IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 23 July 2024, accepted 7 August 2024, date of publication 19 August 2024, date of current version 29 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3446308

RESEARCH ARTICLE

Analog Flat-Level Circuit Synthesis With Genetic Algorithms

MIGUEL CAMPILHO-GOMES[1,2,3,4,5,6], (Member, IEEE), RUI TAVARES[3,4,5,6], (Member, IEEE), AND JOÃO GOES[3,4,5,6], (Senior Member, IEEE)

[1]Centro de Estudos e Desenvolvimento de Electrónica e Telecomunicações (CEDET), Instituto Superior de Engenharia de Lisboa (ISEL), 1959-007 Lisbon, Portugal
[2]Instituto Superior de Engenharia de Lisboa (ISEL), 1959-007 Lisbon, Portugal
[3]NOVA School of Science and Technology, 2829-516 Caparica, Portugal
[4]Center of Technology and Systems (UNINOVA-CTS), 2829-516 Caparica, Portugal
[5]Associated Laboratory of Intelligent Systems (LASI), 2829-516 Caparica, Portugal
[6]Department of Electrical and Computer Engineering, NOVA University Lisbon, 2829-516 Lisbon, Portugal

Corresponding author: Miguel Campilho-Gomes (mpcg@cedet.isel.ipl.pt)

This work was supported in part by INCD funded by FCT and FEDER through the project 01/SAICT/2016 under Grant 022153, in part by FCT/MCTES through the Project under Grant UIDB/00066/2020 (PEST) and Grant PTDC/CTM-PAM/4241/2020 (IDS-PAPER), in part by Portuguese Foundation for Science and Technology (FCT) within the Scope of CTS Multiannual Strategic Funding under Grant UIDB/00066/2020 and Grant UIDP/00066/2020, and in part by the IDS-Paper Research and Development Project under Grant PTDC/CTM-PAM/4241/2020.

ABSTRACT This paper proposes new techniques for automatic simulation-based analog circuit synthesis using genetic algorithms. This is intended to contribute to the set of electronic design automation tools that use genetic algorithms in circuit synthesis, especially those that use the simulator-in-the-loop paradigm. In this study, a genetic algorithm was employed as the generation engine for analog circuits, and variable-length chromosomes were used to describe circuit topology. The entire process is carried out on a flat level (device level), i.e. using the transistor and other elementary devices (e.g. resistors) as the basic elementary blocks. Circuit synthesis is accomplished without any knowledge of previously defined topologies (or analog block cells). Three techniques are presented for analog circuit synthesis that are incorporated in the genetic algorithm, which contribute to its robustness, leading to better and faster results. These techniques can be summarized as follows: 1) adaptive probability of chromosome acceptance, 2) removal of redundant or useless components, and 3) segmented evolution. The automatic process starts with the circuit input and output specifications and proceeds with the evolution of both circuit topology and component sizing. The results shown in this paper include a 40 dB DC gain amplifier, which, when evaluated with SPECTRE/CADENCE 6.0, using a standard 130 nm technology, with a load capacitor of 10 pF, has a gain of 102 V/V, a GBW product of 70 MHz, and a figure of merit of 1436 MHz.pF/mW.

INDEX TERMS Automatic topology generation, genetic algorithm, electronic design automation, variable length chromosome, simulator-in-the-loop, analog circuit synthesis, ngspice.

I. INTRODUCTION

Digital and analog circuit design for integrated circuits (ICs) classically has three major stages: topology selection, component sizing, and layout generation [1], [2], [3], [4]. For some years now, the synthesis of digital circuits has been almost fully automated [5], but analog ICs, or the analog part of modern Mixed-Signal System-on-a-Chip (MS-SoC), still require a lot of human intervention in the topology choice stage, despite the research efforts of the last decades in the field of automated synthesis. The typical analog circuit design flow [1], [2] includes these three phases in a highly iterative fashion, as shown in Fig. 1.

Typically, in analog circuit design, the topology selection stage is performed by a human designer who, based on experience and knowledge, selects a topology that is believed

The associate editor coordinating the review of this manuscript and approving it for publication was Ludovico Minati.

to provide the desired performance for the given task. The parameters of the circuit components (such as resistance, capacitance, transistor length and width) are then sized so that the circuit achieves the desired performance. This is often an iterative process that involves changing one or more parameters, running a simulation, and evaluating the circuit's new behavior. If the circuit does not achieve the desired performance, it may be necessary to select another topology and the cycle can then include the first stage, i.e. another topology is selected, and the sizing stage is iterated over again. If this cycle converges to generate a circuit that meets the specifications, we then move on to the layout design stage. When things get complicated, it may be necessary to include the sizing stage in this cycle, and in extreme cases even the topology selection stage may be called into play (although it is more common to make only minor adjustments to the topology than to select an entirely new one).

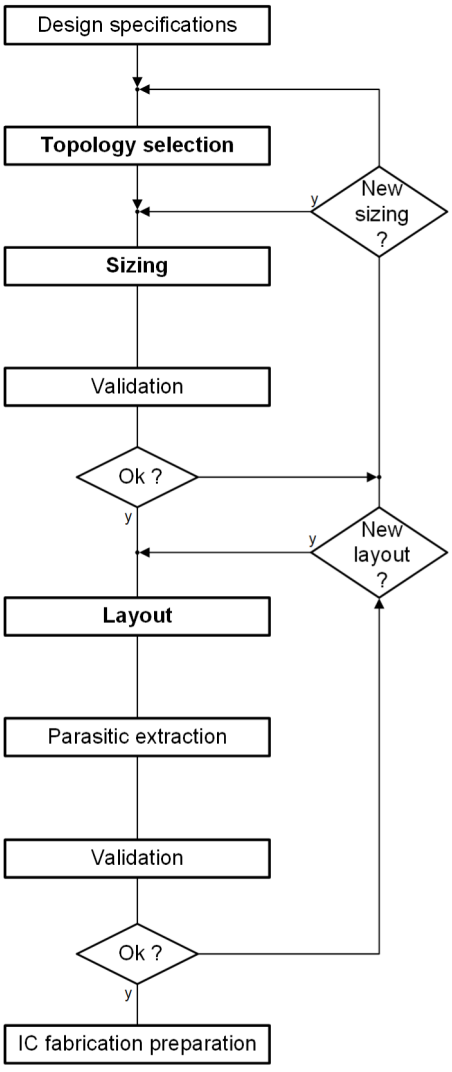

FIGURE 1. Typical analog circuit design flow.

The field of Electronic Design Automation (EDA), a trend in research efforts over the past three decades and targeted in technology roadmaps such as the International Technology Roadmap for Semiconductors (ITRS) [6], focuses on reducing designer effort, minimizing time-to-market, and minimizing production costs. Analog Design Automation (ADA), a subset of EDA, deals specifically with analog circuits and includes the three stages mentioned above: topology synthesis, component sizing, and layout generation, although the layout stage is replaced by a printed circuit board (PCB) generation stage if the circuit is not to be integrated.

The topology synthesis stage is the least developed in ADA. As the authors of [1] point out, "In most of the recent literature, topology selection is viewed as an inferior choice that has stalled since the beginning years due to its severe weakness (e.g., high computational effort and limited beneficiary applications)". Automatic topology synthesis of analog circuits is far from a simple task, and despite various research efforts over the last decades, especially in the field of Evolutionary Algorithms (EAs), it is not yet mature enough to be available in software packages for the integrated circuit industry. Analog circuit design is still essentially based on expert designers and has often been considered an art [7]. This design work is typically accomplished by porting known topologies to new fabrication technologies or circuit functions, or by optimizing existing circuit topologies, and less often by creating new circuit topologies. In fact, due to the high design complexity involved, the huge design space to be explored, the considerable design expertise required, and the trade-offs between conflicting constraints, there are still no widely accepted solutions for automated analog circuit synthesis, even at the research level [8]. According to [8], in terms of implementation strategy, the existing works on circuit topology generation can be classified into three categories: 1) knowledge-based; 2) EA-based; and 3) construction-based; and none of these three categories is absolutely perfect for circuit topology generation [8].

It is difficult to say whether one of these categories is better than the others, and research is still being done in all three categories. The work presented in this paper focuses on a variant of the so-called "string-based topology generation" [1], which is a problem codification scheme for topology generation that falls into the EA-based category. In string-based topology generation, a circuit is mapped onto a chromosome, which is an array of genes corresponding to circuit components, and a Genetic Algorithm (GA) is tasked with generating the topology. This method simultaneously sizes components and generates the topology, a process frequently employed in EA-based methods. The results of this process are then used to decide on future changes to the topology.

The aim of this work is to contribute to the pursuit of this line of research, with the intention of finding out whether it would not be possible to go further with genetic algorithms in the automatic synthesis of circuits, far beyond passive circuits, in particular for the generation of amplifier circuits, without using any previously known building blocks as a way of promoting novelty in topology generation. Thus, this article is focused in: 1) the recovery of genetic algorithms

(GAs) for the synthesis of analog circuits, promoting the use of this class of evolutionary algorithms in the field of analog design automation (ADA), without the use of predefined building blocks as a way to enhance novelty in the generation of topologies; 2) proposing new techniques complementary to the classical GA, which enhances its ability to search for and evaluate new topologies.

The rest of this paper is organized as follows: the next subsection (subsection I-A) presents a summary of the research work on the automatic synthesis of amplifier circuits. Section II presents the techniques proposed for the automatic synthesis of circuits using genetic algorithms. In this work, these techniques have made it possible to generate analog circuits, in particular for the generation of amplifiers, without the use of building blocks. Section III shows the results of amplifier circuits generated using the proposed techniques.

A. AUTOMATIC SYNTHESIS OF AMPLIFIER CIRCUITS

Several published works have pursued the generation of amplifier circuits employing EA-based methods. In [3], very interesting results were obtained using multi-objective selection (NSGA-II [9]) and other techniques (Genetic Programming (GP) [10], Age-Layered Population Structure (ALPS) [11]), while using a library of building blocks (among other types of components that can be used to generate a circuit). In [12], several amplifier topologies are generated with very interesting specifications, using GP and some additional techniques, like a current-flow analysis that is a procedure used to identify and correct any faulty design prior to simulation. Although single components (like transistors, resistors and capacitors) are used to build the circuit, they also use a user-defined library of building blocks.

In [13], circuit synthesis is performed at flat-level (i. e. device- level, or as the author says, ''from scratch''), but using only transistors, available on an FPTA (Field Programmable Transistor Array). This FPTA is used in-the-loop (i. e. inside the evolutionary loop) to evaluate the generated circuits. More than one EA is used (a GA, a variant of a GA, and a Multi-Objective Optimization Algorithm (MOEA) [14]), and the results are compared for the generation of logic gates, comparators, square-wave oscillators, and very simple operational amplifiers (OPAMPs). The latter resemble elementary versions of a differential stage of an OPAMP, although, as the author points out, ''(...) it can be stated that, despite the EA did not discover any new groundbreaking design, it is still an impressive result that the algorithm achieved to synthesize a differential input stage (...) without prior analog design knowledge''.

A classic work in the field of ADA dedicated to the synthesis of CMOS amplifiers with GAs is [15], where, as is typical for GAs, topology selection and circuit sizing are performed simultaneously. Although some very promising results have been obtained, including not-so-simple amplifier topologies, the generation is entirely based on building blocks, yielding results very much based on well-known classical topologies. In [16], several techniques employing Variable Length Chromosomes (VLCs) (also called Variable Length Representation) in GAs are used to generate circuits at flat-level without any use of building blocks. Several case studies are presented, namely for the synthesis of combinational digital circuits (such as multiplexers, comparators and parity functions) and passive filters (using resistors, capacitors, and inductors). However, only one result about amplifiers is shown, an amplifier made of 7 transistors (Bipolar Junction Transistors (BJTs) of NPN and PNP type with unknown characteristics) and although the authors claim a gain of 55dB, nothing is mentioned about linearity, output offset, dynamic range or bandwidth.

Some authors use EAs that are not GAs to generate amplifiers, as in [17], which uses a variant of the Differential Evolution (DE) technique proposed in [18]. The original DE technique uses Fixed Length Chromosomes (FLCs), but the authors of [17] adapted it to use VLCs and presented one amplifier with BJTs and resistors without using building blocks. This amplifier is claimed to have a DC gain of 63.16 dB, an output DC bias of 5.04 V, a power dissipation of 9.34 W, and a 3 dB bandwidth of 458.8 kHz. However, these characteristics are obtained when the amplifier is used in a negative feedback shunt-shunt topology [19], which linearizes the overall system response and increases its bandwidth. This is possible because the amplifier uses an inverter topology, which results in negative gain, and allows the creation of a negative feedback topology by using a two-resistor feedback network. Thus, it is not clear what are the open-loop characteristics of the amplifier, namely nothing is said about its linearity and bandwidth.

Another application of the DE technique and some of its variants [20], [21] is presented in [22], where a transconductance operational amplifier is fully sized and the performance of three EAs is compared. In this study, the selection of the topology is left aside and is done beforehand by a human designer, but the results are interesting because they show that even in isolation, the problem of sizing is in itself a difficult one, since not all the algorithms tested succeeded in all the specifications desired for the amplifier.

In [23], a combination of the divide-and-conquer approach [24] and GP, along with a proposed new circuit representation method based on a two-layer evolutionary scheme, is used to generate circuits without building blocks. The authors refer that the embryo circuit is given as a primitive individual of the circuit, but do not present an example of such a circuit and are not clear about its randomness (or about the relationship, if any, between the primitive individual and the generated circuit). The results presented show an amplifier based on an ideal OPAMP, in a topology that uses both negative and positive feedback achieved by resistor networks that have been synthesized by the algorithm. However, the active components of the amplifier are ''buried'' inside the ideal OPAMP, which is in fact a kind of building block. Another circuit shown is called a low-pass filter by the

authors, but its frequency response is that of a band-pass filter. The circuit consists of two amplifier stages, each with an NPN transistor, the first in a degenerate common emitter topology and the second in a common collector topology. However, it is not clear what exactly has been generated by the algorithm, although the achieved gain (20dB) and high frequency cut-off (30kHz) correspond to the desired (objective) values.

B. AUTOMATIC TOPOLOGY SYNTHESIS AND SIZING WITH GENETIC ALGORITHMS

Many circuit topologies used today have evolved in line with the reduction in CMOS device dimensions and have been optimized according to the currently available technology; however, conventional planar CMOS devices are approaching their scaling limits [25]. It is imperative to search for new topologies, and this article contributes to the automatic generation of circuit topologies, which eventually (and desirably) tend towards unsupervised automatic generation of circuits.

GAs are search and optimization Algorithms based on the mechanics of natural selection and natural genetics [26]. GAs belong to a large group of EAs, which constitute a class of stochastic search and optimization methods that simulate the process of natural evolution [27], and have been used for circuit synthesis research, at least since the works of Lohn [28] and Zebulum [29]. Currently, EAs are unavoidable in electronic design automation (EDA) tools [30] and are used in circuit synthesis, sizing, and optimization for numerous problems [31], [32], [33], [34], [35]. The simulator-in-the-loop paradigm is often used in EDA tools, and is acknowledged the way to go in recent (and also in not so recent) works [34], [35], [36], [37].

As in related and well-known works by other groups [37], we looked at global, stochastic search algorithms for one algorithm to use in our synthesis kernel because of their empirical robustness in the face of highly nonlinear, non-convex cost functions. We chose to use a GA for analog circuit synthesis in our work, in which VLCs [28], [29], [38] codified the circuits, and each gene encoded a single circuit component, such as a transistor or resistor [39]. Circuit synthesis is therefore accomplished at a flat level (device level) without any knowledge of previously defined topologies (or even analog block cells [12], [40] [41]).

In our simulator-in-the-loop analog synthesis evolutionary kernel, fitness evaluation is performed by a spice-like circuit simulator, *NGSPICE* [42], and all candidate circuits are submitted to simulation, i. e., there isn't any predicting technique [35], [43] (often referred as an 'oracle') filtering candidates with poor convergence probability (or with any other acceptance/refusal metric). The current version of this work focuses on the proof of concept that a GA has flat-level circuit synthesis ability. This synthesis task was found to be computationally heavy; therefore, other features often found in sizing EDA tools were not incorporated, such as Monte Carlo (MC) analysis and layout-aware simulation [2]. MC analysis is commonly accepted as a standard method to estimate circuit yield, but it is also known to be highly time-consuming. Although some techniques to reduce the computational burden have been described in the literature [44], their inherent addition to the overall computation times was considered not tractable (for the time being). For similar reasons, a layout-aware simulation was not incorporated in this work (even though related accelerating techniques can be used [35], [43]). Circuit synthesis by means of GAs, especially at flat-level, is a computationally expensive task, but when used in the absence of building blocks, GAs have an intrinsic tendency toward novelty in topology generation, which is a good motivation to devote research efforts to study their possible contributions to the ADA field.

II. TECHNIQUES FOR AUTOMATIC SIMULATION-BASED ANALOG CIRCUIT SYNTHESIS WITH GENETIC ALGORITHMS

Using a canonical GA, as originally described by Holland [45], the synthesis of analog circuits is extremely difficult, if possible. Therefore, every attempt to use GAs to synthesize analog circuits incorporates features (or techniques) that were not included in the original description of the algorithm. In this section, we describe some of the techniques used in our work that we consider important facilitators of the obtained results.

A. FITNESS FUNCTION

The global fitness function *fit* is a merit function (higher is better, bounded to [0..1]) composed by a product of partial merit-like functions fit_i [46], as shown in *(1)*:

$$fit = fit_1 \times fit_2 \times \ldots \times fit_N \tag{1}$$

We call this a global fitness function because several of those functions fit_i are themselves fitness functions (related to partial objectives, such as amplifier gain). Others can be viewed as penalty constraint functions according to their role in the problem being solved by the GA [47].

Most functions fit_i in *(1)* have the format of *(2)* [46], in which $achieved_i$ represents an objective function that should be minimized (if $k = 1$) or maximized (if $k = -1$) towards $desired_i$, the desired value, and $weight_i$ is a weight factor used to distinguish the relevance of each factor in *(1)* [48].

$$fit_i = 1 - exp(\text{-}weight_i^{-1} \times (desired_i/achieved_i)^k) \tag{2}$$

B. ADAPTING PROBABILITY OF ACCEPTANCE IN MUTATION OPERATOR

Crossover and mutation operators are often adapted [49], [50], [51] to work in populations with chromosomes that don't have a constant length, i. e., when VLCs are used the canonical operators proposed by Holland [45] are not used. In our study, we used a variant of the mutation operator that has features oriented toward VLCs, such as *insertion*,

replication, and *deletion* operations [49], but also has features applicable to both VLCs and FLCs.

One such feature of the mutation operator is the ability to refuse a mutation if it worsens the chromosome fitness. The probability p_r of rejection of a mutated chromosome is not constant. Instead, it is adapted during the evolution of the population. The probability p_a of acceptance ($p_a = 1 - p_r$) starts high and decreases over time, similar to a simulated annealing process [52], allowing broader regions of space to be searched in early generations, but later narrowing this search to small regions around the solution(s) already obtained (local exploitation). However, it is difficult to determine the rate at which the probability of acceptance should decrease over time (because it is difficult to determine the rate at which the temperature should decrease in a simulated annealing process to obtain an optimal solution [53]).

Instead of decreasing the probability of acceptance over time (the standard procedure in classic simulated annealing), we propose a heuristic that adapts that probability according to a function f_Q that measures the quality of the solutions obtained thus far. This measure is obtained by the average fitness of the N best chromosomes in the actual population, smoothed by a 2nd order discrete low pass filter (LPF). Function f_Q is bounded [0..1], where 0 represents the worst possible case and 1 represents the best possible case. The LPF is implemented by cascading two first-order IIR low-pass filters described by *(3)*, and function f_Q is described by *(5)*, where $fit(P_i)$ is the fitness of the population in iteration i given by *(4)*.

$$y[n] = (1-\alpha)\,x[n] + \alpha y[n-1] \tag{3}$$

$$fit(P_i) = \frac{1}{N}\sum_{j=1}^{N} fit\left(C_j\right) \tag{4}$$

$$f_Q(P_i) = LPF(fit(P_i)) \tag{5}$$

In *(4)*, N is the number of best chromosomes being considered ($N = 1$ if only the best chromosome is considered, $N = 2$ if only the best and second-best chromosomes are considered, and so on), and $fit(C_j)$ is the fitness of chromosome j.

To map a fitness value (obtained by f_Q) to a probability value (the probability p_r of rejection), we use a simple nonlinear heuristic of an exponential nature, function f_{map}, described by *(6)* and depicted in Fig. 2.

$$f_{map}(x) = \frac{(y_M - y_m)\left(e^{kx} - 1\right)}{e^k - 1} + y_m \tag{6}$$

This function f_{map} uses parameter k to adjust the steepness of its elbow shape, where $x \in [0..1]$, and $y \in [y_m .. y_M]$, which, in this case, was set to $y_m = 0$ and $y_M = 1$.

In our tests for the generation of digital or analog circuits, we used parameter k in the range of 1 to 4; however, this may require further adjustment in other contexts.

Both methods for measuring the quality of the solutions were tested, and both performed similarly well in our tests, either for the synthesis of digital or analog circuits, making this approach a robustness enhancer of the adaptive process by decreasing its sensitivity to the convergence speed of the GA (which is known to be highly irregular and dependent on numerous factors, such as the initial solution and general GA parameterization). This process can be summarized by the following pseudocode:

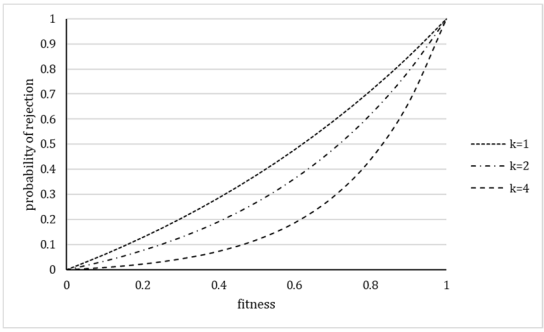

FIGURE 2. Function used to map finess to probability p_r.

```
function chrom_mutation:
  if(it is time to mutate this chrom) then
    Make a copy of this chrom
    Mutate this chrom
    Evaluate mutated chrom
    if(mutated chrom has better fitness) then
      Keep mutated chrom in the population
    else
      Calculate f_Q using population info
      Calculate f_map
      Calculate probability p_a of acceptance
      if(p_a is high enough) then
        Keep mutated chrom in the population
      else
        Keep unmutated chrom in the population
      end if
    end if
  end if
```

One relevant benefit of this technique is lowering the sensitivity of the GA to the probability parameters used by the mutation operator (we use the plural *parameters* because when using VLCs there are quite often several different probability parameters involved in the mutation operator).

C. ADAPTING PENALTY FUNCTIONS WEIGHTS

There are several methods for handling constraints in GAs [14], [26], [47]. Static penalty functions are frequently used in analog circuit synthesis. Common examples are the penalty functions for the power supplied to the circuit, total chip area, and bandwidth. When using VLCs, another possible constraint is the number of genes, that is, the number of components in the circuit (a well-known problem often referred to as "bloating" [4], [16], [54], [55]). However, when implemented using a penalty function, it is difficult to choose its weight for the entire circuit evolution. If the weight is too low, the GA produces circuits with too many redundant or useless components. However, if it is too high, the GA will hardly converge to a useful solution because it lacks exploration capability. From our results, it seems that for some penalty functions, a low weight is mandatory in the initial generations of the GA (so that there is sufficient

exploration capability), but it must be significantly higher in later stages of the GA so that the GA "cleans" the circuit by removing unnecessary components. This weight should be increased after a good solution is found, that is, after the fitness assigned to the main objective(s) is sufficiently high. Therefore, the goal is to adapt this weight according to a measure of the quality of the solutions obtained by the GA. Note that it is not useful to adapt this weight as a function of the number of generations elapsed so far because it is hardly a measure of the state of achievement of the actual circuit being synthesized by the GA (which is usually the circuit encoded by the best chromosome in the population).

To achieve this goal, we use a technique in which another function of the form of *(6)* is used (Fig. 4). However, instead of mapping fitness to a probability, fitness is mapped to the weight of the penalty constraint function associated with the number of genes (components) on the chromosome (circuit). In this case, $y_m = 0.005$ and $y_M = 0.01$, which are the minimum and maximum values intended for the weight, respectively. In our tests we used parameter k the range 4 to 16.

With this technique, the GA "starts cleaning" the circuit by removing unnecessary components without losing "sight" of the main objectives of the problem, as the global fitness function still accounts for those objectives. It appears that the GA still manages to find an optimum (or near-optimum) circuit because the weight of this penalty constraint maintains a low value during the initial exploration stage.

To illustrate the use of this technique, Fig. 3 shows some data collected in the preliminary generations of one run of the GA while attempting to synthesize an analog amplifier (circuit not shown here owing to lack of space). Fig. 3 shows the evolution of the number of genes of the best chromosome along with its fitness, as well as the mean number of genes used in the population. The number of genes in the best chromosome reached a maximum of 37 (in iteration 110 for the first time and in iteration 148 for the last time) before decreasing to its stationary value of 19 (at iteration 502).

However, the number of genes in the final circuit of this run (not shown in Fig. 3 but can be seen in Fig. 7) is 10, so that a maximum of 37 is 3.7 times the number of genes that were needed to achieve this circuit's objective. This number of components (37) significantly penalizes the population simulation times and may seem exaggerated and unnecessary; however, from our perspective, it is not. This high number of genes allows sufficient space exploration for the GA to find good solutions to the problem, as evidenced by the increase in the fitness of the best chromosome. A large and unlimited increase in the number of components must be avoided. Therefore, the attained increase in fitness increases the weight of the constraint function, which forces the number of genes to stop growing excessively, as shown in Fig. 3. This is the desired effect of this technique.

The need to control circuit bloating is certainly not new in the field of EDA using evolutionary algorithms. However, to the best of the authors' knowledge, the technique described above, which successfully deletes unnecessary devices while still allowing enough "circuit bloating" (i.e., space exploration), is a novel technique.

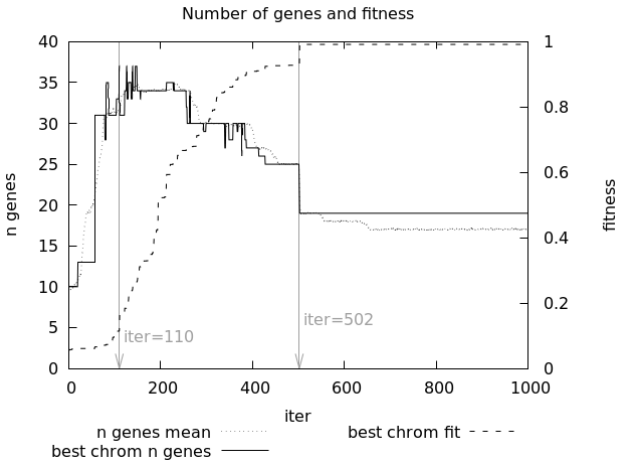

FIGURE 3. Number of genes and best chromosome fitness in initial generations.

D. MULTIPHASE PARAMETRIZATION

Extending the use of variable weights for penalty constraints, we also propose to segment the evolution of the GA in sections (i.e., in sets of generations of the GA) that allow different parameterizations in each, namely with discontinuous changes in some weights between sections.

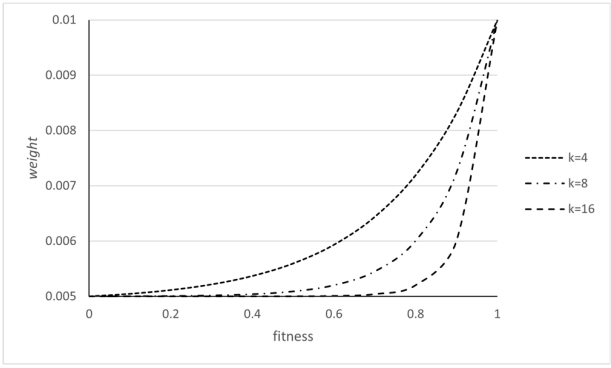

FIGURE 4. Function used to map fitness to constraint weight.

These sections, entitled "phases," in which one component of the global fitness function is given more importance than the others, allow for some steering of the GA, directing its search and optimization effort towards a specific optimization target, such as the number of components or the total area used in transistors. The criterion for a phase change is either phase stagnation or reaching a fitness threshold.

This process is illustrated by the diagram in Fig. 5, which shows an example of the three phases used in several of our tests. Each phase had a set of parameters adjusted to fit the intended steering for each phase. Typically, two contiguous phases share most parameters and weights, and only a few

are changed (eventually only one is changed) when the phase changes.

In this example, Phase 0 used a set of parameters (Param. set 0) that "tunes" the GA to find a circuit topology that fulfills some main goal (e.g., maximizing gain), without being penalized too much by secondary goals, such as minimization of the total number of components in the circuit or minimization of the total area used by transistors.

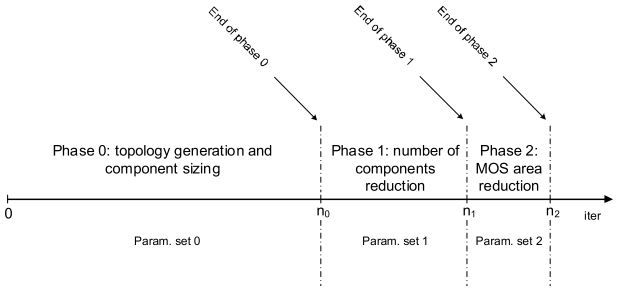

FIGURE 5. Multiphase diagram: example for 3 phases.

When a stopping criterion is reached (such as a given fitness threshold being achieved or stagnation being established), Phase 0 is terminated, and another parameter set (Param. Set 1), which has an extra penalization for a specific constraint (the total number of components used in the circuit). When this phase stops, a third parameter set (Param. Set 2) is used, which shifts the optimization effort of the GA by adding extra penalization for another constraint (the total area used by the transistors).

Considering again the aforementioned example, whose data are plotted in Fig. 3, combining the continuous adaptation of the weight of the constraint on the number of components during Phase 0 with small discontinuous increases of the same weight at the moments of change to later phases (an increase in the transition from Phase 0 to Phase 1, and a smaller increase in the transition from Phase 1 to Phase 2), the GA was able to reduce the total number of circuit components from 35 to 10, as shown in Fig. 7, which contains data from the complete run already partially plotted in Fig. 3.

This process is described in Fig. 6, supplemented by the following piece of pseudocode:

```
function Setup phase 0:
  Setup weights of all partial fitness functions
  fit_i accordingly, making sure that gain and
  THD fitness functions must have more weight
  relatively to functions that evaluate the number
  of components and the total area used by
  transistors in the circuit.
  Setup other parameters used by the GA.
function Setup next phase:
  if (phase == 1) then
    Increase weight of partial fitness function fit_i
    that evaluates the number of components used by
    the circuit
else if (phase == 2) then
   Increase weight of partial fitness function fit_i
   that evaluates the total area used by
   transistors in the circuit
end if
// insert here changes to other weights if more
// phases are used
```

In this run, there were 18381 iterations and Phase 1 occurred between iterations 2017 and 10364 (Fig. 7). Although Phase 2 is mainly committed to decreasing the total area used by transistors in the circuit, the GA still managed to lower the number of genes in this phase by removing several unnecessary components from the circuit.

This multiphase parameterization technique can be used for any number of phases and partial goals other than those mentioned in the previous example. According to knowledge gathered by our experimental results, this technique, complemented by the use of adaptive weights along the evolution (i.e., inside phases), achieved better results for all tests in which a comparison was carried out without using it.

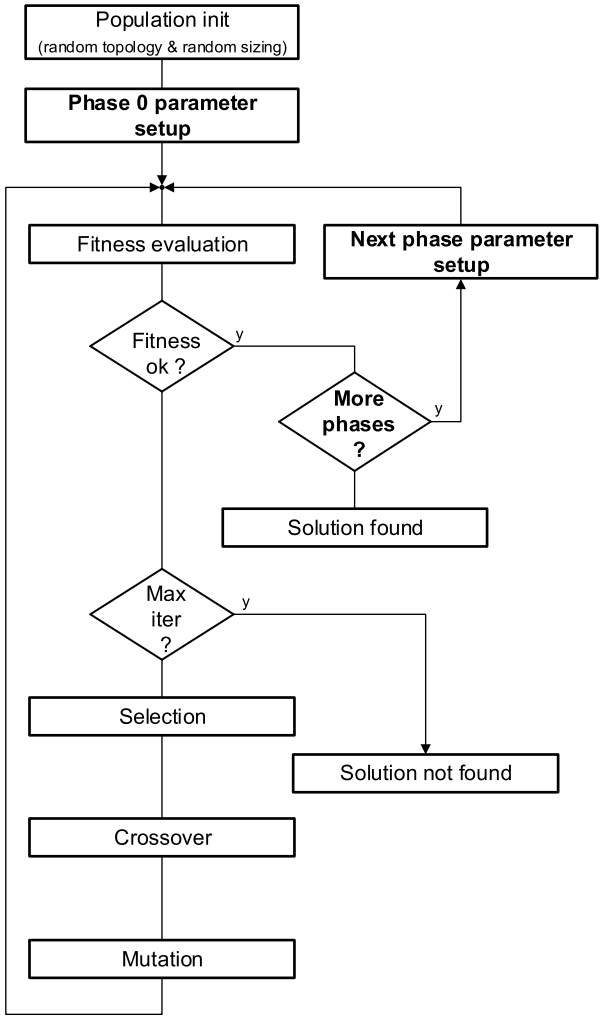

FIGURE 6. Diagram of the GA with multiphase parametrization.

III. EXPERIMENTAL RESULTS

To exemplify the use of some of the previously described techniques in the synthesis of analog circuits, experimental results are presented. To demonstrate the effectiveness of the proposed methodology, examples of single-ended-input and single-ended-output amplifiers were chosen because it was

felt that this choice would not compromise the validity of the proof of concept intended in this article and would reduce the complexity of the circuit (with a consequent reduction in execution time). Moreover, making it easier to identify and understand the automatically generated circuit topologies and compare them with those in the existing literature, to assess their novelty. Furthermore, it is a standard approach to analyze a fully differential circuit by considering only half of the circuit, i.e. the single-input, single-output version.

By focusing on single-ended designs, fundamental principles and innovations can be illustrated without the added complexity of a fully differential topology. Nevertheless, testing the synthesis of fully-differential topologies would only require a change in the embryo circuit, e.g. two-input and two-output, and then perform circuit synthesis considering the CMRR with the proposed techniques.

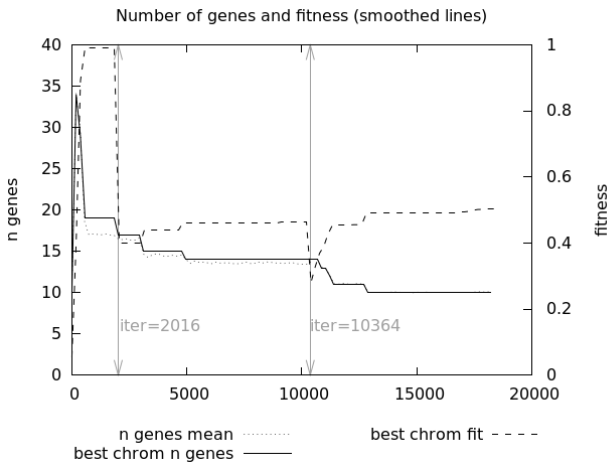

FIGURE 7. Number of genes and best chromosome fitness in one entire run.

Fig. 8 shows the embryo circuit used in our runs for the synthesis of the amplifiers. The box entitled "Evolvable circuit" surrounds the part of the circuit where the GA can interact, either by adding, removing, or sizing components such as transistors and resistors (and capacitors if their use is intended) and by adding or removing wires interconnecting components. The GA can also connect components located inside the box to the circuit outside the box, which we call the "context circuit". In this example, the "context circuit" comprises a symmetrical power supply ($V_{DD} = -V_{SS}$), two load resistors (R_{Load1} and R_{Load2}), a bias network (M_{pbias}, M_{nbias}, and I_{bias}) that is available for the GA to use (if needed), and the input voltage source (V_A). The biasing network is an auxiliary circuit that aims to facilitate the creation of current sources (by the GA), often implemented with current mirrors, which are frequently used in the design of amplifier circuits. The existence of these two load resistances aims to promote the synthesis of solutions (by the GA) that can sink or source current at the amplifier output.

Table 1 lists some GA parameters that are common to all phases of a run, or that are used in the GA initial state.

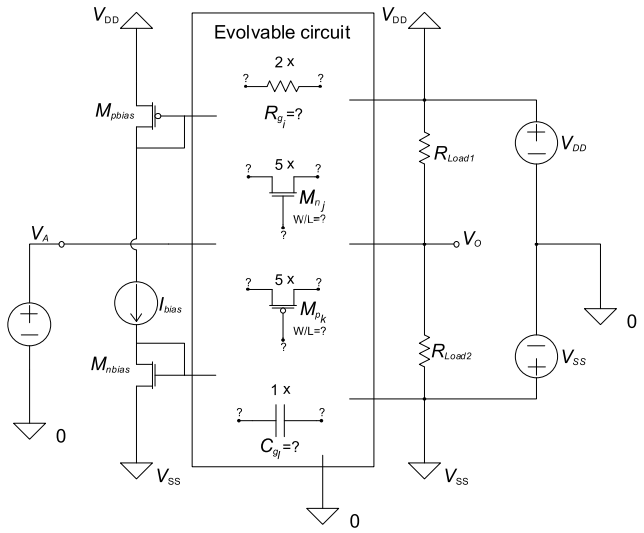

FIGURE 8. The embryo circuit.

As expected, the GA uses several other parameters that are not described here because of the lack of space (and relevance).

Some parameters not mentioned in Table 1 but often used in Phase 0 of our runs are listed in Table 2. In this phase, the mutation deletion probability is 0.02, which is a value that has been experimentally found to work well when combined with the insertion and replication probability values shown in Table 1, providing a good balance between the expansionary trend given by the insertion and replication operators (which add components to the circuit) and the regressive trend of the deletion operator (which removes components from the circuit). The weight used in the penalty function associated with the number of gene constraints was adapted between 0.004 and 0.015, using the technique described in Section II-C. Similarly, the weight used in the penalty function associated with the total transistor area constraint is adapted between 0.001 and 0.01.

TABLE 1. Some GA parameters common to all phases.

Description	Value	Comment
Chromosomes in population	240	May vary during evolution due to some operators.
Chromosomes for elitism	2	
Initial number of genes	10	Applies to each chrom.
Max n of iterations	infinite	
Stopping criterion		Stagnation
Crossover probability	0.7	
Mutation probability	0.16	Applies to values and nodes
Mut. insertion probability	0.01	
Mut. replication probability	0.01	
Mut. rejection probability	Adaptive	$f_{map} : y \in [0.0 .. 1.0]$ f_Q : fitness of only the best chrom (low-pass filtered).

In both cases, the function used to measure the quality of the population was the fitness of the best chromosome.

TABLE 2. Some GA parameters for phase 0.

Description	Value	Comment
Mut. deletion probability	0.02	
Weight for number of genes penalty function	Adaptive	f_{map} : $y \in [0.004 \,..\, 0.015]$, k = 3.0 f_Q : fitness of best chrom.
Weight for area penalty function	Adaptive	f_{map} : $y \in [0.001 \,..\, 0.01]$, k = 5.0 f_Q : fitness of best chrom.

TABLE 3. Circuit specifications desired for the 32dB amplifier.

Description	Value	Comment
Voltage gain	40	40 ≡ 32dB
THD@1kHz	*NA*	Minimize
Bandwidth	Any	f_L=0 but f_H minimum not specified: any value is accepted.
Max. input DC current	100 nA	Upper limit for input voltage source V_A.
Max. power supply current	4 mA	Upper limit for both V_{DD} and V_{SS} power sources.
Number of components	*NA*	Minimize
Area used by transistors	*NA*	Minimize

In our runs, when attempting to synthesize an amplifier, the main objective was to achieve some gain while attempting to keep the THD as low as possible. It is also necessary to satisfy other specifications, such as minimizing the number of components, minimizing the area occupied by transistors, ensuring that the amplifier input current is sufficiently low, and that the power consumed is below a certain threshold (values for these specifications are shown in the following sections, where examples of circuits generated by the GA are presented).

A. EXAMPLE 1: SYNTHESIS OF AN AMPLIFIER USING TRANSISTORS WITH L = 10μm AND A SPICE LEVEL 1 MOS MODEL

In this section, experimental data are presented and discussed in the context of the synthesis of a 32 dB amplifier. To reduce the simulation time, keeping the focus of the tests on the synthesis ability of the GA, the simulation was greatly simplified, without considering corners or layout effects, and using a simple model for transistors. For these, the simplest (and fastest) spice model (Level 1 MOS transistor model) was used, with the smallest recommended length for this model ($L = 10\mu$m) preset for all transistors, whereas their width (W) remained available for the GA to size. Other model parameters have their default values superseded: $\lambda = 0.04\text{V}^{-1}$, $V_{TO_n} = 1.6\text{V}$, $V_{TO_p} = -1.6\text{V}$, $K_{P_n} = 20\mu\text{A/V}^{-2}$, $K_{P_p} = 7.67\mu\text{A/V}^{-2}$ (providing slightly more realistic transistor characteristics). The remaining parameters retain their default values. While searching for a suitable circuit topology using resistors and transistors, the GA was allowed to size the values of the resistors and W of each transistor.

Fig. 9 shows a circuit synthesized by the GA after 84 iterations of a run, at the end of its Phase 0, whose main objective was to obtain a 32 dB DC amplifier (other desired specifications for the circuit are shown in Table 3). In Fig. 9, the input voltage source that generates V_A is omitted for clarity (owing to lack of space), and the W/L ratio of each transistor is depicted next to the transistor symbol, near the transistor naming label.

The circuit synthesized by the GA at this stage already had a gain of 39.4 ≅ 31.9 dB, which is very close to the intended objective. Other simulated results are listed in Table 4 (all measurements were obtained using *NGSPICE*).

Bandwidth, gain – bandwidth product (*GBW*), and Figure of Merit (*FoM*) were not specifications of the amplifier but are shown here to better characterize the synthesized circuit and were measured with a load capacitor of 10pF. The *FoM* [Mhz.pF/mW] was calculated using (7), where *GBW* is the gain-bandwidth product (in MHz), C_{Load} is the load capacitance (in pico Farad), and *Pwr* is the power supplied to the amplifier (in mW).

$$FoM = \frac{GBW.C_{Load}}{Pwr} \tag{7}$$

The circuit shown in Fig. 9 has several redundant or useless components but already has much fewer components than many intermediate circuits synthesized during GA evolution in this phase. This effect, of exploring the search space with more components than are really needed, but avoiding an uncontrolled growth of the number of components and even reducing it when a solution starts to have a good evaluation, is achieved by the technique already described in section II-C.

When the GA progressed from Phase 0 to Phase 1, some parameters were adjusted, as listed in Table 5. The mutation deletion probability was significantly increased, whereas the insertion and replication probabilities remained unaltered, which led to a regression trend in the total number of components used in the circuit. In addition, the weight of the penalty function that deals with the number of components increases between the last iteration of Phase 0 and the first iteration of Phase 1. The intention of these adjustments is to steer the GA to eliminate useless components without significantly deteriorating what has already been achieved, namely, the amplifier gain. As expected, during Phase 1, the number of components is further reduced, as shown in Fig. 10. However, in this phase, which lasted until iteration 2081, there was no great concern about reducing the total area used by the circuit transistors, although this already occurred because of the reduction in the number of transistors.

Reducing the number of components in one phase and reducing the total transistor area in another phase has proven

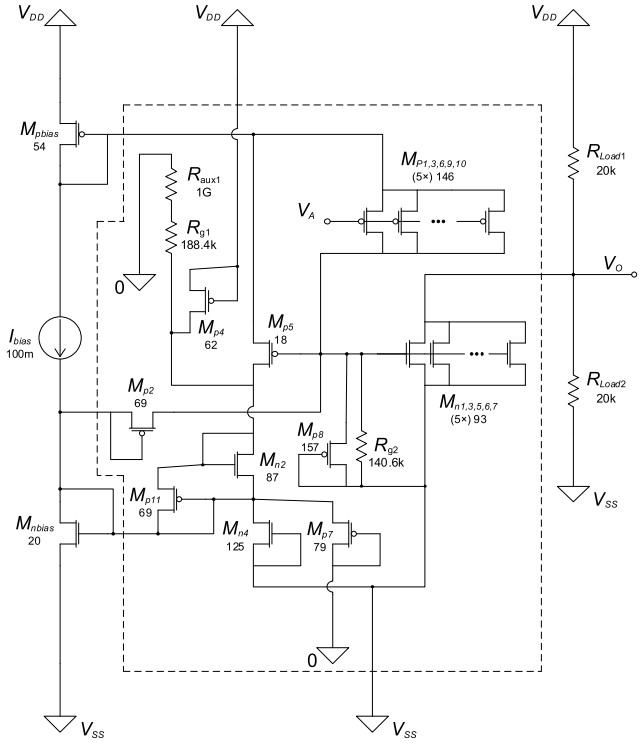

FIGURE 9. 32dB amplifier: circuit synthesized at the end of Phase 0.

TABLE 4. Circuit specifications obtained in phase 0 for the 32dB amplifier.

Description	Value	Comment
Voltage gain	39.4	39.4 ≅ 31.9dB
THD@1kHz	0.13%	V_{iampl} = 2.55mV V_{oampl} ≈ 100mV
Bandwidth	1.8 MHz	$f_L = 0$ f_H ≈ 1.8 MHz@-3dB
GBW	71 MHz	
Input DC current	≈ 0	
Power supply current	2 × 1.067 mA	$I_{DD} \approx I_{SS} \approx 1.067$ mA
FoM	66.5 MHz.pF/mW	C_{Load} = 10 pF
Number of components	21	
Total transistor area	0.186 mm^2	Estimated area used by transistors.

TABLE 5. Some GA parameters for phase 1.

Description	Value	Comment
Mut. deletion probability	0.12	Significantly increase the value of the previous phase.
Weight for number of genes penalty function	Fixed	Use the last value from the previous phase plus an increment of 0.05.
Weight for area penalty function	Fixed	Use the last value from the previous phase.

to be more efficient than trying to accomplish both goals simultaneously.

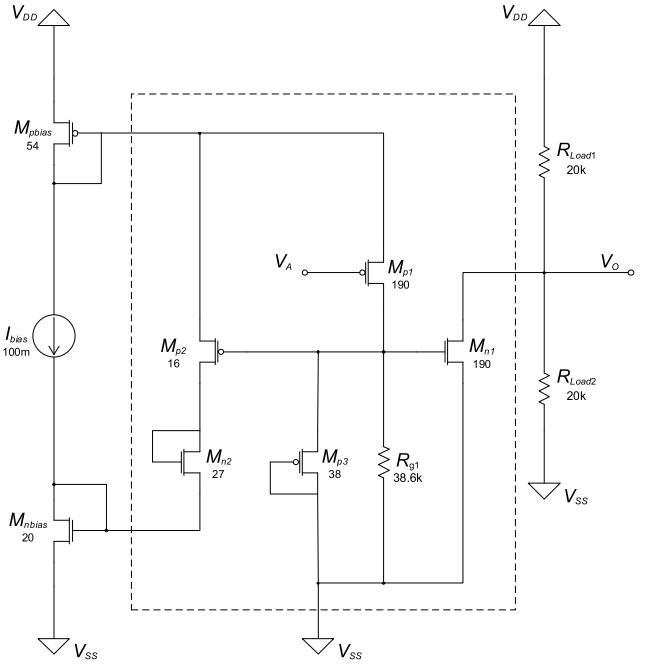

FIGURE 10. 32dB amplifier: circuit synthesized at the end of phase 1.

As before, when the GA progressed from Phase 1 to Phase 2, some parameters were adjusted, as listed in Table 6. The mutation deletion probability is significantly reduced, and the weight of the penalty function that deals with the total area used by transistors is increased between the last iteration of Phase 1 and the first iteration of Phase 2. The intention of these adjustments is to steer the GA to minimize the total area used by transistors without significantly deteriorating what has already been achieved, namely, the gain and the reduced number of components.

TABLE 6. Some GA parameters for phase 2.

Description	Value	Comment
Mut. deletion probability	0.01	Use a value significantly lower than Phase 1.
Weight for number of genes penalty function	Fixed	Use the same value from the previous phase.
Weight for area penalty function	Fixed	Use the same value from the previous phase plus an increment of 0.05.

The circuit synthesized in Phase 2, after a total of 5642 iterations, is shown in Fig. 11, where the circuit topology has been maintained, but most transistor sizes have been reduced (some operating point currents and voltages measured in *NGSPICE* are shown as complementary information about the circuit).

This circuit has virtually the same gain as the circuit synthesized in Phase 0 and uses less transistor area, which was estimated to be 0.046 mm^2 for the circuit obtained in Phase 1 and 0.017 mm^2 for the circuit obtained in Phase 2. These areas

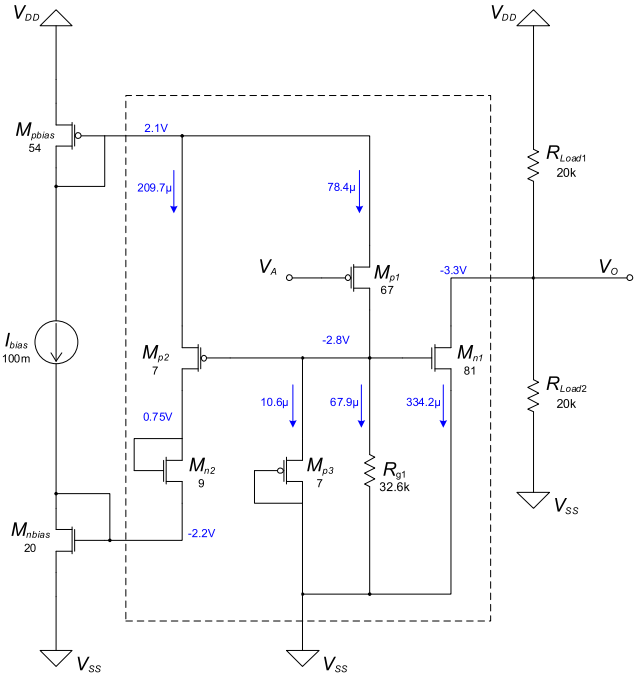

FIGURE 11. A 32dB amplifier: circuit synthesized at the end of Phase 2.

are estimated simply by summing all areas resulting from the product $W \times L$ for all transistors.

The bandwidth and *GBW* are virtually the same as those of Phase 0, but the *FoM* is better because the power supplied to the circuit decreases, as shown in Table 7, where the specifications of this circuit are presented.

Using this example, we demonstrate that circuit synthesis is possible using the techniques described above.

TABLE 7. Results obtained in phase 2 for the 32dB amplifier.

Description	Value	Comment
Voltage gain	40.2	40.2 ≅ 32.09dB
THD@1kHz	0.25%	$V_{i_{ampl}}$ = 2.50mV $V_{o_{ampl}}$ ≈ 100mV
Bandwidth	1.8 MHz	f_L = 0 , f_H ≈ 1.8 MHz
GBW	72 MHz	
Input DC current	≈ 0	
Power supply current	2 × 805.3 µA	$I_{DD} \approx I_{SS} \approx$ 805.3 µA
FoM	89.4 MHz.pF/mW	C_{Load} = 10 pF
Number of components	6	
Total transistor area	0.017 mm^2	Estimated area used by transistors.

B. EXAMPLE 2: SYNTHESIS OF AN AMPLIFIER USING TRANSISTORS WITH L = 0.5µm AND A SPICE BSIM3 MOS MODEL

In this section, we present another example of the synthesis of a DC amplifier using the aforementioned techniques.

The objective was to obtain a DC amplifier made with smaller L transistors (transistors with $L = 0.5\mu$m were used), simulated with a better SPICE model (a BSIM3v3 "Level 8 Version = 3.2.4" model was used with a parameterization close to a 130 nm technology), with a higher *GBW* than the previous one (at least 30 MHz, preferably higher than 40 MHz), using power supplies with lower voltages ($V_{DD} = 0.8$V and $V_{SS} = -0.8$V were used while both load resistors were increased to $R_{\text{load1}} = R_{\text{load2}} = 200$ kΩ and a 10pF load capacitor was added in anticipation of more realistic applications and commonly used test conditions).

The circuit shown in Fig. 12 was generated by the GA, which, when evaluated by SPECTRE/CADENCE 6.0, using a standard 130 nm technology, has the results depicted in Table 8.

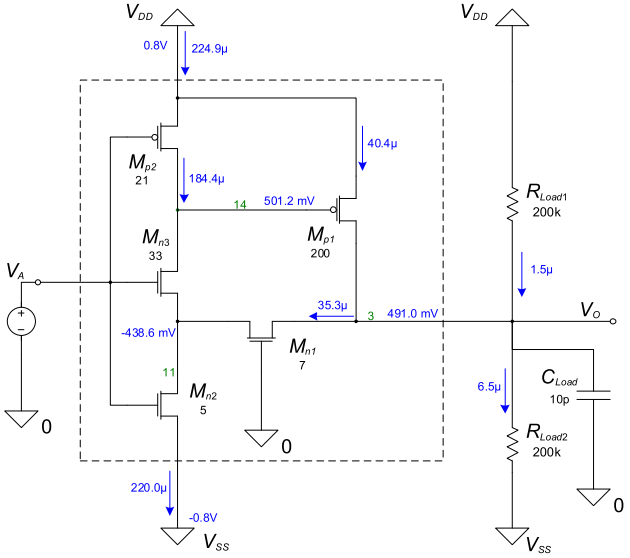

FIGURE 12. A 40dB amplifier using transistors with L = 0.5μm.

TABLE 8. Results obtained for the 40dB amplifier of fig. 12.

Description	Value	Comment
Voltage gain	40.2	102.3 ≅ 40.2dB
THD@1kHz	0.67%	$V_{i_{ampl}}$ = 1.0mV $V_{i_{os}}$ = -6mV $V_{o_{ampl}}$ ≈ 102.5mV
Bandwidth	684 kHz	f_L = 0 f_H ≈ 684 kHz@-3dB
GBW	70 MHz	
Input DC current	≈ 0	
Power supply current	2 × 243.6 µA	$I_{DD} \approx I_{SS} \approx$ 243.6 µA
FoM	1436 MHz.pF/mW	C_{Load} = 10 pF
Number of components	5	
Total transistor area	66.5 µm^2	Estimated area used by transistors.

This circuit fulfills the objectives. Therefore, from the point of view of the subject of circuit synthesis with GAs

it is a success case. The proof of concept is performed: GA synthesis of analog amplifiers is feasible, particularly if auxiliary techniques such as those proposed here are used.

IV. CONCLUSION

In analog circuit synthesis, the topology generation stage is the least developed. A significant portion of the research effort devoted to topology synthesis employs EAs, with a smaller proportion using GAs, usually trying to simultaneously generate an appropriate topology while sizing the circuit components. However, the use of GAs (in circuit synthesis) has somewhat stalled in the last decade.

Many works, either employing EAs or other paradigms, utilize some kind of knowledge-based synthesis, typically in the form of building blocks. However, this knowledge-based synthesis tends to limit the novelty of the generated topologies. Additionally, a significant number of works dedicate effort to generating passive circuits (the synthesis of filters is very common in the literature), but the generation of active circuits, such as amplifiers, is much less often attempted.

Since it is not clear to the authors why better results have not been obtained with GAs in the synthesis of active circuits, namely amplifiers, this work follows the work of many others in exploring the synthesis of circuits with GAs, focusing on the case of amplifiers, and without the use of building blocks, thus possibly increasing the probability of novelty in the generation of topologies.

To produce useful results in the context of flat-level circuit synthesis, GAs must innovate their canonical form and incorporate new techniques that enable them to address the new challenges imposed by these problems.

This paper presents three techniques that are incorporated into a GA kernel for analog circuit synthesis, which contribute to its robustness, leading to better and faster results. The robustness is enhanced because the GA is less sensitive to the initial pseudorandom generator seed and to some parameters. Better results were obtained because the circuits were produced with fewer redundant or useless components. Faster results were obtained because less time is spent in circuit simulation (because the number of components in intermediate circuits is high only during a set of iterations that are strictly necessary).

The contributions of this study are summarized as follows: 1) adaptive probability of chromosome acceptance, 2) circuit cleaning, and 3) segmented evolution.

A heuristic is used to implement an adaptive probability of acceptance of mutated chromosomes according to a measure of the quality of the obtained solutions. This technique enhances the local exploitation capabilities of the GA and reduces its sensitivity to mutation probability. A circuit-cleaning technique is used to eliminate redundant components in the circuit, in which we dynamically adapt the weight of a penalty function to steer the algorithm towards the sub-objective of minimizing the number of components. In addition to these techniques, a segmentation procedure of the GA evolution was presented, where several phases were used, each with different parameterizations, which we consider a valuable technique to extend the steering of the GA towards sub-objectives. Circuits synthesized by the GA are shown using all the techniques described.

Although the authors consider this article to be an important contribution to the field of ADA, and in particular to the efforts being made in this area with GAs, it is by no means intended to present a finished solution for the automatic generation of analog circuits. There is still a lot of work to be done, and it will be a long time before we have a tool that completely automates the automatic generation of any type of analog circuit.

REFERENCES

[1] S. E. Sorkhabi and L. Zhang, "Automated topology synthesis of analog and RF integrated circuits: A survey," *Integration*, vol. 56, pp. 128–138, Jan. 2017.

[2] N. Lourenço, R. Martins, A. Canelas, R. Póvoa, and N. Horta, "AIDA: Layout-aware analog circuit-level sizing with in-loop layout generation," *Integration*, vol. 55, pp. 316–329, Sep. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926016300128

[3] T. McConaghy, P. Palmers, M. Steyaert, and G. G. E. Gielen, "Variation-aware structural synthesis of analog circuits via hierarchical building blocks and structural homotopy," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 9, pp. 1281–1294, Sep. 2009.

[4] Ž. Rojec, Á. Bűrmen, and I. Fajfar, "Analog circuit topology synthesis by means of evolutionary computation," *Eng. Appl. Artif. Intell.*, vol. 80, pp. 48–65, Apr. 2019. [Online]. Available: https://www.sciencedirect.com/-science/article/pii/S0952197619300119

[5] C. Faragó, A. Lodin, and R. Groza, "An operational transconductance amplifier sizing methodology with genetic algorithm-based optimization," *Acta Technica Napocensis. Electronica-Telecomunicatii*, vol. 55, no. 1, pp. 15–20, 2014.

[6] ITRS. (2019). *International Technology Roadmap for Semiconductors*. [Online]. Available: http://www.itrs2.net

[7] C. Ferent and A. Doboli, "Measuring the uniqueness and variety of analog circuit design features," *Integr. VLSI J.*, vol. 44, no. 1, pp. 39–50, Jan. 2011.

[8] Z. Zhao and L. Zhang, "An automated topology synthesis framework for analog integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 4325–4337, Dec. 2020.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[10] J. R. Koza, J. R. Koza, and J. P. Rice, *Genetic Programming: On the Programming of Computers By Means of Natural Selection* (A Bradford Book). Bradford, U.K.: MIT Press, 1992. [Online]. Available: https://books.google.pt/books?id=Bhtxo60BV0EC

[11] G. S. Hornby, "ALPS: The age-layered population structure for reducing the problem of premature convergence," in *Proc. 8th Annu. Conf. Genet. Evol. Comput.*, Jul. 2006, pp. 815–822.

[12] T. Sripramong and C. Toumazou, "The invention of CMOS amplifiers using genetic programming and current-flow analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 11, pp. 1237–1252, Nov. 2002.

[13] M. A. Trefzer, "Evolution of transistor circuits," Ph.D. dissertation, Ruperto-Carola-Univ., Heidelberg, Germany, 2006.

[14] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms* (Wiley Interscience Series in Systems and Optimization). Hoboken, NJ, USA: Wiley, 2001.

[15] D. L. W. Kruiskamp, "DARWIN: CMOS opamp synthesis by means of a genetic algorithm," in *Proc. 32nd Design Autom. Conf.*, Jun. 1995, pp. 433–438.

[16] R. S. Zebulum, M. Vellasco, and M. A. Pacheco, "Variable length representation in evolutionary electronics," *Evol. Comput.*, vol. 8, no. 1, pp. 93–120, Mar. 2000, doi: 10.1162/106365600568112.

[17] H. Yuan and J. He, "Evolutionary design of operational amplifier using variable-length differential evolution algorithm," in *Proc. Int. Conf. Comput. Appl. Syst. Modeling (ICCASM)*, vol. 4, Oct. 2010, pp. 610–614.

[18] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[19] A. S. Sedra and K. C. Smith, *Microelectronic Circuits*. London, U.K.: Oxford Univ. Press, 2014.

[20] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw. (ICNN)*, vol. 4, 1995, pp. 1942–1948.

[21] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 687–697, Jan. 2008. [Online]. Available: https://www.sciencedirect.com/science/-article/pii/S1568494607000531

[22] S. L. Sabat, K. S. Kumar, and S. K. Udgata, "Differential evolution and swarm intelligence techniques for analog circuit synthesis," in *Proc. World Congr. Nature Biol. Inspired Comput. (NaBIC)*, Dec. 2009, pp. 469–474.

[23] F. Wang, Y. Li, K. Li, and Z. Lin, "A new circuit representation method for analog circuit design automation," in *Proc. IEEE Congr. Evol. Comput., IEEE World Congr. Comput. Intell.*, Jun. 2008, pp. 1976–1980.

[24] J. Torresen, "A divide-and-conquer approach to evolvable hardware," in *Evolvable Systems: From Biology to Hardware*, M. Sipper, D. Mange, and A. Pérez-Uribe, Eds., Berlin, Germany: Springer, 1998, pp. 57–65.

[25] R. K. Ratnesh, A. Goel, G. Kaushik, H. Garg, M. Singh, and B. Prasad, "Advancement and challenges in MOSFET scaling," *Mater. Sci. Semicond. Process.*, vol. 134, Nov. 2021, Art. no. 106002. [Online]. Available: https://www.sciencedirect.com/science/-article/pii/S1369800121003498

[26] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed., Boston, MA, USA: Addison-Wesley, 1989.

[27] G. Nicosia, S. Rinaudo, and E. Sciacca, "An evolutionary algorithm-based approach to robust analog circuit design using constrained multi-objective optimization," in *Research and Development in Intelligent Systems XXIV*. London, U.K.: Springer, 2008, pp. 7–20.

[28] J. D. Lohn and S. P. Colombano, "Automated analog circuit synthesis using a linear representation," in *Proc. Int. Conf. Evolvable Syst.* Lausanne, Switzerland: Springer, 1998, pp. 125–133.

[29] R. S. Zebulum, M. A. Pacheco, and M. Vellasco, "Comparison of different evolutionary methodologies applied to electronic filter design," in *Proc. IEEE Int. Conf. Evol. Comput., IEEE World Congr. Comput. Intell.*, May 1998, pp. 434–439.

[30] E. Afacan, N. Lourenço, R. Martins, and G. Dündar, "Review: Machine learning techniques in analog/RF integrated circuit design, synthesis, layout, and test," *Integration*, vol. 77, pp. 113–130, Mar. 2021. [Online]. Available: https://www.sciencedirect.com/science/-article/pii/S0167926020302947

[31] F. Passos, R. Martins, N. Lourenço, E. Roca, R. Castro-López, R. Póvoa, A. Canelas, N. Horta, and F. V. Fernández, "Handling the effects of variability and layout parasitics in the automatic synthesis of LNAs," in *Proc. 15th Int. Conf. Synth., Modeling, Anal. Simulation Methods Appl. Circuit Design (SMACD)*, Jul. 2018, pp. 1–164.

[32] F. Passos, E. Roca, R. Castro-López, N. Horta, and F. V. Fernandez, "Synthesis of mm-wave circuits using-EM-simulated passive structure libraries," in *2019 Int. Conf. Synth., Modeling, Anal. Simulation Methods Appl. Circuit Design (SMACD)*, Jul. 2019, pp. 57–60.

[33] P.-O. Beaulieu, É. Dumesnil, F. Nabki, and M. Boukadoum, "Analog RF circuit sizing by a cascade of shallow neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 12, pp. 4391–4401, Jun. 2023.

[34] E. Afacan and G. Dundar, "A comprehensive analysis on differential cross-coupled CMOS LC oscillators via multi-objective optimization," *Integration*, vol. 67, pp. 162–169, Jul. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/-S0167926018305030

[35] J. Domingues, A. Gusmão, N. Horta, N. Lourenço, and R. Martins, "Accelerating voltage-controlled oscillator sizing optimizations with ANN-based convergence classifiers and frequency guess predictors," in *Proc. 18th Int. Conf. Synth., Modeling, Anal. Simulation Methods Appl. Circuit Design (SMACD)*, Jun. 2022, pp. 1–4.

[36] M. Krasnicki, R. Phelps, R. A. Rutenbar, and L. R. Carley, "MAELSTROM: Efficient simulation-based synthesis for custom analog cells," in *Proc. Design Autom. Conf.*, Jun. 1999, pp. 945–950.

[37] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums, "Anaconda: Simulation-based synthesis of analog circuits via stochastic pattern search," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 6, pp. 703–717, Jun. 2000.

[38] I. Kajitani, T. Hoshino, M. Iwata, and T. Higuchi, "Variable length chromosome GA for evolvable hardware," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1996, pp. 443–447.

[39] M. Campilho-Gomes, R. Tavares, and J. Goes, "Automatic flat-level circuit generation with genetic algorithms," in *Proc. 11th Adv. IFIP WG 5.5/SOCOLNET Doctoral Conf. Comput., Electr. Ind. Syst. (DoCEIS)*. Costa de Caparica, Portugal: Springer, Jul. 2020, pp. 101–108.

[40] R. Harjani, R. A. Rutenbar, and L. R. Carley, "OASYS: A framework for analog circuit synthesis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 8, no. 12, pp. 1247–1266, Dec. 1989.

[41] A. Doboli and R. Vemuri, "Exploration-based high-level synthesis of linear analog systems operating at low/medium frequencies," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 11, pp. 1556–1568, Nov. 2003.

[42] SourceForge. (2019). *Mixed Mode-Mixed Level Circuit Simulator Based on Berkeley's SPICE 3F5*. [Online]. Available: http://ngspice.sourceforge.net/

[43] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanovic, "BagNet: Berkeley analog generator with layout optimizer boosted with deep neural networks," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[44] A. Canelas, R. Póvoa, R. Martins, N. Lourenço, J. Guilherme, J. P. Carvalho, and N. Horta, "FUZYE: A fuzzy *c*-means analog IC yield optimization using evolutionary-based algorithms," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 1, pp. 1–13, Jan. 2020.

[45] J. H. Holland, *Adaptation in Natural and Artificial Systems*, 2nd ed., Ann Arbor, MI, USA: Univ. of Michigan Press, 1992.

[46] N. Paulino, J. Goes, and A. Steiger-Garcao, "Design methodology for optimization of analog building blocks using genetic algorithms," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 5, May 2001, pp. 435–438.

[47] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Comput. Methods Appl. Mech. Eng.*, vol. 191, nos. 11–12, pp. 1245–1287, Jan. 2002.

[48] R. Santos-Tavares, "Time-domain optimization of amplifiers based on distributed genetic algorithms," Ph.D. thesis, Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa, Lisbon, Portugal, 2010.

[49] B. Hutt and K. Warwick, "Synapsing variable-length crossover: Meaningful crossover for variable-length genomes," *IEEE Trans. Evol. Comput.*, vol. 11, no. 1, pp. 118–131, Feb. 2007.

[50] D. S. Deif and Y. Gadallah, "Wireless sensor network deployment using a variable-length genetic algorithm," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2014, pp. 2450–2455.

[51] S. N. Pawar and R. S. Bichkar, "Genetic algorithm with variable length chromosomes for network intrusion detection," *Int. J. Autom. Comput.*, vol. 12, no. 3, pp. 337–342, Jun. 2015.

[52] M. Barros, J. Guilherme, and N. Horta, "Analog circuits optimization based on evolutionary computation techniques," *Integr. VLSI J.*, vol. 43, no. 1, pp. 136–155, Jan. 2010.

[53] Y. Nourani and B. Andresen, "A comparison of simulated annealing cooling strategies," *J. Phys. A: Math. Gen.*, vol. 31, no. 41, pp. 8373–8385, Oct. 1998, doi: 10.1088/0305-4470/31/41/011.

[54] C. Mattiussi and D. Floreano, "Analog genetic encoding for the evolution of circuits and networks," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 596–607, Oct. 2007.

[55] S. Ando and H. Iba, "Analog circuit design with a variable length chromosome," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 2, 2000, pp. 994–1001.

MIGUEL CAMPILHO-GOMES (Member, IEEE) received the bachelor's and M.Sc. degrees in electrical and computer engineering (ECE) from the Instituto Superior Técnico (IST), Lisbon, in 1988 and 1999, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the School of Sciences and Technology, NOVA University of Lisbon (NOVA), Portugal.

Since December 1989, he has been with the Department of Electronics, Telecommunications, and Computer Engineering (DEETC), Instituto Superior de Engenharia de Lisboa (ISEL), Instituto Politécnico de Lisboa (IPL), where he is currently an Adjunct Professor. From 1985 to 1993, he was a Research Assistant with the Institute of Systems and Computer Engineering (INESC), Lisbon, previously a Student (until 1989), and later a Graduate Researcher. His research interests include artificial neural nets, analog and digital electronics in integrated circuits, evolutionary algorithms, and electronic design automation of analog circuits and systems (EDA/CAD).

Mr. Campilho-Gomes has membership in the IEEE Circuits and Systems Society (CAS), IEEE Computer Society, and IEEE Education Society. He joined the IEEE Council on Electronic Design Automation.

RUI TAVARES (Member, IEEE) was born in Oeiras, Portugal, in 1975. He received the bachelor's, M.Sc., and Ph.D. degrees from the NOVA University of Lisbon (UNL), Portugal, in 1998, 2001, and 2010, respectively.

Since September 1998, he has been with the Department of Electrical Engineering and Computers (DEEC), Faculdade de Ciências e Tecnologia (FCT), UNL, where he is currently an Assistant Professor. Since 1998, he has also been a Senior Researcher with the Centre for Technology and Systems (CTS), UNINOVA. From 1998 to 2001, he was an Assistant Researcher with the Interoperability Supported by Standards Group (GRIS), UNINOVA, where he actively participated in several national and joint European cooperative projects in science and technology, such as ESPRIT IV 22056-FunSTEP. Since 2001, he has been with the Analog Microelectronics Design Group, CTS/UNINOVA, where he has been actively participating in several national and joint European cooperative projects in science and technology, such as H2020-Proteus. He has published several papers in leading international conferences, international journals, and a book. His research interests include electronic design automation of analog circuits and systems (EDA/CAD), and the design of low-power and low-voltage analog integrated circuits.

Prof. Tavares has been a member of the IEEE Circuits and Systems (CAS) Society and IEEE Solid-State Circuits (SSC) Society. He participated in the organizing committee of the Seasonal Schools: CAS4IoT'2016, CAS4IIoT' 2018, and the international conferences ESSCIRC/ESSDERC'2023 and IEEE ISCAS'2015. He has been serving as a reviewer for many IEEE Conferences.

JOÃO GOES (Senior Member, IEEE) received the bachelor's degree in electrical and computer engineering (ECE) from Instituto Superior Técnico (IST), Lisbon, in 1992, the M.Sc. and Ph.D. degrees in ECE from the Technical University of Lisbon, in 1996 and 2000, respectively, and the Habilitation degree in electronics from the NOVA University of Lisbon (NOVA), in 2012.

Since 1998, he has been with the Department of Electrical and Computer Engineering (DEEC), School of Sciences and Technology (FCT), NOVA, where he has been a Full Professor, since 2017. From 2012 to 2019, he headed the DEEC, comprising 50 professors and over 1000 students. From 2012 to 2017, he was the Director of the Centre of Technology and Systems (CTS), Research Institute for New Technologies (UNINOVA), leading nearly 50 Senior Researchers with a Ph.D., over 70 collaborators, and more than 90 Ph.D. students. Since 2023, he has been the Executive Director of UNINOVA. He has been elected as a member of the Scientific Council of FCT NOVA for 12 years and the General Council of NOVA, the top governing body of the University, from 2022 to 2025. In 2003, he co-founded and served as the CTO of ACACIA Semiconductor S.A., a Portuguese engineering company specializing in high-performance analog front-end products (now RENESAS). From 1997 to 1998, he was the Project Manager at CHIPIDEA S.A. (now SYNOPSYS). He was the first engineer to be hired. From 1993 to 1997, he was a Researcher at the Integrated Circuits and Systems Group (GCSI), IST, conducting research on data converters and filters. Since 1992, he has participated in and led several National and European projects related to science, technology, networking, and training. He has been the Primary Investigator for over 20 projects. He supervised (concluded) 20 Ph.D. students and over 50 M.Sc. Theses. He has published over 200 papers in international journals (more than 50) and IEEE leading conferences (ISSCC, VLSI, CICC, and ESSCIRC), and holds four international patents. He is the co-author of eight books.

Dr. Goes is a member of the IEEE Circuits and Systems (CAS) Society and the IEEE Solid-State Circuits (SSC) Society. He is the co-author of the journal article recipient of the 2012 IEEE CASS Outstanding Young Author Award and a Co-Winner (1st Place) of the first edition of the ''Innovation Award INCM,'' in 2016. He was the Chairperson of the IEEE CASS Analog Signal Processing Technical Committee, from 2013 to 2015. He is the General Chairperson of IEEE ESSCIRC-ESSDERC'2023, held in Lisbon, Portugal, in September 2023. He was one of the key organizers and the TPC Co-Chairperson of IEEE ISCAS'2015, held in Lisbon, in May 2015, as well as the TPC Co-Chairperson of PRIME'2016. He was Associate Editor (AE) of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, from 2016 to 2023. Since 2024, he has been an AE of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS.