AUTOMATIC CMOS ANALOG CIRCUIT DESIGN USING ABC (ARTIFICIAL BEE COLONY) ALGORITHM

[1] MR. J.R.CHAUHAN, 2 PROF. R.V.BUTANI

[1]M.E [E.C] Student, DEPT. OF ELECTRONICS AND COMMUNICATION, MARWADI EDUCATION FOUDATION GROUP OF INSTITUTIONS, RAJKOT, GUJARAT

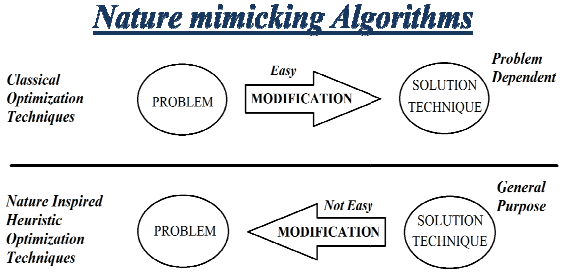
[2] ASST. PROF AT DEPT. OF ELECTRONICS AND COMMUNICATION, MARWADI EDUCATION FOUDATION GROUP OF INSTITUTIONS, RAJKOT, GUJARAT

Jayesh.chauhan7890@gmail.com, ravi_butani@yahoo.com

Abstract: *There is arising complexity of the MOSFET model due to the downscaling of the MOSFET during past few years. So it is difficult to capture the large no model parameter accurately .Due to downscaling Complexity of the circuit and also process of effect, supply voltage, temperature (PVT) varies. To solve this problem we can use the Nature Inspired algorithm. Circuit designing problem treat as non linear numerical optimizing problem. So that Nature inspired algorithm GA, PSO, ANT can be use to solve problem. Now here ABC algorithm can be use as solve non linear circuit design problem because its provide better search space and also explore parameter from extraction and find out the better result compare to other.*

Keywords: ABC (artificial bee colony), PSO (particle swarm optimization), benchmark function, MLPSO

Introduction: There is a trend to solve complex optimizing problem through the nature heuristic algorithm because the inefficiency of the classical optimization algorithm to solve non linear optimizing problem. Generally classical optimization algorithm can handle simple problem, so we have to convert our problem in manner that the classical optimization algorithm can solve it. So for that we have to take many assumptions in that specific problem parameter. But it is not easy to take assumption in every situation. So overcome this problem general purpose algorithm can be use. It is the mainly base on the nature inspired. So it is also called nature inspired algorithm or swarm optimization algorithm. This algorithms has better solution compare to the classical optimization algorithm[1].



Now the circuit design is also the non linear numerical optimization problem. So for that we can use this nature inspired optimization algorithm to solve our problem and extract circuit parameter.

Examples of different nature inspired algorithm:

1. GA: In the computer science field of artificial intelligence, a genetic algorithm (GA) is a search heuristic that mimics the process of natural evolution. This heuristic is routinely used to generate useful solutions to optimization and search problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover.

2. ANT: Ant colony optimization is a class of optimization algorithms modeled on the actions of an ant colony. ACO methods are useful in problems that need to find paths to goals. Artificial 'ants' simulation agents locate optimal solutions by moving through a parameter space representing all possible solutions. Real ants lay down pheromones directing each other to resources while exploring their environment. The simulated 'ants' similarly record their positions and the quality of their solutions, so that in later simulation iterations more ants locate better solutions. One variation on this approach is the bees algorithm, which is more analogous to the foraging patterns of the honey bee [4].

3. PSO: Particle swarm optimization (PSO) is a global optimization algorithm for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Hypotheses are plotted in this space and seeded with an initial velocity, as well as a communication channel between the particles. Particles then move through the solution space, and are evaluated according to some fitness criterion after each time step. Over time, particles are accelerated towards those particles within their communication grouping which have better fitness values. The main advantage of such an approach over other global minimization strategies such as simulated annealing is that the large numbers of members that make up the particle swarm make the technique impressively resilient to the problem of local minima[6][1][2].

4. ABC: In ABC, a population based algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of the employed bees is as same as number of solutions in the population. 1st a randomly distributed initial food source positions is generated. Then after initialization, the population is subjected to repeat the cycles of the search processes of the employed, onlooker, and scout bees. An employed bee produces a modification on the source position in her memory and discovers a new food source. Provided that the nectar amount of the new one is higher than that of the previous source, the bee memorizes the new source position and forgets previous. so she keeps the position of the one in her memory. After all employed bees complete the search process, they share the position information of the sources with the onlookers on the dance area. Each onlooker evaluates the nectar information taken from all employed bees and then chooses a food source depending on the nectar amounts of sources. As in the case of the employed bee, she produces a modification on the source position in her memory and checks its nectar amount. Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the previous one. The sources empty are determined and new sources are randomly produced to be replaced with the empty ones by artificial scouts [1].

Artificial Bee Colony optimization algorithm

ABC Algorithm is one of the member of Swarm Intelligence Algorithm for global optimization based on Foraging behavior of Honey Bee. ABC Algorithm was proposed by Karaboga in 2005 [1]. ABC is found Promising for Constrained Nonlinear Optimization Problems(CNOP).

Artificial Bee colony contain three group of Honey Bees

1. Employed Bees(Exploration)
2. Onlooker Bees(Exploitation)
3. Scout Bees(Exploration)

Half of colony contain Employed Bees & other half of colony contain Onlooker Bees. One food randomly source per one Employed Bee. Employed Bee of abandoned food source become Scout Bee.A bee waiting on the dance area for making decision to choose afood source is named an onlooker and a bee going to the food source visited by itself previously is named an employed bee. A bee carrying out random search is named a scout. In the ABC algorithm. First half of the colony consists of employed bees and the second half constitutes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source is exhausted by the employed and onlooker bees becomes a scout. The main steps of the ABC algorithm are given below[1]

• Initialize.
• REPEAT.
(a) Place the employed bees on the food sources in the memory;
(b) Place the onlooker bees on the food sources in the
memory;
(c) Send the scouts to the search area for discovering new food sources.
• UNTIL (requirements are met).

In the ABC algorithm, each cycle of the search consists of three steps: sending the employed bees onto the food sources and then measuring their nectar amounts; selecting of the food sources by the onlookers after sharing the information of employed bees and determining the nectar amount of the foods, determining the scout bees and then sending them onto possible food sources. At the initialization stage, a set of food source positions are randomly selected by the bees and their nectar amounts are determined. Then, these bees come into the hive and share the nectar information of the sources with the bees waiting on the dance area within the hive.

At the second stage, after sharing the information, every employed bee goes to the food source area visited by herself at the previous cycle since that food source exists in her memory, and then take a new food source by means of visual information in the neighborhood of the present one.

At the third stage, an onlooker prefers a food source area depending on the nectar information distributed by the employed bees on the dance area. As the nectar amount of a food source increases, the probability with which that food source is taken by an onlooker more. Hence, the dance of

employed bees carrying higher nectar recruits the onlookers for the food source areas with higher nectar amount. After arriving at the selected area, she chooses a new food source in the neighborhood of the one in the memory depending on visual information. Visual information is based on the comparison of food source positions. When the nectar of a food source is empty by the bees, a new food source is randomly found by a scout bee and replaced with the empty one. In our model, at each cycle at most one scout goes outside for searching a new food source and the number of employed and onlooker bees were equal.

In the ABC algorithm, the position of a food source represents a possible solution of the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population. At the first step, the ABC generates a randomly distributed initial population P(G = 0) of SN solutions (food source positions), where SN denotes the size of population. Each solution (food source) xi (i = 1, 2, . . . , SN) is a D-dimensional vector. Here, D is the number of optimization parameters. After initialization, the population of the positions (solutions) is subjected to repeated cycles, (C = 1, 2, . . . ,Cmax), of the search processes of the employed bees, the onlooker bees and scout bees. An artificial employed or onlooker bee probabilistically produces a modification on the position (solution) in her memory for finding a new food source and tests the nectar amount (fitness value) of the new source (new solution). In case of real bees, the production of new food sources is based on a comparison process of food sources in a region depending on the information gathered, visually, by the bee. In our model, the production of a new food source position is also based on a comparison process of food source positions. However, in the model, the artificial bees do not use any information in comparison. They randomly select a food source position and produce a modification on the one existing in their memory as described[1]. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. After all employed bees complete the search process, they share the nectar information of the food sources (solutions) and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. As in the case of the employed bee, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one. An onlooker bee chooses a food source depending on the probability value associated with that food source, pi, calculated by the following expression

$$P = \frac{fit(i)}{\sum_1^{sn} fit(i)} \qquad -1$$

Where fit(i) is the fitness value of the solution i evaluated by its employed bee, which is proportional to the nectar amount of the food source in the position(i… SN) is the number of food sources which is equal to the number of employed bees (BN). In this way, the employed bees exchange their information with the onlookers. In order to produce a candidate food position from the old one, the ABC uses the following expression

$$v_{i,j}^{g} = x_{i,j}^{g-1} + \emptyset_j * (x_{i,j}^{g-1} - x_{k,j}^{g-1}) \qquad \textbf{-2}$$

where k {1, 2, . . . , BN} and j {1, 2, . . . ,D} are randomly chosen indexes. Although k is determined randomly, it has to be different from i. $\emptyset_j$ is a random number between [−1, 1]. It controls the production of a neighbor food source position around and the modification represents the comparison of the neighbor food positions visually by the bee. Equation shows that as the difference between the parameters of the and decreases, the perturbation on the position decreases, too. Thus, as the search approaches to the optimum solution in the search space, the step length is adaptively reduced. If a parameter produced by this operation exceeds its predetermined limit, the parameter can be set to an acceptable value. In this work, the value of the parameter exceeding its limit is set to its limit value. The food source whose nectar is abandoned by the bees is replaced with a new food source by the scouts. In the ABC algorithm this is simulated by randomly producing a position and replacing it with the abandoned one. In the ABC algorithm, if a position cannot be improved further through a predetermined number of cycles called limit then that food source is assumed to be abandoned. After each candidate source position is produced and then evaluated by the artificial bee, its performance is compared with that of. If the new food has equal or better nectar than the old source, it is replaced with the old one in the memory. Otherwise, the old one is retained. In other words, a greedy selection mechanism is employed as the selection operation between the old and the current food sources.

Flow of ABC algorithm:[6]

BEGIN
Initialize the set of the food source $x_{i,}$ i=1……SN
Evaluate each $x_{i,}$ i=1……SN
g=1
REPEAT
FOR i=1 **TO** SN
Generate v_i^g with x_i^{g-1} by use eq (2)
Evaluate v_i^g
IF v_i^g is better **than** x_i^{g-1}
$x_i^g = v_i^g$
ELSE
$x_i^g = x_i^{g-1}$
END IF
END FOR
FOR i-1 **TO** SN
Selected based on fitness proportional selection food source x_i^g
Generate v_i^g with x_i^g by use eq (2)
Evaluate v_i^g
IF v_i^g is better **than** x_i^g
$x_i^g = v_i^g$
END IF
END FOR
Generate new food source at random for those limit to be improved has been reached keep the best solution so far
g=g+1
UNTIL
g=MCN
End

BENCHMARK FUNCTIONS

A function is multimodal if it has two or more local optima. A function of variables is separable if it can be rewritten as a sum of functions of just one variable[2]. The reparability is closely related to the concept of epitasis or interrelation among the variables of the function[6][2].
The problem is even more difficult if the function is also multimodal. The search process must be able to avoid the regions around local minima in order to approximate, as far as possible, to the global optimum. The most complex case appears when the local optima are randomly distributed in the search space.

The dimensionality of the search space is another important factor in the complexity of the problem. In order to compare the performance of the proposed ABC with PSO, and ML-PSO seven classical benchmark functions used in this work as given in table 2.

Table 1: parameter of benchmark function

Fun.	Dim. (n)	Initialization range	Search space	Goal (f_0)
F_1	30	$(50, 100)^n$	$(-100,100)^n$	10^{-9}
F_2	30	$(15, 30)^n$	$(-100, 100)^n$	10^{-6}
F_3	30	$(2.56, 5.12)^n$	$(-10, 10)^n$	10^{-9}
F_4	30	$(300, 600)^n$	$(-600, 600)^n$	10^{-9}
F_5	30	$(15, 32)^n$	$(-32, 32)^n$	10^{-9}
F_6	02	$(15, 30)^n$	$(-100, 100)^n$	0.9940069
F_7	02	$(15, 30)^n$	$(-100, 100)^n$	0.999999

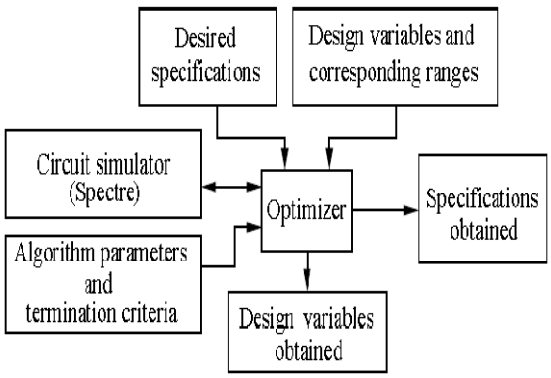
Table 2 Benchmark function

Function, Globally optimum solution, Type
F_1, Sphere, Minimize, Origin, Unimodal $$f_1(x) = \sum_{i=1}^{n} x_i^2$$
F_2, Rosenbrock, Minimize, $[1, 1, \ldots, 1]^n$,Unimodal $$f_2(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$$
F_3, Rastrigin, Minimize, Origin, Multimodal $$f_3 = 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) \right]$$
F_4, Griewank, Minimize, Origin, Multimodal $$f_4(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$$
F5, Ackley, Minimize, Origin, Multimodal $$f_5(x) = 20 + \exp(1) - 20\exp\left(\frac{-\|x\|_2}{5\sqrt{n}} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right)$$
F6, Schaffer, Maximize, Origin, Multimodal $$f_6(x) = \frac{\sin^2\left[\sqrt{x_1^2 + x_2^2} \right]}{1.0 + 10^{-3}\left[x_1^2 + x_2^2 \right]^2}$$
F7, Schaffer, Maximize, Origin, Multimodal $$f_7(x) = 0.5 - \frac{\left(\sin\left[\sqrt{x_1^2 + x_2^2} \right] \right)^2 - 0.5}{\left(1.0 + 10^{-3}\left[x_1^2 + x_2^2 \right] \right)^2}$$

Table 3 comparison of ABC, PSO AND MLPSO

Fun.		ABC	PSO[2]	MLPSO	
	w	-	0.9-0.4	0.9 - 0.4	0.8 - 0.4
F_1	$\bar{\varepsilon}$	0	0	0	0
	σ	0	0	0	0
	S	100	100	100	100
	$\bar{N}_f$	40119	67623	75314	43059
F_2	$\bar{\varepsilon}$	5.91	16.45	0.95	0.016
	σ	8.27	23.11	7.28	0.08
	S	-	–	–	20
	$\bar{N}_f$	-	–	–	197532
F_3	$\bar{\varepsilon}$	0	43.95	0	0
	σ	0	11.05	0	0
	S	100	–	100	100
	$\bar{N}_f$	83491	–	98560	88751
F_4	$\bar{\varepsilon}$	0	0.014	0.012	0.012
	σ	0	0.014	0.015	0.014
	S	100	32	39	34
	$\bar{N}_f$	34603	69213	76218	50698
F_5	$\bar{\varepsilon}$	0	0.15	0	0
	σ	0	0.45	0	0
	S	100	89	100	100
	$\bar{N}_f$	83130	82366	105064	74816
F_6	$\bar{\varepsilon}$	0	0	0	0
	σ	0	0	0	0
	S	100	100	100	100
	$\bar{N}_f$	3150	10129	10864	3463
F_7	$\bar{\varepsilon}$	0	9.77×10^{-5}	1.94×10^{-4}	8.74×10^{-4}
	σ	0	9.66×10^{-4}	1.36×10^{-3}	2.78×10^{-3}
	S	100	99	98	91
	$\bar{N}_f$	25012	22843	30478	20276

Automatic CMOS Analog Circuit Design Block [3]



In this table comparison with ABC ,PSO and ML-PSO algorithm shown[6].

$$F_\epsilon = \sqrt{\sum \left(\frac{Spec_{Desired} - Spec_{Sim}}{Spec_{Desired}}\right)^2}$$

where $Spec_{Desired}$ represents desired performance measures (lower or upper limit as specified) and $Spec_{Sim}$ denotes the performance measures returned by a circuit simulator for a particular set of design variables provided by the optimizer. During the circuit design, those performance measures which satisfy the conditions given in Eq.341, will not contribute to the error function in Equation [3].

Resistor voltage divider circuit : This is very simple circuit used at initial stage for test the proper functioning of ABC optimizer then apply on CMOS Circuit . In electronics, a voltage divider (also known as a potential divider) is a simple linear circuit that produces an the fixed voltages at different nodes depends upon value of resistors used in circuit. Below figure show the voltage divider where R1,R2 and R3 are three circuit variables. Loop current I and node voltages (V2 and V3) are desired specifications.

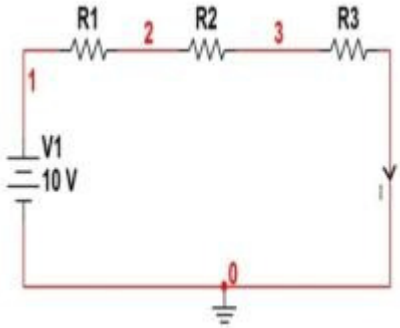


Table 4.1 Voltage divider desired specifications

Specs name	V(2,3)	V(3,0)	I
Desired specs	2V	4V	1mA

Table 4.2 Voltage divider variable search space and obtained value

Variable name	Search Space	Achieved value
R1(Ω)	(1,100k)	3.999996e+03
R2(Ω)	(1,100k)	2.000013e+03
R3(Ω)	(1,100k)	3.999991e+03

Table 4.3 Obtained specification for voltage divider

V(2,3)	V(3,0)	I
1.9999124 V	4.0000178 V	0.999912 mA

CMOS VOLTAGE DIVIDER CIRCUIT

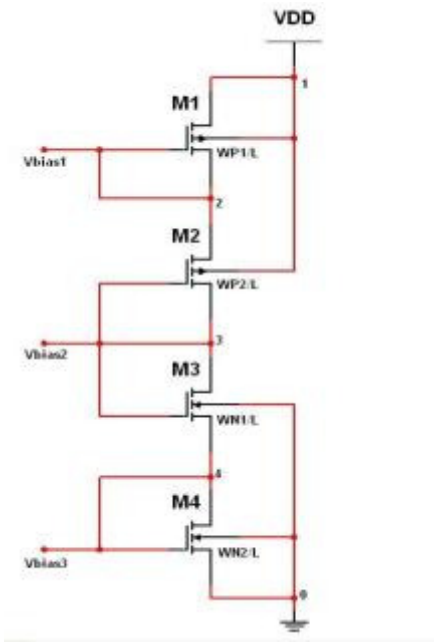


Table 5. CMOS voltage divider design variable search space and obtained value

Design variables	Input range (um)	Obtained Value (um)
WP1	0.1 to 2000	246.3891000
WP2	0.1 to 2000	1681.410950
WN1	0.1 to 2000	6.710750
WN2	0.1 to 2000	2.158700

Conclusion: ABC a new proposed algorithm is used for automatic CMOS analog circuit design application. Performance of ABC is compared with the PSO for automatic circuit design. The current starved VCO circuit can design and optimized with ABC and PSO and comparison results.ABC based optimizer can be used for any other numerical optimization problems. The work can be further improved by using the variants of ABC algorithm in order to achieve better results. The work can be extended for automatic circuit design with nano-scale novel devices like tunnelFET and FinFET. Further the process voltage and temperature (PVT) variation can be included in this work for improve reliability of design.

Reference:

1. D.Karaboga, B. Basturk,"A Powerful And Efficient Algorithm For Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm",Journal of Global Optimization, Volume:39,Issue:3,pp: 459-471, Springer Netherlands, 2007.

2. Rajesh A. Thakker, M. Shojaei Baghini and M. B. Patil "Automatic Design of Low-Power Low-Voltage Analog Circuits Using Particle Swarm Optimization with Re-Initialization", Journal of Low Power Electronics, Vol. 5, 1–12, 2009.

3. R. A. Thakker,Sanjay B. Prajapati and M. B. Patil, "Particle Swarm Optimization with Memory Loss Operation" Department of Electrical Engineering, Indian Institute of Technology – Bombay, India.

4. Goldberg, David E., "Genetic Algorithms in Search Optimization and Machine Learning", Addison Wesley. 1989.pp. 41. ISBN 0201157675

5.Andreas Fester "Electronic circuit simulation with gEDA and NG-Spice by Example", May 25, 2004

6. Ravi C. Butani "Thesis on Automatic CMOS Analog Circuit Design using Artificial Bee Colony Algorithm" Department of Electronics and communication Engineering L.D.Engineering college Ahmedabad June 2011.