Fuzzy Selection Based Differential Evolution Algorithm for Analog Cell Sizing Capturing Imprecise Human Intentions

Bo Liu, Francisco V. Fernández, Georges Gielen

Abstract—In this paper, a fuzzy selection-based differential evolution algorithm (FSBDE) for analog cell sizing is investigated. By combining the selection-based constraint handling method and fuzzy membership functions, a new selection methodology for handling fuzzy constraints is proposed and is integrated with the differential evolution (DE) algorithm to construct FSBDE. FSBDE specializes in solving analog sizing problems capturing imprecise human intentions, both avoiding the inflexibility of crisp constraint sizing methods and the excessive relaxation of available fuzzy sizing approaches. The high optimization ability of the DE algorithm is also inherited in this approach. Comparisons are carried out with the crisp selection-based differential evolution algorithm (SBDE) and DE in conjunction with available fuzzy optimization methods, showing that the proposed FSBDE algorithm presents important advantages in terms of fuzzy constraint handling ability and optimization quality.

I. INTRODUCTION

Driven by the advances in integrated circuit (IC) fabrication technologies, the increasing complexity of modern analog ICs and the ever-demanding time-to-market pressures, automated synthesis methodologies for analog circuits have received much attention recently. The analog design procedure consists of topology design and parameter design (also called circuit sizing) [1],[2]. This paper concentrates on the latter, aiming at automated parameter selection and optimization to improve the performances for a given circuit topology.

In the past 20 years, numerous analog circuit sizing strategies, frameworks and programs have been reported [3]-[8]. Many synthesis tools formulate the analog circuit design problem in a single-objective constrained optimization form, and then solve it by using numerical optimizers, typically in combination with a SPICE-like simulator. The design goal is usually to minimize/maximize one objective function (e.g., minimize power), subject to some constraints (e.g., slew rate larger than a certain value). In analog design, when function evaluation is performed by making use of an electrical simulator, Kirchhoff's laws are enforced. Thus, the equality constraints generated by the circuit equations are automatically handled. Besides single objective sizing, Pareto based multi-objective optimization for analog circuit has

appeared in literature in recent years [9]. This paper focuses on the single objective model.

Electrical simulation-based tools [6], [10]-[11] yield superior accuracy, generality, and ease of use, as they rely on direct electrical simulations to evaluate the circuit performances. Thus, the approach proposed in this paper is based on simulation and has been implemented via a link between HSPICE and MATLAB.

In optimization-based approaches, the quality of the solution is determined to a large extent by the optimization ability of the algorithm. Therefore, a lot of research has been done in the field of optimization algorithms for analog synthesis [3]-[11]. Evolutionary computation (EC)-based analog synthesis methods [7]-[11] are widely accepted in recent years, because EC algorithms do not require a good starting point, do not require differentiability, have global optimization abilities and can easily be parallelized on a cluster of computers. Research on optimization engines and constraint handling methods are two focal points in EC-based analog sizing methodologies. Many tools use a genetic algorithm (GA), or genetic programming (GP) as the search engine, and static penalty functions (PF) to handle the constraints. Differential evolution (DE) [12] was first successfully used for analog synthesis in [13]. [10] proposed a co-evolutionary differential evolution (CODE) algorithm by using DE as the core optimizer and an augmented Lagrangian method to handle the constraints, which shows significant advantages over common sizing algorithms in terms of optimization and constraint handling ability. [11] proposed an improved methodology, called memetic single objective evolutionary algorithm (MSOEA), by integrating different operators from DE, GA, simulated annealing (SA) algorithms and an improved selection-based constraint handling approach [14].

Though these methods are important, all of them are designed to handle crisp constraints. In many cases, this is a too rigid approach to capture actual human intentions [15]. The designer is often interested in the neighborhood of the obtained solution; he would like to know which constraint can have an acceptable relaxation in order to obtain a better objective function, rather than strictly conforming to the, often arbitrarily, pre-defined specifications. But such information cannot be provided by methodologies based on crisp constraints. A possible solution is to set all performances of interest as design objectives and apply a multi-objective optimization algorithm. This provides a Pareto-optimal front, exhibiting the trade-offs among all performances. Then, the decision on an appropriate balance of performances can be taken a posteriori. The major drawback is that a lot of computational resources are wasted

Manuscript received Nov.10, 2008.

Georges Gielen and Bo Liu are with the ESAT-MICAS, Katholieke Universiteit Leuven, Leuven, Belgium. (e-mail: {Georges.Gielen, Bo.Liu}@esat.kuleuven.be).

Francisco Fernández is with IMSE, CSIC and University of Sevilla, Sevilla, Spain. (e-mail: pacov@imse.cnm.es).

in getting solutions in regions of the performance space without any current interest for the designer. This drawback is solved by introducing fuzzy constraints in the optimization problem. Fuzzy sets theory and fuzzy logic have appeared in the analog circuit synthesis field [15]-[19], covering various topics, e.g. symbolic modeling, optimization, topology selection, etc. Fuzzy constraints-based analog sizing methodologies are described in [15], [16]. However, the experiments in this paper show that these methods may generate solutions that are far from the designer's requirements. To capture the human intentions in a more complete and accurate way, a new methodology, fuzzy selection-based differential evolution (FSBDE), is proposed.

The paper is organized as follows. Section II reviews related works and motivates the strategy of FSBDE. The differential evolution algorithm and its implementation are discussed in Section III. Section IV introduces the design of fuzzy numbers in FSBDE. Section V formulates the fuzzy selection-based constraint handling approach. The proposed algorithm FSBDE is then elaborated. Section VI provides a practical example to show the effectiveness of the proposed approach. Comparisons with other methods are also carried out. The concluding remarks are offered in Section VII.

II. PREVIOUS WORKS AND MOTIVATIONS

A couple of fuzzy circuit optimization approaches have appeared in literature [15], [16]. These methods take into account performance tolerances and allow varying degrees of acceptability of a particular solution. By using fuzzy numbers to represent constraints, imprecision and vagueness of the designer's intentions are supported in analog synthesis.

For fuzzy constraints and objectives, membership functions $\mu(x)$ are defined, reflecting the fulfillment of the fuzzy constraints $g(x)$ or objective $f(x)$. The constrained optimization problem is transformed into the maximization of the weighted-sum of the membership function values, which can be described by the following equation:

$$\mu_D(x) = \sum_{i=1}^m w_i \mu_{f_i} + \sum_{j=1}^n w_j \mu_{g_j} \quad (1)$$

where m is the number of objectives, n is the number of constraints, $\mu_{f_i}(x)$ and $\mu_{g_j}(x)$ are their membership function value and w_i are the weights to prioritize objectives and constraints. The membership function for minimization constraints in the published methods is defined as [15]:

$$\mu_{g_j}(x) = \begin{cases} 1 & g_j \leq spec_j \\ \exp\left\{-\left(\frac{g_j(x) - spec_j}{p_j/2}\right)^2\right\} & g_j > spec_j \end{cases} \quad (2)$$

where $p_j > 0$ is a tolerance interval, that can be controlled by the designer, as will be discussed in Section IV.

The advantage of these methods is that they allow tolerance intervals for the constraints, which solves the problems of using crisp constraints. However, some drawbacks limit their

applicability. First, a weighted-sum method is used. The result is very sensitive to the weights, but there is no systematic approach to choose the proper weights. In other words, it leaves uncertainty to the designer whether the chosen weights appropriately reflect their preferences [20]. Second, the designer is often only interested in the neighborhood of the desired specifications. But for large p_j , design parameter values leading to solutions far away from the expected specifications may have an acceptable membership function value. In addition, the weighted-sum method will worsen this problem if the weights are not appropriate. If p_j are small, the evolution may fail because the entire fitness of the population is 0. This aspect is further elaborated in section IV and is demonstrated in section V.

Therefore, a new method, FSBDE, is proposed, whose purpose is to: (1) find a better way to integrate imprecise designer's intentions into analog sizing, and (2) enhance the ability of optimization by employing more effective search algorithms.

III. DIFFERENTIAL EVOLUTION ALGORITHM

The DE [21] algorithm is selected as the search engine in FSBDE, so a brief introduction of DE is first provided.

DE is a population-based evolutionary computation (EC) technique, which uses a simple differential operator to create new candidate solutions and a one-to-one competition scheme to greedily select new candidates. The theoretical framework of DE is simple and it can easily be coded. Besides, it is computationally inexpensive in terms of memory requirements and CPU times. Thus, DE has attracted much attention and found applications in various fields [21]-[23]. We will use it here for analog circuit sizing.

DE starts with the random initialization of a population of individuals in the search space and works on the cooperative behaviors of the individuals in the population. Therefore, it finds the global best solution by utilizing the distance and direction information according to the differentiations among the population. However, the searching behavior of each individual in the search space is adjusted by dynamically altering the differentiation's direction and step length in which this differentiation is performed.

The i -th individual in the d -dimensional search space at generation t can be represented as

$$X(t) = [x_{i,1}, x_{i,2}, \dots, x_{i,d}], (i = 1, 2, \dots, NP),$$

where NP denotes the size of the population. At each generation t , the *mutation* and *crossover* operators are applied to the individuals, and a new population arises. Then, *selection* takes place, and the corresponding individuals from both populations compete to comprise the next generation.

For each target individual, according to the *mutation* operator, a *mutant vector*: $V_i(t+1) = [v_{i,1}(t+1), \dots, v_{i,d}(t+1)]$ is generated. It is generated by adding the weighted difference between a defined number of individuals randomly selected from the previous population to another individual, which is described by the following equation:

$$V_i(t+1) = X_{best}(t) + F(X_{r_1}(t) - X_{r_2}(t)) \quad (3a)$$

where indices r_1 and r_2 ($r_1, r_2 \in \{1, 2, \dots, N\}$) are randomly chosen and mutually different, and also different from the current index i . $F \in (0, 2]$ is a constant called scaling factor, which controls the amplification of the differential variation $X_{r_1}(t) - X_{r_2}(t)$, and the population size NP is at least 4, so that the mutation can be applied. $X_{best}(t)$, the base vector to be perturbed, is the best member of the current population so that the best information can be shared among the population.

After the *mutation* phase, the *crossover* operator is applied to increase the diversity of the population. Thus, for each target individual, a *trial vector*

$U_i(t+1) = [u_{i,1}(t+1), \dots, u_{i,d}(t+1)]$ is generated as follows:

$$u_{i,j}(t+1) = \begin{cases} v_{i,j}(t+1), & \text{if } (rand(j) \leq CR) \text{ or } j = randn(i), \\ x_{i,j}(t), & \text{otherwise} \end{cases} \quad (3b)$$

$j = 1, 2, \dots, d$

where $rand(j)$ is an independent random number uniformly distributed in the range $[0, 1]$. $randn(i)$ is a randomly chosen index from the set $\{1, 2, \dots, d\}$. $CR \in [0, 1]$ is a constant called the crossover parameter, that controls the diversity of the population.

Following the *crossover* operation, the *selection* arises to decide whether the *trial vector* $U_i(t+1)$ will be a member of the population of the next generation $t+1$ or not. For a minimization problem, $U_i(t+1)$ is compared to the initial target individual $X_i(t)$ by the following one to one based greedy selection criterion:

$$X_i(t+1) = \begin{cases} U_i(t+1), & \text{if } f(U_i(t+1)) < f(X_i(t)), \\ X_i(t), & \text{otherwise} \end{cases} \quad (3c)$$

$X_i(t+1)$ is the individual of the new population. The procedure described above is considered as DE/best/1/bin. Several variants of DE have been proposed, depending on the selection of the base vector to be perturbed, the number and selection of the differentiation vectors and the type of crossover operators [12].

IV. DESIGN OF FUZZY NUMBERS

As stated above, in many engineering optimization problems, strict or crisp constraints are used, including traditional analog sizing methods. For example, suppose the specification of the phase margin is 60 degrees and the objective is power minimization. Assume there are two candidate solutions. The first one has a 59 degrees phase margin and a power consumption of 1.5mW, while the second one has 65 degrees and 2.5mW, respectively. According to the crisp constraint handling method, the first solution will be eliminated because it does not meet the strict phase margin specification. However, most designers will choose the first

one.

A possible way to capture such human intentions is to use fuzzy sets theory, which has been successfully applied in various fields [24]. In this paper, fuzzy numbers (also called ‘fuzzy sets’) are integrated in the optimization process. A fuzzy number is a set of values, each having a membership function $\mu(x)$ that reflects the degree of acceptability of the required value. $\mu(x) = 1$ means that the value is fully acceptable, and $\mu(x) = 0$ means that the value is not acceptable at all.

Fig.1. Membership functions

Previous works have defined a kind of fuzzy numbers that are described by eqn (2), and whose shape can be controlled by the tolerance interval parameter p_i . However, this kind of fuzzy number, illustrated in Fig.1, has two drawbacks. Suppose that the specification of DC gain is 60dB. First, as shown in Fig. 1 (a), a large p_i provides a large span of the fuzzy sets, which makes $\mu(x) > 0$ for values quite far from the designer’s anticipation. This may cause the sizing to be “too soft”. In contrast, in Fig.1 (b), a small p_i reflects the designer’s intentions well, but this will highly affect the evolution, because in the first a few iterations, it is possible that no candidate reaches the $\mu(x) > 0$ region of any fuzzy number, so all the fitness values are 0 in eqn (1), and the evolution will fail. (Actually, $\mu(x)$ is not exactly 0 if using an exponential membership function, but there is a large region with $\mu(x) \approx 0$ and there is no distinct improvement, as shown by the points with DC gain less than 40dB in Fig. 1 (b). The fitness values of these points have little difference, and from these points it is difficult to reach regions with $\mu(x)$ distinctly larger than 0. Therefore, $\mu(x) = 0$ is used for this condition in the following passage.) This problem is solved by a new fuzzy selection-based constraint handling method in this work.

Two kinds of fuzzy numbers are used in FSBDE. The first one is a linear fuzzy number, as shown in Fig. 1 (c). The membership increases linearly in the $\mu(x) \geq 0$ and $\mu(x) \leq 1$ area. The designer needs to specify the upper and lower

bounds of the anticipated specifications. The advantage of a linear fuzzy number is that it is straightforward and easy to design, and can reflect the designer's intentions in many cases. The drawback is that its slope is a constant, while the designer may need a higher slope in some region and a lower slope in another region. Therefore, the fuzzy number from [15], [16] is also used, but two modifications are applied. First, p_j is restricted to a certain interval, in order to restrict the fuzzy number to be within the neighborhood of the anticipated specifications. Second, upper and lower bounds also exist. For values below the lower bound, $\mu(x)$ is set to 0. Though the membership function is not continuous at $\mu(x) = 0$, as shown in Fig. 1 (d), the membership function value near the lower bound is very close to 0, so it has little disruptive effect on the evolution procedure. The advantage of this kind of membership function is that the shape can be modified by changing p_j . The above two membership functions are typical examples; designers can decide their own fuzzy membership functions according to their own needs.

V. FUZZY SELECTION BASED DIFFERENTIAL EVOLUTION ALGORITHM

A. Fuzzy selection based constraint handling method

As said above, restricting the fuzzy number to the neighborhood of the specified constraints can reflect the designer's intentions, but may cause all the fitness values to be 0 during the evolution because of the weighted-sum method used and therefore may ruin the optimization. Consequently, an appropriate constraint handling method is needed as a key requirement. A constraint handling algorithm for genetic algorithms proposed by K. Deb [14] has proven to be effective in crisp constrained optimization methods. Given two candidates in the population, there may be, at most, three situations:

- (1) Both solutions are feasible;
- (2) Both solutions are infeasible;
- (3) One solution is feasible, but the other is not.

Accordingly, the selection rules are:

- (1) Given two feasible solutions, select the one with the better objective function value;
- (2) Given two infeasible solutions, select the solution with the smaller constraint violation;
- (3) If one solution is feasible and the other is not, select the feasible solution.

Constraint violation is calculated as $\sum_{i=1}^n |\min(g_i(x), 0)|$ for

constraints $\{g_1(x) > 0, g_2(x) > 0, \dots, g_n(x) > 0\}$. We argue that unlike the weighted-sum method, a selection-based constraint handling method is also effective when the fuzzy numbers are restricted to a relatively small region near the specifications. Two candidates with $\sum \mu(x) = 0$ can still be compared with their violation of the constraints to direct the selection procedure in the selection-based methods, while

$\sum \mu(x) = 0$ is the only information that can be obtained in the weighted-sum method. Therefore, the constraint handling method we proposed in this paper is also selection-based.

In order to derive the selection rules, the designer's intentions are first analyzed, which can be summarized as follows. Firstly, the designer may allow a certain extent of tolerance for the specifications, but only in the neighborhood of the specifications. Secondly, the designer only accepts constraint degradation when the objective function value can effectively be improved. If there is no improvement to the objective function, the relaxation of the constraints is of little sense, because the new solution is inferior to the solution that did not allow tolerance on objective and constraints. Thirdly, solutions that have many performances with $\mu(x) > 0$ but close to $\mu(x) = 0$ may appear in the sizing result. Although being in the allowed region, such solutions with obviously many degradations can hardly be acceptable for the designer. Fourthly, although the designer can allow a tolerance, he may also wish that the performances are forced to meet the specifications if the objective function value is not harmed.

To address the third rule, besides the allowed region of the fuzzy number (the lower and upper bounds), a λ -cut [24] is defined. A λ -cut set is a clear set from the fuzzy set, which can be expressed as $A_\lambda = \{x | \mu_{A(x)} \geq \lambda\}$. When the obtained performance of the candidate has a membership function value larger than λ , it is considered as comparatively satisfactory. Consequently, there are 4 levels to judge an obtained performance, as shown in Table I. With this classification, the designer may set a positive number m to require more than m performances to reach the comparatively satisfactory value.

Table I. Fuzzy identifications of the performances

Level of performances	Judgment
Fully satisfactory	$\mu(x) = 1$
Comparatively satisfactory	$\mu(x) \geq \lambda$ AND $\mu(x) < 1$
Allowed but not satisfactory	$\mu(x) > 0$ AND $\mu(x) < \lambda$
Not acceptable	$\mu(x) = 0$

According to the above analysis, the final selection rule is defined as follows. Considering an analog sizing problem with n constraints and m performances, which are required to reach the comparatively satisfactory level,

- (1) Given two solutions with $\min(\mu_i(x)) = 0$, $i = 1, \dots, n$, select the solution with the smaller constraint violation;
- (2) Given one solution with $\min(\mu_i(x)) = 0$, $i = 1, \dots, n$ and one solution with $\min(\mu_i(x)) > 0$, select the solution with $\min(\mu_i(x)) > 0$;
- (3) Given two solutions with $\min(\mu_i(x)) > 0$, $i = 1, \dots, n$, compute the number of constraints with $\mu_i(x) \geq \lambda$ (noted as $\bar{m}_1, \bar{m}_2$, $\bar{m}_A = \min(\bar{m}_1, \bar{m}_2)$, $\bar{m}_B = \max(\bar{m}_1, \bar{m}_2)$)

(3.1) if $\bar{m}_A \geq m$, select the solution with the better objective function;

(3.2) if $\bar{m}_B < m$, select the solution with the better $\sum_{i=1}^n \mu_i(x)$.

(3.3) if $\bar{m}_A < m \leq \bar{m}_B$, select the solution with more constraints that reach the comparatively satisfactory level.

It can be seen that the selection mechanism achieves the four designer's intentions described above. Firstly, the tolerance is restricted to the neighborhood of the obtained performances, because infeasible solutions are only considered to be selected when the minimum membership function value is larger than 0. Secondly, only when a better objective function value appears, the candidate with lower $\sum_{i=1}^n \mu_i(x)$ can be selected. Thirdly, a sizing result with many performances near the value of $\mu(x) = 0$ that is considered as an unsatisfactory case, will not occur. The reason is that if the number of satisfactory performances is not enough, the mechanism will select results with better $\sum_{i=1}^n \mu_i(x)$. Forthly, besides considering objective function values of infeasible solutions, $\sum_{i=1}^n \mu_i(x)$ is also optimized when the number of performances in the λ -cut set is not enough. This corresponds to the intention of, the designer wishing the sizing to meet his / her expected specifications as much as possible, without harming the objective function value to get satisfactory results.

B. FSBDE algorithm

With the above new selection mechanism, the fuzzy selection-based differential evolution (FSBDE) algorithm can be constructed by replacing the original selection mechanism in DE with the newly presented fuzzy selection mechanism. The flow diagram is shown in Fig. 2.

The algorithm consists of the following steps:

Step 1: Initialize the population by randomly selecting values of the decision variables within the allowed ranges.

Step 2: Evaluate the objective value of all individuals, and select the individual with the best objective value X_{best} .

Step 3: Perform the mutation operation for each individual according to eqn (3a) in order to obtain each individual's mutant counterpart.

Step 4: Perform the crossover operation between each individual and its corresponding mutant counterpart according to eqn (3b) in order to obtain each individual's trial individual.

Step 5: Evaluate the objective value of the trial individuals.

Fig.2. Flow diagram of FSBDE algorithm

Step 6: Perform the fuzzy selection operation between each individual and its corresponding trial counterpart according to the fuzzy selection rule described above, so as to generate the new individuals for the next generation.

Step 7: Determine the best individual of the current new population with the best objective value. If the objective value is better than the objective value of X_{best} , then update X_{best} and its objective value with the value and objective value of the current best individual.

Step 8: If the stopping criterion is met (e.g. a convergence criterion or a maximum number of generations), then output X_{best} and its objective value; otherwise go back to Step (2).

VI. EXPERIMENTAL RESULTS AND COMPARISONS

In this section we tested FSBDE based on a practical analog circuit sizing problem: the two-stage telescopic cascode amplifier in Fig. 3. The load capacitance is 1.6pF, and the technology used is a 0.25 μ m CMOS process with 2.5V power supply. The design variables are the transistor widths (after appropriate matching conditions were enforced): W1, W1C, W3, W3C, W5, W7, W9, W11, Wbn, with a range of 0.5 μ m to 1000 μ m; the compensation capacitance (cc), with a range of 0.1pF to 50pF; the bias current (ibb), with a range of 0.5 μ A to 10 mA; and the bias voltage (vvcp, vvcn), with a range of 0.05V to 1.1V. The transistor lengths are set to 0.25 μ m. The DE step size F is 1, the crossover probability is 0.8, and the population size is 60. All the examples were run on a Pentium IV PC at 3GHz with 2GB RAM and Linux operating system, in the MATLAB environment.

Comparison between the results by the crisp SBDE [25], the fuzzy weighted-sum method from [15], [16] in

conjunction with DE, and the FSBDE method without and with λ -cut are carried out. 20 random runs are performed for each experiment. The specifications and 3 typical results of the different algorithms are shown in Table II to Table VI. The so-called “dm” parameters represent the value of the drain-source voltage with respect to the drain-source saturation voltage, whose purpose is to constrain the transistors to work in the saturation region. The ‘dm’ constraints are considered as crisp constraints.

Fig. 3 Two-stage telescopic cascode amplifier

Table II. Typical results by crisp SBDE with high requirements

Specifications	constraints	Run 1	Run 2	Run 3
DC gain (dB)	≥ 70	70.28	70.03	70.17
GBW (MHz)	≥ 250	251.20	314.01	250.32
Phase margin (°)	≥ 60	60	60.06	60.12
Output swing (V)	≥ 4	4.06	4.06	4.06
Area (μm^2)	≤ 3600	3470.4	3600	3429
dm1	≥ 1.2	8.98	4.41	4.73
dm3	≥ 1.2	2.14	1.74	2.25
dm5	≥ 1.2	11.81	5.64	12.32
dm7	≥ 1.2	2.58	2.85	2.56
dm9	≥ 1.2	3.63	5.60	3.55
dm11	≥ 1.2	9.89	5.10	10.40
dm13	≥ 1.2	5.23	8.05	1.61
Power (mW)	objective	15.48	51.71	14.90
Time (s)		1887	1829	1906

Table III. Typical results by crisp SBDE with low requirements

Specifications	constraints	Run 1	Run 2	Run 3
DC gain (dB)	≥ 65	66.07	65.01	65.07
GBW (MHz)	≥ 220	226.94	222.91	220.93
Phase margin (°)	≥ 30	31.66	33.50	30.98
Output swing (V)	≥ 3	4.32	4.68	4.73
Area (μm^2)	≤ 4900	3901.3	4511.4	4137.2
dm1	≥ 1.2	4.83	5.44	6.50
dm3	≥ 1.2	1.68	1.99	1.81
dm5	≥ 1.2	13.70	12.01	13.98
dm7	≥ 1.2	2.89	6.39	7.53
dm9	≥ 1.2	5.65	17.32	21.74
dm11	≥ 1.2	10.62	14.42	16.04
dm13	≥ 1.2	6.15	8.31	4.61
Power (mW)	objective	4.97	6.30	6.31
Time (s)		1699	1652	1646

As can be seen in Table II and Table III, the selection-based differential evolution algorithm can handle the crisp constraints, but cannot integrate human intentions. If the high requirements are used, the algorithm cannot have an acceptable relaxation in order to obtain a better objective function. If the requirements are relaxed, the algorithm can

obtain a better objective function value, but the other performances may be too low. Both behaviors do not correspond to human intentions, who want something in between.

Two sets of fuzzy numbers used in the weighted-sum fuzzy optimization method are shown in Fig. 4 and Fig. 5. Fig. 4 shows the fuzzy membership functions with relatively large p_j , whereas low p_j is used in Fig. 5. The weights to different constraints: DC gain, GBW, phase margin, output swing and square root of area, are 1, 4, 2, 3, 1, respectively.

Fig. 4. Fuzzy membership functions with large p_j

The results of three typical runs of the fuzzy weighted-sum optimization method with large p_j values are shown in Table IV. The notation of n defines a fuzzy constraint. It can be seen that the results are totally unsatisfactory. Because of the large p_j , “too soft performances” appear, such as the GBW in solution 3. The membership function value is an acceptable 0.43 for 176.33 MHz in Fig. 4 (b), but it is far from the requirement of 250MHz. Clearly, the p_j for the GBW also seems not large enough for the first and second results. The membership function values of 0.006 and 5.66 are very close to 0. It can be seen that the evolution of GBW failed, because no solution with a large membership function has been found. Therefore, the algorithm tried to find larger membership function values for other performances, such as phase margin, to obtain a better weighted-sum result of the sum of the membership functions of all constraints.

A new experiment by using the same constraints with the fuzzy numbers shown in Fig. 5 yields no solution: the membership function values of most constraints are very close to 0. The reason is that at the initial steps of the optimization, the performances have difficulty to reach the lower bounds. If the lower bounds are relatively moderate (this can be achieved by using large p_j), the membership function values of those constraints are distinctly larger than 0 and can be enhanced if better solutions are found (previous experiment). If the weighted-sum method is used with tight lower bounds (small p_j), the only information that can be

obtained is that all the memberships are approximately 0, so the selection cannot be executed. However, if the fuzzy selection-based methods are used, though the membership function values are all approximate 0 in the first steps, the violation of constraints are used to select better individuals.

Table IV. Typical results by weighted-sum fuzzy optimization with large p_i

Specifications	constraints	Run 1	Run 2	Run 3
DC gain (dB)	≥ 70	54.41	38.85	56.05
GBW (MHz)	≥ 250	0.006	5.66	176.33
Phase margin ($^\circ$)	≥ 60	88.88	23.28	0.47
Output swing (V)	≥ 4	2.78	2.49	2.75
Area (μm^2)	≤ 3600	4243.9	44680	1201.2
dm1	≥ 1.2	10.44	2.02	12.42
dm3	≥ 1.2	6.13	1.31	1.83
dm5	≥ 1.2	1.83	1.68	1.63
dm7	≥ 1.2	2.46	2.55	2.47
dm9	≥ 1.2	1.64	2.29	1.56
dm11	≥ 1.2	1.60	1.76	3.84
dm13	≥ 1.2	19.96	1.59	9.55
Power (mW)	objective	2.67	72.99	3.22
Time (s)		1201	1237	1073

Table V. Typical results by FSBDE without λ -cut

Specifications	constraints	Run 1	Run 2	Run 3
DC gain (dB)	≥ 70	75.11	75.31	83.44
GBW (MHz)	≥ 250	240.68	230.13	280.23
Phase margin ($^\circ$)	≥ 60	45.50	48.89	49.68
Output swing (V)	≥ 4	4.60	3.21	4.30
Area (μm^2)	≤ 3600	2279.9	1204.7	1576.8
dm1	≥ 1.2	2.64	1.85	5.90
dm3	≥ 1.2	5.21	6.06	6.80
dm5	≥ 1.2	8.92	5.08	12.70
dm7	≥ 1.2	6.81	3.63	3.14
dm9	≥ 1.2	17.45	7.05	5.68
dm11	≥ 1.2	9.61	3.94	9.57
dm13	≥ 1.2	5.86	11.24	11.72
Power (mW)	objective	3.99	3.67	4.39
Time (s)		1628	1683	1719

Fig. 5. Fuzzy numbers with small p_j

The fuzzy numbers used in FSBDE with and without λ -cut are shown in Fig. 6. Notice that this is an example of fuzzy numbers used in sizing, and that different designers can use

different fuzzy numbers according to their own interests.

Fig. 6. Fuzzy membership functions in the FSBDE experiments

The results of FSBDE without λ -cut shown in Table V are much better than the crisp constrained optimization algorithm and the weighted-sum method. FSBDE without λ -cut obtains better objective function values and less degradations compared with Table III. The method generates solutions with good objective function and the degradation on the other performances can be accepted by the designer, such as solution 3. However, the method also generated “embarrassing” solutions, where although the objective function is good, many constraints are near the lower bounds, such as solution 2. The reason is that although the fuzzy number from the designer reflects his intentions, there is no distinction of “marginally accepted degradation” and “relatively satisfactory degradation”. A designer cannot allow too many “marginally accepted degradations”. This cannot be controlled if not using λ -cuts.

The final experiment is the full FSBDE algorithm with λ -cuts, as shown in Table VI. The λ and m values are determined by the analog designer according to his specific needs. In this example, we require at least 4 of the 5 performances to reach a comparatively satisfactory level. The satisfactory levels are 67dB, 240MHz, 50° , 3.5V, $4096 \mu\text{m}^2$ for DC gain, GBW, phase margin, output swing and area, respectively. Here, λ can be decided by combining the values of satisfactory levels and the membership functions. Apart from directly defining λ , this is an alternative way. Three typical results are shown in Table VI.

It can be seen that the results of the FSBDE algorithm conform to the four designer’s intentions summarized in section V. No performance has excessive relaxation. In all the results, at least four of the performance constraints reached the relatively satisfactory level. The objective function is competitive with the results of any other method in this example.

Table VI. Typical results by FSBDE with λ -cut

Specifications	constraints	Run 1	Run 2	Run 3
DC gain (dB)	≥ 70	78.19	74.38	75.85

GBW (MHz)	≥ 250	240.36	257.28	249.45
Phase margin (°)	≥ 60	53.60	50.94	60.79
Output swing (V)	≥ 4	4.10	4.02	3.99
Area (μm^2)	≤ 3600	1203.6	1162.2	3503.8
dm1	≥ 1.2	4.92	4.22	5.26
dm3	≥ 1.2	2.80	1.58	4.83
dm5	≥ 1.2	6.58	6.36	9.21
dm7	≥ 1.2	2.92	2.99	2.54
dm9	≥ 1.2	5.57	5.28	3.63
dm11	≥ 1.2	5.57	4.98	7.79
dm13	≥ 1.2	9.22	6.69	11.51
Power (mW)	objective	4.25	3.68	4.90
Time (s)		1650	1681	1659

VII. CONCLUSION

In this paper the FSBDE algorithm has been proposed for solving analog integrated circuit sizing problems. Thanks to the proposed fuzzy selection-based constraint handling method and the powerful differential evolution algorithm, FSBDE combines the human's flexibility and the computer's high optimization ability to obtain an intelligent automatic analog sizing methodology. Future work will be on integrating fuzzy control theory in analog sizing to handle more complex sub-block synthesis and hierarchical synthesis of analog systems.

Acknowledgements

This work is supported by a special bilateral agreement scholarship of the Katholieke Universiteit Leuven, Leuven, Belgium with Tsinghua University, Beijing, China.

REFERENCES

- [1] D. Nam, Y. Seo, L. Park, C. Park, and B. Kim, "Parameter optimization of an on-chip voltage reference circuit using evolutionary programming", *IEEE Transactions on Evolutionary Computation*, vol 5, no 4, pp 414-421, 2001
- [2] E. Martens and G. Gielen, "Classification of analog synthesis tools based on their architecture selection mechanisms," *Integration – the VLSI Journal*, Vol. 41, No. 2, pp. 238-252, 2008.
- [3] E. Ochotta, R. Rutenbar, and L.R. Carley, "Synthesis of High-Performance Analog Circuits in ASTRX/ OBLX," *IEEE Trans. Computer Aided-Design*, vol. 15, pp. 273-294, 1996.
- [4] G. Gielen, H. Walsharts, and W. Sansen, "Analog Circuit Design Optimization based on Symbolic Simulation and Simulated Annealing," *IEEE J. Solid-State Circuits*, vol. 25, no. 3, 1990.
- [5] F. Medeiro, R. Rodríguez-Macias, F.V. Fernández, R. Domínguez-Castro, J.L. Huertas and A. Rodríguez-Vázquez, "Global design of analog cells using statistical optimization techniques," *Analog Integrated Circuits and Signal Processing*, Vol. 6, No. 3, pp. 179-195, 1994.
- [6] R. Castro-López, F.V. Fernández, O. Guerra and A. Rodríguez-Vázquez. Reuse based methodologies and tools in the design of analog and mixed-signal integrated circuits. Springer, 2006.
- [7] C. Goh, and Y. Li, "GA automated design and synthesis of analog circuits with practical constraints", *IEEE Congress on Evolutionary Computation*, vol. 1, pp. 170--177, 2001
- [8] J. Koza, F. Bennett, D. Andre, M. Keane, and F. Dunlap, "Automated synthesis of analog electrical circuits by means of genetic programming", *IEEE Trans. Evolutionary Computation*, Vol 1, No.2, pp 109-128, 1997.
- [9] T. McConaghy, P. Palmers, G. Gielen, and M. Steyaert, "Simultaneous multi-topology multi-objective sizing across thousands of analog circuit topologies", in *Design Automation Conference*, pp. 944-947, 2007.
- [10] B. Liu, Y. Wang, Z. Yu, L. Liu, M. Li, Z. Wang, J. Lu and F. V. Fernández, "Analog optimization system based on hybrid evolutionary algorithms", *Integration, the VLSI Journal*. Vol. 42, No. 2, pp.137-148, 2009
- [11] B. Liu, F. V. Fernández, G. Gielen, R. Castro-López, and E. Roca, "A Memetic Approach to the Automatic Design of High Performance Analog Integrated Circuits", *ACM Transactions on Design Automation of Electronics Systems*, in press.
- [12] K. Price, R. Storn and J. Lampinen, *Differential evolution. A practical approach to global optimization*. Springer, 2005.
- [13] P. Vancorenland, G. V. Plas, M. Steyaert, G. Gielen and W. Sansen, "A layout-aware synthesis methodology for RF circuits", *Proceedings of international conference on computer-aided design*, pp.358-362, 2001
- [14] K. Deb, "An efficient constraint handling method for genetic algorithm", *Computer Methods in Applied Mechanics and Engineering*, pp. 311-338, 2000.
- [15] B. Sahu and A. K. Dutta, "Automatic synthesis of CMOS operational amplifiers: a fuzzy optimization approach", *Proceedings of 15th International Conference on VLSI Design*, pp. 366-371, 2002.
- [16] M. Fares and B. Kaminska, "FPAD: a fuzzy nonlinear programming approach to analog circuit design", *IEEE Trans. Computer Aided-Design*, pp. 785-793, 1995.
- [17] G. Oltean, C. Miron, S. Zahan and M. Gordan, "A fuzzy optimization method for CMOS operational amplifier design", 5th seminar on neural network applications in electrical engineering, pp. 152-157, 2000.
- [18] A. Torralba, J. Gháñez and L. G. Franquelo, "Circuit performances modeling by means of fuzzy logic", *IEEE Trans. Computer-Aided Design*, pp.1391-1398, 1996.
- [19] A. Torralba, J. Gháñez and L. G. Franquelo, "FASY: A fuzzy-logic based tool for analog synthesis", *IEEE Trans. Computer-Aided Design*, 705-715 ,1996.
- [20] C. Coello Coello, G. Lamont and D. Veldhuizen, "Evolutionary algorithms for solving multi-objective problems (second edition)", 2007, Springer.
- [21] J. Lampinen, A Bibliography of Differential Evolution Algorithm. <http://www.lut.fi/~jlampine/debiblio.htm>, Oct 14, 2002.
- [22] R. Storn, K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. *J Global Optim*, pp. 341-359, 1997.
- [23] U. K. Chakrabarty (Ed.) *Advances in Differential Evolution*, Springer Verlag, 2008.
- [24] T. J. Ross, "Fuzzy logic with engineering applications", McGraw-Hill press, 1995.
- [25] K. Zielinski and R. Laur, "Constrained single-objective optimization using differential evolution", In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 223-230, 2006.