A new asynchronous parallel global optimization method based on simulated annealing and differential evolution

Jernej Olenšek, Tadej Tuma, Janez Puhan, Árpád Bűrmen

University of Ljubljana, Faculty of electrical engineering, Tržaška 25, SI-1000 Ljubljana, Slovenia

Abstract

This paper presents a new asynchronous parallel global optimization method and its application to the automated device sizing in analog integrated circuit (IC) design. The method is based on the simulated annealing algorithm (SA), but incorporates features from differential evolution (DE) to improve the sampling efficiency and avoid the problems involved with the cooling schedule selection. A simple local search procedure is also incorporated to improve the fine tuning capabilities of the method. To reduce the optimization time, the method is designed as an asynchronous master-slave parallel system that allows simultaneous evaluation of several trial solutions. Comparison with simple SA and DE on a set of well known analytical test functions confirms the method's efficiency. The parallel efficiency of the method is also verified by optimizing the functions with 1, 2, 4, and 8 processors. The proposed approach is also applied to several real world cases of device sizing in analog IC design. The optimization results indicate that the method is capable of finding near optimal circuits. The parallel efficiency of the method is confirmed with optimization runs on 1, 2, 4, and 8 processors.

Key words: simulated annealing, differential evolution, parallel global optimization, analog integrated circuit design

Email address: jernej.olensek@fe.uni-lj.si (Jernej Olenšek)

1. Introduction

Global optimization problems arise in almost every field of scientific research, engineering, chemistry, economy, etc. Many real world problems can be formulated as global optimization problems of the following form:

$$\begin{gathered} x^* = \arg\min_{x \in S} f(x) \\ f : S \to \mathbb{R} \\ S = \left\{ x, x \in \mathbb{R}^{N_v}, l_i \leq x_i \leq u_i, i = 1, ...N_v \right\} \end{gathered} \tag{1}$$

where $f(x)$ is the so-called cost function (CF), x is an N_v-dimensional vector of optimization variables, and l_i and u_i are the lower and the upper bound for the i-th variable, respectively. x^* denotes the global minimum of the CF. In this paper we only consider problems with simple box constraints.

In practical applications the actual shape of the CF is usually unknown. Often the CF values are the result of expensive and time consuming measurements or simulations. In such cases problem (1) can't be solved analytically. Many different classes of optimization methods have been developed to solve the problem numerically. Gradient methods are the fastest, but they require the derivatives of the CF and work only on differentiable functions. They are also extremely local by nature and sensitive to noise. This reduces their suitability for many practical applications. The alternative to the gradient methods are the direct search methods. They do not require gradients of the CF and can handle noisy and multimodal functions. Optimization methods can also be classified as local or global. The former are designed to find a minimum as fast as possible, even though it may not be the true global minimum. The latter are usually slower but can find the true global minimum with high probability. There are also many different hybrid methods that try to exploit the fast convergence of the local methods and the global search capabilities of the global methods [1, 2, 3, 4, 5, 6].

The basis of the method presented in this paper is a hybrid method (DESA)[7] that combines simulated annealing (SA) and differential evolution (DE). The random sampling and the Metropolis criterion from SA [8] are combined with the population of points and the sampling mechanism from DE [9] to balance global and local search. We extend DESA so that it can be run in parallel on a cluster of computers. The new method is referred to as Parallel Simulated Annealing Differential Evolution (PSADE).

The paper is organized as follows. In sections 2 and 3 the simple SA and the basic DE algorithms are summarized. Section 4 presents a brief classification of parallel optimization approaches. In section 5 PSADE is presented in more detail. Section 6 compares PSADE with the original DE and simple SA on a set of 23 well known mathematical test functions. In section 7 PSADE is applied to the problem of device sizing in analog integrated circuit (IC) design. Section 8 gives the concluding remarks.

1.1. Notation

We use $U(1, N)$ to denote a uniformly distributed random integer from $\{1, 2, ..., N\}$, and $U(0, 1)$ to denote a uniformly distributed random number from the (0,1) interval. Superscripts denote different vectors, while subscripts denote vector components. Parentheses are used to denote iteration numbers. $x_n^i(k)$ for example denotes the n-th component of the i-th vector in k-th iteration.

2. Simulated annealing

SA is a very popular stochastic sequential global optimization algorithm that performs random sampling of the search space [8]. Its main feature is the so-called Metropolis criterion that occasionally allows the acceptance of inferior solutions. The probability of making the transition from the current point x^c to a trial point x^t generated by randomly perturbing x^c is defined as:

$$P = \min(1, e^{-\frac{f(x^t)-f(x^c)}{T}}) \tag{2}$$

where $f(x^t)$ and $f(x^c)$ denote the CF values at x^t and x^c, respectively. SA always accepts trail points with lower CF value, while the probability of accepting inferior points is controlled by the temperature parameter T. For a high value of T this probability is close to one. During the annealing process, T is reduced according to some predefined cooling schedule, which decreases the acceptance probability for low quality solutions. This mechanism allows the method to escape from a local minimum when T is large and to fine tune the solutions when T is small.

SA has a few drawbacks. In order to guarantee convergence to a global minimum, careful selection of the sampling mechanism and the cooling schedule is required. In [10] for example, several convergent versions of SA are

derived. In practice however such algorithms are usually too slow and inefficient. Often modifications are required that sacrifice global convergence for speed. This way acceptable results can be obtained in a reasonable amount of time. The optimal sampling mechanism and cooling schedule depend on the problem and finding them is not a trivial task. If the cooling schedule is too fast, SA gets trapped in a local minimum. If it is too slow, the optimization takes too much time.

In our implementation of SA the initial temperature was determined by the maximal CF difference of 100 random initial points. We allowed $2 \cdot N_v$ steps in every temperature stage. Trial steps were generated independently for every optimization parameter according to the Cauchy probability distribution (3) with range parameter R:

$$p(r_i) = \frac{R}{\pi \cdot ({r_i}^2 + R^2)}, \ i = 1, 2, ...N_v \tag{3}$$

To simplify the implementation all optimization parameters are normalized to the [0,1] interval so only one R parameter is needed. Its initial value is set to 1. SA was run with a predefined limit on the number of CF evaluations (CFE). The temperature and range cooling coefficients were determined separately so that T (R) reached $T_{min} = 10^{-10}$ ($R_{min} = 10^{-6}$) when the CFE limit was reached. This setup does not guarantee the convergence to the global minimum but allows SA to reach the fine tuning stage (where T and R are small) even for those cases where the initial temperature must be set to a very large value.

3. Differential evolution

DE is another very popular optimization method. Unlike the serial SA, DE uses a population of N_p points to guide the search process [9]. In its various forms DE has been applied to many real world problems (e.g. [11, 12, 13, 14]). In our experiments we used the scheme classified as DE/rand/bin [9]. Algorithm 1 represents the used DE algorithm.

f and p_x denote the weight factor and the crossover probability. n_x ensures that at least one component of x^m is used in the trial point x^t. DE also has some drawbacks. The user is required to provide f and p_x, which are problem dependent and difficult to determine. The distribution of trial points depends on the distribution of the population, thus the search radius

Algorithm 1 Differential evolution

Initialize population
$k := 0$
while stopping conditions not met
    for $i := 1, 2, ...N_p$
        $a := U(1, N_p)$,$b := U(1, N_p)$,$c := U(1, N_p)$, $a \neq b \neq c \neq i$
        $x^m := x^a(k) + (x^b(k) - x^c(k)) \cdot f$
        $n_x := U(1, N_v)$
        for $n := 1, 2, ...N_v$
            if $U(0, 1) < p_x$ **or** $n = n_x$
                $x_n^t := x_n^m$
            else
                $x_n^t := x_n^i(k)$
            end
        end
        if $f(x^t) < f(x^i(k))$
            $x^i(k + 1) := x^t$
        else
            $x^i(k + 1) := x^i(k)$
        end
    end
    $k := k + 1$
end

is limited to a region around the current population. If the entire population converges to a neighborhood of a local minimum, DE can no longer sample points far from that region. The greedy selection that only allows superior solutions to reach the next generation also prevents the population to gradually climb out of a local minimum.

The values of DE parameters used in our experiments are based on the values suggested in literature [9, 15]. We used $N_p = 100$, $F = 0.5$, $P_x = 0.9$.

4. Parallel optimization methods

In many practical applications the computationally most expensive part of the optimization is the CF evaluation. Since the number of CF evaluations (CFE) required to obtain high quality solutions is often very large, the entire optimization can take a very long time. Parallel methods are capable of distributing the workload among several processing units and can achieve considerable speedups when compared to sequential methods. There are two major approaches to parallelization, synchronous and asynchronous. The main feature of synchronous methods are the so-called synchronization points. When a worker reaches a synchronization point it stops and waits until all workers have reached that point. This means that workers must wait for the slowest worker. When the CF evaluation times vary considerably, such idle time can seriously affect the achievable speedup. Idle time can be avoided by using the asynchronous approach where workers do not wait for each other. When a worker finishes a task, it reports the results and immediately starts a new task.

Another classification of parallel methods is based on the type of communication. In master-slave (MS) configuration a single processing unit called the master controls the optimization and distributes the workload among workers. There is no direct communication between workers. In peer-to-peer (P2P) configuration all workers are in constant communication so there is no need for a master process. MS systems are easier to implement and require less communication between processing units, but are sensitive to the response time of the master. P2P systems are more complex and robust, but they require more communication between workers and are more difficult to implement. The proposed hybrid method operates in asynchronous MS configuration.

5. Parallel Simulated Annealing Differential Evolution - PSADE

With PSADE we wish to improve the random sampling of SA with some kind of memory that would allow more efficient sampling of the search space. We replace the serial random search of SA with the population of N_p points and augment the random sampling with a mechanism similar to the original DE. To avoid the difficulties regarding the selection of problem dependent DE parameters (weight factor f and crossover probability p_x), PSADE assigns different parameter values to every population member ($F^i, PX^i, i = 1, 2, ...N_p$). The method modifies these values during the optimization in order to adapt to the specifics of the problem at hand. To avoid the selection of the cooling schedule for SA, PSADE assigns a different constant temperature and constant random sampling radius ($T^i, R^i, i = 1, 2, ...N_p$) to every population member. In every iteration an individual is selected from the population and its T and R values are used in the process of trial point generation and acceptance (Metropolis criterion). The temperature and random step length changes are achieved by exchanging the values of T and R between population members. PSADE also includes a simple local search mechanism to speed up the convergence. To simplify the implementation of PSADE, all optimization variables are normalized to the [0,1] interval. The following sections provide a detailed description of PSADE.

5.1. Parallel asynchronous implementation

PSADE uses the asynchronous MS configuration of workers to parallelize the search process. It was implemented in MATLAB. Communication between the master and the slaves is handled by the DP toolbox [16], that uses the PVM message passing library [17]. The master process maintains the population and all parameters. It is responsible for trial point generation, the acceptance of the evaluated points into the population (Metropolis criterion or greedy acceptance if local step was performed), selection of the search direction for local step, and the termination of the optimization. Algorithm 2 depicts the master process. The slaves perform a single CF evaluation or a one-dimensional local step depending on the instructions received from the master.

5.2. User input

The user must provide the population size N_p, the minimal temperature T_{min}, and the minimal random sampling radius R_{min}. In the experiments

Algorithm 2 PSADE - Master Process

initialization
for all slaves
    generate a trial point and request CF evaluation from slave i
    mark slave i as busy
end
while stopping condition not satisfied
    wait for fresh result
    let i denote the slave that returned the result
    mark slave i as idle
    if slave performed local step
        greedy acceptance
    else
        metropolis acceptance
        if local step needed
            select direction
            request local step from slave i
            mark slave i as busy
        end
    end
    if slave i is idle
        generate a trial point and request CF evaluation from slave i
        mark slave i as busy
    end
end

these parameters were set to $N_p = 20, T_{min} = 10^{-10}, R_{min} = 10^{-6}$. The algorithm also requires two probabilities. τ_1 denotes the probability of a local one-dimensional step after the verification of the Metropolis criterion and τ_2 denotes the probability of adapting the parameters for trial point generation. Their default values are $\tau_1 = 0.01$ and $\tau_2 = 0.1$.

5.3. Initial population

The initial population can be chosen randomly. PSADE uses the Latin hypercube approach to ensure that the entire search space is uniformly covered by the initial population. First the feasible range for every variable is divided into N_p equal subintervals. N_p points are then randomly generated so that every subinterval for every optimization variable is included in the initial population. The values of parameters inside subintervals are chosen randomly.

5.4. Initial parameters of the method

The temperature and the random sampling radius are initialized in the following way:

$$\begin{aligned} c_t &= \tfrac{1}{N_p - 1} \cdot \log(\tfrac{T_{max}}{T_{min}}) \\ T^i &= T_{max} \cdot e^{-c_t \cdot (i-1)}, i = 1, 2, ...N_p \\ c_r &= \tfrac{1}{N_p - 1} \cdot \log(\tfrac{R_{max}}{R_{min}}) \\ R^i &= R_{max} \cdot e^{-c_r \cdot (i-1)}, i = 1, 2, ...N_p \end{aligned} \tag{4}$$

T_{max} is the maximal temperature. It is set to the CF difference between the worst and the best point in the initial population. R_{max} is set to one.

5.5. Trial point generation

First two randomly selected individuals (denoted by i_1, i_2) compete for better T and R parameters. They exchange their T and R values with the following probability:

$$P_T = \min(1, e^{(f(x^{i_1}) - f(x^{i_2})) \cdot (1/T^{i_1} - 1/T^{i_2})}) \tag{5}$$

where y^{i_1}, y^{i_2} denote the CF values of the two individuals, and T^{i_1}, T^{i_2} denote their temperatures. The mechanism tries to assign low T and R to individuals with small CF values while still allowing the opposite from time to time.

After competition, one of the individuals is selected to control the generation and the acceptance of the trial point into the population. The probability (P_S^i) that the i-th individual is selected as the controlling individual (denoted by superscript ic) is based on rank ordering. Individuals with small CF values have smaller rank and higher probability of being selected:

$$P_S^i = \frac{e^{-r^i}}{\sum_{i=1}^{N_p} e^{-r^i}} \tag{6}$$

where r^i denotes the rank of the i-th individual ranging from 1 to N_p. Combined with the competition described in the previous paragraph, this mechanism ensures that low T and R are used more often which indicates exploitation of promising regions. It is possible, however, that individuals with low CF values have large T and R. It is also possible that an individual with large CF value is selected to control the search. In both cases large T and R values are used which enhances global search and allows the method to escape from local minima.

In every iteration a single individual (denoted by it) is selected randomly as the target for replacement. In the original DE the user must provide the crossover probability p_x and the weight factor f (e.g. $p_x = 0.9, f = 0.5$). Since these values are problem dependent and are difficult to select in practice, PSADE assigns a separate set of parameters ($F^i, P_X^i, i = 1, 2, ...N_p$) to every individual. These values can be adapted during the optimization. The values that are used in trial point generation are determined in the following way. In most cases the it-th individual's P_X^{it} and F^{it} value take the role of p_x and f. It is also possible (with some small probability τ_2) that p_x and f are chosen randomly from the [0.1,0.9] and [0.5,1.5] interval, respectively. If the generated trial point is accepted into the population by the Metropolis criterion, the used control parameters replace the existing values of the it-th individual (F^{it}, P_X^{it}). This allows the method to automatically find the parameters that produce acceptable solutions.

A trial point x^t is generated by combining the random step of SA with the DE operator. In contrast with the original DE, PSADE uses two differential vectors to obtain the mutated point. The random step is generated independently for every optimization variable according to the Cauchy probability distribution. The range parameter of the selected controlling individual (R^{ic}) determines the random search radius. The random step increases population diversity and allows the method to escape from local minima, while the DE

operator exploits the knowledge accumulated by the entire population to guide the search process. The pseudo code is depicted by Algorithm 3.

Algorithm 3 Generation of x^t

$a := U(1, N_p)$
$x^m := x^a$
for $id := 1, 2$
    $b := U(1, N_p)$,$c := U(1, N_p)$, $b \neq c$
    $\alpha := U(0, 1) \cdot f$
    $x^m := x^m + (x^b - x^c) \cdot \alpha$
end
for $i := 1, 2, ...N_v$
    if $U(0, 1) < p_x$
        $x_i^t := x_i^m$
    else
        $x_i^t := x_i^{it}$
    end
    $x_i^t := x_i^t + R^{ic} \cdot \tan(\pi \cdot (U(0, 1) - 1/2))$
    if $x_i^t > 1$ **or** $x_i^t < 0$
        $x_i^t = U(0, 1)$
    end
end

5.6. Local search

Any kind of local search procedure can be applied to improve the convergence speed of global optimization methods. PSADE performs a local step with fixed probability τ_1 after the acceptance criterion for it-th individual is checked. The local step is always performed if x^{it} is the best individual in the population. In the local step stage PSADE constructs a one-dimensional quadratic model function using three collinear points. The master first randomly selects two population members (denoted by a and b). They define the search direction $d = x^a - x^b$. The starting point x^{it} and the direction d are sent to a slave that must select two additional points x^{L1} and x^{L2}. x^{L1} is obtained as:

$$x^{L1} = x^{it} + U(0, 1) \cdot d \tag{7}$$

If $f(x^{L1}) < f(x^{it})$, x^{L2} is obtained as:

$$x^{L2} = x^{L1} + 2 \cdot U(0,1) \cdot d \tag{8}$$

otherwise it is obtained as:

$$x^{L2} = x^{it} - 2 \cdot U(0,1) \cdot d \tag{9}$$

Any point violating the box constraints is repeatedly contracted toward x^{it} (by halving its distance) until the violation is resolved. With three collinear sample points (x^{it}, x^{L1}, and x^{L2}), the one-dimensional quadratic model function is constructed. If the model function is not convex, the best of the three sample points is returned to the master. Otherwise the minimum of the model function x^m is calculated. If x^m violates box constraints it is repeatedly contracted toward x^{it} until the violation is resolved. The point with the lowest CF value among x^{it}, x^{L1}, x^{L2}, and x^m is then returned to the master.

5.7. Acceptance criterion

Once the slave reports its results back to the master, the acceptance criterion is applied. Greedy acceptance is applied to local steps. For all other steps the Metropolis criterion (2) is used. The temperature of the controlling individual (T_{ic}) determines the probability of accepting inferior solutions. An exception is the case when the target individual is the best point in the current population. In this case the temperature used in the Metropolis criterion is set to 0 which ensures that the best point can only be replaced by superior solutions. Such elitism is necessary to ensure probabilistic convergence to the global minimum.

Beside replacing the population members, PSADE also uses the Metropolis criterion to search for DE parameters that produce acceptable solutions. If the trial solution is accepted, the DE parameters of the it-th individual are replaced with the values selected in the parameters selection phase (i.e. $P_X^{it} = p_x, F^{it} = f$).

5.8. Convergence

Global convergence of PSADE can be derived in a similar manner as for DESA [7]. The positive random sampling radius ensures that no part of the parameter space is completely excluded from the search process. The positive temperature allows PSADE to climb out of local minima in order to

Table 1: Comparison of PSADE, DE, and SA.

	N_v	f^*	CFE_{max}	f_{min}-SA	f_{min}-DE	f_{min}-PSADE
f_1	30	0	100000	9.99e-06	5.29e-08	3.32e-13
f_2	30	0	100000	1.39e-03	2.39e-04	1.85e-02
f_3	30	0	100000	3.88e-04	30.41	8.71e-01
f_4	30	0	100000	1.80e-03	3.64e-01	1.06e-02
f_5	30	0	100000	98.11	22.41	19.67
f_6	30	0	100000	0	0	0
f_7	30	0	100000	1.66e-02	1.38e-02	7.83e-03
f_8	30	-12569.5	100000	-11858.9	-4980.45	-12569.5
f_9	30	0	100000	8.95	190.75	2.59e-03
f_{10}	30	0	100000	1.70	5.54e-05	3.26e-05
f_{11}	30	0	100000	4.64e-02	1.64e-07	1.05e-10
f_{12}	30	0	100000	2.43e-08	2.38e-09	2.28e-17
f_{13}	30	0	100000	3.51e-07	2.50e-08	1.74e-16
f_{14}	2	0.998	20000	0.998	0.998	0.998
f_{15}	4	3.075e-04	30000	1.520e-03	3.075e-004	3.203e-04
f_{16}	2	-1.0316	20000	-1.0316	-1.0316	-1.0316
f_{17}	2	0.398	20000	0.398	0.398	0.398
f_{18}	2	3	20000	3	3	3
f_{19}	3	-3.863	20000	-3.863	-3.863	-3.863
f_{20}	6	-3.322	20000	-3.239	-3.310	-3.322
f_{21}	4	-10.153	20000	-4.826	-10.153	-10.153
f_{22}	4	-10.402	20000	-5.294	-10.402	-10.403
f_{23}	4	-10.536	20000	-6.189	-10.536	-10.536

find better solutions. The acceptance criterion ensures that the best point in the population can only be replaced by an even better solution. With all these features it is possible to show that PSADE can find the true global minimum of the CF with probability 1.

6. Optimization of mathematical functions

To examine the performance of PSADE, a set of 23 well known mathematical test functions was used. The definitions of the test functions can be found in [18]. The set contains unimodal and multimodal functions with dimensionality ranging from 2 to 30.

Table 1 shows the optimization results for SA, DE, and PSADE with a limited number of CF evaluations (CFE_{max}). Every function was optimized 10 times with different random seeds. For every test function the table lists the number of variables (N_v), the known global minimum (f^*), the CFE limit (CFE_{max}), and the average minimal CF value f_{min} for SA, DE, and PSADE. The results indicate that no method outperforms all others on all functions. For the unimodal functions $f_2 - f_5$ and for the deceptive f_{15} PSADE had

Table 2: Parallel performance of PSADE.

	f^*	f_{stop}	t_1 (CFE)	S_2	S_4	S_8
f_1	0	1e-10	1180.9 (60487)	2.2	5.8	7.9
f_2	0	0.1	904.7 (46507)	1.8	5.2	8.7
f_3	0	15	1325.9 (67744)	2.3	4.6	7.8
f_4	0	0.1	1327.6 (68219)	1.9	3.8	7.4
f_5	0	30	686.1 (35251)	1.8	3.6	6.2
f_6	0	0	344.5 (17699)	2.0	4.2	8.1
f_7	0	0.02	705.9 (36311)	2.0	4.0	8.2
f_8	-12569.5	-12569.5	735.8 (37694)	2.0	4.1	8.1
f_9	0	0.1	1621.7 (83334)	2.1	4.1	7.9
f_{10}	0	1e-04	1032.9 (53038)	1.9	5.4	7.8
f_{11}	0	1e-09	1537.6 (78767)	2.1	6.1	7.6
f_{12}	0	1e-10	988.6 (50701)	2.0	4.0	7.5
f_{13}	0	1e-10	1060.6 (54261)	1.9	4.2	7.7
f_{14}	0.998	0.998	13.9 (714)	1.0	2.6	5.9
f_{15}	3.075e-04	4e-04	358.0 (18462)	1.8	3.3	8.2
f_{16}	-1.0316	-1.0316	19.1 (985)	2.3	3.8	8.8
f_{17}	0.398	0.398	16.5 (846)	3.1	4.7	10.9
f_{18}	3	3	76.3 (3946)	1.9	3.7	7.8
f_{19}	-3.863	-3.863	10.6 (547)	2.2	3.5	6.2
f_{20}	-3.322	-3.322	56.8 (2928)	1.4	3.5	5.1
f_{21}	-10.153	-10.153	160.1 (8229)	2.1	3.4	8.0
f_{22}	-10.402	-10.402	130.8 (6749)	1.5	3.8	7.1
f_{23}	-10.536	-10.536	151.2 (7753)	1.8	3.2	8.1

problems fine tuning the solution and would require more CFE. Since it is a global stochastic method that uses random search, this is not surprising. For the remaining functions, however, PSADE produced solutions that are better or at least as good as solutions found by SA and DE. The results are especially encouraging for the most difficult highly multimodal functions $f_8 - f_{13}$. PSADE produced results in close proximity of the corresponding global minima and by far outperformed SA and DE on these functions. This makes it appropriate for use in areas such as parameter fitting [19] and training of neural networks [20].

To test the parallel performance of PSADE, every function was optimized 10 times (with different random seeds) using 1, 2, 4, and 8 slaves. Runs with a different number of slaves were started with the same random seed and consequently with the same initial population. To simulate the variations of the CF evaluation time and the communication overhead, an artificial random delay between 10 and 20 milliseconds was added to every CF evaluation. The optimization results are presented in Table 2. The table lists the known global minimum (f^*) and the target CF value (f_{stop}). t_1 denotes the aver-

age time in seconds required to reach f_{stop} when a single slave is used. The corresponding average number of required CFE is given in parentheses. For runs with 2, 4, and 8 workers the speedups (S_2, S_4, S_8) with respect to the run with a single worker are given. The speedup is defined as:

$$S_i = \frac{t_i}{t_1} \quad i = 2, 4, 8 \tag{10}$$

where t_i denotes the average time required to reach f_{stop}, when i slaves are used. PSADE reached f_{stop} in all runs for all test functions.

It is evident that the asynchronous parallel approach can result in considerable speedups when the number of processors is increased. Due to the stochastic nature of PSADE it is unrealistic to expect the same speedups for all functions or even for several runs with the same function. It is possible that a run using multiple slaves finds the minimum with fewer CFE that a run using a single slave. Combined with parallel evaluation of trial solutions, this can lead to speedups greater than the number of slaves. In contrast it is also possible that the run using a single slave finds the minimum using fewer CFE, which can result in speedups that are smaller than the number of slaves. On the average, however, the results confirm that PSADE is capable of successfully exploiting multiple processors to speed up the optimization.

7. Optimization of analog integrated circuits

Analog IC design is a very difficult and time consuming task. It consists of two major steps. The first step is the selection of the circuit topology which depends mostly on the knowledge and the experience of the designer. In the second step, referred to as parametric optimization the device parameters (transistor dimensions, capacitances, resistances, etc.) must be determined so that the final circuit satisfies the design requirements. In this paper we are only concerned with the parametric optimization of the ICs and we assume that the topology is fixed.

There are several ways to evaluate the performance of analog ICs. In [21, 22], for example surveys of analog synthesis strategies are presented, and in [23, 24] the authors present an overview of more recent design methodologies for the design of large systems on chip. The methods involve the extraction of nonlinear macromodels of the circuits which are then used to evaluate the circuit performance. In this paper we use an approach more similar to [25]. We use the external SPICE OPUS circuit simulator [26] that performs full

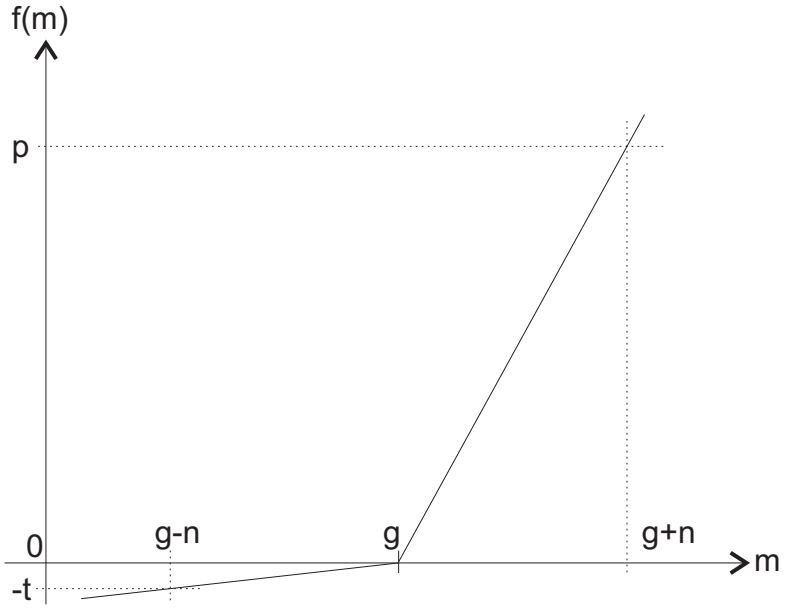

Figure 1: Penalty function example.

device model circuit simulation. This results in accurate values of circuit performance measures but is also very time consuming, so it can only be used for medium sized circuits.

7.1. The cost function

In analog IC design there are usually several conflicting design goals (e.g. gain, speed, size, power consumption, noise, etc.) and a large number of optimization variables. Multiobjective optimization methods (e.g. [27, 28]) are one way to handle all design goals simultaneously but have so far mostly been used for problems with 2-3 objectives. In IC design this is rarely enough. In the present work we use the penalty function approach [29] that combines all the specified design goals into a single real valued function. All deviations of the performance measures from their specified target values are penalized and result in an increase of the overall CF value. Figure 1 depicts an example of a penalty function for a performance measure m. The goal (target value) g, norm n, penalty factor p, and tradeoff factor t determine the shape of the function.

In order to obtain robust circuits we also consider the problem of varying environmental conditions (e.g. the temperature, supply voltage, device

model parameters, etc.) by simulating the circuits across several corners [29]. Every corner represents a combination of these parameters. The corners are usually selected so that they represent the most extreme combinations of the environmental conditions. If the circuit achieves the design specifications for the worst environmental conditions, it is likely that it will perform at least as well under normal conditions. For every corner a separate circuit simulation is required and a separate penalty function is constructed. The corner with the largest penalty value determines the contribution of a particular performance measure to the final CF value:

$$CF = \sum_{i=1}^{N_m} \max_{j=1,2,\ldots N_c} f_{i,j}(m_{i,j}) \tag{11}$$

where N_m and N_c denote the number of measurements and corners, respectively. Cases with numerous corners are usually much more difficult to optimize and also take considerably more time.

In our experiments we ignored some other important aspects of analog IC design, such as worst case and mismatch issues, yield estimation, and layout issues (e.g. [30, 31, 32]). These can be included in the existing framework as separate analyses and measurements but the corner analysis employed in this paper should be sufficient to test the performance of PSADE on real world problems.

7.2. Test cases

PSADE was tested on several real world cases of analog IC design. We present the detailed case setup and detailed optimization results only for the first two cases (AMP and AMP5C). Both cases deal with the same circuit topology depicted in Figure 2. It is an amplifier circuit with 27 optimization parameters (variables):

RR2,RR3,RR4	→ 3 resistances
CC1,CC0	→ 2 capacitances
NM0=NM1	→ width and length
NM3=NM5=NM7=NM8	→ width and length
PM0=PM1	→ width and length
PM2=PM3=PM5=PM10	→ width and length
PM9=PM11	→ width and length
NM2,NM4,NM6,PM4,PM6,PM7	→ 6 × (width and length)

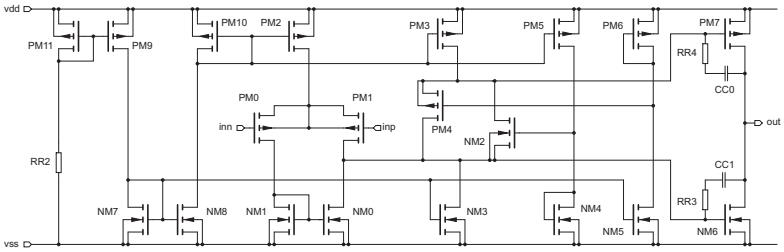

Figure 2: Topology for the AMP and the AMP5C test case.

Table 3: AMP5C - corner parameters.

corner	temperature [^oC]	supply voltage [V]	device models
typical	27	1.8	nominal PMOS,NMOS
ff	50	2.0	fast NMOS,PMOS
ss	0	1.6	slow NMOS,PMOS
ffx	0	1.6	fast NMOS,PMOS
ssx	50	2.0	slow NMOS,PMOS

where NM0=NM1 indicates that transistors NM0 and NM1 should be matched (identical in size). In the AMP case we consider a single nominal corner with typical device model parameters, $T = 27^o$C, and $V_{dd} = 1.8$V. The AMP5C case deals with 5 different corners in order to obtain a more robust circuit. The parameters for all 5 corners are given in Table 3.

The case setup was the same for both cases. Table 4 describes all 15 design requirements with their target values, norm and penalty factors that are used to construct the penalty functions. The tradeoff factors were all set to 0. The table also contains the performance measures at the final solution found by PSADE for both cases.

For the AMP case PSADE was able to find a solution that satisfies all design requirements except the output swing requirement, which was slightly below the desired value. The optimization of the AMP5C case is more difficult. The unfavorable environmental conditions affect the circuit performance and the output swing of the final circuit was 0.58V below the desired value. Additionally, the required DC gain and fall time were also not achieved. Other design requirements were satisfied.

The remaining cases will be described more briefly. The circuit topology for the BUFF and BUFF5C cases is depicted in Figure 3. Both cases consider the same circuit with 36 optimization variables, and 13 design goals.

Table 4: Test cases AMP and AMP5C: setup and results.

Property	Goal	norm	penalty	$f_{min,AMP} = 0.063$	$f_{min,AMP5C} = 1.900$
area [m^2]	< 1e-08	1e-09	1	7.418e-09	5.23e-09
isupply [A]	< 1e-03	1e-05	3	4.40e-04	5.30e-04
acgain [dB]	> 70	70	5	71.20	73.58
ugbw [Hz]	> 5e+06	5e+06	3	6.52e+06	7.97e+06
bw [Hz]	> 500	500	1	1074.30	1685.26
pm [^{o}]	> 60	60	5	72.31	62.20
gm [^{o}]	> 10	10	5	24.71	16.42
gainderiv	< 0	1	5	-1.72e-07	-5.70e-09
swing [V]	> 1.6	1.6	5	1.58	1.02
dcgain [dB]	> 60	60	5	68.44	59.91
settling [s]	< 3e-07	3e-07	1	1.14e-07	9.91e-08
overshoot [%]	< 1	1	1	0.00	0.60
slewrate [V/s]	> 5e+06	5e+06	1	1.17e+07	1.09e+07
rise time [s]	< 2e-07	2e-07	1	4.12e-08	4.39e-08
fall time [s]	< 2e-07	2e-07	1	1.37e-07	2.13e-07

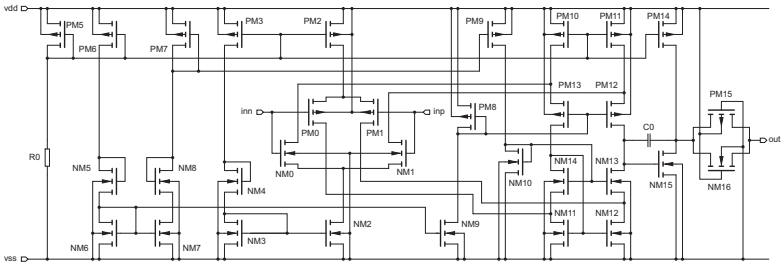

Figure 3: Topology for the BUFF and the BUFF5C test case.

The BUFF case considers only the nominal corner while the BUFF5C case considers 5 different corners. The last case (DAMP) is also an amplifier circuit with 15 optimization parameters, 13 design goals, and 14 corners. The topology of the circuit is depicted in Figure 4.

The optimization results are summarized in Table 5. PSADE proved to be a promising approach for automated IC device sizing. Even though very little information about the CF is available, PSADE was able to find solutions that satisfied most of the design goals for all cases. The parallel approach proved to be quite successful. Due to the stochastic nature of the method it is unrealistic to expect a linear relationship between the speedup and the number of slaves. It is even possible that the asynchronous approach in some cases results in a shorter path toward the minimum of the CF. In general PSADE was very successful at exploiting several processors and achieved considerable speedups when the number of slaves was increased.

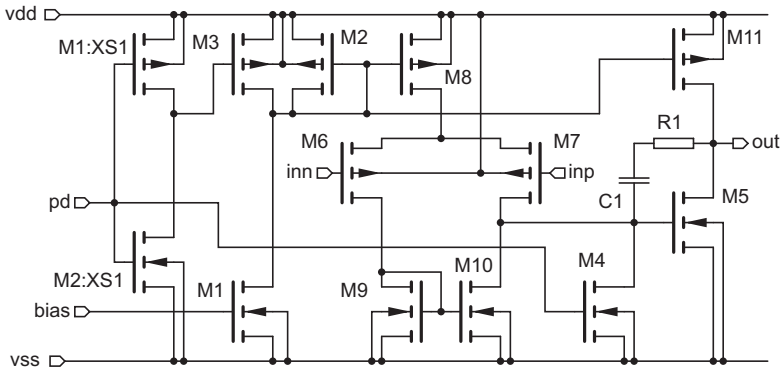

Figure 4: Topology for the DAMP test case.

Table 5: IC optimization results.

	N_v	N_c	N_g	f_{stop}	t_1	S_2	S_4	S_8
AMP	27	1	15	7.00e-02	7979.88	0.92	4.28	6.24
AMP5C	27	5	15	1.90e+00	158587.11	1.93	3.99	7.48
BUFF	36	1	13	0.00e+00	20023.60	2.12	4.43	9.38
BUFF5C	36	5	13	3.00e+00	629250.63	1.88	4.16	9.55
DAMP	15	14	13	5.30e+00	52155.32	1.99	3.94	8.04

8. Conclusion

A new hybrid asynchronous parallel global optimization method (PSADE) was presented. It combines features from simulated annealing and differential evolution to efficiently sample the parameter space. The method was designed as an asynchronous parallel algorithm that allows simultaneous evaluation of several trial solutions. This can greatly reduce the time needed for the optimization especially in applications where the CF evaluation times are long and vary with time. Optimization of 23 well known mathematical test functions confirmed that PSADE is capable of producing near optimal solutions with high probability. Tests with 1,2,4, and 8 processors also confirmed, that it is capable of achieving remarkable speedups when the number of slaves is increased. PSADE was also used to optimize 5 real world analog integrated circuits. The results confirm that it is a very promising approach for analog IC device sizing. Since IC optimization is a very time consuming task, the parallel approach is very appealing because it can reduce the design times from several days to just a few hours.

9. Acknowledgement

The research has been supported by the Ministry of Higher Education, Science and Technology of the Republic of Slovenia within programme P2-0246 - Algorithms and optimization methods in telecommunications.

References

[1] P. Preux, E. G. Talbi, Towards hybrid evolutionary algorithms, International Transactions in Operational Research 6 (1999) 557–570.

[2] U. M. Garcia-Palomares, F. J. Gonzales-Castaño, J. C. Burguillo-Rial, A combined global & local search CGLS approach to global optimization, Journal of Global Optimization 34 (2006) 409–526.

[3] H. Schmidt, G. Thierauf, A combined heuristic optimization technique, Advances in Engineering Software 36 (2005) 11–19.

[4] S. L. Ho, S. Yang, G. Ni, J. M. Mochado, A modified ant colony optimization algorithm modeled on tabu-search methods, IEEE Transactions on Magnetics 42 (4) (2006) 1195–1198.

[5] Y. T. Kao, Z. E., A hybrid genetic algorithm and particle swarm optimization for multimodal functions, Applied Soft Computing 8 (2) (2008) 849–857.

[6] C. L. Huang, J. F. Dun, A distributed PSO-SVM hybrid system with feature selection and parameter optimization, Applied Soft Computing 8 (4) (2008) 1381–1391.

[7] J. Olenšek, Á. Bűrmen, J. Puhan, T. Tuma, DESA: A new hybrid global optimization method and its application to analog integrated circuit sizing, Journal of Global Optimization 44 (1) (2009) 53–77.

[8] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 1277–1292.

[9] R. Storn, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (1997) 341–359.

[10] R. L. Yang, Convergence of the simulated annealing algorithm for continuous global optimization, Journal of Optimization Theory and Applications 104 (3) (2000) 691–716.

[11] I. De Falco, A. Della Cioppa, D. Maisto, et al., Differential Evolution as a viable tool for satellite image registration, Applied Soft Computing 8 (4) (2008) 1453–1462.

[12] L. D. Coelho, F. A. Guerra, B-spline neural network design using improved differential evolution for identification of an experimental nonlinear process, Applied Soft Computing 8 (4) (2008) 1513–1422.

[13] M. Varadarajan, K. S. Swarup, Differential evolution approach for optimal reactive power dispatch, Applied Soft Computing 8 (4) (2008) 1549–1561.

[14] H. Nobakhti, A. nad Wang, A simple self-adaptive Differential Evolution algorithm with application on the ALSTOM gasifier, Applied Soft Computing 8 (1) (2008) 350–370.

[15] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Žumer, Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems, IEEE Transactions on Evolutionary Computation 10 (6) (2006) 646–657.

[16] http://www.mb.hs-wismar.de/cea/dp/, DP toolbox homepage, 2008.

[17] http://www.csm.ornl.gov/pvm/pvm_home.html, PVM homepage, 2008.

[18] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation 3 (2) (1999) 82–102.

[19] M. Možek, D. Vrtavcnik, D. Resnik, U. Aljančič, S. Penič, S. Amon, Digital self-learning calibration system for smart sensors, Sensors and Actuators A-Physical 141 (1) (2008) 101–108.

[20] A. Krenker, M. Volk, U. Sedlar, J. Bešter, A. Kos, Bidirectional Artificial Neural Networks for Mobile-Phone Fraud Detection, ETRI Journal 31 (1) (2009) 92–94.

[21] E. Ochotta, T. Mukherjee, R. Rutenbar, L. Carley, Practical Synthesis of High-Performance Analog Circuits, Norwell, MA: Kluwer Academic .

[22] G. G. E. Gielen, R. A. Rutenbar, Computer-Aided Design of Analog and Mixed-Signal Integrated Circuits, Proceedings of the IEEE 88 (12) (2000) 1825–1854.

[23] G. G. E. Gielen, CAD tools for embedded analogue circuits in mixed signal integrated systems on chip, Proceedings, IEE, Computer and Digital Techniques 152 (3) (2005) 317–332.

[24] R. A. Rutenbar, G. G. E. Gielen, J. Roychowdhury, Hierarchical Modeling, Optimization, and Synthesis for System-Level Analog and RF Designs, Proceedings of the IEEE 95 (3) (2007) 640–669.

[25] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, J. R. Hellums, Simulation-Based Synthesis of Analog Circuits Via Stochastic Pattern Search, IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems 19 (6) (2000) 703–717.

[26] http://www.spiceopus.si/, SPICE OPUS circuit simulator homepage, 2010.

[27] H. Abbass, R. Sarker, C. Newton, PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems, in Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001) 2 (2001) 971–978.

[28] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[29] A. Bűrmen, D. Strle, F. Bratkovič, J. Puhan, I. Fajfar, T. Tuma, Automated robust design and optimization of integrated circuits by means of penalty functions, AEU-International Journal of Electronics and Communication 57 (1) (2003) 47–56.

[30] K. J. Antreich, H. E. Graeb, C. U. Wieser, Circuit Analysis and Optimization Driven by Worst-case Distances, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems 13 (1) (1994) 703–717.

[31] R. Schwencker, F. Schenkel, M. Pronath, H. Graeb, Analog Circuit Sizing using Adaptive Worst-Case Parameter Sets, Proceedings, Design, Automation and Test 4-8 (2002) 581–585.

[32] A. Žemva, B. Zajc, Functionality fault model: A basis for technology-specific test generation, Microelectronics Reliability 38 (4) (1998) 597–604.