Electronics Research institute

CrossMark

Journal of Electrical Systems and Information Technology 2 (2015) 219–241

www.elsevier.com/locate/jesit

Optimal CMOS inverter design using differential evolution algorithm

Bishnu Prasad De [a], R. Kar [a,*], D. Mandal [a], S.P. Ghoshal [b]

[a] Department of Electronics and Communication Engg., NIT Durgapur, India
[b] Department of Electrical Engg., NIT Durgapur, India

Received 3 March 2014; accepted 13 March 2015
Available online 12 September 2015

Abstract

The inverter is known to be the nucleus of all digital designs. Evolutionary computation may be a competent implement for automatic design of digital integrated circuits (IC). In this paper, optimal switching characteristics of a CMOS inverter are realized using an evolutionary optimization approach called differential evolution (DE) algorithm. The real coded genetic algorithm (RGA) and particle swarm optimization (PSO) have been adopted for the sake of comparison. DE based design results have been compared also with those of the PSPICE results. The comparative simulation results establish the DE as a more competent candidate to other aforementioned evolutionary algorithms in terms of accuracy and convergence speed.

Keywords: CMOS inverter; Switching characteristics; Rise time; Fall time; Differential evolution; Evolutionary optimization technique

1. Introduction

Optimization refers to the method of input adjustments for proper characterizing a device; a mathematical process, with an objective to get minimum or maximum outputs as desired. The input may consist not only of variables and processes, but also functions including cost functions, objective functions or fitness functions. The output to such functions can be the cost or fitness; on the contrary if the process is an experiment, then the variables are restricted to physical inputs provided to the experiment. An optimization algorithm can be illustrated as combination of three key components: formulation of the optimization problem, performance evaluation engine and optimization engine. With the advancement of nano-technology in very large scale integration (VLSI), the realization of more and more complex integrated electronic circuits and systems is now a days possible. Scaling of semiconductor devices to the extent of nanometers has led to the miniaturization of the critical feature sizes resulting in ultra high integration density

* Corresponding author. Tel.: +91 9434788056; fax: +91 343 2547375.
E-mail addresses: rajibkarece@gmail.com (R. Kar), durbadal.bittu@gmail.com (D. Mandal), spghoshalnitdgp@gmail.com (S.P. Ghoshal).

Peer review under the responsibility of Electronics Research Institute (ERI).

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jesit.2015.03.014

and drastic circuit size reduction. However, at nanometric scale, various effects of manufacturing process variations and design optimization process have pushed the entire phenomena from deterministic to stochastic domain, and the inter-relationships among the key analysis parameters (delay, power, reliability, noise and area) have become more interleaved and complex. Extensive research in developing new methodologies to minutely examine these stochastic metrics in a unified manner by various researchers and scientists has expedited the necessity of further development of multi-metric, stochastic, evolutionary optimization techniques.

Different evolutionary optimization techniques aptly used for different optimization problems are Genetic Algorithm (GA) which is inspired by the Darwin's "Survival of the Fittest" strategy (Ma and Cowan, 1996), Biological evolutionary strategy adopted in the development of differential evolution (DE) algorithm (Storn and Price, 1995; Karaboga, 2005) and swarm intelligence mimicked in particle swarm optimization (PSO) and its variants (Luitel and Venayagamoorthy, 2010; Mondal et al., 2012). Conventional PSO has mimicked the behaviour of bird flocking or fish schooling (Luitel and Venayagamoorthy, 2010; Hussain et al., 2011; Fang et al., 2009; Krusienski and Jenkins, Nov 2003; Mandal et al., 2011; Saha et al., 2011). GA is a probabilistic heuristic search optimization technique developed by Holland (1975). The features such as multi-objective, coded variable and natural selection made this technique distinct, suitable and very popular amongst researchers for finding the near global solution. GA is applied for the optimal design of FIR filters (Mastorakis et al., 2003; Lu and Tzeng, 2000). Genetic Algorithm has also been used for the synthesis of passive analogue circuits to get optimal values of R, L and C elements from a given set of specifications (Das and Vemuri, 2007). Circuit bi-partitioning (Gill et al., 2009), placement and area optimization of soft modules in VLSI floor plan design (Tang and Lau, 2007) have been developed using Genetic Algorithm.

DE algorithm was first introduced by Storn and Price in 1995 (Storn and Price, 1995). Like GA, it is a randomized stochastic search technique enriched with the operations of crossover, mutation and selection but unlike GA, stagnation and entrapment to local minima are less associated to it (Karaboga, 2005). Karaboga has proposed DE algorithm (Karaboga and Cetinkaya, 2006) for the design of digital finite impulse response filters for different filter orders. It has been established that the performance of the design is better than those obtained by RGA and other classical approaches. DE algorithm has been useful in different fields of electrical power system optimization, such as economic load dispatch problem (Chiou, 2007), short-term hydrothermal scheduling problem (Mandal and Chakraborty, 2008), etc. It is established that DE yields superior results in terms of cost and computation time. A new image segmentation method using DE as reported in Pei et al. (2009) shows that DE presents good segmentation results in noisy images. Synthesis of unequally spaced linear antenna arrays by using DE has been carried out in Lin et al. (2010). Synthesis results show that DE algorithm exhibits stronger synthesis capability, higher reliability and efficiency compared to other available optimization algorithms adopted in that paper. The use of DE algorithm to the restrictive channel routing problem in VLSI Circuit Design has been investigated in Vijayakumar et al. (2009).

PSO is swarm intelligence based algorithm developed by Eberhart et al. (Kennedy and Eberhart, 1995; Eberhart and Shi, 1998). Several attempts have been made towards the system identification problem with basic PSO and its modified versions (Luitel and Venayagamoorthy, 2010; Mondal et al., 2012). The key advantage of PSO is its simplicity in computation and a few steps are required in the algorithm. PSO was effectively utilized in various application areas. Design of digital IIR filter using particle swarm optimization was proposed in Chen and Luk (2010). PSO was adopted for the placement and routing of the field programmable gate arrays (FPGA) to reduce the distances between Configurable Logic Blocks (CLBs) (Gudise and Venayagamoorthy, 2004). PSO was chosen for image segmentation to estimate the parameters in the mixture density function for minimization of the square error between the density function and the actual histogram (Lai, 2006). PSO was used for the synthesis of micro strip coupler and single shunt stub matching circuits (Ulker, 2008). Optimal designs of two basic analogue circuits such as differential amplifier with current mirror load and two stage operational amplifiers were carried out using PSO algorithm (Vural and Yildirim, 2012).

Fall time (t_{f}) of output voltage for a CMOS inverter is estimated using PSO in Vural et al. (2011). Design of CMOS inverter having symmetric output waveform with equal rise time (t_r) and fall time (t_{f}) is investigated using PSO in Vural et al. (2011), Vural et al. (2010) and Mukhopadhyay and Pandit (2012). Design of CMOS inverter with equal output voltage delay times (t_f, t_r), and propagation delay times (t_{pHL}, t_{pLH}) using PSO is reported in Vural et al. (2011).

In this paper, optimal switching characteristics of CMOS inverter are presented using real coded genetic algorithm (RGA) and DE algorithm. It has been realized that RGA is incapable for local searching (Karaboga, 2005) in a multidimensional search space and also suffers from premature convergence and easy entrapment to sub-optimal solution (Karaboga, 2009). Simulation results obtained from RGA and DE algorithms are compared to those of PSO based results to demonstrate the effectiveness and superiority of the performance of DE in achieving the near-global optimal solutions.

The rest of the paper is organized as follows: In Section 2, different meta-heuristic optimization techniques under consideration namely RGA and DE are explained briefly. Switching characteristics of CMOS inverter are described in Section 3. In Section 4, the objective functions used in this paper are formulated and RGA, DE based inverter design examples are discussed comprehensively. Discussion of results, comparison of results with PSPICE simulator and PSO based reported results (Vural et al., 2011, 2010; Mukhopadhyay and Pandit, 2012) are given in Section 5. Finally, Section 6 concludes the paper.

2. Evolutionary algorithms employed

Evolutionary algorithms stand upon the platform of meta-heuristic optimization methods, which are characterized as stochastic, adaptive and learning in order to produce intelligent optimization schemes. Such schemes have the potential to adapt to their ever changing dynamic environment through the previously acquired knowledge. Here, real coded genetic algorithm (RGA) and differential evolution (DE) Algorithm are briefly discussed.

2.1. Real coded genetic algorithm (RGA)

Standard Genetic Algorithm (also known as real coded GA) is mainly a probabilistic search technique, based on the principles of natural selection and evolution built upon the Darwin's "Survival of the Fittest" strategy (Holland, 1975). Each encoded chromosome that constitutes the population is a solution to the problem under consideration. These solutions may be good or bad, but are tested rigorously through the genetic operations such as crossover and mutation to evolve to a global optimal or near global optimal solution to the problem at hand. Chromosomes are constructed over some particular alphabet $\{0, 1\}$, so that chromosomes' values are uniquely mapped onto the real decision variable domain. Each chromosome is evaluated by a function known as error fitness function, which is usually the cost function or objective function of the corresponding optimization problem. Each chromosome has a probability of selection and has to take part in the genetic operation based upon the Roulette's wheel strategy. In the genetic operations, crossover and mutation bring the variation in alleles of gene in the chromosome population along with the effort to alleviate trapping to local optimal solution. GA has mainly two drawbacks: lack of good local search capability and premature convergence. The algorithmic steps of RGA are as follows:

Step 1: Initialize the real coded chromosome strings (ω) of n_p population, each consisting of the number of parameters need to be optimized (dimension of the optimization problem (D)). Each parameter has a maximum bound and a minimum bound and is randomly generated within this range. Maximum iteration/genetic cycles (=250 or 500 depending on the case study) is defined. Mutation probability = 0.003; Crossover ratio = 0.8; Selection probability = 1/3.
Step 2: Decoding of the strings and evaluation of cost function (CF).
Step 3: Selection of elite strings in order of increasing cost function values from the minimum value.
Step 4: Copying the elite strings over the non-selected strings.
Step 5: Crossover and mutation to generate offspring.
Step 6: Genetic cycle/iteration cycle updating.
Step 7: The iteration stops when the termination criteria of maximum genetic cycles are satisfied. The grand minimum CF and its corresponding chromosome string or the desired solution is finally obtained.

2.2. Differential evolution (DE) algorithm

Differential Evolution or DE fits into the class of evolutionary algorithms. DE is a stochastic; population based heuristic approach, having the capability to solve global optimization problems. The crucial idea behind DE algorithm is a scheme for generating trial parameter vectors and adds the weighted difference between two population vectors to a third one. Like any other evolutionary algorithm, DE algorithm aims at evolving a population of n_p, D-dimensional parameter vectors, so-called individuals, which encode the candidate solutions, i.e.,

$$\vec{x}_{i,g} = \{x_{1,i,g}, x_{2,i,g}, \ldots, x_{D,i,g}\} \tag{1}$$

where $i = 1, 2, 3, \ldots, n_p$. The initial population (at $g = 0$) should cover the entire search space as much as possible by uniformly randomizing individuals within the search constrained by the prescribed minimum and maximum parameter bounds: $\vec{x}_{\min} = \{x_{1,\min}, \ldots, x_{D,\min}\}$ and $\vec{x}_{\max} = \{x_{1,\max}, \ldots, x_{D,\max}\}$.

For example, the initial value of the jth parameter of the ith vector is

$$x_{j,i,0} = x_{j,\min} + rnd * (x_{j,\max} - x_{j,\min}), \quad \text{where } j = 1, 2, 3, \ldots, D \tag{2}$$

The random number generator *rnd* returns a uniformly distributed random number from within the range [0,1]. After initialization, DE enters a loop of evolutionary operations: mutation, crossover, and selection.

(i) *Mutation*

Once initialized, DE mutates and recombines the population to produce new population. For each trial vector $x_{i,g}$ at generation g, its associated mutant vector $\vec{v}_{i,g} = \{v_{1,i,g}, v_{2,i,g}, ..., v_{D,i,g}\}$ can be generated via certain mutation strategy. Five most frequently used mutation strategies in the DE codes are listed as follows:

$$"DE/rand/1" : \vec{v}_{i,g} = \vec{x}_{r_1',g} + F\left(\vec{x}_{r_2',g} - \vec{x}_{r_3',g}\right) \tag{3}$$

$$"DE/best/1" : \vec{v}_{i,g} = \vec{x}_{best,g} + F\left(\vec{x}_{r_1',g} - \vec{x}_{r_2',g}\right) \tag{4}$$

$$"DE/rand-to-best/1" : \vec{v}_{i,g} = \vec{x}_{i,g} + F\left(\vec{x}_{best,g} - \vec{x}_{i,g}\right) + F\left(\vec{x}_{r_1',g} - \vec{x}_{r_2',g}\right) \tag{5}$$

$$"DE/best/2" : \vec{v}_{i,g} = \vec{x}_{best,g} + F\left(\vec{x}_{r_1',g} - \vec{x}_{r_2',g}\right) + F\left(\vec{x}_{r_3',g} - \vec{x}_{r_4',g}\right) \tag{6}$$

$$"DE/rand/2" : \vec{v}_{i,g} = \vec{x}_{r_1',g} + F\left(\vec{x}_{r_2',g} - \vec{x}_{r_3',g}\right) + F\left(\vec{x}_{r_4',g} - \vec{x}_{r_5',g}\right) \tag{7}$$

The indices $r_1', r_2', r_3', r_4', r_5'$ are mutually exclusive integers randomly chosen from the range $[1,n_p]$, and all are different from the base index i. These indices are randomly generated once for each mutant vector. The scaling factor F is a positive control parameter for scaling the difference vector. $x_{best,g}$ is the best individual vector with the best fitness value in the population at generation 'g'.

(ii) *Crossover*

To complement the differential mutation search strategy, crossover operation is applied to increase the potential diversity of the population. The mutant vector $v_{i,g}$ exchanges its components with the target vector $x_{i,g}$ to generate a trial vector:

$$\vec{u}_{i,g} = \left\{u_{1,i,g}, u_{2,i,g}, \ldots, u_{D,i,g}\right\} \tag{8}$$

In the basic version, DE employs the binomial (uniform) crossover defined as

$$u_{j,i,g} = \begin{cases} v_{j,i,g} & if(rnd_{i,j} \geq C_r \text{ or } j = j_{rand}) \\ x_{i,j,g} & \text{otherwise} \end{cases} \tag{9}$$

where $j = 1, 2, \ldots, D$; $rnd_{i,j}$ returns a uniformly distributed random number from within the range [0,1]. The crossover rate C_r is user-specified constant within the range $(1, 0)$, which controls the fraction of parameter values copied from the mutant vector. j_{rand} is a randomly chosen integer in the range $[1,D]$. The binomial crossover operator copies the jth parameter of the mutant vector $\vec{v}_{i,g}$ to the corresponding element in the trial vector $\vec{u}_{i,g}$, if $rnd_{i,j} \leq C_r$ or $j = j_{rand}$. Otherwise, it is copied from the corresponding target vector $\vec{x}_{i,g}$.

(iii) *Selection*

To keep the population size constant over subsequent generations, the next step of the algorithm calls for selection to determine whether the target or the trial vector survives to the next generation, i.e., at $g = g + 1$. The selection operation is described as (10).

$$\vec{x}_{i,g+1} = \begin{cases} \vec{u}_{i,g} & \text{if} \quad f\left(\vec{u}_{i,g}\right) \leq f\left(\vec{x}_{i,g}\right) \\ \vec{x}_{i,g} & \text{otherwise} \end{cases} \tag{10}$$

where $f(x)$ is the objective/cost function to be minimized. So, if the new vector yields an equal or lower value of the objective function, it replaces the corresponding target vector in the next generation; otherwise the target is retained in the population. Hence, the population gets either better (with respect to the minimization of the objective function) or remains the same in fitness status, but never deteriorates.

The above three steps are repeated generation after generation until some specific termination criteria are satisfied.

2.2.1. Control parameter selection of DE

Proper selection of control parameters is very important for the success and performance of an algorithm. The optimal control parameters are problem-specific. Therefore, the set of control parameters that best fit each problem has to be chosen carefully. Values of F lower than 0.5 may result in premature convergence, while values greater than 1 tend to slow down the convergence speed. Large populations help maintaining diverse individuals, but also slow down convergence speed. In order to avoid premature convergence, F or n_p should be increased or crossover rate C_r should be decreased. Larger values of F result in larger perturbations and better probabilities to escape from local optima, while lower C_r preserves more diversity in the population, thus avoiding local optima.

The main advantages of DE are simple in concept; few control parameters, high convergence characteristics. The algorithmic steps of DE are as follows:

Step 1. *Generation of initial population*: Set the generation counter $g = 0$ and randomly initialize D-dimensional n_p individuals (parameter vectors/target vectors), $\vec{x}_{i,g} = \{x_{1,i,g}, x_{2,i,g}, ..., x_{D,i,g}\}$; where D is equal to the dimension of the parameter vector need to be optimized. Here the range of i is described as $\{1, 2, 3, \ldots, n_p\}$. The initial population (at $g = 0$) should cover the entire search space as much as possible by uniformly randomizing individuals within the search constrained by the prescribed minimum and maximum parameter bounds: $\vec{x}_{\min} = \{x_{1,\min}, \ldots, x_{D,\min}\}$ and $\vec{x}_{\max} = \{x_{1,\max}, \ldots, x_{D,\max}\}$. Maximum iteration cycles and target error are defined.

Step 2. *Mutation*: For $i = 1$ to n_p, generate a mutated vector, $\vec{v}_{i,g} = \{v_{1,i,g}, v_{2,i,g}, ..., v_{D,i,g}\}$ corresponding to the target vector $\vec{x}_{i,g}$ via any one of 5 mutation strategies mentioned earlier.

Step 3. *Crossover*: Generation of a trial vector $\vec{u}_{i,g}$ for each target vector $\vec{x}_{i,g}$, where $\vec{u}_{i,g} = \{u_{1,i,g}, u_{2,i,g}, ..., u_{D,i,g}\}$ for $i = 1$ to n_p; $\mathrm{j_{rand}} = [rnd^*\mathrm{D}]$; for $j = 1$ to D.

$$u_{j,i,g} = \begin{cases} v_{j,i,g} & \text{if} \quad \left(rnd_{i,j} \geq C_r \text{ or } \quad j = j_{rand}\right) \\ x_{i,j,g} & \text{otherwise} \end{cases}$$

'rnd_{ij}' is an uniformly distributed random number generated within [0,1]. The crossover rate C_r is the user-specified constant within the range [1,0], which controls the fraction of parameter values copied from the mutant vector. j_{rand} is a randomly chosen integer in the range [1,D]. The binomial crossover operator copies the jth parameter of the mutant vector $\vec{v}_{i,g}$ to the corresponding element in the trial vector $\vec{u}_{i,g}$ if $rnd_{i,j} \geq C_r$ or $j = j_{rand}$. Otherwise, it is copied from the corresponding target vector, $\overrightarrow{x_{i,g}}$.

Step 4. *Selection*: for $i = 1$ to n_p,

$$\vec{x}_{i,g+1} = \begin{cases} \vec{u}_{i,g} & \text{if} \quad f\left(\vec{u}_{i,g}\right) \leq f\left(\vec{x}_{i,g}\right) \\ \vec{x}_{i,g} & \text{otherwise} \end{cases}.$$

Increment the generation count $g = g + 1$. The last three steps are repeated generation after generation until termination criteria are satisfied.

2.3. Statistical comparison of accuracy between two algorithms

Two-sample t-test is a hypothesis testing method for determining the statistical significance of the difference between two independent samples of an equal sample size (Walpole and Myer, 1978). The t-test value will be positive if the

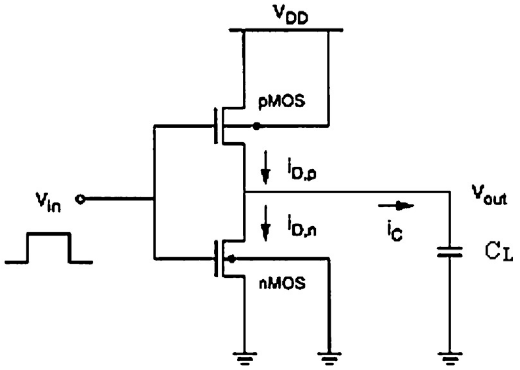

Fig. 1. Schematic diagram of a CMOS inverter.

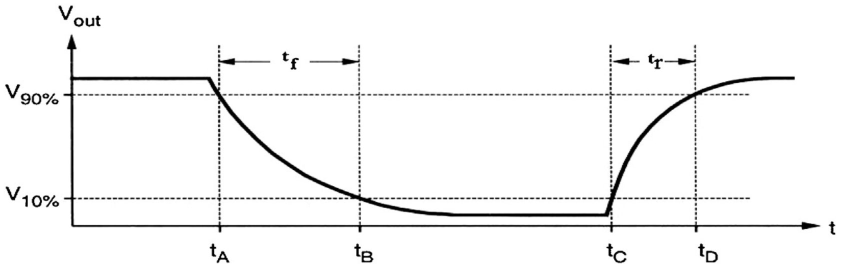

Fig. 2. Output voltage rise time (t_r) and fall time (t_f).

second algorithm is better than the first, and it is negative if it is poorer. The t-value is defined as given in the following equation

$$t = \frac{\bar{\alpha}_1 - \bar{\alpha}_2}{\sqrt{\left(\frac{\sigma_1^2}{\beta+1}\right) + \left(\frac{\sigma_2^2}{\beta+1}\right)}} \tag{11}$$

where $\overline{\alpha}_1$ and $\overline{\alpha}_2$ are the mean values of the first and the seconds methods, respectively; σ_1 and σ_2 are the standard deviations of the first and the second methods, respectively; and β is the value of the degree of freedom. When the t-value is higher than 1.645 ($\beta = 49$), there is a significant difference between the two algorithms with a 95% confidence level. The t-value is larger than 2.15 (degree of freedom = 49), meaning that there is a significant difference between the two algorithms with a 98% confidence level. In our all the Case studies, RGA and DE are considered as algorithm 1 and algorithm 2, respectively.

3. Switching characteristics of CMOS inverter

The speed of operation of a digital system is determined by the switching characteristics of the logic gates used to construct the system. Since the inverter is the basic logic gate of any digital IC technology, the switching characteristics of the inverter are the fundamental parameters in characterizing the technology. Therefore, the switching speed of the circuit must be approximated and optimized at an early design phase to ensure circuit reliability and performance. Here, the optimal switching characteristics of CMOS inverter are investigated using afore-mentioned evolutionary optimization techniques. The schematic diagram of a CMOS inverter is shown in Fig. 1. Rise time (t_r) and fall time (t_f) of the output voltage are shown in Fig. 2. The input and output voltage waveforms of CMOS inverter circuit are shown in Fig. 3.

The switching operation of the CMOS inverter is analyzed to determine its fall time (t_f), rise time (t_r) and propagation delay times (t_{pHL}, t_{pLH}). It is presumed that a pulse waveform is applied to the input of the inverter. The fall time (t_f), is the time required for the output voltage to drop from $V_{90\%}$ level to $V_{10\%}$ level. Similarly, the rise time (t_r) is defined as

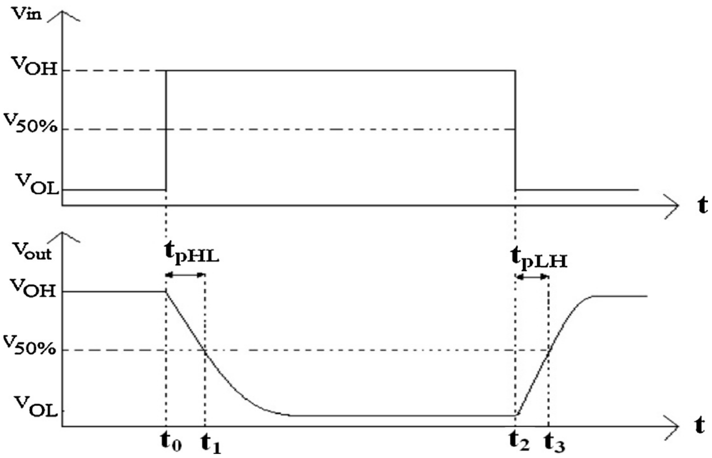

Fig. 3. Input and output voltage waveforms of CMOS inverter and definitions of propagation delay times.

the time required for the output voltage to rise from $V_{10\%}$ level to $V_{90\%}$ level. The propagation delay times t_{pHL} and t_{pLH} establish the input to output signal delays during high-to-low and low-to-high transitions of the output, respectively. The high-to-low propagation delay (t_{pHL}) is defined as the time delay between the $V_{50\%}$ transition of the rising input voltage and $V_{50\%}$ transition of the falling output voltage. Similarly, the low-to-high propagation delay (t_{pLH}) is the time delay between the $V_{50\%}$ transition of the falling input voltage and $V_{50\%}$ transition of the rising output voltage.

To compute fall time (t_f) of the output voltage, output load capacitance (C_L) should be discharged through the active NMOS transistor, considering PMOS transistor is in cut-off region. The fall time is given as follows (DeMassa and Ciccone, 1996):

$$t_f = \frac{C_L}{\mu_n C_{OX}\left(\frac{W}{L}\right)_n (V_{DD} - V_{tn})}\left[\frac{2(V_{tn} - 0.1V_{DD})}{(V_{DD} - V_{tn})} + \ln\left(\frac{(2(V_{DD} - V_{tn})) - 0.1V_{DD}}{0.1V_{DD}}\right)\right] \tag{12}$$

To calculate rise time (t_r) of the output voltage, output load capacitance (C_L) should be charged through the active PMOS transistor, considering NMOS transistor is in cut-off region. The rise time is given in (13) (DeMassa and Ciccone, 1996).

$$t_r = \frac{C_L}{\mu_p C_{OX}\left(\frac{W}{L}\right)_p (V_{DD} - |V_{tp}|)}\left[\frac{2(|V_{tp}| - 0.1V_{DD})}{(V_{DD} - |V_{tp}|)} + \ln\left(\frac{(2(V_{DD} - |V_{tp}|)) - 0.1V_{DD}}{0.1V_{DD}}\right)\right] \tag{13}$$

Study of propagation delay times t_{pHL} and t_{pLH} involves discharging output load (C_L) capacitance through active NMOS transistor and charging output load capacitance (C_L) through active PMOS transistor, respectively. To simplify the analysis and the derivation of delay expressions, the input voltage waveform is usually supposed to be ideal step pulse with zero rise and fall times. Under this consideration, t_{pHL} becomes the time required for the output voltage to fall from $V_{\rm OH}$ to the $V_{50\%}$ level, and t_{pLH} becomes the time required for the output voltage to rise from $V_{\rm OL}$ to the $V_{50\%}$ level. For CMOS inverter, $V_{\rm OH} = V_{\rm DD}$ and $V_{\rm OL} = 0$ are considered. The propagation delay times are given as follows (Sung-Mo-Kang and Leblebici, 2003):

$$t_{pHL} = \frac{C_L}{\mu_n C_{OX}\left(\frac{W}{L}\right)_n (V_{DD} - V_{tn})}\left[\frac{2V_{tn}}{(V_{DD} - V_{tn})} + \ln\left(\frac{(4(V_{DD} - V_{tn}))}{V_{DD}} - 1\right)\right] \tag{14}$$

$$t_{pLH} = \frac{C_L}{\mu_p C_{OX}\left(\frac{W}{L}\right)_p (V_{DD} - |V_{tp}|)}\left[\frac{2|V_{tp}|}{(V_{DD} - |V_{tp}|)} + \ln\left(\frac{(4(V_{DD} - |V_{tp}|))}{V_{DD}} - 1\right)\right] \tag{15}$$

4. Problem formulation

This paper considers three different case studies for characterization of the CMOS inverter switching. In Case study-1, the fall time (t_f) of output voltage for a CMOS inverter is investigated. Case study-2 is for the design of CMOS

Table 1
Delay limits and design parameters bound for the Case study-1.

Design set no.	Specified ranges		
	C_L (pF)	(W/L)	t_f (ns)
1	0.1–2.4	0.3–3.3	0.5–6.7
2	0.2–5.6	0.4–2.3	0.3–6.0
3	0.6–3.4	0.9–5.0	0.6–8.6
4	0.5–3.6	1.2–4.1	0.9–11.0
5	0.7–1.8	0.7–4.9	1.2–15.0
6	0.3–2.4	2.2–3.2	1.4–12.0
7	0.7–2.3	0.7–3.0	1.6–5.7
8	0.6–1.9	1.5–3.5	1.0–8.15

inverter having symmetric output voltage with equal rise time (t_r) and fall time (t_f). In Case study-3, a CMOS inverter is designed with improved symmetry at output voltage with equal rise time (t_r) and fall time (t_f) and equal high-to-low propagation delay (t_{pHL}) and low-to-high propagation delay (t_{pLH}).

4.1. Case study-1

In this case, the main objective is to estimate the fall time of output voltage of an inverter, as given in (12), with the least error value. During the design phase, values of design parameters such as fall time (t_f), output load capacitance (C_L) and aspect ratio (W/L) of MOS structures (identical sized NMOS and PMOS) should be reserved in certain ranges. RGA and DE algorithms are individually employed to find out the solution set that consists of the accurate values of t_f, C_L and (W/L) ratio for the given ranges. The cost/error function (CF) is defined as

$$CF = \left|\mu_n C_{OX}\left(\frac{W}{L}\right)_n t_f - \frac{C_L}{(V_{DD}-V_{tn})}\left[\frac{2\,(V_{tn}-0.1V_{DD})}{(V_{DD}-V_{tn})} + \ln\left(\frac{(2\,(V_{DD}-V_{tn}))-0.1V_{DD}}{0.1V_{DD}}\right)\right]\right| \tag{16}$$

The error fitness function is given as

$$J = 10\log_{10}(CF) \tag{17}$$

To order to obtain the accurate values of the design parameters, CF is set to a value very close to zero. Equating CF to zero means that error is equal to zero and t_f is successfully estimated depending on the design parameters. Here, TSMC 0.25 micron fabrication technology parameters (MOSIS) as follows: $V_{DD} = 2.5$ V, $V_{tn} = 0.3655$ V and $\mu_n C_{ox} = 243.6\ \mu\text{A/V}^2$. All optimization programmes were run in MATLAB 7.5 version on core (TM) 2 duo processor, 3.00 GHz with 2 GB RAM.

For both the RGA and DE, the size of the initial population matrix is 10 × 3. Rows specify the number of particle vectors in the population and columns specify the dimension of each particle vector, defined as $x = [C_L, (W/L), t_f]$, assuming $(W/L) = (W/L)_n = (W/L)_p$ (identical aspect ratios for the NMOS and PMOS transistors). In this case study, (5) is chosen for mutation strategy and F is set to 0.5 for DE.

The algorithms are run with an upper limit of 250 iteration/generation cycles for all design sets, individually and independently. In this case study, estimation of fall time (t_f) of output voltage is executed for eight different ranges of design parameters (C_L, $(W/L)_n$) and design criterion (t_f). Specified ranges and synthesized results of RGA, DE and PSO (Vural et al., 2011) are shown in Table 1 and Table 2, respectively.

4.2. Case study-2

To achieve a symmetric switching response, it is expected to have equal rise time (t_r) and fall time (t_f) of output voltage. Due to some second order effects, a definite error between t_r and t_f is always observed. In this case, the main objective is to estimate the design parameters which minimize the difference between t_r and t_f. The design problem can be specified as follows:

Table 2
Synthesized results of RGA, DE and PSO (Vural et al., 2011) for the Case study-1.

Design set no.	RGA based results				DE based results				PSO based reported results (Vural et al., 2011)
	C_L (pF)	(W/L)	t_f (ns)	Error (CF)	C_L (pF)	(W/L)	t_f (ns)	Error (CF)	t_f (ns)
1	1.2165	1.5458	4.3680	2.1652×10^{-15}	0.4189	2.7076	0.8560	0.98343×10^{-15}	1.777
2	1.0840	1.1057	5.4413	1.8396×10^{-15}	0.5628	1.0095	3.0924	0.56276×10^{-15}	3.536
3	3.3646	2.3197	8.0324	4.4488×10^{-15}	0.8825	1.8661	2.6195	0.82621×10^{-15}	3.885
4	3.3416	1.7120	10.7982	8.9002×10^{-15}	1.0773	1.7850	3.3446	0.37518×10^{-15}	4.627
5	1.6772	1.0914	8.5011	4.6743×10^{-15}	1.6439	4.0936	2.2265	0.52067×10^{-15}	2.805
6	2.1880	2.2381	5.4335	7.8466×10^{-15}	0.9140	2.2312	2.2690	0.95327×10^{-15}	2.719
7	2.2418	2.8412	4.3650	6.1363×10^{-15}	0.8869	2.5695	1.9132	0.16305×10^{-15}	2.028
8	1.3954	1.5224	5.0620	6.9919×10^{-15}	0.8041	1.8959	2.3525	0.64131×10^{-15}	3.709

Minimize

$$CF = \left|\left(t_f\left(C_L, \left(\frac{W}{L}\right)_n\right) - t_r\left(C_L, \left(\frac{W}{L}\right)_p\right)\right)\right| \tag{18}$$

subject to

$$(t_f)_{\min} \le t_f \le (t_f)_{\max}$$

$$(t_r)_{\min} \le t_r \le (t_r)_{\max}$$

where

$$(C_L)_{\min} \le C_L \le (C_L)_{\max};$$

$$\left(\left(\frac{W}{L}\right)_n\right)_{\min} \le \left(\frac{W}{L}\right)_n \le \left(\left(\frac{W}{L}\right)_n\right)_{\max};$$

$$\left(\left(\frac{W}{L}\right)_p\right)_{\min} \le \left(\frac{W}{L}\right)_p \le \left(\left(\frac{W}{L}\right)_p\right)_{\max}.$$

TSMC 0.25 μm fabrication technology parameters (MOSIS) are also used for this design as follows: $V_{DD} = 2.5$ V, $V_{tn} = 0.3655$ V, $V_{tp} = -0.5466$ V, $\mu_p C_{ox} = 51.6\ \mu A/V^2$ and $\mu_n C_{ox} = 243.6\ \mu A/V^2$.

For both the RGA and DE, initial population matrix size is taken as 10 × 3. The number of particles in the population is defined as rows and column of any row is the dimension of each particle vector, denoted as $x = [C_L, (W/L)_n, (W/L)_p]$. In this case study, (5) is chosen for mutation strategy and F is set to 0.5 for DE. Each algorithm is run with an upper bound of 250 iteration cycles for all the design sets, individually and independently. Delay limits and bounds of design parameters are shown in Table 3. RGA, DE and PSO (Vural et al., 2011, 2010) based results for each specified range is reported in Table 4.

4.3. Case study-3

In Case study-3, the main objective is to achieve an improved symmetrical output voltage of the CMOS inverter, having equal t_r and t_f and equal t_{pHL} and t_{pLH}. RGA and DE algorithms are employed to find out the optimal design parameters which minimize the error between t_f and t_r of the output voltage and the error between propagation delay times (t_{pHL}, t_{pLH}). The design problem can be specified as follows:

$$CF = \left|\left(t_f\left(C_L, \left(\frac{W}{L}\right)_n\right) - t_r\left(C_L, \left(\frac{W}{L}\right)_p\right)\right)\right| + \left|\left(t_{pHL}\left(C_L, \left(\frac{W}{L}\right)_n\right) - t_{pLH}\left(C_L, \left(\frac{W}{L}\right)_p\right)\right)\right| \tag{19}$$

Table 3
Delay limits and design parameters bound for the Case study-2.

Design set no.	Specified ranges				
	C_L (pF)	$(W/L)_\mathrm{n}$	$(W/L)_p$	t_f (ns)	t_r (ns)
1	0.33–2.3	1–3	2–18	1–12	1–12
2	0.6–1.5	0.5–2.5	1.6–19.3	0.5–7.6	0.5–7.6
3	0.3–3	0.3–1.9	1.76–7.65	0.56–8.7	0.56–8.7
4	0.11–1.34	1.5–3.5	2.65–18.9	0.77–7.89	0.77–7.89
5	0.5–1.5	1–2.5	2–13.75	0.1–15	0.1–15
6	0.5–1.5	1–3	2–21	0.1–15	0.1–15
7	1.0–3.0	1.5–3.5	3.75–21	0.1–15	0.1–15
8	1.5–3.5	1.5–3	3–19.2	0.1–10	0.1–10

subject to

$$(t_f)_{\min} \le t_f \le (t_f)_{\max}; \quad (t_r)_{\min} \le t_r \le (t_r)_{\max}; \; (t_{pHL})_{\min} \le t_{pHL} \le (t_{pHL})_{\max}; \; (t_{pLH})_{\min} \le t_{pLH} \le (t_{pLH})_{\max}$$

where

$$(C_L)_{\min} \le C_L \le (C_L)_{\max};$$

$$\left(\left(\frac{W}{L}\right)_n\right)_{\min} \le \left(\frac{W}{L}\right)_n \le \left(\left(\frac{W}{L}\right)_n\right)_{\max};$$

$$\left(\left(\frac{W}{L}\right)_p\right)_{\min} \le \left(\frac{W}{L}\right)_p \le \left(\left(\frac{W}{L}\right)_p\right)_{\max}.$$

Fabrication technology parameters and the control parameters of RGA and DE are the same as used in the previous case studies. The dimension of particle vector is denoted as $x = [C_L, (W/L)_n, (W/L)_p]$. The algorithms are run with an upper bound of 500 iteration cycles, individually and independently. Delay limits, bounds of design parameters and PSO based results are shown in Table 5. RGA and DE based results are shown in Table 6.

5. Results and discussion

In this work, two popular evolutionary optimization algorithms called RGA and DE are employed to achieve the near global optimal solutions for the switching characteristics of CMOS inverter circuit. Three different design cases are considered. Firstly, for the Case study-1, RGA and DE algorithms are applied to estimate the fall time (t_f) of output voltage. Here, identical aspect ratios for NMOS and PMOS transistors are considered. Eight different ranges of design parameters and design criterion are considered in the study.

Fig. 4 shows the plot of J versus iteration cycle for the seventh design set of the Case study-1.

J becomes −142.1209 dB at convergence in 61 cycles, 3.767 s. For DE, J converges to −157.8768 dB at iteration cycle 67 in 3.545 s. So, DE is better and faster than RGA.

Table 2 depicts all simulation results for the Case study-1. RGA results in the minimum error for design set numbers 1 and 2. RGA based approach for the first design set of the Case study-1 results in $C_L = 1.2165$ pF, $(W/L) = 1.5458$, $t_f = 4.3680$ ns and the error $= 2.1652 \times 10^{-15}$. PSO based approach as reported recently in Vural et al. (2011) for this design set results in $t_f = 1.777$ ns. So, RGA based fall time (t_f) is more than PSO based result for this design set.

RGA based approach for the second design set of the Case study-1 results in $C_L = 1.0804$ pF, $(W/L) = 1.1057$, $t_f = 5.4413$ ns and error $= 1.8396 \times 10^{-15}$. PSO based switching characterization as reported in Vural et al. (2011) for this design set results in $t_f = 3.536$ ns. Therefore, PSO based t_f is less than RGA based t_f for this design set. Similarly, RGA based t_f values are more than PSO based t_f values for all the design sets of the Case study-1.

For the same Case study-1, DE achieves the least errors for the design set numbers 4 and 7. RGA based results for the fourth design set are $C_L = 3.3416$ pF, $(W/L) = 1.7120$, $t_f = 10.7982$ ns and error $= 8.9002 \times 10^{-15}$. DE based results

Table 4
Synthesized results of RGA, DE and PSO (Vural et al., 2011, 2010) for the Case study-2.

Design set no.	RGA based results						DE based results						PSO based results (Vural et al., 2011, 2010)	
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	Error (CF) (ps)	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	Error (CF) (ps)	t_f (ns)	t_r (ns)
1	1.2874	1.5557	8.4922	4.5367	4.5267	10.031	0.5433	2.0307	11.0782	1.4667	1.4644	2.2776	1.86	1.86
2	1.2577	1.0886	5.9107	6.3333	6.3535	20.231	0.6135	1.6043	8.6990	2.0965	2.1060	9.4555	2.31	2.31
3	1.9176	1.2087	6.6391	8.6975	8.6244	73.110	0.8355	0.6638	3.6120	6.9001	6.9068	6.6610	7.07	7.07
4	1.3297	2.6852	14.6899	2.7148	2.7029	11.901	0.4762	3.0791	16.7597	0.84778	0.84836	0. 58846	0.87	0.87
5	1.4851	1.6222	8.8055	10.038	10.072	34.404	0.7134	1.0670	5.8258	3.6652	3.6565	8.6944	3.77	3.78
6	1.4934	1.0106	5.4471	8.1016	8.1866	85.044	0.9265	1.0234	5.5782	4.9630	4.9593	3.6514	6.18	6.19
7	2.9905	1.5011	8.1406	10.921	10.969	47.963	1.4241	1.6304	8.8805	4.7885	4.7815	7.0859	5.54	5.54
8	3.4164	1.8942	10.3060	9.8874	9.8983	10.896	1.7222	1.5237	8.2925	6.1966	6.2015	4.8699	7.79	7.79

Table 5
Delay limits, design parameters bound and PSO reported results for the Case study-3.

Design set no.	Specified ranges							PSO reported results (Vural et al., 2011)			
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)
1	0.2–4	1.1–6.1	2.8–19.3	1.1–13	1.1–13	0.5–10	0.5–10	1.18	1.18	0.55	0.50
2	0.1–5.1	1.6–7.1	1.8–18	1.1–15	1.1–15	0.5–8	0.5–8	1.18	1.18	0.55	0.50
3	0.47–2	1.4–6.7	3.2–38	0.5–12	0.5–12	0.2–9	0.2–9	0.50	0.50	0.23	0.21
4	0.1–1.1	1.2–7	1.5–17.5	0.5–5	0.5–5	0.2–4	0.2–4	0.50	0.50	0.23	0.21
5	0.2–14	1.9–5.0	2.7–17	0.7–6	0.7–6	0.4–5	0.4–5	0.94	0.94	0.44	0.40
6	0.3–3.6	1.3–3.5	3.5–16.2	0.25–7	0.25–7	0.3–4.5	0.3–4.5	0.78	0.78	0.36	0.33
7	0.2–4.9	1.1–5.8	2.2–25.3	0.5–6.6	0.5–6.6	0.2–7.7	0.2–7.7	0.53	0.53	0.25	0.23
8	0.2–3.5	0.3–7.6	1.3–39	0.3–6.6	0.3–6.6	0.1–4.4	0.1–4.4	0.32	0.32	0.15	0.14

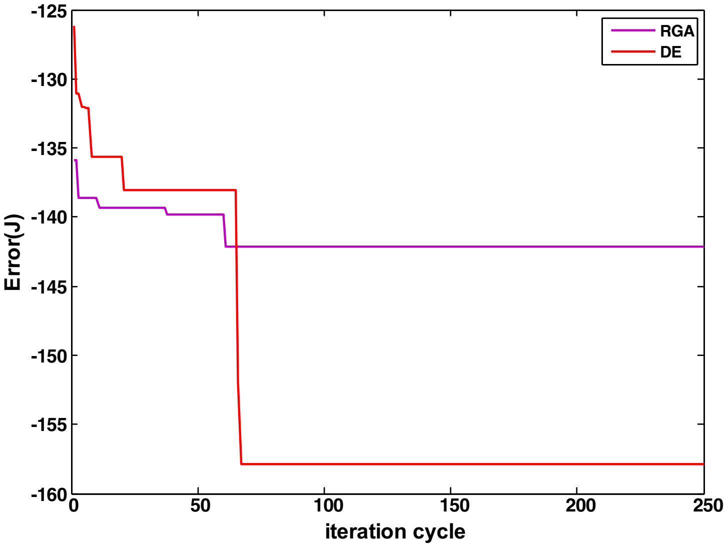

Fig. 4. Plot of J versus iteration cycle for the seventh design set in the Case study-1.

for this design set are $C_L = 1.0773$ pF, $(W/L) = 1.7850$, $t_f = 3.3446$ ns and error = 0.37518×10^{-15}. PSO based results (Vural et al., 2011) for this design set are $t_f = 4.627$ ns. So, DE based t_f is the least one as compared to RGA and PSO based t_f values for this design set.

RGA based results for the seventh design set of the same Case study are $C_L = 2.2418$ pF, $(W/L) = 2.8412$, $t_f = 4.3650$ ns and error = 6.1363×10^{-15}. DE based results for this design set are $C_L = 0.8869$ pF, $(W/L) = 2.5695$, $t_f = 1.9132$ ns and error = 0.16305×10^{-15}. PSO based results as reported in Vural et al. (2011) for this design set are $t_f = 2.028$ ns. Therefore, DE based t_f is the least one as compared to RGA and PSO based t_f values for this design set. Table 2 shows DE based t_f is the best compared to RGA and the PSO based t_f values for all the design sets of the Case study-1. Thus, the proposed DE has proven to be the best optimizer in this case study.

For the Case study-2, the CMOS inverter having equal t_r and t_f of output voltage is designed. RGA and DE algorithms are applied to minimize the difference between t_f and t_r of the output voltage. Eight different design sets are considered to encompass a wide range of CMOS parameters. Aspect ratios of PMOS and NMOS transistors have been found from the synthesized results.

Fig. 5 shows the plot of J versus iteration cycle for the fourth design set of the Case study-2. At iteration cycle 61, J converges to −109.2442 dB and it remains fixed up to 250 iteration cycles. Execution time taken by RGA up to the convergence is 3.871 s. For DE, at iteration cycle 52, J converges to −122.3028 dB and it remains constant up to maximum iteration cycles. Execution time required by DE up to the convergence is 3.214 s. So, DE proves to be better and faster than RGA in this case study also.

Table 6
RGA and DE based results for the Case study-3.

Design set no.	RGA based results								DE based results							
	C_L (pF)	$(W/L)_\mathrm{n}$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (CF) (ps)	C_L (pF)	$(W/L)_\mathrm{n}$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (CF) (ps)
1	1.6080	3.4587	18.7995	2.5487	2.5540	1.0814	1.1235	47.524	0.6479	3.0105	16.5350	1.1798	1.1700	0.50058	0.51469	23.946
2	1.2625	2.7157	14.8586	2.5485	2.5370	1.0813	1.1161	46.239	0.5574	2.5901	14.2747	1.1799	1.1660	0.50059	0.51295	26.171
3	1.2872	6.0924	34.4854	1.1583	1.1146	0.49144	0.49031	44.836	0.5318	5.7294	31.4609	0.50885	0.50474	0.21590	0.22204	10.257
4	0.6901	2.3398	12.6490	1.6169	1.6291	0.68601	0.71666	42.862	0.2835	3.0718	16.6818	0.50601	0.50752	0.21469	0.22326	10.081
5	0.7487	2.2411	12.4224	1.8316	1.7997	0.77709	0.79172	46.441	0.5245	3.0381	16.9208	0.94638	0.92553	0.40153	0.40715	26.472
6	0.8194	2.2130	12.2291	2.0298	2.0007	0.86120	0.88011	48.033	0.3355	2.3897	13.1027	0.76957	0.76447	0.32651	0.33630	14.881
7	1.229	4.1733	23.2164	1.6144	1.5880	0.68495	0.69539	44.063	0.3407	3.6425	19.9899	0.51278	0.50893	0.21756	0.22388	10.169
8	2.2245	6.2761	34.7111	1.9431	1.9136	0.82440	0.84180	46.881	0.2911	5.2430	28.7531	0.30434	0.30227	0.12912	0.13297	5.9165

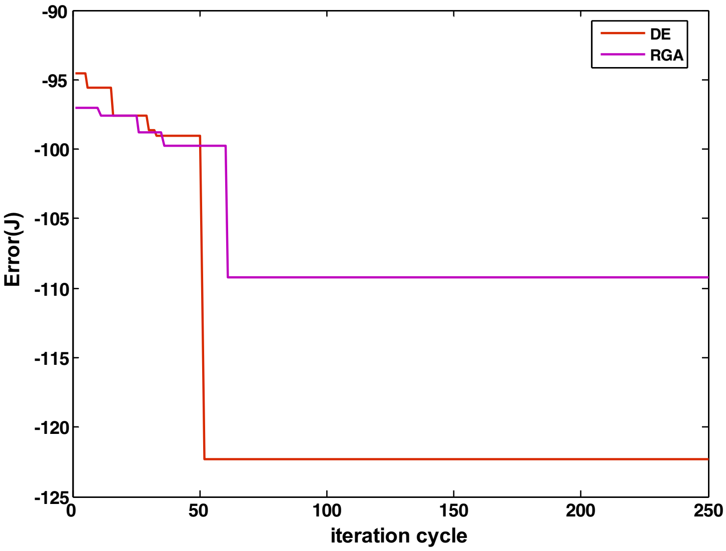

Fig. 5. Plot of J versus iteration cycle for the fourth design set of the Case study-2.

Table 4 shows all simulation results for the Case study-2. RGA gives minimum error for design set numbers 1and 8. RGA based approach for the first design set results in $C_L = 1.2874$ pF, $(W/L)_n = 1.5557$, $(W/L)_p = 8.4922$, $t_f = 4.5367$ ns, $t_r = 4.5267$ ns and error = 10.031 ps. PSO based approach (Vural et al., 2011) for this design set results in $t_f = 1.86$ ns, $t_r = 1.86$ ns. Therefore, RGA shows inferior results than PSO based results for t_f and t_r.

RGA based approach for the eighth design set of the Case study-2 results in $C_L = 3.4164$ pF, $(W/L)_n = 1.8942$, $(W/L)_p = 10.3060$, $t_f = 9.8874$ ns, $t_r = 9.8983$ ns and error = 10.896 ps. PSO based approach as reported in a recent literature (Vural et al., 2010) for this design set results in $t_f = 7.79$ ns, $t_r = 7.79$ ns. So, RGA shows worse results in terms of t_f and t_r than PSO for this design set. RGA based t_f and t_r are more than PSO based corresponding results for all the design sets of the Case study-2.

For the same Case study, DE yields the minimum error for design set numbers 1 and 4. RGA based results for the first design set are $t_f = 4.5367$ ns, $t_r = 4.5267$ ns and error = 10.031 ps. DE based results for this design set are $C_L = 0.5433$ pF, $(W/L)_n = 2.0307$, $(W/L)_p = 11.0782$, $t_f = 1.4667$ ns, $t_r = 1.4644$ ns and error = 2.2776 ps. PSO based results (Vural et al., 2011) for this design set are $t_f = 1.86$ ns, $t_r = 1.86$ ns. This is to be noted that (Vural et al., 2011) has considered only two decimal points for the calculation of t_r and t_f and hence, the error between t_r and t_f is apparently zero. The DE algorithm based inverter design yields the best results in terms of t_f and t_r than those of RGA based design and PSO based design as reported in Vural et al. (2011) for this design set.

RGA based results for the fourth design set of the Case study-2 are $C_L = 1.3297$ pF, $(W/L)_n = 2.6852$, $(W/L)_p = 14.6899$, $t_f = 2.7148$ ns, $t_r = 2.7029$ ns and error = 11.901 ps. DE based results for this design set are $C_L = 0.4762$ pF $(W/L)_n = 3.0791$, $(W/L)_p = 16.7597$, $t_f = 0.84778$ ns, $t_r = 0.84836$ ns and error = 0.58846 ps. PSO based results (Vural et al., 2011) for this design set are $t_f = 0.87$ ns, $t_r = 0.87$ ns. So, DE based design shows the best results in terms of t_f and t_r than RGA and PSO based results (Vural et al., 2011) for this design set. Similarly, DE based t_f and t_r are the least as compared to RGA and PSO based results (Vural et al., 2011) for all other design sets of the Case study-2. So, DE based results can be used for faster switching operation of CMOS inverter.

For the Case study-3, the CMOS inverter having improved symmetrical output voltage with equal t_r and $\mathrm{t_f}$ and equal t_{pHL} and t_{pLH} is designed. RGA and DE algorithms are applied for the eight different design sets.

Fig. 6 shows the plot of J versus iteration cycle for the eighth design set of the Case study-3. It is observed that J converges to −103.2900 dB at iteration cycle 41 and it remains constant upto 500 maximum iteration cycles. Execution time taken by RGA up to convergence is 4.691 s. For DE, J converges to −112.2794 dB at iteration cycle 31 and it remains fixed up to the maximum iteration cycles. Execution time required by DE up to convergence is 3.982 s. So, DE becomes better and faster than RGA for this case study also.

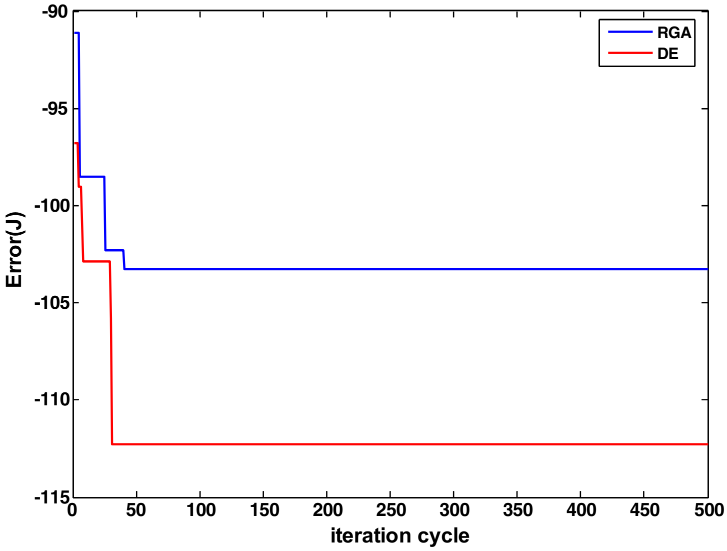

Fig. 6. Plot of J versus iteration cycle for the eighth design set of the Case study-3.

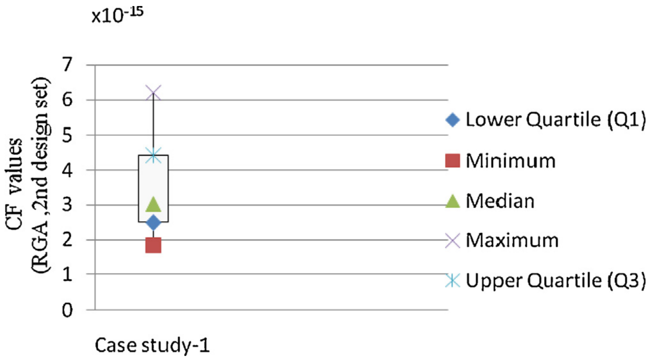

Fig. 7. Box and whisker plots of RGA for the second design set of Case study-1 over 50 runs.

Table 5 shows PSO based results (Vural et al., 2011) Table 6 shows RGA and DE based results for the Case study-3. RGA results in the minimum error for design set numbers 4 and 7. RGA based approach for the fourth design set results in $C_L = 0.6901$ pF, $(W/L)_n = 2.3398$, $(W/L)_p = 12.4690$, $t_f = 1.6169$ ns, $t_r = 1.6291$ ns, $t_{pHL} = 0.68601$ ns, $t_{pLH} = 0.71666$ ns and error = 42.862 ps. PSO based results (Vural et al., 2011) for this design set are $t_f = 0.50$ ns, $t_r = 0.50$ ns, $t_{pHL} = 0.23$ ns, $t_{pLH} = 0.21$ ns. Therefore, RGA shows inferior results than PSO based results for t_f, t_r and propagation delay times (t_{pHL}, t_{pLH}).

RGA based approach for the seventh design set of the Case study-3 results in $C_L = 1.2290$ pF, $(W/L)_n = 4.1733$, $(W/L)_p = 23.2164$, $t_f = 1.6144$ ns, $t_r = 1.5880$ ns, $t_{pHL} = 0.68495$ ns, $t_{pLH} = 0.69539$ ns and error = 44.063 ps. PSO based results (Vural et al., 2011) for this design set are $t_f = 0.53$ ns, $t_r = 0.53$ ns, $t_{pHL} = 0.25$ ns, $t_{pLH} = 0.23$ ns. So, RGA shows worse results than PSO for t_f, t_r and t_{pHL}, t_{pLH}. Similarly, RGA based t_f, t_r and t_{pHL}, t_{pLH} are more than PSO based corresponding results for all the design sets of the Case study-3.

For the same case study, DE yields the least errors for design set numbers 4 and 8. RGA based results for the 4th design set are $t_f = 1.6169$ ns, $t_r = 1.6291$ ns, $t_{pHL} = 0.68601$ ns, $t_{pLH} = 0.71666$ ns and error = 42.862 ps. DE based results for this design set are $C_L = 0.2835$ pF, $(W/L)_n = 3.0718$, $(W/L)_p = 16.6818$, $t_f = 0.50601$ ns, $t_r = 0.50752$ ns, $t_{pHL} = 0.21469$ ns, $t_{pLH} = 0.22326$ ns and error = 10.081 ps. PSO based results (Vural et al., 2011) for this design set are $t_f = 0.50$ ns, $t_r = 0.50$ ns, $t_{pHL} = 0.23$ ns, $t_{pLH} = 0.21$ ns. The error in PSO based result is 20 ps. Therefore, DE shows the best results as compared to RGA based results for t_f, t_r, t_{pHL}, t_{pLH}. As error for DE based result is lesser than PSO based result, so DE based results produce a much improved symmetric output waveform for the CMOS inverter.

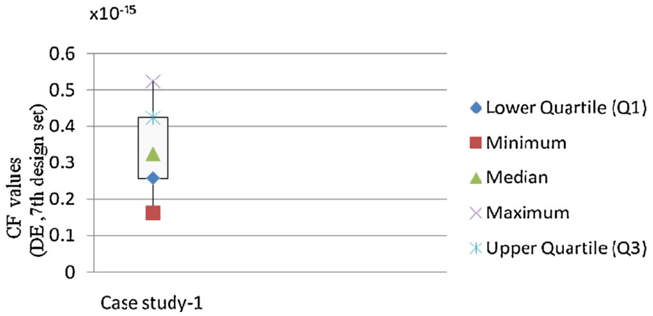

Fig. 8. Box and whisker plots of DE for the seventh design set of Case study-1 over 50 runs.

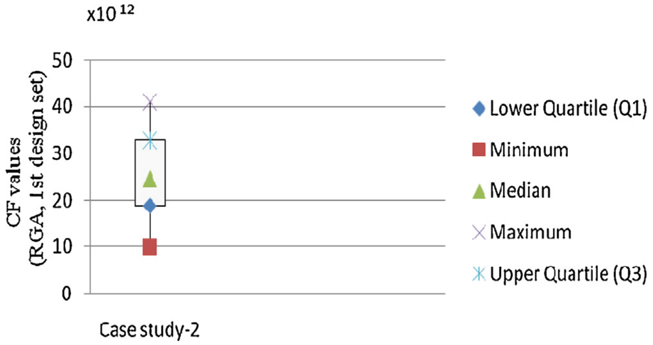

Fig. 9. Box and whisker plots of RGA for the first design set of Case study-2 over 50 runs.

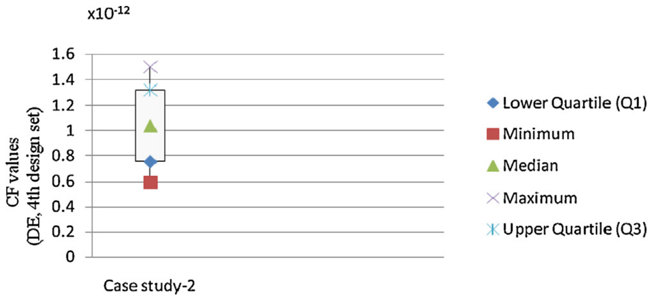

Fig. 10. Box and whisker plots of DE for the fourth design set of Case study-2 over 50 runs.

RGA based results for the eighth design set of the Case study-3 are $C_L = 2.2245$ pF, $(W/L)_n = 6.2761$, $(W/L)_p = 34.7111$, $t_f = 1.9431$ ns, $t_r = 1.9136$ ns, $t_{pHL} = 0.82440$ ns, $t_{pLH} = 0.84180$ ns and error = 46.881 ps. DE based results are $C_L = 0.2911$ pF, $(W/L)_n = 5.2430$, $(W/L)_p = 28.7531$, $t_f = 0.30434$ ns, $t_r = 0.30227$ ns, $t_{pHL} = 0.12912$ ns, $t_{pLH} = 0.13297$ ns and error = 5.9165 ps. PSO based results (Vural et al., 2011) for the same design set are $t_f = 0.32$ ns, $t_r = 0.32$ ns, $t_{pHL} = 0.15$ ns, $t_{pLH} = 0.14$ ns. So, DE shows the best results than RGA and PSO (Vural et al., 2011) based results in terms of t_f, t_r, t_{pHL}, t_{pLH}. Hence, DE produces much better symmetric output waveform for the CMOS inverter than RGA and PSO for all the design sets of the Case study-3.

5.1. Two statistical tests

5.1.1. Box and Whisker plots

RGA and DE have been run 50 times for each design set of all the Case studies and the resulting CF value obtained in each run has been used for box and whisker plots. Figs. 7–12 shows the box and whisker plots of the best design set of RGA and DE for all case studies, respectively. Upper and lower ends of boxes represent the 75th and 25th percentiles. Median is represented by the green colour. The whiskers are lines extending from each end of the boxes to show the

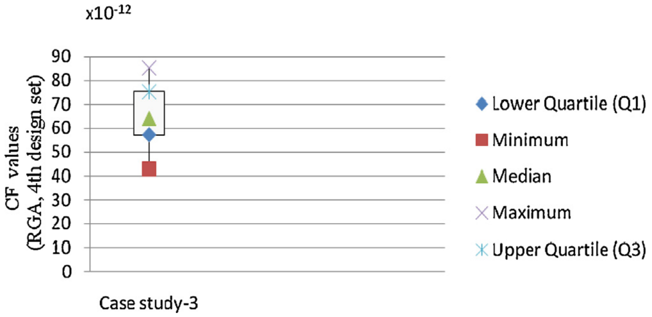

Fig. 11. Box and whisker plots of RGA for the fourth design set of Case study-3 over 50 runs.

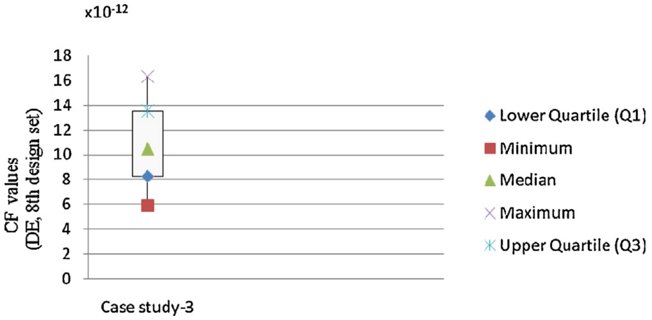

Fig. 12. Box and whisker plots of DE for the eighth design set of Case study-3 over 50 runs.

Table 7
PSPICE results vs. RGA based results for the Case study-1.

Design set no.	PSPICE inputs		PSPICE results	RGA based result
	C_L (pF)	(W/L)	t_f (ns)	t_f (ns)
1	1.2165	1.5458	7.1999	4.3680
2	1.0840	1.1057	8.5670	5.4413
3	3.3646	2.3197	13.871	8.0324
4	3.3416	1.7120	17.745	10.7982
5	1.6772	1.0914	13.385	8.5011
6	2.1880	2.2381	9.3984	5.4335
7	2.2418	2.8412	7.9902	4.3650
8	1.3954	1.5224	8.3109	5.0620

extent of the rest of the data. Outliers are data with values beyond the ends of the whiskers. From Figs. 7–12, it is evident that the lowest value of CF obtained by DE is lower than the CF obtained using RGA for all the case studies. The median of the CF values obtained by DE is lower than that of RGA. So, DE performs more stably.

5.1.2. *t-test*

The *t*-values between the best design sets for RGA and DE with different case studies are shown in Table 13. The *t*-values of all the case studies are larger than 2.15 (degree of freedom = 49), meaning that there is a significant difference between RGA and DE with a 98% confidence level. Thus, from statistical analysis, it is clear that the DE based optimization technique is a much better algorithm than RGA with the 98% confidence level; DE offers more robust and promising results.

Table 8
PSPICE results vs. DE based results for the Case study-1.

Design set no.	PSPICE inputs		PSPICE results	DE based results
	C_L (pF)	(W/L)	t_f (ns)	t_f (ns)
1	0.4189	2.7076	1.6047	0.8560
2	0.5628	1.0095	4.9084	3.0924
3	0.8825	1.8661	4.4061	2.6195
4	1.0773	1.7850	5.6780	3.3446
5	1.6439	4.0936	4.5074	2.2265
6	0.9140	2.2312	3.9477	2.2690
7	0.8869	2.5695	3.4649	1.9132
8	0.8041	1.8959	4.0196	2.3525

Table 9
PSPICE results vs. RGA based results for the Case study-2.

Design set no.	PSPICE inputs			PSPICE results			RGA based results		
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	Error (ns)	t_f (ns)	t_r (ns)	Error (ps)
1	1.2874	1.5557	8.4922	7.5624	6.6399	0.9225	4.5367	4.5267	10.031
2	1.2577	1.0886	5.9107	10.095	9.0060	1.089	6.3333	6.3535	20.231
3	1.9176	1.2087	6.6391	13.870	12.375	1.495	8.6975	8.6244	73.110
4	1.3297	2.6852	14.6899	4.9882	3.9594	1.0288	2.7148	2.7029	11.901
5	1.4851	1.6222	8.8055	16.603	14.513	2.090	10.038	10.072	34.404
6	1.4934	1.0106	5.4471	12.827	11.668	1.159	8.1016	8.1866	85.044
7	2.9905	1.5011	8.1406	17.817	15.745	2.072	10.921	10.969	47.963
8	3.4164	1.8942	10.3060	16.722	14.234	2.488	9.8874	9.8983	10.896

Table 10
PSPICE results vs. DE based results for the Case study-2.

Design set no.	PSPICE inputs			PSPICE results			DE based results		
	C_L (pF)	$(W/L)_n$	$(W/L)_p$	t_f (ns)	t_r (ns)	Error (ns)	t_f (ns)	t_r (ns)	Error (ps)
1	0.5433	2.0307	11.0782	2.6076	2.2515	0.3561	1.4667	1.4644	2.2776
2	0.6135	1.6043	8.6990	3.5568	3.1448	0.4120	2.0965	2.1060	9.4555
3	0.8355	0.6638	3.6120	10.771	9.7678	1.0032	6.9001	6.9068	6.6610
4	0.4762	3.0791	16.7597	1.6675	1.5193	0.1482	0.84778	0.84836	0. 58846
5	0.7134	1.0670	5.8258	5.9707	5.3196	0.6511	3.6652	3.6565	8.6944
6	0.9265	1.0234	5.5782	7.9253	7.1466	0.7787	4.9630	4.9593	3.6514
7	1.4241	1.6304	8.8805	7.9999	7.0167	0.9832	4.7885	4.7815	7.0859
8	1.7222	1.5237	8.2925	10.197	8.9318	1.2652	6.1966	6.2015	4.8699

5.2. *PSPICE based results*

In order to authenticate the results achieved through RGA and DE optimizations, the inverters are redesigned using PSPICE with synthesized values of output load capacitor and transistor aspect ratios as inputs. PSPICE results are shown in Tables 7–12, respectively. The dissimilarity between PSPICE results and RGA/DE based design results occur from the fact that PSPICE calculates the rise times and fall times using more complex circuit equation sets. The delay expressions, used in RGA, DE based designs are very simple and derived from the simple current–voltage relationships of long-channel transistors. So, the effects of channel velocity saturation and small-geometry effect of transistor are not considered. Thus, PSPICE results in greater delay times compared to RGA, DE based inverter designs.

Table 11
PSPICE results vs. RGA based results for the Case study-3.

Design set no.	PSPICE inputs			PSPICE results					RGA based result				
	C_L (pF)	$(W/L)_{\mathrm{n}}$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (ns)	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (ps)
1	1.6080	3.4587	18.7995	5.0402	3.8420	2.6045	1.5905	2.2122	2.5487	2.5540	1.0814	1.1235	47.524
2	1.2625	2.7157	14.8586	4.6814	3.6916	2.4359	1.6852	1.7405	2.5485	2.5370	1.0813	1.1161	46.239
3	1.2872	6.0924	34.4854	3.3885	1.7639	1.9868	0.89849	2.71291	1.1583	1.1146	0.49144	0.49031	44.836
4	0.6901	2.3398	12.6490	2.8627	2.4943	1.5716	1.1932	0.7468	1.6169	1.6291	0.68601	0.71666	42.862
5	0.7487	2.2411	12.4224	3.2725	2.7016	1.7663	1.3098	1.0274	1.8316	1.7997	0.77709	0.79172	46.441
6	0.8194	2.2130	12.2291	3.6087	2.9753	1.9260	1.4444	1.115	2.0298	2.0007	0.86120	0.88011	48.033
7	1.2290	4.1733	23.2164	4.6131	2.4104	2.6348	1.2041	3.6334	1.6144	1.58808	0.68495	0.69539	44.063
8	2.2245	6.2761	34.7111	5.6578	2.8724	3.1773	1.4272	4.5355	1.9431	1.9136	0.82440	0.84180	46.881

Table 12
PSPICE results vs. DE based results for the Case study-3.

Design set no.	PSPICE inputs			PSPICE results					DE based result				
	C_L (pF)	$(W/L)_\mathrm{n}$	$(W/L)_p$	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (ns)	t_f (ns)	t_r (ns)	t_{pHL} (ns)	t_{pLH} (ns)	Error (ps)
1	0.6479	3.0105	16.5350	2.2815	1.9297	1.2976	0.78088	0.86852	1.1798	1.1700	0.50058	0.51469	23.946
2	0.5574	2.5901	14.2747	2.2184	1.8691	1.2571	0.84869	0.75771	1.1799	1.1660	0.50059	0.51295	26.171
3	0.5318	5.7294	31.4609	1.5350	0.98906	0.97114	0.48180	1.03528	0.50885	0.50474	0.21590	0.22204	10.257
4	0.2835	3.0718	16.6818	1.0829	1.1324	0.70500	0.35765	0.39785	0.50601	0.50752	0.21469	0.22326	10.081
5	0.5245	3.0381	16.9208	1.8534	1.6220	1.0921	0.61351	0.70999	0.94638	0.92553	0.40153	0.40715	26.472
6	0.3355	2.3897	13.1027	1.4671	1.3502	0.89751	0.62555	0.38886	0.76957	0.76447	0.32651	0.33630	14.881
7	0.3407	3.6425	19.9899	1.1988	1.1658	0.74919	0.24554	0.53665	0.51278	0.50893	0.21756	0.22388	10.169
8	0.2911	5.2430	28.7531	0.99595	0.82751	0.67489	0.33563	0.5077	0.30434	0.30227	0.12912	0.13297	5.9165

Table 13
t-values between RGA and DE for different Case studies over 50 runs.

CF values	Case study-1		Case study-2		Case study-3	
	RGA (2nd design set)	DE (7th design set)	RGA (1st design set)	DE (4th design set)	RGA (4th design set)	DE (8th design set)
Minimum	1.8396×10^{-15}	0.16305×10^{-15}	10.031×10^{-12}	0.58846×10^{-12}	42.862×10^{-12}	5.9165×10^{-12}
Maximum	6.1976×10^{-15}	0.5251×10^{-15}	40.926×10^{-12}	1.498×10^{-12}	85.147×10^{-12}	16.279×10^{-12}
Mean	3.0904×10^{-15}	0.3512×10^{-15}	25.044×10^{-12}	1.105×10^{-12}	64.201×10^{-12}	10.754×10^{-12}
Standard deviation	2.2598×10^{-15}	0.0458×10^{-15}	19.591×10^{-12}	0.0459×10^{-12}	49.963×10^{-12}	0.0519×10^{-12}
t-value	8.57 for DE		8.6404 for DE		7.5641 for DE	

6. Conclusion

In this work the utilization of evolutionary algorithms like RGA and DE is investigated to achieve the optimal switching characteristics of CMOS inverter. The algorithms are applied to three different inverter design cases with different ranges of design parameters. DE algorithm confirmed its efficiency in finding the least errors for all design cases. DE yields the best symmetric output waveform of the designed CMOS inverter.

References

Chen, S., Luk, B.L., 2010. Digital IIR filter design using particle swarm optimization. Int. J. Model. Ident. Control 9 (4), 327–335.

Chiou, J.P., 2007. Variable scaling hybrid differential evolution for large-scale economic dispatch problems. Electr. Power Syst. Res. 77 (3–4), 212–218.

Das, A., Vemuri, R., 2007. An automated passive analog circuit synthesis framework using genetic algorithms. In: IEEE Computer Society Annual Symposium on VLSI (ISVLSI'07), pp. 145–152.

DeMassa, T.A., Ciccone, Z., 1996. Digital Integrated Circuits. John Wiley & Sons, New York.

Eberhart, R., Shi, Y., 1998. Comparison between genetic algorithm and particle swarm optimization. In: Evolutionary Programming VII. Springer, pp. 611–616.

Fang, W., Sun, J., Xu, W., 2009. A new mutated quantum behaved particle swarm optimizer for digital IIR filter design. EURASIP J. Adv. Signal Process. 2009, 367465, 1–7.

Gill, S.S., Chandel, R., Chandel, A., 2009. Comparative study of ant colony and genetic algorithms for VLSI circuit partitioning. World Acad. Sci. Eng. Technol. 28, 890–894.

Gudise, V.G., Venayagamoorthy, G.K., 2004. FPGA placement and routing using particle swarm optimization. In: IEEE Computer Society Annual Symposium on VLSI, USA, February 19–20, pp. 307–308.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. Univ. Michigan Press, Ann Arbor, MI.

Hussain, Z.M., Sadik, A.Z., O'Shea, P., 2011. Digital Signal Processing – An Introduction with MATLAB Applications. Springer-Verlag.

http://www.mosis.com/pages/Technical/Testdata/tsmc-025-prm.

Karaboga, N., 2005. Digital IIR filter design using differential evolution algorithm. EURASIP J. Appl. Signal Process. Hindwai Publishing Corp. 2005, 8, 1269–1276.

Karaboga, N., 2009. A new design method based on artificial bee colony algorithm for digital IIR filters. J. Franklin Inst. 346, 328–348.

Karaboga, N., Cetinkaya, B., 2006. Design of digital FIR filters using differential evolution algorithm. Circ. Syst. Signal Process. 25 (5), 649–660.

Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. Proc. IEEE Int. Conf. Neural Netw. 4, 1942–1948.

Krusienski, D.J., Jenkins, W.K., 2003. Adaptive filtering via particle swarm optimization. In: Proc. 37th Asilomar Conf. on Signals, Systems and Computers, vol. 1, November, pp. 571–575.

Lai, C., 2006. A novel image segmentation approach based on particle swarm optimization. IEICE Trans. Fund. Electron. Commun. Comput. Sci. E89-A (January (1)), 324–327.

Lin, C., Qing, A., Feng, Q., 2010. Synthesis of unequally spaced antenna arrays by using differential evolution. IEEE Trans. Antenna Propag. 58 (8), 2553–2561.

Lu, Hung-Ching, Tzeng, Shian-Tang, 2000. Design of arbitrary FIR log filters by genetic algorithm approach. Signal Process. 80 (3), 497–505.

Luitel, B., Venayagamoorthy, G.K., 2010. Particle swarm optimization with quantum infusion for system identification. Eng. Appl. Artif. Intell. 23, 635–649.

Ma, Q., Cowan, C.F.N., 1996. Genetic algorithms applied to the adaptation of IIR filters. Signal Process. 48 (2), 155–163.

Mandal, K.K., Chakraborty, N., 2008. Differential evolution technique based short-term economic generation scheduling of hydrothermal systems. Electric Power Syst. Res. 78 (11), 1972–1979.

Mandal, S., Ghoshal, S.P., Kar, R., Mandal, D., 2011. Design of optimal linear phase FIR high pass filter using craziness based particle swarm optimization technique. J. King Saud Univ. – Comput. Inf. Sci., http://dx.doi.org/10.1016/j.jksuci.2011.10.007.

Mastorakis, N.E., Gonos, I.F., Swamy, M.N.S., 2003. Design of two dimensional recursive filters using genetic algorithms. IEEE Trans. Circuits Syst. – I: Fundam. Theory Appl. 50 (5), 634–639.

Mondal, S., Ghoshal, S.P., Kar, R., Mandal, D., 2012. Craziness based particle swarm optimization algorithm for FIR band stop filter design. J. Swarm Evol. Optim. 7, 58–64.

Mukhopadhyay, J., Pandit, S., 2012. Design of a nano-scale CMOS inverter with symmetric switching characteristics using particle swarm optimization algorithm. In: IEEE 2nd Annual International Conference on Innovative Techno-Management Solutions for Social Sector (IEMCON 2012), India, January 17–18.

Pei, Z., Zhao, Y., Liu, Z., 2009. Image Segmentation Based on Differential Evolution Algorithm. In: International Conference on Image Analysis and Signal Processing, IASP 2009, April 11–12, pp. 48–51.

Saha, S.K., Kar, R., Mandal, D., Ghoshal, S.P., 2011. IIR filter design with craziness based particle swarm optimization technique. World Acad. Sci. Eng. Technol. 60, 1628–1635.

Storn, R., Price, K., 1995. Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report. International Computer Science Institute, Berkley.

Kang, S.-M., Leblebici, Y., 2003. CMOS Digital Integrated Circuits Analysis and Design. Tata McGraw-Hill Edition, New Delhi.

Tang, M., Lau, R.Y.K., 2007. A Parallel genetic algorithm for floorplan area optimization. In: Seventh International Conference on Intelligent Systems Design and Application, Brazil, pp. 801–806.

Ulker, S., 2008. Particle swarm optimization applications to microwave circuits. Microwave Opt. Technol. Lett. 50 (5), 1333–1336.

Vijayakumar, S., Sudhakar, J.G., Muthukumar, G.G., Victoire, T.A.A., 2009. A differential evolution algorithm for restrictive channel routing problem in VLSI circuit design. In: World Congress on Nature & Biologically Inspired Computing, NaBIC 2009, Coimbatore, India, December 9–11, pp. 1258–1263.

Vural, R.A., Yildirim, T., 2012. Analog circuit sizing via swarm intelligence. Int. J. Electron. Commun. (AEÜ) 66 (9), 732–740.

Vural, A., Der, O., Yildirim, T., 2010. Particle swarm optimization based inverter design considering transient performance. Digital Signal Process. 20 (4), 1215–1220.

Vural, A., Der, O., Yildirim, T., 2011. Investigation of particle swarm optimization for switching characterization of inverter design. Expert Syst. Appl. 38 (5), 5696–5703.

Walpole, R.E., Myer, R.H., 1978. Probability and Statistics for Engineers and Scientists. Macmillan, New York.