# Massively Multi-Topology Sizing of Analog Integrated Circuits

Pieter Palmers, Trent McConnaghy, Michiel Steyaert, Georges Gielen
K.U.Leuven - Department of Electrical Engineering - ESAT/MICAS
Kasteelpark Arenberg 10
B-3001 Heverlee (Leuven, Belgium)
georges.gielen@esat.kuleuven.be

## Abstract

*This paper demonstrates a system that performs multi-objective sizing across 100,000 analog circuit topologies simultaneously, with SPICE accuracy. It builds on a previous system, MOJITO, which searches through 3500 topologies defined by a hierarchically-organized set of 30 analog blocks. This paper improves MOJITO's results quality via three key extensions. First, it enlarges the block library to enable symmetrical transconductance amplifiers and more. Second, it improves initial topology diversity via optimization-based constraint satisfaction. Third, it maintains topology diversity during search via a novel multi-objective selection mechanism, dubbed TAPAS. MOJITO+TAPAS is demonstrated on a problem with 6 objectives, returning a tradeoff holding 17438 nondominated designs. The tradeoff is comprised of 152 unique topologies that include the newly-introduced topologies. 59 designs across 12 topologies designs outperform an expert-designed reference circuit.*

## 1. Introduction

Front-end analog circuit design involves determining a circuit topology and its device sizes. Automating aspects of front-end design has promise of improved time-to-market, productivity, and / or quality of designs [1]. While there are now industrially palatable tools for automated cell-level sizing such as [2], so far no industrial tools give broad support for topology design. Such a tool needs: to return results that are trustworthy enough to commit to silicon, consider a sufficiently rich set of topologies so that the designer does not have to intervene, easily adapt to new technology nodes, be general enough for a variety of circuit types, have low setup time for the designer, have low runtime, and have results with quality comparable to manual design.

There has been much research in cell-level topology design tools. Open-ended approaches like [3–6] search across unstructured combinations of transistors, but results are not trustworthy [7]. [8–13] use rule-based systems or pre-set behavior-to-structure mappings, which requires excessive setup effort. DARWIN [14] and MINLP [15] define a flat combinatorial space of possible topologies which need just structural information, but the flat approach generalizes poorly and has only been shown on $<$100 topologies. In contrast, MOJITO [16] uses a *hierarchically*-defined set of structural blocks to define a space of thousands of topologies; but it has not yet outperformed manual designs.

**Table 1. Topology Synthesis Approaches**

| **Technique** | **# Topologies** (op amp space) | **Trustworthy**? | **Shown outperform manual**? |
|---|---|---|---|
| Open-ended [3–6] | $>> 10^{12}$ | NO | NO |
| DARWIN [14] | 24 | YES | NO |
| MINLP [15] | 64 | YES | NO |
| MOJITO [16] | 3528 | YES | NO |
| **MOJITO+TAPAS (this work)** | **101904** | **YES** | **YES** |

This paper's key contribution enhances MOJITO to generate designs competitive with manual across the whole performance tradeoff. To do so, the library is extended to include all expected final Pareto-optimal topologies, leading to 100,000 possible topologies. Also, search is enhanced to maintain diversity so that final Pareto-optimal topologies optimize long enough to reach their full potential, via Topology Aware Pareto Age-layered Structure (TAPAS). Table 1 summarizes.

This paper is organized as follows. Section 2 specifies the problem, and section 3 reviews MOJITO. Section 4 describes the enlarged library. Sections 5-7 illuminate the topology diversity issue, propose algorithms to fix it, and validate the fix, respectively. Section 8 examines final designs' quality, and section 9 concludes.

## 2. Problem Specification

The topology design problem is:

$$\begin{array}{rll} minimize & f_i(\phi) & i = 1..N_f \\ s.t. & g_j(\phi) \leq 0 & j = 1..N_g \\ & h_k(\phi) = 0 & k = 1..N_h \\ & \phi \in \Phi & \end{array} \quad (1)$$

where $\Phi$ is the space of possible topologies and sizings. The algorithm traverses $\Phi$ to return a Pareto Optimal Set (POS) $Z = \{\phi_1^*, \phi_2^*, \cdots, \phi_{N_{ND}}^*\}$ on $N_f$ objectives, $N_g$ inequality constraints, and $N_h$ equality constraints. We can minimize all objectives, have inequality constraints aim $\leq 0$, and equality constraints aim $= 0$, without loss of generality.

## 3. MOJITO Review

MOJITO defines a set of possible topologies via hierarchically-organized analog blocks, and searches across the library with NSGA-II [17], a multi-objective evolutionary algorithm (MOEA). It exploits the hierarchy to mix topology sub-blocks [18]. It uses SPICE in-the-loop, and parallel computing. Table 2 gives the top-level algorithm. An Age Layered Population Structure (ALPS) [19] ensures reliable convergence and continual exploration of new regions. Each age layer $P_k$, $k = 1..K$, holds $N_L$ individuals. $P_1$ allows individuals with genetic age 0-9, $P_2$ allows age 0-19, and so on. If an individual gets too old for a fitness layer, it gets removed from that layer. Selection at layer $k$ uses the individuals from layer $k$ and $k-1$ as candidates, so younger high-fitness individuals can propagate to higher layers. Every $N_a$ generations (line 3), a new age layer may be added (lines 4-5), and initial individuals enter layer $k$=0 via random sampling, $P_{0,i} \sim \Phi$ (line 6). Line 8 runs one generation of NSGA-II. An external archive maintains $Z$.

**Table 2. Procedure MojitoSynthesis()**

**Inputs:** $\Phi$, $N_a$, $K$, $N_L$
**Outputs:** $Z$
1. $N_{gen} = 0$; $Z = \emptyset$; $P = \emptyset$
2. while stop($N_{gen}, \ldots$) $\neq True$:
3. if $(N_{gen} \% N_a) = 0$:
4. if $\|P\| < K$:
5. $P_{\|P\|+1} = \emptyset$
6. $P_{0,i}$ = InitialCircuit($\Phi$), $i = 1..N_L$
7. for $k$ = 1 to $\|P\|$:
8. $(P_k, Z)$ = OneMOEAGen($P_k$, $P_{k-1}$, $Z$)
9. $N_{gen} = N_{gen} + 1$
10. return $Z$

## 4. Enlarged Building Block Library

The MOJITO library of [16] included 1- and 2-stage amplifiers, PMOS vs. NMOS loads, PMOS vs. NMOS inputs, stacked vs. folded cascode vs. non-cascode inputs, cascode vs. non-cascode vs. resistor loads, level shifting, several different current mirrors, and single-ended and differential inputs. Two of the most frequently used blocks in CMOS are folded and symmetrical operational transconductance amplifiers (F/S OTAs), as shown in Figure 1 left and center. While the original library supported F OTAs, it did not support S OTAs. So, support was given by adding current mirror folding blocks. Support was also added for cascoding of folding transistors, as shown in Figure 1 right. This increases topology count by 30x, to about 100,000.

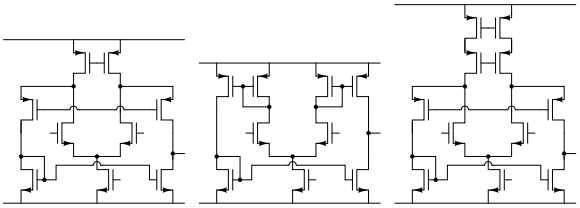

**Figure 1. Left to right: Folded OTA, symmetrical OTA, folded cascoded folder OTA**

## 5. Experiments: Diversity Issues

With the enlarged library, we ran MOJITO using the setup of Table 3. Having 6 objectives, we expect a large final Pareto Optimal Set (POS) with high topological diversity, including the presence of the new OTAs. Figure 2 tracks topology count over time. The count in the population spikes every $10^{th}$ generation, when randomly-generated individuals are added. But after each spike, the count dives as poorly-performing topologies are replaced by well-performing topologies. The feasible and nondominated topology counts are strikingly low. Once the algorithm finally finds a feasible topology (generation 40), that topology and variants dominate the population. By generation 80, the POS has 3000 individuals but only 5 unique topologies.

Figure 3 shows that the count of F/S OTA topologies in the top layer dimishes over time, which is surprising because these topologies should be good enough to survive. This is because despite being more complex circuit, a 2-stage amplifier is actually *easier* to stabilize than a 1-stage (the 1-stage's nondominant pole is a function of several parasitic capacitances, which are harder to control) [21].

The lack of expected F/S topologies is our "canary in the mine." MOJITO needs to generate and protect topologies

**Table 3. Experimental setup parameters**

| | |
|---|---|
| Search Space | 101,904 topologies + sizes & biases |
| Objectives | maximize DC gain, max. GBW, minimize power, min. area, max. dynamic range (DR), max. slew rate (SR) |
| Constraints | Device operating constraints [20], DC gain $>$ 20 dB, GBW $>$ 10 MHz, power $<$ 100 mW, $10^{-14} \leq$ area $\leq 10^{-4}m^2$, PM $> 65°$, DR $> 0.1$ V, SR $> 10^6$ V/s |
| Tech. & Test harness | 0.18$\mu$m CMOS, 1 pF load capacitance, 1.8 V supply voltage, HSPICE simulator |
| Search Algorithm | $K$ = 10 age layers, 200 individuals per age layer, $N_a$ = 10 generation intervals |

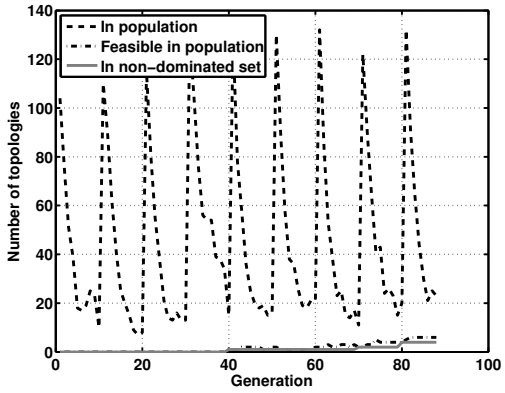


**Figure 2. Number of topologies vs. generation (status quo MOJITO)**

long enough for them to optimize to their full potential.

## 6. Algorithms to Maintain Diversity

This section describes two approaches to maintain topology diversity: one for initial generation of individuals, and one for MOEA selection.

### 6.1 Maintaining Initial Diversity

The algorithm of Table 4 replaces the random sampling in step 6 of Table 2. It aims to defer competition among randomly-generated topologies until each topology is at least close to feasible. It does so by optimizing sizings & biasings in a series of constraint-satisfaction "gates" that are successively more expensive to evaluate: from function device operating constraints (DOCs) (lines 2-5), to simulation-based DOCs (lines 6-9), and finally to performance constraints (lines 10-13). In all three gates, mutateSizings() applies Gaussian mutation to all sizing & biasing

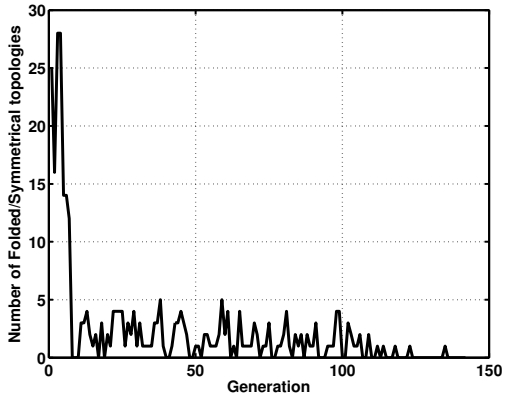


**Figure 3. Number of unique F/S OTAs in top age layer, vs. generation**

parameters of the design.

**Table 4. Procedure InitialCircuit()**

**Inputs:** $\Phi$
**Outputs:** $\phi \in \Phi$
1. $\phi \sim \Phi$
2. while meetsFuncDOCs($\phi$) $\neq True$:
3. $\quad \phi'$ = mutateSizings($\phi$)
4. $\quad$ if funcDOCsCost($\phi'$) $<$ funcDOCsCost($\phi$):
5. $\quad\quad \phi = \phi'$
6. while meetsSimDOCs($\phi$) $\neq True$:
7. $\quad \phi'$ = mutateSizings($\phi$)
8. $\quad$ if simDOCsCost($\phi'$) $<$ simDOCsCost($\phi$):
9. $\quad\quad \phi = \phi'$
10. while meetsPerfConstraints($\phi$) $\neq True$:
11. $\quad \phi'$ = mutateSizings($\phi$)
12. $\quad$ if perfCost($\phi'$) $<$ perfCost($\phi$):
13. $\quad\quad \phi = \phi'$
14. Return $\phi$

### 6.2 Maintaining Diversity During Search

This section first discusses NSGA-II diversity issues, a promising alternative called MOEA/D and its issues, and finally a topology-preserving enhancement called TAPAS.

**NSGA-II.** In a topologically rich search space, NSGA-II's "nondominated sort" operation results in a relatively rapid domination of easy-to-optimize topologies, as we saw in section 5. To maintain a sufficiently high chance that difficult topologies are also considered, impractically large populations have to be used.

**MOEA/D** is a promising MOEA [22]. Its basic idea is to run $N_L$ single-objective local optimizations simul-

taneously, where each local optimization $i$ minimizes a weighted sum across objective costs $\boldsymbol{w_i}^T \boldsymbol{f}(\phi)$. Each local optimization points to a different direction $\boldsymbol{w_i}$, and directions are well-spread $\boldsymbol{W} = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_{N_L}}\}$. Selection and crossover for a given direction $\boldsymbol{w_i}$ considers the individuals of neighboring directions $j \in N(\boldsymbol{w_i})$. The MOEA/D version of OneMOEAGen() is in Table 5, and illustrated in Figure 4.

**Table 5. Procedure OneMOEA/DGen()**

**Inputs:** $P_k$, $P_{k-1}$, $Z$, $\boldsymbol{W}$
**Outputs:** $P'_k$, $Z'$
1. $P'_k = P_k$
2. for $i$ in $\{1, 2, \ldots, N_L\}$:
3. $B$ = {best ind. according to $\boldsymbol{w_i}$} $\cup$ {best ind. acc. to $\boldsymbol{w_j}$}$\forall j, j = N(\boldsymbol{w_i})$
4. $P_{sel} = \{\sim unif(B), \sim unif(B)\}$
5. $\phi_{child}$ = ApplyOperators($P_{sel}$)
6. $\phi_{child}$ = Evaluate($\phi_{child}$)
7. $Z'$ = NondominatedFilter($Z \cup \phi_{ch}$)
8. for $j$ in $N(\boldsymbol{w_i})$:
9. if $\phi_{child}$ is better than $P'_{k,j}$ acc. to $\boldsymbol{w_j}$:
10. replace ind. $P'_{k,j}$ with $\phi_{child}$
11. return $(P'_k, Z')$

In experiments, we found that MOJITO+MOEA/D could efficiently generate a smooth Pareto Front without performance gaps, but it contained far fewer nondominated individuals than NSGA-II. In a single-topology space this may not be a problem, since the retained individuals are the "best" for one (or more) weights. However, in a topology-rich search space this is very undesirable, because MOEA/D's local-optimization perspective biases towards easy-to-optimize topologies, and the more difficult topologies do not survive.

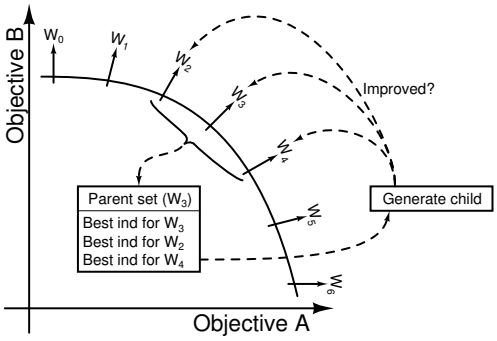


**Figure 4. MOEA/D in action**

**TAPAS.** We designed TAPAS to incorporate a topology-preservation mechanism. It is like the MOEA/D algorithm of Table 5, except for line 3's mechanism to compute $B$: for a weight $\boldsymbol{w_i}$ or $\boldsymbol{w_j}$, instead of choosing one best individual according to the weight, the $M$ best *unique topologies* are chosen. Figure 5 illustrates (for $M$ = 3). This ensures a much larger topological diversity by design, and guarantees that at least M topologies will be present in the active population. If they are not present from the initial population, mutation and crossover operators will introduce new topologies along the way.

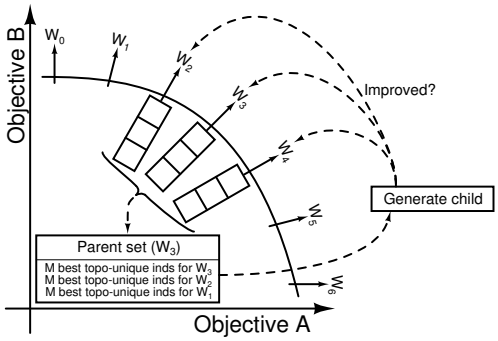


**Figure 5. TAPAS in action**

## 7. Experiments: Improved Diversity?

In this section, we validate that MOJITO+TAPAS can indeed improve diversity. The first experiment investigates how much constraint satisfaction improves initial individuals. 200 individuals were generated with constraint satisfaction (section 6.1), and 200 without. The populations were merged, then filtered into a Pareto Optimal Set (POS). Table 6 shows that constraint-satisfaction is superior.

**Table 6. Comparison of initial generation approaches**

| Approach | # inds in merged POS | % inds in merged POS | # unique topos. in merged POS |
|---|---|---|---|
| no constraint satisfaction | 0 | 0% | 0 |
| **constraint satisfaction** | 200 | 100% | 10 |

The next experiment investigates the effect of NSGA-II vs. TAPAS selection, by doing one run of MOJITO + NSGA-II and one of MOJITO + TAPAS. For a fair comparison, they had identical problem setup, identical initial constraint-satisfaction individuals, and settings giving equal runtime (see Table 7). Figure 6 tracks the total F/S topology count in the population. TAPAS clearly succeeds in maintaining more F/S topologies. When looking only at the top

age layer, shown in Figure 7, the improvement offered by TAPAS is even more apparent: whereas the NSGA-II top age layer gradually loses its F/S topologies, TAPAS always retains them.

These experiments confirm that the new topology diversity mechanisms are indeed performing according to their design.

**Table 7. Experimental parameters**

| | |
|---|---|
| NSGA-II | $K$ = 10, $N_a$ = 10, $N_L$ = 200 inds. per age layer, 200 inds. in age layer initialization |
| TAPAS | $K$ = 10, $N_a$ = 10, 200 weights, 5 neighbours per weight, $M$ = 20 topologies per weight, 20 inds. in age layer initialization |
| Runtime | 7 days across five 2x 4-core Xeon machines |

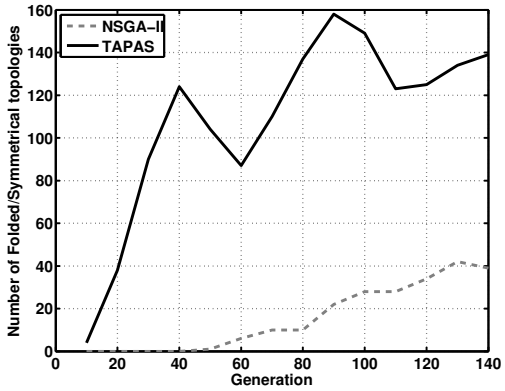


**Figure 6. Total number of F/S topologies in population, vs. generation**

## 8. Experimental Results on Quality of Designs

In this section, we investigate the quality of the results generated by MOJITO. First, we compare the quality of MOJITO + NSGA-II results with MOJITO + TAPAS results by doing one run for each as in section 7, then merging each run's POS into an overall POS. Table 8 shows that the TAPAS results cover more of the tradeoff than NSGA-II.

**Table 8. Compare designs in final POS**

| Approach | # inds in merged POS | % inds in merged POS |
|---|---|---|
| MOJITO+NSGAII | 3991 | 39.5% |
| **MOJITO+TAPAS** | 6092 | 60.5% |

Figure 8 shows a cross section of the MOJITO + TAPAS Pareto Front in the Gain-GBW plane. It has 17438 designs,

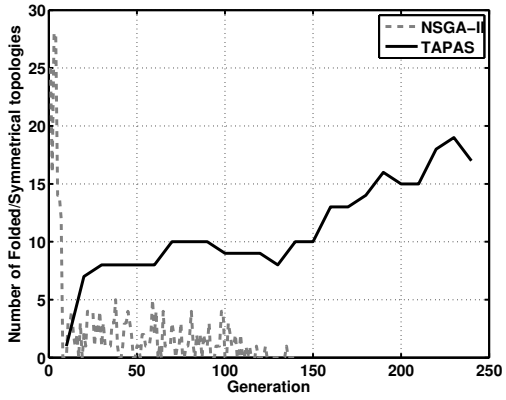


**Figure 7. Number of F/S topologies in top age layer, vs. generation**

comprising 152 unique topologies having 1 or 2 stages. For comparison, a reference design was extracted from a CMOS data converter designed by an expert analog designer; it is also shown. We see that MOJITO designs compete with the reference design along these axes. In fact, MOJITO results were highly competitive on all performances, as Table 9 shows. Indeed, MOJITO + TAPAS found 59 designs that were better than the manual reference design, distributed over 12 unique topologies.

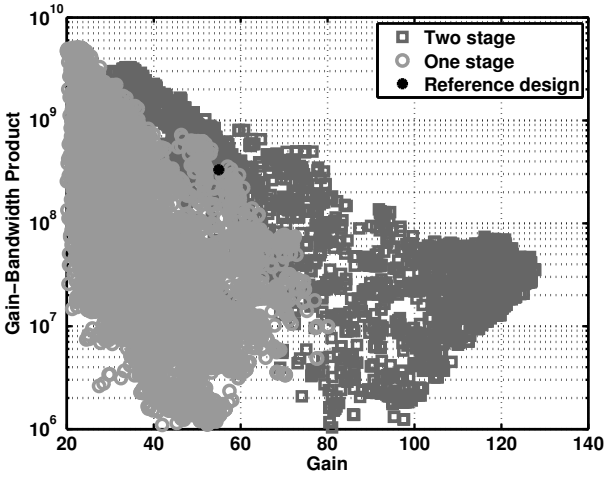


**Figure 8. Pareto Front cross-section for Gain vs GBW, showing all results**

## 9. Conclusion

This paper presented MOJITO + TAPAS, a topology design tool that considers 100,000 possible analog circuit

**Table 9. Comparison to Reference Design**

| Performance | Aim | Manual | MOJITO | MOJITO |
|---|---|---|---|---|
| **Topology** | - | Sym-metrical | Tele-scopic | Miller |
| **Gain (dB)** | max. | 55 | 53.06 | 56.24 |
| **GBW (MHz)** | max. | 330 | 474 | 413 |
| **DR (V)** | max. | 1.2 | 1.01 | 1.53 |
| **SR (V/$\mu s$)** | max. | 380 | 389 | 554 |
| **Power (mW)** | min. | 2.26 | 1.02 | 7.40 |
| **Area ($\mu m^2$)** | min. | - | 218 | 3974 |
| **PM (°)** | $\geq 65$ | 65 | 67 | 65.18 |
| **DOCs** | $met?$ | YES | YES | YES |

topologies to generate a set of Pareto-Optimal sized topologies. It has SPICE accuracy, low setup cost, and reasonable runtime. To maintain high topology diversity, a constraint-satisfaction mechanism and a novel multi-objective selection mechanism were introduced. In the results tradeoff across 6 objectives, there were 17438 designs across 152 unique topologies; and among them, 59 designs across 12 topologies outperformed an expert-designed reference circuit.

## References


[1] R.A. Rutenbar, G.E. Gielen, and B.A.A. Antao, Eds. *Computer aided design of analog integrated circuits and systems*. IEEE Press, 2002, pp. 3–30.

[2] Cadence Design Systems Inc., Virtuoso NeoCircuit Product, *http://www.cadence.com*, 2008.

[3] J.R. Koza, D. Andre, F.H. Bennett III, and M. Keane. *Genetic programming 3: Darwinian invention and problem solving*. Morgan Kaufman, 1999.

[4] T. Sripramong and C. Toumazou, "The invention of CMOS amplifiers using genetic programming and current-flow analysis," *IEEE Trans. CAD*, 21(11), 2002, pp. 1237-1252.

[5] T.R. Dastidar, P.P. Chakrabarti, and P. Ray, "A synthesis system for analog circuits based on evolutionary search and topological reuse." *IEEE Trans. Ev. Comp.*, 9(2), 2005, pp. 211–224.

[6] A. Das and R. Vemuri, "Topology synthesis of analog circuits based on adaptively generated building blocks," *Proc. DAC*, 2008, pp. 44-49.

[7] T. McConaghy and G.E. Gielen, "Genetic programming in industrial analog CAD: Applications and challenges," *Genetic Prog. Theory and Practice III*, T. Yu et al., Eds., ch. 19, Springer, 2005, pp. 291–306.

[8] R. Harjani, R.A. Rutenbar, and L.R. Carley, "OASYS: A framework for analog circuit synthesis," *IEEE Trans. CAD*, vol. 8, no. 12, 1992, pp. 1247–1266.

[9] F.M. E1-Turky and R.A. Nordin, "BLADES: An expert system for analog circuit design," *Proc. ISCAS*, 1986, pp. 552–555.

[10] B. De Smedt and G.E. Gielen, "WATSON: Design space boundary exploration and model generation for analog and RF/IC design," *IEEE Trans. CAD*, 22(2), 2003, pp. 213–223.

[11] H.Y. Koh, C.H. Séquin, and P.R. Gray, "OPASYN: A compiler for CMOS operational amplifiers," *IEEE Trans. CAD*, vol. 9, 1990, pp. 113–125.

[12] Z. Ning, A.J. Mouthaan, and H. Wallinga, "SEAS: A simulated evolution approach for analog circuit synthesis," *Proc. CICC*, 1991, pp. 5.2-1-4.

[13] C. Toumazou, C.A. Makris, and C.M. Berrah, "ISAID: A methodology for automated analog IC design," *Proc. ISCAS*, vol. 1, 1990, pp. 531–555.

[14] W. Kruiskamp and D. Leenaerts, "DARWIN: CMOS opamp synthesis by means of a genetic algorithm," *Proc. DAC*, 1995.

[15] P.C. Maulik, L.R. Carley, and R.A. Rutenbar, "Integer programming based topology selection of cell level analog circuits," *IEEE Trans. CAD*, 14(4), 1995.

[16] T. McConaghy, P. Palmers, G.E Gielen, M. Steyaert, "Simultaneous multi-topology multi-objective sizing across thousands of analog circuit topologies," *Proc. DAC*, 2007.

[17] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Ev. Comp.*, 6(2), 2002.

[18] J.R. Koza. *Genetic programming: On the programming of computers by means of natural selection*. Cambridge, MA: MIT Press, 1992.

[19] G.S. Hornby, "ALPS: The age-layered population structure for reducing the problem of premature convergence," *Proc. GECCO*, 2006, pp. 815–822.

[20] H.E. Graeb, S. Zizala, J. Eckmueller, and K. Antreich, "The sizing rules method for analog integrated circuit design," *Proc. ICCAD*, 2001, pp. 343–349.

[21] W.M.C. Sansen. *Analog design essentials*. Springer. ISBN: 0387257462, 2006.

[22] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Trans. Ev. Comp.*, 11(6), 2007.