A Genetic Algorithm-Based Multiobjective Optimization for Analog Circuit Design

Gabriel Oltean, Sorin Hintea, and Emilia Sipos

Technical University of Cluj-Napoca, Bases of Electronics Departament,
C-tin Daicoviciu 15, 400020 Cluj-Napoca, Romania
{Gabriel.Oltean,Sorin.Hintea,Emilia.Sipos}@bel.utcluj.ro

Abstract. Multiple, often conflicting objectives are specific to analog design. This paper presents a multiobjective optimization algorithm based on GA for design optimization of analog circuits. The fitness of each individual in the population is determined using a multiobjective ranking method. The algorithm found a set of feasible solutions on the Pareto front. Thus, the circuit designers can explore more possible solutions, choosing the final one according to further preferences/constraints. The proposed algorithm was shown to produce good solutions, in an efficient manner, for the design optimization of a CMOS amplifier, for two different sets of requirements.

Keywords: genetic algorithm, multiobjective optimization, Pareto ranking, Pareto front, analog circuit design.

1 Introduction

Designing the analog part of a mixed-signal complex electronic circuit requires a long time of the overall design time, even if the analog part represent only a small fraction of the circuit. Unlike its digital counterpart, the analog design domain is not blessed with powerful tools that simplify the design process [1]. The role of CAD techniques for circuit analysis and optimization became essential to obtain solutions that satisfy the requested performance with the minimum time effort [2]. Due to the complexity of analog circuits, global and local optimization algorithms have to be extensively employed to find a set of feasible solutions that satisfies all the objectives and constraints required by a given application.

Traditional single objective optimization does not allow multiple competing objectives to be accounted for explicitly; moreover they do not give circuit designer the freedom to choose among different, equally feasible solutions. A big step forward in this direction can be achieved using a multiobjective approach. This technique allows different objectives to be treated separately and simultaneously during the optimization process [2].

A multiobjective optimization algorithm should provide a set of nondominated individuals (Pareto front), or optimal solution set. Generating the Pareto set can be computationally expensive and it is often infeasible, because the underlying application prevents exact methods from being applicable. A number of stochastic search strategies such as evolutionary algorithms, tabu search, simulated annealing, and ant

J.D. Velásquez et al. (Eds.): KES 2009, Part II, LNAI 5712, pp. 506–514, 2009.

colony optimization have been developed [3]. As evolutionary algorithms are assumed to yield good results on complex problems without explicit knowledge of the detailed interdependencies involved, they seem to be a tempting choice [1].

Genetic algorithms (GA) performing multiobjective optimization (MOO) have previously been used in analog circuit design to generate a set of Pareto optimal solutions. In [2] the problem of analog IC design is formulated as a constrained MOO problem defined in a mixed integer/discrete/continuous domain. In [4] analog circuit satisfying a specific frequency response, using free circuit structures and including some parasitic effects, are produced in a single design stage. A coding scheme where the structure of the circuit and parameter values are encoded in a single chromosome, and a multiobjective GA is used in [5] to search for an optimal design of a CMOS operational amplifiers. A multiobjective evolutionary design methodology is used in [6] for the design of a 7-block hierarchical decomposition of a complex high-speed Delta/Sigma A/D modulator.

The purpose of this paper is to develop and implement an algorithm for the design optimization of analog circuits, based on a multiobjective GA. The underlying MOO engine makes use of a genetic algorithm where the fitness of each individual in the population is computed using a Pareto multiobjective raking. Our proposed design method provides a set of optimal solutions (Pareto front) giving the possibility for the designer to select the final one in accordance to further preferences or constrains. The objective functions are formulated using fuzzy sets, while the evaluation of the current design is performed by means of neuro-fuzzy models of circuit performances.

2 Multiobjective Optimization

Multiobjective optimization is concerned with the minimization of a vector of objectives *f(x)* that may be subject to a number of constrains or bounds:

$$\text{Find a vector } x \text{ that} \quad \text{minimizes} \quad \{f_1(x), f_2(x), \ldots, f_n(x)\} \text{ subject to:} \\ g_j(x) \le 0, \quad j = 1,\ldots,m \; ; \quad h_q(x) = 0, \quad q = 1,\ldots,p \; ; \quad x_l \le x \le x_u \; . \tag{1}$$

where $g_j(x) \le 0$ are inequality constrains, $h_q(x) = 0$ are equality constraints, and x_l and x_u are the lower and upper bounds of the variable vector *x*.

Because *f(x)* is a vector, if any of its components are competing, there is no unique solution to this problem. Instead, the concept of noninferiority (also called Pareto optimality) must be used to characterize the objectives [7].

Following the well known concept of Pareto dominance, in the case of all objective functions minimization, an objective vector $f(x^1)$ is said to dominate another objective vector $f(x^2)$, if no component of $f(x^1)$ is greater that the corresponding component of $f(x^2)$ and at least one component is smaller. Accordingly, we can say that a solution x^1 is better than another solution x^2, i.e., x^1 dominates x^2 if $f(x^1)$ dominates $f(x^2)$ [3].

A solution x^* is said to be Pareto optimal, or a nondominated solution for a multiobjective optimization problem (all objectives minimization) if and only if there is no *x* such that

$$\begin{array}{l} f_i(x) \le f_i(x^*) \text{ for } \forall i = 1,...,n \\ \exists j \text{ for that } f_j(x) < f_j(x^*) \end{array} . \quad (2)$$

A nondominated solution is one in which an improvement in one objective requires the degradation of another. Optimal solution, i.e., solution nondominated by any other solution, may be mapped to different objective vectors. In other words, several optimal objective vectors representing different trade-offs between the objectives may exist. The set of optimal solutions is usually denoted as Pareto set, and its image in the objective space is denoted as Pareto front. With many multiobjective optimization problems, knowledge about this set helps the decision maker (circuit designer) in choosing the best compromise solution. In the following, we will assume that the goal of optimization is to find or approximate the Pareto front.

3 The Proposed Algorithm

Design optimization of an electronic circuit is a technique used to find the design parameter values in such a way that the final circuit performances meet the design requirements as close as possible.

To solve this multiobjective optimization problem our approaches consider a genetic algorithm to find or to approximate the Pareto set. Formulating the design objectives for a real design is not always a simple task. The designers can usually accept a certain degree of fulfillment of the design objectives.

In this paper, fuzzy sets are used to define the objective functions [8]. We will associate with each requirement one or two fuzzy sets whose membership functions will represent the corresponding fuzzy objective functions. The fuzzy objective functions became:

$$\mu_k(f_k(x)): \ D_{f_k} \to [0,1] \ . \quad (3)$$

where D_{f_k} is the range of possible values for $f_k(x)$, x is the vector of the design parameters, and f_k is the k^{th} performance function. $\mu_k(f_k(x))$ indicates the error degree in accomplishing the k^{th} requirement. A value μ_k=0 means full achievement of fuzzy objective, while a value μ_k=1 means that the fuzzy objective is not achieved at all. The formulation of the multiobjective optimization problem now becomes:

$$\text{Find } x \text{ that minimizes } \ \{\mu_1(f_1(x)), \mu_2(f_2(x)),..., \mu_n(f_n(x))\} . \quad (4)$$

where n is the number of requirements.

The performance functions used in our algorithm consist of neuro-fuzzy models of circuit performances. These neuro-fuzzy models (first order Takagi-Sugeno neuro-fuzzy systems [9]) are built up based on input-output data sets using ANFIS [10].

The heart of the whole algorithm is the optimization engine. A genetic algorithm (GA) is responsible for the exploration of the solution space in the quest for the optimal solutions. Generally, the best individuals of any population tend to reproduce and survive, thus improving successive generations [11]. However, inferior individuals

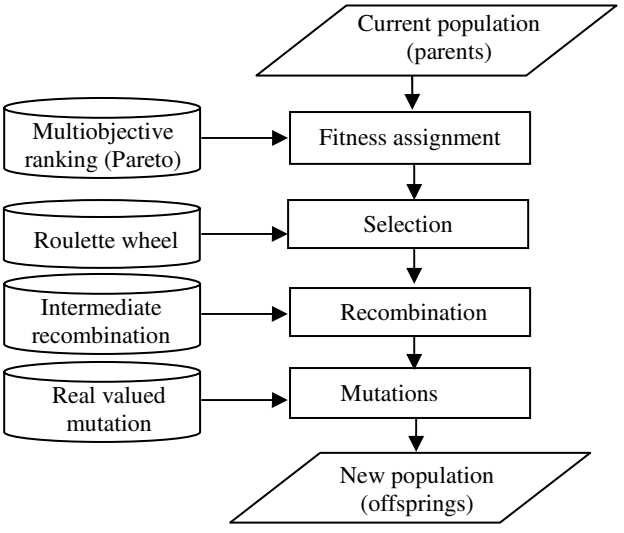

Fig. 1. Genetic algorithm multiobjective optimization

can, by chance, survive and reproduce. In our case, the individuals consist of different versions (same topology, but different parameter values) which can evolve until a set of optimal solutions is reached (in terms of requirements accomplishment). The underlying procedure of our GA for multiobjective optimization is presented in Fig.1.

The evolution of the population starts with a random at uniform initialization. Each individual in the current population receives a reproduction probability depending on its own objective function values and the objective function values for all other individuals in the current population.

As long as we are concerned with multiobjective optimization, for the fitness assignment the multiobjective ranking is used, in order to evaluate the quality of an individual. Each individual within a population receives a rank according to its quality. All solutions that are found during optimization and are not dominated by a different solution constitute the Pareto optimal solutions set. All these nondominated solutions will receive a maximum value for their rank.

$$Rank_{max}=N_{Ind}-1 \ . \tag{5}$$

where N_{Ind} represents the number of individuals.

For all the other solution the rank is computed using the relation:

$$Rank=Rank_{max}-N_{Dominating} \ . \tag{6}$$

where $N_{Dominating}$ is the number of individuals dominating the individual under consideration. All nondominated solutions have a high selection probability, while the dominated solutions have a lower selection probability, decreasing with the number of dominating individuals.

For the selection, our approach uses the roulette-wheel method. Even if this method is the simplest selection scheme it provides good results, without significant loss of population diversity, when it is used in conjunction with a rank-based fitness assignment (as is the case in this paper), instead of proportional fitness assignment. For each individual *j* a selection probability is computed as:

$$Selection\_probability_j = \frac{Rank_j}{\sum_{i=1}^{N} Rank(i)} \ . \tag{7}$$

where *N* is the number of individuals.

The individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its selection probability. A uniformly distributed random number is generated and the individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained (called mating population).

Recombination produces new individuals by combining the information contained in two or more parents. For our real valued variables the intermediate recombination method was chosen. Offsprings are produced according to the rule [12]:

$$Var_j^O = a_j Var_j^{P1} + (1 - a_j) Var_j^{P2}, \quad j = 1, 2, ..., Nvar \ . \tag{8}$$

where Var_j^O represent the j^{th} variable of the offspring, Var_j^{P1} represent the j^{th} variable of the first parent, and Var_j^{P2} represent the j^{th} variable of the second parent. The scaling factor a_j is chosen uniformly at random over an interval $[-d, 1+d]$, for each variable. A value of $d = 0.25$ ensures that the variable area of offspring is the same as variable area spanned by the variables of the parents [12].

By mutation, individuals are randomly altered. In [13] it is shown that a mutation rate of 1/*m* (*m*: number of variables of an individual) produced good results for a wide variety of test functions. That means that per mutation only one variable per individual is changed/mutated. Such an operator [12] was considered here:

$$\begin{aligned} &Var_j^{Mut} = Var_j + s_j r_j a_j, \quad j = 1, 2, ..., m \\ &s_j \in \{-1, +1\} \text{ uniform at random} \\ &r_j = r \cdot domain_j, \ r \text{ - the mutation range (standard 10\%)} \\ &a_j = 2^{-uk}, u \in [0, 1] \text{ uniform at random; } \quad k \text{ - mutation precision} \end{aligned} \tag{9}$$

In the above equations, $domain_j$ represent the domain of the variable Var_j and *k* parameter defines indirectly the minimal step-size possible and the distribution of mutation steps in the mutation range. Typical values for *k* are $k \in \{4,5,...,20\}$ [12].

Our GA uses the pure reinsertion scheme: produce as many offsprings as parents and replace all parents with offsprings. Every individual lives one generation only.

4 Experimental Results

Our optimization algorithm is developed in the Matlab. It accepts three types of requirements "greater than", "equal" and "smaller than" for each design requirement.

We used our algorithm to design a CMOS simple transconductance amplifier (SOTA). Due to the lack of space, is not given here, but it can be found easily in the literature [14]. The circuit is designed for a set of three requirements: voltage gain – *Av*, gain-bandwidth product – *GBW* and common mode rejection ratio – *CMRR*, using four design parameters: three transistor sizes $(W/L)_{12}$, $(W/L)_{34}$, $(W/L)_{56}$, and a biasing current *Ib*. All design parameters have lower and upper bounds, determined so that the transistors in the circuit will remain in their active region regardless the combination of parameter values. These bounds are: *LB*=[20, 0.5, 0.75, 1] and *HB*=[70, 4, 7.5, 100]. The values of GA parameters used in our experimentations are: *d*=0.1, *r*=0.1, *k*=18, *m*=4 and a recombination rate of 1.

The design optimization of SOTA is first illustrated for a set of "equal" requirements as they are presented in Table 1. The optimization was run for a population of 400 individuals for 1000 generations (iterations).

Table 1. Performances and objective functions for "equal" type requirements

Requirements		*Av* =40	*GBW* =5000 [kHz]	*CMRR* =500000
Indiv.1	Performances	40.46	4891	487418
	Obj. function	*0.0008*	*0.0012*	*0.0013*
Indiv.2	Performances	40.04	4613	472018
	Obj. function	*7.3897e-6*	*0.0156*	*0.0063*
Indiv.3	Performances	40.81	4980	466923
	Obj. function	*0.0025*	*4.09479e-5*	*0.0088*
Indiv.4	Performances	41.37	4716	500158
	Obj. function	*0.0071*	*0.0084*	*2.96298e-5*

At the end of the optimization our algorithm found 34 individuals on the Pareto front. Due to the lack of space we present the performances and the values of objective functions for four of them in Table 1. Indiv.1 was selected as the one with minimum average objective function (0.0011). Indiv.2 is the better one from the point of view of *Av* requirement, meaning that it has a minimum value of the objective function for *Av* in the entire Pareto set (7.3897e-6). Indiv.3 is the one having the minimum objective function for *GBW* requirement in the final Pareto set (4.09479e-5). From the point of view of *CMRR* the best individual is Indiv.4 whose objective function is 2.96298e-5. Each individual constitutes a feasible design solution, the final decision being made by the circuit designer.

The individuals (values of the design parameters) are presented in Table 2. The dynamical behavior of our optimization algorithm is presented in Fig.2. The quality of the entire population is improved generation by generation especially at the beginning of the optimization. The average of the objective functions in the entire population decreases continuously from an initial value of 0.4109 down to 0.0046.

Table 2. Individuals for "equal" type requirements

	Design parameters			
	$(W/L)_{12}$	$(W/L)_{34}$	$(W/L)_{56}$	Ib[μA]
Indiv.1	62.3	2.6	7.3	100
Indiv.2	59	2.5	7.4	94.1
Indiv.3	63.1	2.7	7.3	94.1
Indiv.4	57.4	2.7	7.1	94.4

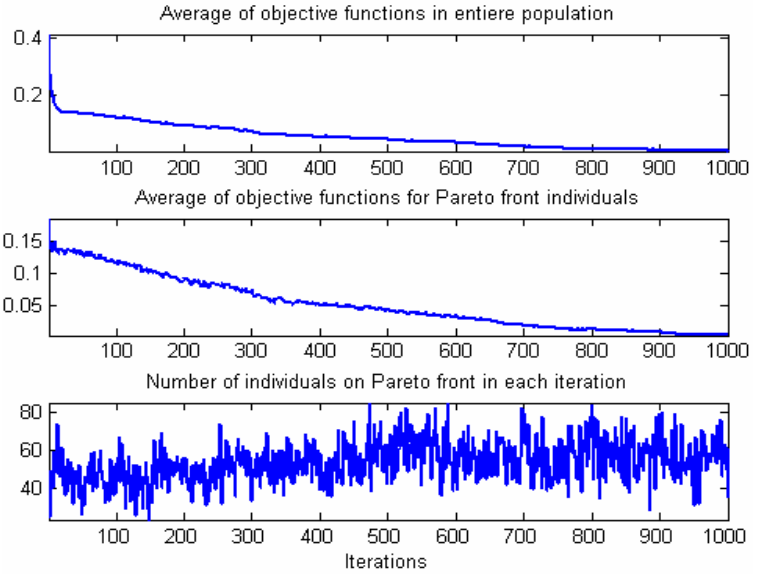

Fig. 2. Dynamic behavior of the optimization algorithm

The quality of the Pareto front is improved during optimization in a slightly oscillating manner, meaning that not always a new Pareto front is better than the previous one. The average of the objective functions in the Pareto front decreases from an initial value of 0.18254 down to a final value of 0.0038. The dimension of the Pareto front varies during the evolution (Fig.2), with a minimum of 23 individuals in generation 149 and a maximum of 85 individuals in generations 474 and 588.

The design optimization of SOTA is then illustrated for a set of "greater than" requirements: Av>50, GBW>3000[KHz], $CMRR$>1200000.The optimization was run for a population of 100 individuals for only 50 generations. In the population evolution there was only 1 individual on the Pareto front up to the iteration 41. From that point forward, the number of individuals on the Pareto front was increased continuously up to the final value of 97 individuals (out of 100). The performances of five individuals and the corresponding individuals from the final Pareto front are presented in Table 3; all design requirements being accomplished. The individuals are quite similar to each other, a possible interpretation being that during the evolution some diversity of

Table 3. Optimization results for "greater than" type requirements

Performances			Individuals			
Av	*GBW*	*CMRR*	$(W/L)_{12}$	$(W/L)_{34}$	$(W/L)_{56}$	*Ib*[μA]
59.83	3369.16	1252521.98	20.20	3.60	5.70	84.20
59.68	3389.05	1234074.67	20.30	3.60	5.80	82.50
60.90	3468.18	1242884.89	20.20	3.80	5.70	82.10
59.84	3369.10	1220738.45	20.20	3.60	5.60	82.30
58.98	3363.40	1211318.90	20.40	3.50	5.90	81.20

population was lost. The genetic algorithm can be improved if some condition to preserve the population diversity is introduced.

5 Conclusions

A method to design analog circuits using a GA-based multiobjective optimization was presented in this paper. The method uses a multiobjective ranking procedure to compute the fitness of individuals. The algorithm was used to design a CMOS amplifier for different sets of requirements. The algorithm always produces a set of Pareto optimal solutions, regardless the type of requirements ("equal" or "greater than"). The algorithm is an efficient one, the individuals in the Pareto front being permanently improved by evolution.

Further research work should be performed to improve the algorithm by introducing an elitist solution and to maintain population diversity.

References

[1] Greenwood, G.W., Tyrrell, A.M.: Introduction to Evolvable Hardware. IEEE Press Series on Computational Intelligence. Wiley&Sons Inc., Los Alamitos (2007)

[2] Nicosia, G., Rinaudo, S., Sciacca, E.: An Evolutionary Algorithm-based Approach to Robust Analog Circuit Design using Constrained Multi-objective Optimization. Knowledge-based Systems 21(3), 175–183 (2008)

[3] Zitzler, E., Laumanns, M., Bleuler, S.: A Tutorial on Evolutionary Multiobjective Optimization. In: Proceedings of the Workshop on Multiple Objective Metaheuristics, pp. 3–38. Springer, Heidelberg (2004)

[4] Yaser, M.A.K., Badar, K., Faisal, T.: Multiobjective Optimization Tool for a Free Structure Analog Circuits Design Using Genetic Algorithms and Incorporating Parasitics. In: Proc. of the Conference Companion on Genetic and Evolutionary computation, pp. 2527–2534 (2007) ISBN 978-1-59593-698-1

[5] Goh, C., Li, Y.: Multi-objective Synthesis of CMOS Operational Amplifiers using a Hybrid Genetic Algorithm. In: Proc.of the 4th Asia-Pacific Conf. on Simulated Evolution and Learning, pp. 214–218 (2002)

[6] Eeckelaert, T., Schoofs, R., Gielen, G., Steyaert, M., Sansen, W.: Hierarchical Bottom-up Analog Optimization Methodology Validated by a Delta-Sigma A/D Converter Design for the 802.11a/b/g standard. In: Design Automation Conf., 43rd ACM/IEEE, pp. 25–30 (2006)

[7] Boyd, S., Vandeberghe, L.: Introduction to Convex Optimization with Engineering Application. Stanford University (1999)
[8] Oltean, G.: FADO - A CAD Tool for Analog Modules. In: Proc. of the International Conference on Computer as a Tool, EUROCON 2005, Belgrade, pp. 515–518 (2005) ISBN 1-4244-0050-3, IEEE catalog number: 05EX1255C
[9] Oltean, G.: Fuzzy Logic Toolbox 2, User's Guide, on line version, The Math Works, Inc. (2007)
[10] Jang, R.J.-S.: ANFIS, Adaptive-Network-Based Fuzzy Inference System. IEEE Transaction on System, Man, and Cybernetics 23(3), 665–685 (1993)
[11] Cecilia, R., Machado, J.A.T., Cunha, J.B., Pires, E.J.S.: Evolutionary Computation in the Design of Logic Circuits. In: IEEE International Conference on Systems, Man and Cybernetics, pp. 1664–1669 (2007)
[12] Pohlheim, H.: GEATbx Introduction to Evolutionary Algorithms: Overview, Methods and Operators, version 3.7 (November 2005), http://www.geatbx.com/
[13] Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive models for the breeder genetic algorithm in continuous parameter optimization. In: Evolutionary Computation, vol. 1(1), pp. 25–49 (1993) ISSN 1063-6560
[14] Oltean, G.: Fuzzy Techniques in Analog Circuit Design. WSEAS Transactions on Circuits and Systems 7(5), 402–415 (2008)