = x_{best,n}^g + F(x_{k2,n}^g + x_{k3,n}^g - x_{k4,n}^g - x_{k5,n}^g) \quad (11)$$

where F is the mutation factor, which is generally set to 0.5. In order to achieve good convergence effect, the differential weight is adaptively adjusted and the five chosen individuals are ranked by Algorithm 2 and get $x_{best}, x_{better}, x_{mid}, x_{worse}, x_{worst}$ corresponding to the fitness values $f_{best}, f_{better}, f_{mid}, f_{worse}, f_{worst}$ [27]. F is finally improved by:

$$F = F_l + (F_u - F_l)\left|\frac{f_{mid} - f_{best}}{f_{worst} - f_{best}}\right| \quad (12)$$

where F_u and F_l are the upper and lower limits of F, respectively.

Algorithm 4 Population Variation

Require: $\boldsymbol{x}$
Ensure: $v_{i,n}$
for i in $range(len(\boldsymbol{x}))$ **do**
  $k_1, k_2, k_3, k_4, k_5 \leftarrow random.randint \in [0, len(\boldsymbol{x})], (k_1 \neq k_2 \neq k_3 \neq k_4 \neq k_5)$
  $x_{best}, x_{better}, x_{mid}, x_{worse}, x_{worst} \leftarrow Sort\_reverse(x_{k1,n}, x_{k2,n}, x_{k3,n}, x_{k4,n}x_{k5,n})$ by $f(x)$
  $$F = F_l + (F_u - F_l)\left|\frac{f_{mid} - f_{best}}{f_{worst} - f_{best}}\right|$$
  $$v_{i,n}^{g+1} = x_{best} + F(x_{better} + x_{mid} - x_{worse} - x_{worst})$$
end for
return $v_{i,n}$

Algorithm 5 Crossover Operation

Require: $\boldsymbol{v}$
Ensure: $\boldsymbol{U}$
$\bar{f} \leftarrow average(f(\boldsymbol{v}))$
for i in $range(len(\boldsymbol{v}))$ **do**
  $r_i \leftarrow random(0, 1)$
  if $f_i \geq \bar{f}$ **then**
    $$Cr = Cr_l + (Cr_u - Cr_l)\frac{f(v_{i,n}) - f_{min}}{f_{max} - f_{min}}$$
  else
    $Cr = Cr_l$
  end if
  for j in n **do**
    if $Cr \geq r_i$ **then**
      $U_{i,j} = v_{i,j}$
    else
      $U_{i,j} = x_{i,j}$
    end if
  end for
end for
return $\boldsymbol{U}$

In proposed differential evolution algorithm, parameters in the search space are expressed as vectors, and the genetic operators operate on bit strings. The vector expression is more effective and more suitable for the parameters optimization because the direct differential operation of vectors can better retain the trend of circuit parameters. Similarly, the crossover is also a branch exchange of vector-based chromosomes or segments.

2) CROSSOVER OPERATION

The crossover operation performs between the individual $x_{i,n}^g$ and mutated individual $v_{i,n}^{g+1}$ in the g-generation population. The rate or probability is controlled by the cross parameter $Cr \in [0, 1]$ in the binomial or exponential way. The binomial crossover, which is used in this paper, intersects each component by generating random numbers $r_i \in [0, 1]$ that obey

Algorithm 6 Selection Operation

Require: $\boldsymbol{U}, \boldsymbol{x}$
Ensure: $\boldsymbol{U}$
$\boldsymbol{t} \leftarrow extend(\boldsymbol{U}, \boldsymbol{x})$
$\boldsymbol{t}^{win}, \boldsymbol{t}^{loss} \leftarrow SelectSolution(\boldsymbol{t})$
if $\boldsymbol{t}^{win} = \emptyset$ **then**
  for i in $range(len(\boldsymbol{x}))$ **do**
    $\boldsymbol{x}^{new}.append(t_i^{loss})$
  end for
else
  $\boldsymbol{P} \leftarrow Non-dominatedSorting(\boldsymbol{t}^{win})$
  $\boldsymbol{Cd} \leftarrow CrowdingDistance(P(\boldsymbol{t}^{win}))$
  for j in $range(len(P(\boldsymbol{t}^{win})))$ **do**
    $P_j^{new} = Sort\_reserve(P_j)$ by Cd_j
    for k in P_j^{new} **do**
      if $len(\boldsymbol{x}^{new}) < \boldsymbol{x}$ **then**
        $\boldsymbol{x}^{new}.append(P_{j,k})$
      end if
    end for
  end for
  if $len(\boldsymbol{x}^{new}) < len(\boldsymbol{x})$ **then**
    $num = 0$
    while $len(\boldsymbol{x}^{new}) < len(\boldsymbol{x})$ **do**
      $\boldsymbol{x}^{new}.append(t_{num}^{loss})$
      $num++$
    end while
  end if
end if

uniform distribution [28].

$$U_{i,j}^{g+1} = \begin{cases} v_{i,j}^{g+1} & \textbf{if } Cr \geq r_i \\ x_{i,j}^{g} & otherwise \end{cases} \tag{13}$$

Similarly, Cr is also adaptively adjusted as:

$$Cr = \begin{cases} Cr_l + (Cr_u - Cr_l)\dfrac{f(v_{i,n}^g) - f_{min}}{f_{max} - f_{min}} & \textbf{if } f_i \geq \overline{f} \\ Cr_l & \textbf{if } f_i < \overline{f} \end{cases} \tag{14}$$

where f_{min} and f_{max} are the fitness of the worst and the best individuals in the current population, respectively. $\overline{f}$ is the average fitness. Cr_l and Cr_u are the lower and upper limits of Cr, respectively.

3) SELECTION OPERATION

As shown in algorithm 6, the better individual is selected as the new individual by the greedy selection strategy which is given as:

$$x_{i,n}^{g+1} = \begin{cases} U_{i,n}^{g+1}, & \textbf{if } u_{i,n}^{g+1} \textbf{ dominate } x_{i,n}^{g} \\ x_{i,n}^{g} & otherwise \end{cases} \tag{15}$$

IV. SIMULATION RESULTS AND DISCUSSION

To estimate the performances of the proposed automatic design system, some simulations have been implemented using a 0.18 μm standard CMOS process.

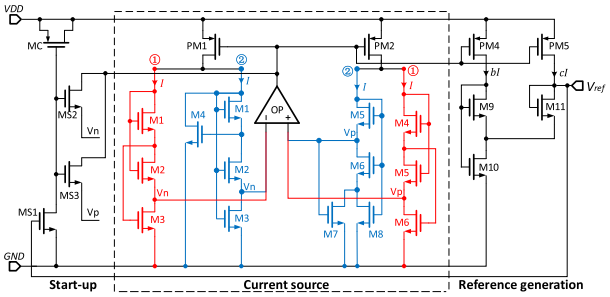


FIGURE 10. The generated circuits.

A. CIRCUIT STRUCTURES AND PERFORMANCES

With the reference generation circuit shown in Fig.1 and bias current in the line with Eq.(5), the reference voltage V_{ref} can be obtained as follows.

$$\begin{aligned} V_{ref} &= V_{GS10} - {V_{GS9}}^* + {V_{GS11}}^* \\ &= V_{TH} + \eta V_T \ln \frac{K_9(b+c)c\alpha}{K_{10}K_{11}b(\eta - 1)} \end{aligned} \tag{16}$$

V_{ref} can be independent of temperature because V_{TH} has a negative temperature coefficient while V_T has the positive temperature coefficient.

Two reference circuits are generated by the proposed automatic structure generation and parameter optimization algorithm, as shown in Fig.10. It shows two voltage reference circuits drawn in two different colors, in which the generated current source circuit consists of six and eight transistors. The current expressions of the two circuits are derived as follows:

1) THE FIRST CIRCUIT

$$V_{GS3} + {V_{GS1}}^* - {V_{GS2}}^* = V_{GS6} + {V_{GS4}}^* - {V_{GS5}}^* \tag{17}$$

Simplifying the Eq.(17), the I can be expressed as:

$$I = \frac{1}{2}(\ln \frac{K_1K_5}{K_2K_4})^2 \frac{K_3K_6}{(\sqrt{K_6} - \sqrt{K_3})^2} \mu C_{ox} {V_T}^2 \tag{18}$$

2) THE SECOND CIRCUIT

$$V_{GS4} + {V_{GS1}}^* - {V_{GS2}}^* = V_{GS7} \tag{19}$$

Simplifying the Eq.(17) [29], [30], the I can be expressed as:

$$I = \frac{1}{2}(\ln \frac{K_2}{K_1})^2 \frac{K_4K_7}{(\sqrt{K_7} - \sqrt{K_4})^2} \mu C_{ox} {V_T}^2 \tag{20}$$

According to Eq.(18) and (20), it is obvious that all the two circuits general the current with the same form of Eq.(5) and all the circuits can normally work to obtain the reference voltage without using the special devices.

The results comparison of all the two circuits and other designs are shown in Table. (2) and (3). Reference [9] proposed a CMOS reference circuit without using the special devices in artificial design. Reference [10] is artificial but using the special devices, while [7] is a parameter optimization of circuit in [10] by other algorithm. Compared with [9], LS of the two circuits are improved by 85.7% at least.

TABLE 2. The results comparison of all the two circuits and other designs.

Circuit	1	2	[9]	[10]	[7]
Year	2019	2019	2013	2018	2018
Process (μm)	0.18	0.18	0.18	0.18	0.18
Supply Voltage (V)	0.85-3	0.85-3	0.85-2.5	0.75-3.5	0.75-3.5
Current (nA)	15.6	10	214	12.8	12.8
V_{ref} (mV)	541	467	633	319	321
TC (ppm/°C)	6.5	9.7	19.4	7.2	7.6
LS (ppm/V)	12.3	34.7	242	752	554
Temperature (°C)	-20 80	-20 80	-20 80	-20 80	-20 80
PSRR (dB) 100Hz/1MHz	-81 -62	-71 -48	-76 -30	-78 -56	-79 -56
Sum of $W * L$ (μm^2)	728	786	1941	1231	1158

TABLE 3. Transistor sizes of all the two circuits and other designs.

Circuit	1	2	[9]	[10]	[7]
MC	0.22/0.18	0.22/0.18	-	20/5	36.4/4.5
MS1	0.22/0.18	0.22/0.18	3/10	1.2/1	1.5/1.0
MS2	0.22/0.18	0.22/0.18	1.2/1	3/10	3/9.8
MS3	0.22/0.18	0.22/0.18	1.2/1	1/5	1/5
MS4	-	-	20/5	-	-
PM0	-	-	-	0.22/60	0.2/60.4
PM1	24.3/0.18	22.1/0.24	-	10/20	10/20.5
PM2	24.3/0.18	22.1/0.24	-	6/20	6/20.0
PM3	0.67/0.18	1.67/0.24	-	1/20	1/20
PM4	8.92/0.18	30/0.24	-	-	-
MA1	0.22/10	0.22/0.95	3/10	-	-
MA2	7.7/5.19	1.07/9.58	3/10	-	-
MA3	3.62/7.8	17.18/10	0.22/60	-	-
MA4	28.48/0.18	1.45/8.97	10/5	-	-
MA5	1.26/9.88	30/8.47	10/5	-	-
M1	(2.26/6.64)×10	15/7.31	1.65/10	4/10	4/10
M2	3.53/0.18	15.75/0.18	0.5/10	4/10	4/9.9
M3	(2.58/9.52)×4	7.12/1.6	(30/5)×2	55/10	55/10
M4	(8.08/2.04)×15	(3.7/4.86)×3	(30/5)×5	0.22/66	0.22/6.0
M5	(0.54/0.18)×15	(0.22/3.8)×8	(20/1)×17	1/10	1.58/7.0
M6	(0.68/7.16)×6	8.91/1.67	5/10	0.22/65	0.25/64.8
M7	-	(0.22/10)×16	3/10	-	-
M8	-	8.57/0.18	20/5	-	-
M9	(5.26/1.94)×8	(0.22/10)×20	20/5	-	-
M10	(0.33/8.12)×4	3.13/10	5/5	-	-
M11	(0.42/4.96)×5	30/0.24	20/5	-	-

Compared with [10] and [7], LS is improved by 95.4% and 93.7% at least, respectively. Compared the circuit without special device in [9], the proposed circuit achieves a 50% improvement in TC with an 92.7% reduction in current consumption. What's more, the chip areas are indirectly compared by calculating the sum of the products $W * L$ of the MOS transistors. All the two circuits achieve the smaller chip area than other designs. Compared with [9], [10] and [7], the chip area are reduced by 59.5%, 36.1% and 32.1% at least, respectively.

It's also worth mentioning that [7] is actually an parameter optimization of [10], in which but LS is obviously underfitted due to the too many constraints. Besides, the circuit parameters by manual design in [7] are needed as the initial population. The algorithm in this paper reduces the underfitting and make the circuit performance more balanced, which is verified by the simulation results. Such performance optimization is obtained under that the initial population randomly generates instead of manually designing in [7].

TABLE 4. Comparison of the optimization capabilities with other simulation software.

Optimization Capabilities	This Paper	WiCkeD [1]	ADE GXL [2]
Optimizing Way	Full-Auto	Semi-Auto	Semi-Auto
Size Optimization	✓	✓	✓
Multi-MOS Optimization	✓	✓	✓
Multi-Obj optimization	✓	✓	×
Feasibility Check	✓	✓	×
Solve Constraint	✓	✓	×
Chip area optimization	✓	✓	×
Init-Size Production	✓	×	×
Structure generation	✓	×	×
Obj-Optimization Weight	✓	×	×
User Interface	×	✓	✓
Post-Layout Effects	×	✓	×

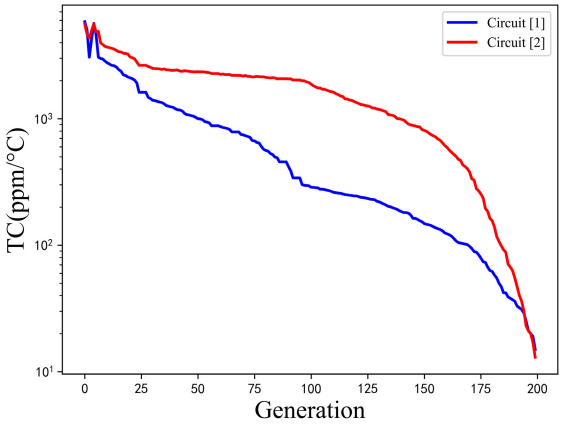


FIGURE 11. TC by fast non-dominated sorting based on weight ranking.

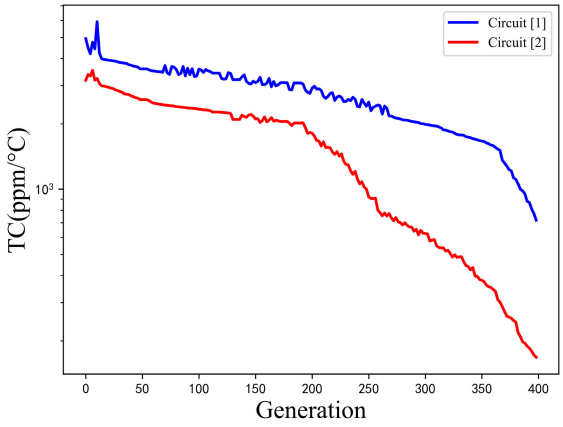


FIGURE 12. TC by fast non-dominated sorting.

Table.4 shows the comparison of the optimization capabilities with other simulation software. Thought the proposed work is an idea or an algorithm system to verify the automatic analog circuit design, which is not and far from to be a commercial software, it shows the superiority on the full-auto optimization not only in the parameters but also in the structure generation.

B. ALGORITHM EFFICIENCY

The optimization processes of TC and LS are shown in Fig. 11 and Fig. 13. During 0th to 20th generations, the solution

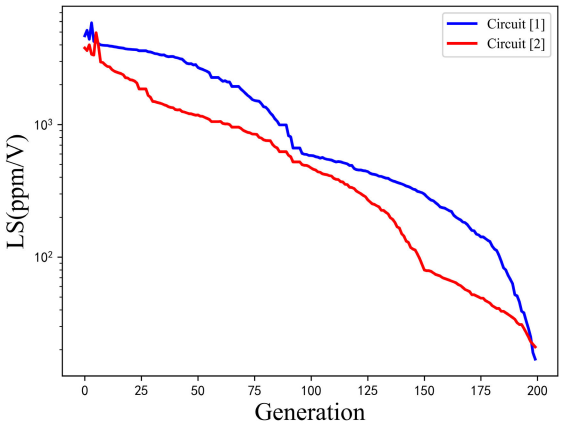


FIGURE 13. LS by fast non-dominated sorting based on weight ranking.

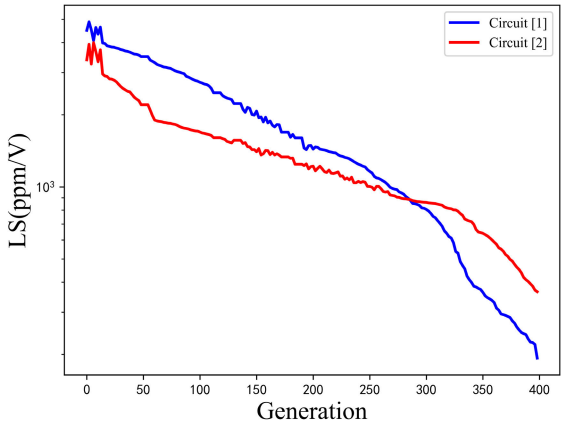


FIGURE 14. LS by fast non-dominated sorting.

resolves the constraint problem and looks chaotic. The solution is rapidly optimized during 50th-250th generations, in which the magnitude of the optimization is proportional to the weight. The algorithm finally converges at about 200th generation. As shown in Fig. 11, Fig. 12, Fig. 13 and Fig. 14, proposed fast non-dominated sorting algorithm based on weight ranking converges faster than the traditional fast non-dominated sorting algorithm. Besides, by combining the improved differential evolution algorithm with the weight-based fast non-dominated algorithm, the fitting speed is accelerated, the number of iterations in the whole optimization process is reduced by 50%, and the performance of circuit parameters is better, and the under-fitting or over-fitting is reduced. In the whole optimization process, we can control the optimization process and the final result by assigning different weights to the non-dominated fast sorting algorithm according to the needs of different circuits. The proposed automatic circuit design system is completed in the server with two E5-2609-V4 (Primary frequency 1.7 Ghz, 8-core 8-thread) CPU and 64G memory. The optimization time depends on the performance of the computer and the number of parameters to be optimized, which is about two hours. It is noteworthy that the results are not affected by hardware conditions and the optimization time.

V. CONCLUSION

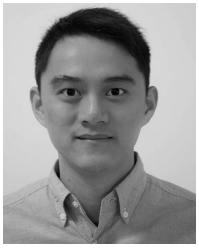
In this paper, an automatic circuit design system with parameter optimization is proposed to achieve high-performance CMOS voltage reference circuits. The operating principle, system implementation, algorithm improvements and simulation results have been described in details. Particular attention has been paid to improving the circuit coding to automatically generate structure and the differential evolution algorithm to efficiently optimize parameter. Finally two CMOS voltage circuits are implemented in a 0.18 μm standard CMOS process. Simulation results show that proposed system obtains better performances, compared with the artificial solutions. As the superiority of automation, good performance and fast-design compatibility, this design system and algorithm have a wide range of application in analog IC design, not only in CMOS voltage reference.

REFERENCES

[1] Berlin, Germany. *MunEDA GmbH Inselkammerstrae Unterhaching*. Accessed: Feb. 19, 2020. [Online]. Available: https://www.muneda.com/solutions_optimization_overview.php
[2] America. *Cadence Design Systems*. Accessed: Feb. 19, 2020 [Online]. Available: https://www.cadence.com/en_US/home/tools/ic-package-design-and-analysis.html
[3] D. Nam, Y. Deuk Seo, L.-J. Park, C. Hoon Park, and B. Kim, "Parameter optimization of an on-chip voltage reference circuit using evolutionary programming," *IEEE Trans. Evol. Comput.*, vol. 5, no. 4, pp. 414–421, Aug. 2001.
[4] J. Lopez-Arredondo, E. Tlelo-Cuautle, and R. Trejo-Guerra, "Optimizing an LDO voltage regulator by evolutionary algorithms consideringtolerances of the circuit elements," in *Proc. 16th Latin-Amer. Test Symp. (LATS)*, Mar. 2015, pp. 1–5.
[5] J. B. Chinchore and R. A. Thakker, "Design of low dropout regulator using artificial bee colony evolutionary algorithm," in *Proc. Int. Conf. Circuits, Power Comput. Technol. (ICCPCT)*, Mar. 2015, pp. 1–8.
[6] O. B. Kchaou, A. Garbaya, M. Kotti, P. Pereira, M. Fakhfakh, and M. H. Fino, "Sensitivity aware NSGA-II based Pareto front generation for the optimal sizing of analog circuits," *Integration*, vol. 55, pp. 220–226, Sep. 2016.
[7] H. Huang, Y. Zeng, J. Liao, R. Chen, and H.-Z. Tan, "Performance optimization for the CMOS voltage reference circuit based on NSGA-II," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Oct. 2018, pp. 82–85.
[8] D. M. Colombo, G. I. Wirth, and C. Fayomi, "Design methodology using inversion coefficient for low-voltage low-power CMOS voltage reference," in *Proc. 23rd Symp. Integr. Circuits Syst. Design (SBCCI)*, 2010, pp. 7–17.
[9] Y. Zeng, Y. Huang, Y. Luo, and H.-Z. Tan, "An ultra-low-power CMOS voltage reference generator based on body bias technique," *Microelectron. J.*, vol. 44, no. 12, pp. 1145–1153, Dec. 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0026269213001675
[10] Y. Zeng, S. Huang, X. Zhang, and H.-Z. Tan, "A 12.8 nA and 7.2 ppm/°C CMOS voltage reference without amplifier," *IEICE Electron. Express*, vol. 15, no. 3, 2018, Art. no. 20171220.
[11] D. Yu and J. He, "A new coding method inspired by the generation process of graph for analog circuits automatic evolutionary design," in *Proc. 9th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2016, pp. 909–914.
[12] C. Mattiussi and D. Floreano, "Analog genetic encoding for the evolution of circuits and networks," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 596–607, Oct. 2007.
[13] J. D. Lohn and S. P. Colombano, "A circuit representation technique for automated circuit design," *IEEE Trans. Evol. Comput.*, vol. 3, no. 3, pp. 205–219, Sep. 1999.
[14] B. Liu, F. V. Fernández, G. Gielen, R. Castro-López, and E. Roca, "A memetic approach to the automatic design of high-performance analog integrated circuits," *ACM Trans. Design Autom. Electron. Syst.*, vol. 14, no. 3, pp. 1–24, May 2009.

[15] L. Bruno de Sa and A. Mesquita, "Synthesis of voltage follower with only CMOS transistors using evolutionary methods," in *Proc. 2nd NASA/ESA Conf. Adapt. Hardw. Syst. (AHS)*, Aug. 2007, pp. 478–485.
[16] J. B. Grimbleby, "Automatic analogue circuit synthesis using genetic algorithms," *IEE Proc.-Circuits, Devices Syst.*, vol. 147, no. 6, pp. 319–323, Dec. 2000.
[17] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using Reference-Point-Based nondominated sorting approach, Part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.
[18] M. R. Bonyadi and Z. Michalewicz, *Locating Potentially Disjoint Feasible Regions of a Search Space with a Particle Swarm Optimizer*. New Delhi, India: Springer, 2015, pp. 205–230.
[19] D. M. Pedroso, M. R. Bonyadi, and M. Gallagher, "Parallel evolutionary algorithm for single and multi-objective optimisation: Differential evolution and constraints handling," *Appl. Soft Comput.*, vol. 61, pp. 995–1012, Dec. 2017.
[20] T. Ray, K. Tai, and K. C. Seow, "Multiobjective design optimization by anevolutionary algorithm," *Eng. Optim.* vol. 33, no. 4, pp. 399–424, 2001.
[21] C. C. Coello, G. B. Lamont, and D. A. van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA, USA: Springer, 2007, pp. 59–99.
[22] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[23] S. Kukkonen and K. Deb, "A fast and effective method for pruning of non-dominated solutions in many-objective problems," *Parallel Problem Solving Nature*, vol. 4193, pp. 553–562, Jun. 2006.
[24] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.
[25] F.-A. Fortin and M. Parizeau, "Revisiting the NSGA-II crowding-distance computation," in *Proc. 15th Annu. Conf. Genetic Evol. Comput. Conf. (GECCO)*, 2013, pp. 623–630.
[26] C. M. Fonseca and P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms. II. application example," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 28, no. 1, pp. 38–47, Jan. 1998.
[27] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Inf. Sci.*, vol. 329, pp. 329–345, Feb. 2016.
[28] M. Yang, C. Li, Z. Cai, and J. Guan, "Differential evolution with auto-enhanced population diversity," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 302–315, Feb. 2015.
[29] B. Razavi, *Design of Analog CMOS Integrated Circuits*. New York, NY, USA: McGraw-Hill, 2005, pp. 17–27.
[30] Y. Zeng, X. Zhang, and H.-Z. Tan, "A 86 nA and sub-1 V CMOS voltage referencewithout resistors and special devices," in *Proc. IFIP/IEEE Int. Conf. Very Large Scale Integr. (VLSI-SoC)*, Oct. 2017, pp. 1–5.

JINTAO LI was born in Guangdong, China, in 1998. He is currently pursuing the degree in optoelectronic information science and engineering with Guangzhou University. His current research interests include automatic design of analog integrated circuits and evolution system design.



YANHAN ZENG (Member, IEEE) received the B.S. degree in electronic engineering from the South China University of Technology, Guangzhou, China, in 2010, and the Ph.D. degree in electronic engineering from Sun Yat-sen University, Guangzhou, in 2015. Since 2018, he has been an Associate Professor with the School of Physics and Electronic Engineering, Guangzhou University, China. His current research interests include power management IC, energy harvesting, and RFID systems.

JIAQI WANG was born in Hubei, China, in 1994. He received the master's degree in electronic and communication engineering from Sun Yat-sen University. His research interests include automatic design of analog integrated circuits and evolution system design.

JINRUI LIAO was born in Guangdong, China, in 1996. He received the B.S. degree in Internet of Things from Guangzhou University, Guangzhou, China, in 2019. He is currently pursuing the M.S. degree with Sun Yat-sen University, Guangzhou. His current research interests include power management IC and the Internet of Things.

JINGCI YANG was born in Guangdong, China, in 1998. She is currently pursuing the degree in electronic information science and technology with Guangzhou University. Her current research interests include power management IC and integrated circuits.

HONG-ZHOU TAN (Senior Member, IEEE) received the Ph.D. degree in electronics engineering from the City University of Hong Kong, Hong Kong, and the South China University of Technology, Guangzhou, China, in 1998. He was with several universities and IT companies in Hong Kong, Singapore, and Canada, from 1998 to 2004. Since 2004, he has been a Full Professor with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou. He is currently the Director of the National Engineering Laboratory for the Internet of Things and the SYSU-CMU Shunde International Joint Research Institute. His current research interests include embedded IC and systems, the Internet of Things, and energy harvesting.