Simultaneous Multi-Topology Multi-Objective Sizing Across Thousands of Analog Circuit Topologies

Trent McConaghy, Pieter Palmers, Georges Gielen, Michiel Steyaert
K.U. Leuven, ESAT-MICAS
Kasteelpark Arenberg 10
B-3001 Leuven, Belgium
trent.mcconaghy, pieter.palmers, georges.gielen, michiel.steyaert @esat.kuleuven.be

ABSTRACT
This paper presents MOJITO, a system which optimizes across thousands of analog circuit topologies simultaneously, and returns a set of sized topologies that collectively provide a performance tradeoff. MOJITO defines a space of possible topologies as a hierarchically organized combination of trusted analog building blocks. To minimize the setup burden: no topology selection rules or abstract behaviors need to be specified, and performance calculations are SPICE-based. The search algorithm is a novel multi-objective evolutionary algorithm that uses an age-layered population structure to balance exploration vs. exploitation. Results are shown for a space having 3528 one- and two-stage operational amplifier topologies.

Categories and Subject Descriptors
B.7.2 [**Integrated Circuits**]: Design Aids

General Terms
Analog, Algorithms, Design, Synthesis

Keywords
Analog, mixed-signal, integrated circuits, computer-aided design.

1. INTRODUCTION
The choice of cell-level analog circuit topology can have a giant impact on the performance of a system, and its implications resonate throughout the rest of the design cycle. Even the best circuit optimizers can only produce as good a result as the chosen topology allows [18]. Unfortunately, a suboptimal choice can occur: it may not suit larger statistical variations or new effects such as proximity [6] when the process changes; functionality requirements may be qualitatively new to the designer; or the designer may unknowingly miss an advance in topology design. The process to choose a topology has traditionally been very iterative, and intertwined with the choice of specifications. As Figure 1 (a) shows, many topologies may be tried for a fixed set of specifications, and if needed, the specs themselves may change. As Figure 1 (b) shows, one can remove the iteration over topology choices by making it part of the search itself; but that still needs iterations over specs, and means that just a *feasible* (not optimal) solution will be found. *Multi-objective* sizers [4] bypass the specs issue by optimizing on >1 goals to generate performance tradeoffs as part of the search task. But so far, multiobjective sizers have only worked on one topology at a time. This means that to get an optimal tradeoff across multiple topologies, one needs one sizing run per topology before merging the topologies, as Figure 1 (c) shows. In system-level design, tradeoffs of topologies are merged; then, search at higher-level blocks implicitly performs topology selection of lower-level blocks [7]. But tradeoff-merging has limits: it would be extremely tedious and time-consuming to do a different sizing run for each of 100 or 1000 topologies. Figure 1 (d) shows the ideal approach, which simultaneously considers a large number of possible topologies and returns a multi-topology tradeoff across specs, all in *one* sizing run.

DAC '07, June 4–8, 2007, San Diego, CA, USA.

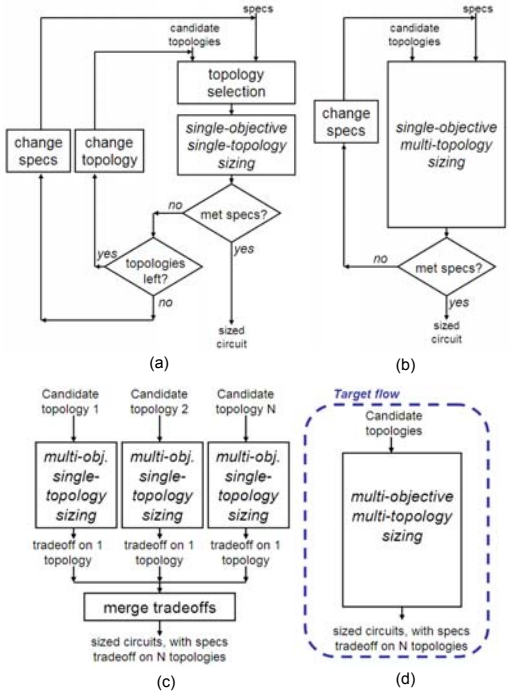

Figure 1: Single- vs. multi-objective and single- vs. multi-topology sizing flows

Cell-level multi-topology sizing approaches also have limits. OASYS [9], BLADES [8], and others [1][2][11][16][21][23] depend on rule-based reasoning or abstract models having transforms to structural descriptions, and therefore have an undesirable amount of up-front setup effort. DARWIN [13] and MINLP [14] only require structural information, but rely on a sneaky definition of a flat combinatorial search space to define possible topologies; they do not show a clear path to generalize

and are restricted to <100 topologies. Approaches like [3][12][20] search across unstructured combinations of transistors, but that flexibility makes them notorious for exploiting missing goals, which compromises designer trust [15].

This paper presents MOJITO, a system for multi -objective and multi–topology sizing. It uses inputs and outputs that are acceptable for industrial single-topology sizing tools (e.g. [19]), with one exception: rather than a single parameterized netlist, it requires a hierarchically organized set of building blocks with respective implementation choices. MOJITO does not need a special decision rule base, or abstract models with mappings to refined structures. This makes it straightforward to switch technologies or add new building blocks. It uses off-the-shelf simulators rather than specially designed performance estimators. Its output is a tradeoff of sized circuits (potentially with different topologies), for final selection by a designer or within a hierarchical methodology like MOBU [7].

This paper is organized as follows. Sections 2 and 3 describe the MOJITO search space and search algorithm, respectively. Section 4 presents experimental results. Section 5 concludes.

2. THE MOJITO SEARCH SPACE

2.1 Search Space Framework

This section describes a topology space (library) that is specified by structural information only, searchable, trustworthy, and flexible. It is defined with hierarchically organized blocks. Like analog HDLs, each block has external ports and parameters for an interface. To fully netlist a given block, the only extra information needed is a value for each parameter of the block. Just three block types are needed:

- **Atomic Blocks**. Have no sub-blocks.
- **Compound Blocks**. Have sub-blocks, which can have internal connections among themselves and to the block's external ports. Sub-block parameters are a function of the block's parameters.
- **Flexible Block**. Have the special topological parameter *chosen_part_index*, which during netlisting, selects one of several candidate sub-blocks and respective connections.

2.2 A Highly Searchable Op Amp Library

We use the framework to define a cell-level library of operational amplifiers. One challenge: if a designer makes a small conceptual change to a circuit that corresponds to a small change in performance, there may be a *drastic* change in the netlist. While this complicates the design of an appropriate search representation, it is needed for changes like folding an input or flipping all NMOS transistors to PMOS. Myriad examples can be found in any analog text [17]. The structural-only op amp approaches [13][14] do cover *some* of these examples, but are designed into a flat space, need special heuristics just to work in their small spaces, and do not readily generalize. An example limitation: [14] had a single parameter to choose whether NMOS vs. PMOS inputs, but did not reconcile that with folded vs. cascode which can flip the load being NMOS vs. PMOS.

To resolve this, we exploit the *chosen_part_index* parameter, which allows highly flexible blocks because it can (a) be a function of one or more higher-level parameters, and (b) choose between sub-blocks that are identical except how those sub-blocks are wired to their parent block. In the MOJITO library, we set the parameter *is_pmos* to become the value of *chosen_part_index* for the MOS4 flexible block having subblock choices of atomic blocks NMOS4 and PMOS4. How *is_pmos* gets set is dependent on the block's context in the library hierarchy and therefore the values of its parent blocks' parameters. Let's see how that affects folding, etc. A block for a 1-stage amplifier is a flexible block with *Vdd* and *Gnd* ports, and has two sub-block choices. Both choices are for the *same* subblock -- the choice specifies only how to *connect* that subblock: one choice ties the subblock's *loadrail* port to *Vdd* and its *opprail* to *Gnd*; for the other choice, vice versa. It also passes on the *chosen_part_index* value as a parameter to lower-level subblocks, mapping it to the appropriate name *loadrail_is_vdd*. Another parameter that starts at the amplifier level and propagates downwards is *input_is_pmos*. To resolve folding, the input cascode block's parameter *is_folded* is calculated by (*input_is_pmos* == *loadrail_is_vdd*). Once that is resolved, the flexible block uses *is_folded* to choose between a folded cascode subblock and a stacked subblock. Each of those cascode subblocks can readily set *is_pmos* parameters for all subblocks.

For space reasons, we do not describe the whole MOJITO amplifier library; but, using the core concepts as a guide, the other building blocks can be defined in a straightforward fashion. For example, a current mirror block is a flexible block that chooses from one of three current mirror choices. The library also includes: 2 level shifter choices (one choice is a wire); 2 choices of how to allocate Vdd/Gnd ports for a 1-stage amplifier and 4 for a 2-stage amplifier; 3 source-degeneration choices; 3 single-ended load choices; and more.

Table 1 shows that MOJITO increases the op amp count by 50x. To calculate that: the count for an atomic block is one; for a flexible block, it's the sum of the counts of each choice block; for a compound block, it's the product of the counts of each of its sub-blocks; but there are subtleties. Subtlety: for a given choice of flexible block, other choice parameters at that level may not matter. Example: if a one-stage amplifier is chosen, do not count choices related to second stage. Subtlety: one higher-level choice might govern >1 lower-level choices, so don't overcount. Example: a two-transistor current mirror should have two choices (nmos vs. pmos), not four (nmos vs. pmos x 2).

Table 1: Size of Op Amp Topology Spaces

Technique	**# topologies**	**Trustworthy?**
Genetic Prog., e.g. [12]	>>billions	NO
DARWIN [13]	24	YES
MINLP [14]	64	YES
MOJITO (this work)	**3528**	**YES**

3. THE MOJITO SEARCH ALGORITHM

MOJITO search is an evolutionary algorithm (EA). To avoid premature convergence, it injects randomness using ALPS [10], which segregates individuals by genetic age as shown in Figure 2.

Selection at a level *l* considers individuals at only level *l* and level *l*-1; therefore younger high-fitness individuals can propagate to higher levels. Genetic age is the number of generations of an individual's oldest genetic material: a random individual is age 0; the age of a child is the maximum of its parents' ages. Putting NSGA-II [5] at each age level makes search multi-objective.

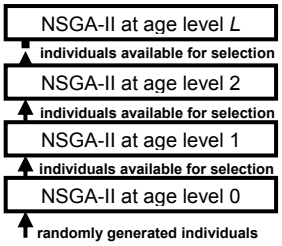

Figure 2: Multi-objective ALPS

MOJITO has a mutation operator and a crossover operator. Mutating continuous-valued parameters follows a Cauchy distribution; integer-valued *chosen_part_index* parameters follow a discrete uniform distribution; other integer and discrete parameters follow discretized Cauchy mutations. Crossover works as follows: given two parent individuals, randomly choose a sub-block in parent A, identify all the parameters associated with that sub-block, and swap those parameters between parent A and parent B. This effectively makes the search a hybrid between tree-based and string-based search. To generate random individuals, MOJITO merely randomly chooses a value for each parameter using a uniform distribution.

4. EXPERIMENTAL RESULTS

This section describes application to two multi-objective multi-op-amp topology sizing problems. The problems were set up as follows. The search space had 50 variables (topology selection variables and sizing variables). Simulator was HSPICE. Technology was 0.18µ CMOS; supply voltage 1.8V; load capacitance 1pF. Search objectives: maximize GBW, minimize power, maximize DC Gain (Experiment Set 2). Constraints: phase margin > 65°, all DOCs, DC Gain > 30dB (Experiment Set 1). EA settings were: 100 individuals per age layer; 10 age layers, maximum age per layer: 9, 19, …, 79, 89, infinity. Each run took approximately 5 days on a single-core 2.0 GHz Linux machine, covering 100,000 search points.

4.1 Experiment Set 1

These runs were to verify the algorithm's ability to traverse the search space and select different topologies. The problem was set up such that the optimization end result was known *a priori*. Three experiments were run, the only difference between them being the common mode voltage ($V_{cmm,in}$) at the input. We know that for $V_{cmm,in}$ = 1.5V, topologies must have an NMOS input

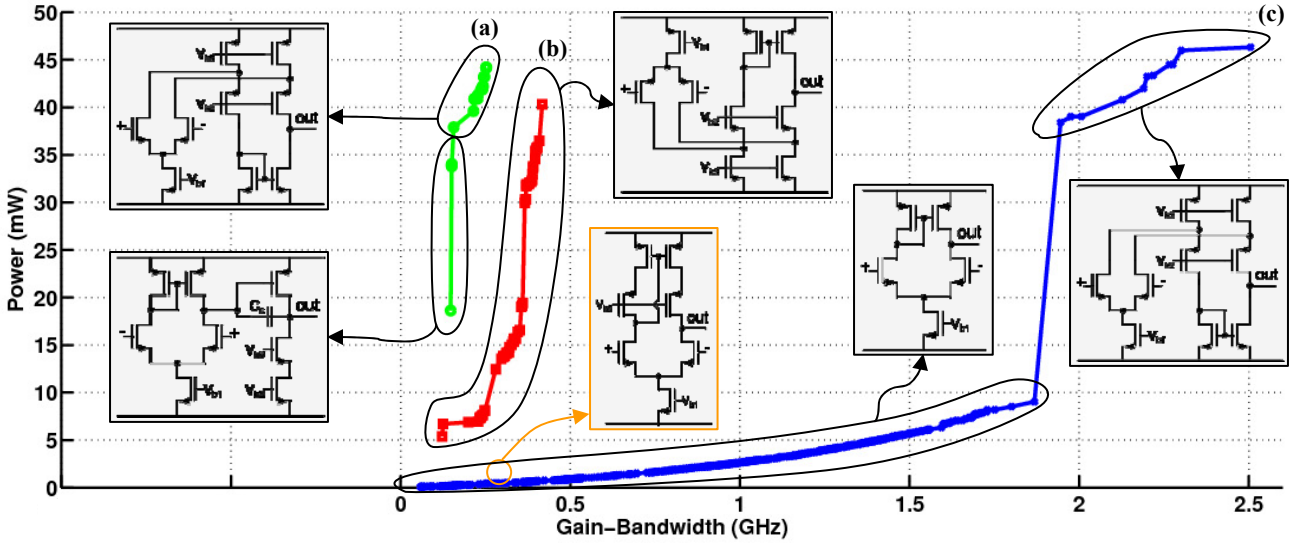

Figure 4: Combined result plot for 3 optimization runs, illustrating some of the selected topologies. Set (a) shows a front for Vin = 1.5, set (b) is for Vin = 0.3V and set (c) is for Vin = 0.9

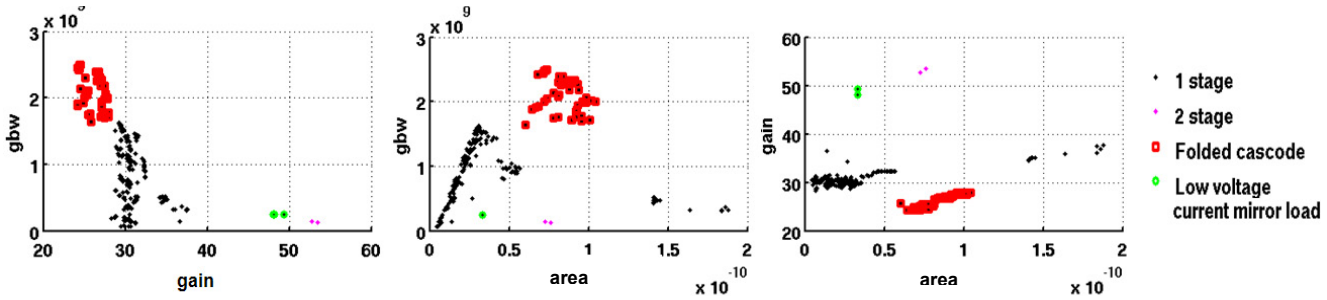

Figure 5: Results for a multi-topology multi-objective sizing run on 3 objectives: gbw, gain, and area.

pair. For $V_{cmm,in}$ = 0.3V, topologies must have PMOS inputs. At $V_{cmm,in}$ = 0.9V, there is no restriction between NMOS and PMOS inputs. Figure 4 illustrates the outcome of the experiments. It contains the combined results of three optimization runs. Result (a) has $V_{cmm,in}$ = 1.5V, and has only topologies with NMOS inputs. It chose to use 1-stage and 2-stage amplifiers, depending on the power-GBW tradeoff. Result (b) has $V_{cmm,in}$ = 0.3V, and MOJITO only returns PMOS input pairs. Note that result (a) is a result before convergence in order to retain the 2-stage amplifier in the result set. Older generations eliminate the 2-stage amplifier in favor of the folded cascode amplifier, as in result (b). For result (c) a $V_{cmm,in}$ = 0.9V has been specified. Though both NMOS and PMOS input pairs might have arisen, the optimization preferred NMOS inputs. The curve clearly shows the switch in topology around GBW=1.9GHz, moving from a folded cascode input to a simple current-mirror amp. Interestingly, the search retained a stacked current-mirror load for about 250MHz GBW. Thus, Experiment 1 validated that MOJITO did find the topologies that we had expected *a priori*.

4.2 Experiment Set 2

The second set of experiments was performed to verify that MOJITO could get interesting groups of topologies in a tradeoff of two or more objectives. The motivation is as follows: whereas a single-objective multi-topology optimization can only return one topology, the more objectives that one has in a multi-topology search, the more opportunity there is for many topologies to be returned, because different topologies naturally lie in different regions of performance space. In this experiment, a single run was performed, having three objectives: area, GBW, and gain. The results are shown in Figure 5. We can see that MOJITO determined (as expected): folded-cascode op amps gave high gain-bandwidth but with high area, 2-stage amps give high gain but at the cost of high area, the low-voltage current mirror load is a 1-stage with high gain, and there are many other 1 stage topologies which give a broad performance tradeoff.

5. CONCLUSION

This paper presented MOJITO, which does multi-topology, multi-objective sizing. It considers thousands of topologies simultaneously, which is possible due to a flexible yet searchable set of trusted building blocks. For industrial relevance, it does not use a rule-base or behavioral abstractions to guide search, and uses SPICE for performance calculation. MOJITO performs search in a hybrid vector/tree space, with a novel multiobjective EA. MOJITO was applied to a space having 3528 different operational amplifier topologies. In one set of experiments, we showed how MOJITO successfully found appropriate topologies trading off power and gain-bandwidth for different common-mode input voltages. In another experiment, we showed how MOJITO evolves different topologies for different regions of the tradeoff among gain, gain-bandwidth, and area.

6. REFERENCES

[1] B.A.A. Antao, A.J. Brodersen, "ARCHGEN: Automated Synthesis of Analog Systems", *IEEE Trans. VLSI* 3(2), June 1995, pp. 231-244

[2] E. Berkcan et al., "Analog Compilation Based on Successive Decompositions," *Proc. DAC*, 1988, pp. 369-375

[3] T.R. Dastidar et al, "A Synthesis System for Analog Circuits Based on Evolutionary Search and Topological Reuse," *IEEE Trans. Ev. Comp.* 9(2), April 2005, pp. 211-224

[4] B. De Smedt and G. Gielen, "WATSON: Design Space Boundary Exploration and Model Generation for Analog and RFIC Design," *IEEE Trans. CAD* 22(2), 2003, pp. 213-224

[5] K. Deb et al., "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II," *IEEE Trans. Ev. Comp.* 6(2), 2002

[6] P. Drennan et al., "Implications of Proximity Effects for Analog Design", *Proc. CICC*, 2006

[7] T. Eeckelaert et al, "An Efficient Methodology for Hierarchical Synthesis of Mixed-Signal Systems with Fully Integrated Building Block Topology Selection," *Proc. DATE*, 2007

[8] F.M. E1-Turky, R.A. Nordin, "BLADES: An Expert System For Analog Circuit Design," *Proc. ISCAS*, 1986, pp.552- 555

[9] R. Harjani et al., "OASYS: A Framework for Analog Circuit Synthesis," *IEEE Trans. CAD* 8(12), pp. 1247-1266, 1992

[10] G.S. Hornby, "ALPS: The Age-Layered Population Structure for Reducing the Problem of Premature Convergence," *Proc. Genetic and Ev. Comp. Conf.* (GECCO), 2006, pp. 815-822

[11] H.Y. Koh et al., "OPASYN: A Compiler for CMOS Operational Amplifiers," *IEEE Trans. CAD* vol. 9, Feb 1990

[12] J.R. Koza et al. Genetic Programming IV. Kluwer, 2003

[13] W. Kruiskamp and D. Leenaerts, "DARWIN: CMOS Opamp Synthesis by Means of a Genetic Algorithm", *DAC*, 1995

[14] P.C. Maulik et al., "Integer Programming Based Topology Selection of Cell Level Analog Circuits", *IEEE Trans. CAD* 14(4), April 1995

[15] T. McConaghy and G. Gielen, "Genetic Programming in Industrial Analog CAD: Applications and Challenges", Genetic Programming Theory and Practice III, Riolo et al, eds., Springer, 2005, ch. 19

[16] Z. Ning et al., "SEAS: A Simulated Evolution Approach for Analog Circuit Synthesis," *Proc. CICC*, 1991

[17] B. Razavi, Design of Analog CMOS Integrated Circuits. McGraw-Hill, 2000

[18] R.A. Rutenbar, G.E. Gielen, and B.A. Antao, eds., Computer-Aided Design of Analog Integrated Circuits and Systems, IEEE Press, Piscataway, 2002

[19] A.H. Shah et al., "High-Performance CMOS-Amplifier Design Uses Front-To-Back Analog Flow," *EDN*, Oct, 2002

[20] 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

[21] K. Swings et al., "HECTOR: a Hierarchical Topology-Construction Program for Analog Circuits Based on a Declarative Approach to Circuit Modeling," *CICC*, 1991

[22] S. Tiwary et al., "Generation of Yield-Aware Pareto Surfaces for Hierarchical Circuit Design Space Exploration," *Proc. DAC*, 2006, pp. 31-56

[23] C. Toumazou et al, "ISAID - A Methodology for Automated Analog IC Design," *Proc. ISCAS*, vol. 1, 1990, pp. 531-555.