2 State-of-the-Art on Analog Design Automation

Abstract. This chapter presents the State-of-the-Art (SOA) in analog circuit design automation. First, the analog design flow is reviewed and the fundamental trends in design automation are discussed. Then, the existing approaches to circuit sizing are presented, outlining in each case their advantages and limitations. Next, a detailed discussion over the existing tools approaches is provided. Finally, conclusions concerning the specification and design of a new analog design automation methodology implementation will be drawn.

2.1 Trends in Design Automation Methodology

A typical design flow for *analog and mixed-signal* IC circuits (AMS) consists of a series of design steps repeated from the system level to the device-level, and bottom-up for layout generation and verification. The steps between any two of these hierarchical levels are: topology selection, circuit sizing, design verification and layout generation task, illustrated in Fig. 2.1.

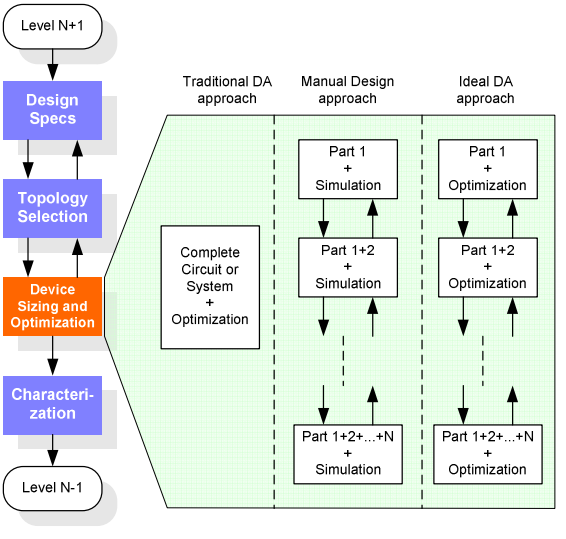

Fig. 2.1 Hierarchical level and design tasks of design flow architectures [1]

M.F.M. Barros et al.: Analog Circuits and Systems Optimization, SCI 294, pp. 19–47.
springerlink.com

In order to handle the increasing complexity of analog and mixed-signal IC design, a clear definition of a hierarchical design flow is essential. Despite the advances made during the last decades, the *design automation* (DA) tools in analog domain cannot support the complete design process, since they either concentrate on specific parts of the design flow or require the intervention of an expert designer. Moreover, they mainly address circuit level design as a whole (traditional design approach), which makes it difficult to apply to highly complex circuits and systems. Therefore, as the SoC complexity increases, the design automation tools must incorporate an hierarchical design decomposition feature in order to apply the well-known divide-to-conquer strategy already applied by most analog designers in a manual design approach.

Trends in this area have been running towards a class of design automation methodology under three aspects, improving:

- Flexibility, allowing the designer to have a higher interaction during the synthesis process and providing a more general approach to deal with multiple architectures or circuit types.
- Modularity, allowing the use of different tools and techniques to address different design tasks, such as topology selection, circuit sizing and layout.
- Hierarchy, allowing the handling of complex system designs and implementing strategies involving several abstraction levels.

2.1.1 Automated Topology Selection

The selection of an adequate architecture is fundamental to achieve a high performance design [2]. The topology selection task receives the performance specifications, for a particular class of circuits or systems, and delivers the most promising topology, traditionally from a predefined library. In IDAC [3] the decision is taken directly by the designer. *Heuristic rules* [4] have been used in the first attempts by TAGUS [5]-[6], OASYS [7], BLADES [8], and OPASYN [9] to automate the topology selection task. The tool FASY [10] uses *fuzzy–logic* based reasoning to select one topology among a fixed set of alternatives. The decision rules are introduced by an expert designer or automatically generated by means of a learning process. Another method comprises computing the *feasible performance space* for each topology within the library and, then compare with the desired performance specs, by AMGIE [2] and [11]. A different method consists of combining the topology selection with the device sizing task and employing an optimization based approach by DARWIN [12] using *genetic algorithms*. This design mechanism illustrated in Fig. 2.2, uses a template rather than an architecture library. This template specifies the topology in terms of blocks, each one with possible different alternatives. In short, this last method is more reliable since it treats the problem in a more deterministic way and at the same time decreases the setup time, as it does not need to rearrange a new set of rules each time a new topology is added to the library; the computation time, however, is worse than in all methods described above.

A new step towards the increase of the automation level is given by a new set of tools where topology selection is performed at a higher abstraction level. Instead of selecting the architecture from a library, a high level functionality of the

system is defined now by a *hardware description language*. Then, an automatic translation is carried out, mapping the functional description into an internal representation and then into a specific topology. The mapping step is implemented after or during the device sizing process. This class of tools usually differs from the type of internal representation used. In the case of [13] the internal representation is a data flow graph, whereas in TAGUS [5]-[6], [14] and Konczykowska [15] it is a *symbolic signal flow graph* and in ARCHGEN [16] a *state-space description* is used. Then, a mapping operation is performed, resulting in a connection of lower-level building blocks whose parameters are optimized, obeying to some design constraints. The operation flow is executed in a top-down basis.

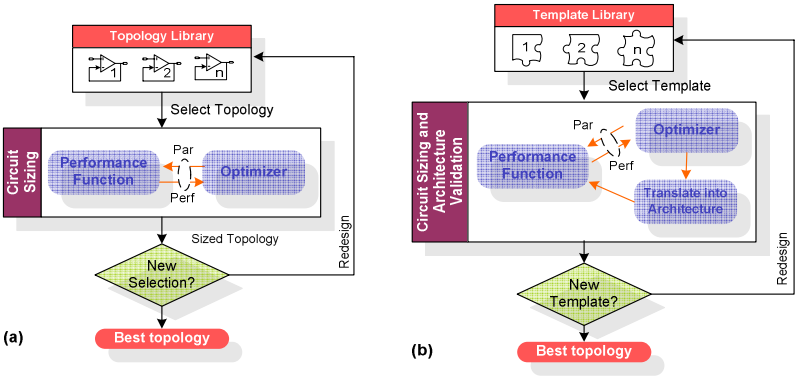

Fig. 2.2 Topology selection mechanism before (a) and during (b) device sizing

Finally, a design methodology able to create new topologies explores the immense potential from low abstraction level. Small elementary blocks are connected bottom-up to each other to form a new topology. The general description of this design methodology illustrated in Fig. 2.3 begins by selecting an initial topology, having in mind the desired specifications. As the design process takes place, an optimizer selects a transformation, adding or deleting a basic entity and/or attributing a value to a parameter. Various fundamental entities can be applied, such as, single transistors, elementary building blocks or node connections. As soon as the architecture is generated, the performance function is evaluated, providing some hints to the optimizer who makes a new selection of transformation. Essentially two exploration methods can be applied in topology generation for analog design. The knowledge-based exploration is based on a systematic or a random strategy where the circuit elements can be added, replaced or removed by an experienced designer with the help of standard CAD tools, like SPICE, and a circuit schematic editor. This method mimics the daily basis design approach supported mainly by simulation tools, and, therefore, suffers from the same drawbacks, i.e., as the number of entities in the system rise, the computational time increases accordingly. The computation time at the circuit description level can become intolerable if no efficient guidance is provided during the exploration step.

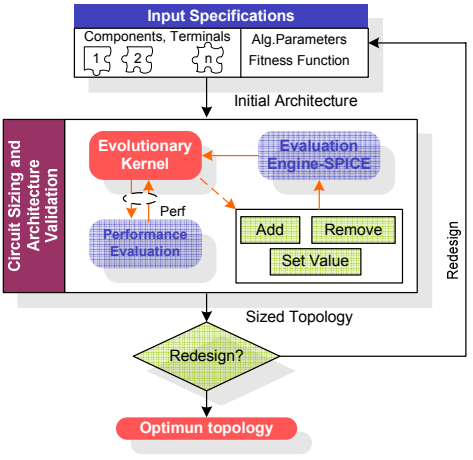

Fig. 2.3 A general description of the topology selection bottom-up methodology

New automation tools appeared, based on stochastic evolutionary computation methods, which apply an appropriate representation for standard circuit-level descriptions and recombination operations. Population-based optimizers provide multiple dimensioned architectures which are then simulated by SPICE-like simulators. In [17] the optimizer is based on a genetic algorithm and in [18]-[19] uses genetic programming techniques. Table 2.1 summarizes the general characteristics of automated topology selection and generation mechanisms.

Table 2.1 General characteristics of automated topology selection and generation

	Topology Selection		**Topology Generation**	
	Heuristic Rules	Feasible Region	Top-down*	Bottom-up
Tools	TAGUS [5]-[6], OASYS [7], BLADES [8], OPASYN [9] and FASY [10]	AMGIE [11], and Gielen [2], DARWIN [12]	Graeb [13], TAGUS[5]-[6], [14], Konc-zykowska [15]	Colombano[17], Koza[18], Toumazou [19]
Drawbacks	(-) Large set up time in order to update the selection rules to a new topology. (-) Qualitative approach and sometimes extremely difficult to codify heuristic rules.	(-) Time consumption	(-) Less generalized.	(-) Large time consumption. (-) No technological param. (-) No corner validation. (-) Not in a mature state.
Advant.	(+) Reduced execution time.	(+) Quantitative and general ap-proach.	(+) Reduced exe-cution time and well defined process	(+) Extremely promising. (+) Generic Approach. (+) No expert knowledge.

* Properties depend on methodology. This column considers knowledge-based approach.

2.1.2 Automated Circuit Sizing/Optimization

The sizing stage receives a topology description, a set of performance specs and a technology reference and, based on these inputs, produces a sizing solution for each block or component depending on the abstraction level. Several solutions were proposed derived from either knowledge-based methods, using some kind of knowledge and heuristics, or optimization-based approaches for both topology selection and specification translation or circuit sizing [1],[20]. The knowledge-based approach requires the expert knowledge of a designer to produce a set of rules and equations for every new circuit topology or technology. Another alternative is obtained considering the circuit sizing as an optimization problem. In these approaches the design problem is first mapped or modeled into an optimization problem and then solved by an appropriate optimization method, as illustrated in Fig. 2.4.

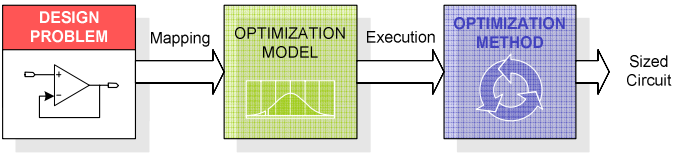

Fig. 2.4 Steps in optimization of circuit design

In this approach, there is a strong correlation between the modeling of a design problem and the way the modeled problem is solved. Since these steps are not independent and have influence on each other, the optimization method will be decided by the chosen model of the problem. For example, if the design problem is formulated in a set of *posynomial* equations the optimization method candidate could be the *geometric programming* (GP) algorithm or other computation algorithm able to process the convex optimization problem defined by posynomial equations [21]. If the design problem is formulated by SPICE models, a simulated annealing or a stochastic pattern algorithm could be used instead. Section 2.2 will explore the main optimization methods and alternative models in the area of analog IC design problems.

2.1.3 Automated Layout Generation

The earliest approaches to automate the layout generation followed a *procedural module generation* [22]-[23] with the codification of the entire circuit layout and its generation during the run time for the parameters attained during the sizing task. The procedural generators define a parametric representation of the geometric layout developed by the designer, accomplished either through a procedural

language or a graphical user interface. The disadvantages of this approach are the lack of flexibility and generality and the high cost of the generation task.

Next, a *template-based approach* was developed [24] allowing the employment of geometric templates, which define the relative position and interconnection of devices. The templates are used to incorporate the designer knowledge into the optimization task. In spite of the low level of reusability achieved by procedural generators, the efficiency of this approach can be improved when user-defined templates are designed to be independent of both technology and specifications [4]. This approach is also suited when modifications in circuit parameters end in small adjustments to the global circuit layout structure, like technology migrations.

Later on the *optimization-based approaches* emerged, using optimization techniques to determine and predefine the layout solution. Small database of procedural cell generators, ANAGRAM [25]-[26], LAYLA [27] and ALDAC [28] synthesize an optimized layout configuration, searching the solution space formed by each cell layout positioning. The ALG [29] approach allows the generation of "optimal layout" of a circuit either automatically or by designer directives. On one hand these approaches require more computation time, but, on the other hand, they are more flexible and general, which compensates largely the weakness mentioned above. Significant technological solutions have resulted from this method [30]-[33], ranging from rule driven to performance driven layout generation tasks [27], reaching a more mature state when compared to what happens in the design automation tasks concerning circuit sizing [1]. The most frequent used optimization techniques in analog IC layout generation tools are simulated-annealing (ILAC [34], KOAN [25]-[26] and LAYLA [27]) and genetic algorithms (LAYGEN [35]-[36]). Simulated-annealing based approaches attained better results but lately the evolutionary approach has become a common option in many situations, like the hybrid solution defined by the genetic approach to simulated-annealing GASA [37] or the combined GA and Tabu Search (TS) used in [38] to develop a polycell placement algorithm for analog LSI chips. As both KOAN and LAYLA employ very simple cells on the database, some highly efficient structures, such as stacked or interdigited transistors, cannot be generated. Recent approaches, however, are tending to hybrid solutions employing optimization on blocks derived from knowledge-based systems. In the case of ALADIN [37],[39], the database usually relies on a hierarchical model where a cell is built using already defined cells. The use of compound cells reduces the search space because the number of cells handled during placement is lower and consequently reduces the computation times. Another knowledge-based approach with optimization is given by IPRAIL [40] and LAYGEN [35]-[36] in which the information presented in the template is defined manually or automatically and used to guide the layout generator during the synthesis procedure. The constraints defined in the template reduce the solution space, and allow the designer a higher control of the layout generation unlike the general optimization approaches [35]. Table 2.2 resumes the general characteristics of layout tools.

Table 2.2 Overview of layout tools

Tool	Year	Description	Techniques	Obs.
KOAN/ANAGRAM [25],[26]	1991	Macro-cell Place and Route; uses pre-defined small module generators data-base; synthesizes an optimized layout configuration from a given Spice netlist with symmetry, matching and tech. specs.	Optimization based with Simu lated Annealing.	The chosen library constitutes a limit of this method since an enormous number of pre-designed layout blocks is required
Layla [27]	1995	It takes into account symmetry constraints, performance degradation due to interconnect parasitics and device mismatches and combines this with geometrical optimization techniques (devices merges, abutment, etc.)	Optimization based with Simulated Annealing.	A performance-driven methodology where all performance constraints are satisfied. Optimize the layout quantifying the performance degradation.
A SKILLTM-based Library for Retargetable Embedded Analog Cores [32]-[33]	2001	Automatic generation and reusability of physical layouts of analog and nixed-signal blocks based on high-functionality pCells that are fully independent of technologies.	Knowledge-based	Parameterized cells (pCells) are organized hierarchically.
ALDAC [28]	2002	This tool providing means to generate multiple versions of full-stacked layout modules for the same circuit. The differences come from different MOS transistor splitting and grouping into stacks that can be performed either fully-automatically or user-controlled	Simulated Annealing	This approach minimizes parasitic diffusion capacitances of the circuit and permits economical post-layout simulation of multiple layouts for performance-driven
IPRAIL [40]	2004	Retargeting is achieved using an automatically extracted template and using a circuit optimizer to size the cells. It uses either a rule or a performance driven approach. It uses optimization based with knowledge-based.	Linear Programming and graph short path on the relational template extracted from the source layout	(+) General approach. (-) Larger run-time required.
ALADIN [25],[32]	2004	The layout generation is based on relatively complex sub-circuits. Designers can construct layouts of parameterizable modules in a technological and application independent way. The placement and routing of modules are performed automatically under the constraints defined by designer	Three phase Place e Route: 1 – GASA e half-perimeter routing 2 – VFSRA e global routing (fine tuning) 3 – Detailed Routing	Design platform for analog circuits, based on a user managed device generators library.

Table 2.2 (*continued*)

Tool	Year	Description	Techniques	Obs.
LAYGEN [35],[36]	2007	Expert knowledge is used to guide an evolutionary algorithm during the automatic generation of the layout The designer provides a high level layout description where position and interconnections are predefined. This template contains placement and routing constrains and is independent from technology. It deals with hierarchically templates for more complex circuits.	Knowledge-based with Evolutionary Computation techniques. Uses a geometric template.	(+) Speeds up retargeting operations or technology migration (-) Works better when changes in circuit parameters result in small adjustments. for the target technology
ALG [29]	2007	ALG is composed by three functional blocks: module generator, placer and router offering performance oriented layout generation in some of these blocks.	Cost function is a weighted sum function parasitics level, aspect ratio and mismatch, etc.	The user may choose the level of automation between full automation and user control.

2.2 Automated Circuit Synthesis Approaches

The computer-aided design methodology for AMS circuits foresees in a short-run the use of design automation tools to accomplish several tasks of the design methodology [41]. This trend began in 80's when the first automation tools applied to different tasks of analog design appeared like LAYLA [42] , IDAC [3],[22], DELIGHT.SPICE [43], BLADES [8] and OASYS [7]. The following sections review some of the most significant approaches for analog IC design including the knowledge-based, optimization-based approaches as well as the first commercial tools.

2.2.1 Knowledge-Based Approach

The knowledge-based approach presented, for instance, in programs like BLADES [8], IDAC [3], OASYS [7] and MDAC/ALSC [44]-[45], was the first to appear and is characterized by including a complete design plan describing how the circuit components must be sized to reach the solution for the design problem, even though, there is no guarantee of finding the optimum solution [2]. For example, the IDAC tool [3] takes advantage of the designer experience to manually derive or rearrange design plans to carry out the circuit sizing. OASYS [7] was built

over a library of design plans defined for each elementary building block allowing the hierarchical representation of topologies, defined as the interconnection of several elementary building blocks. This system also implements a back tracking mechanism in order to recover from a malfunction implementation. The Fig. 2.5 illustrates the general design flow of knowledge-based approach.

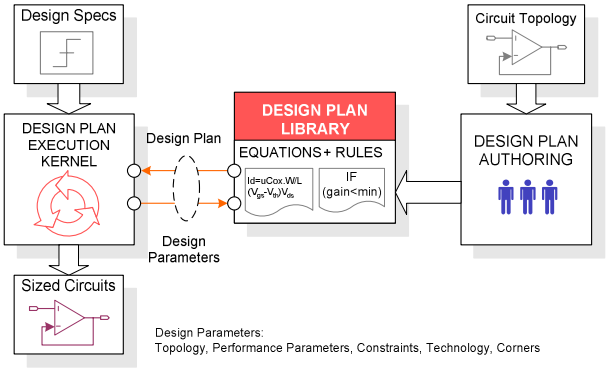

Fig. 2.5 Knowledge-based approach

In these methods, the main purpose is to encapsulate the designer's knowledge, building a pre-design plan with design equations and a design strategy that produce the component sizes in order to meet the performance requirements. This approach presents as major drawbacks the large overhead required to define a new design plan, the reformulation of the entire design plan when expanding the system to new topologies, and, finally, the migration to other technologies. Not only, it is a very time-consuming process to encode design knowledge for a given set of specifications, but design knowledge also has a limited lifetime. The rapid progress in process technologies made the acquired knowledge quickly out-of-date. Therefore, the application of these tools in industrial environments has been limited. However, after defining the design plan, the execution speed associated to the sizing procedure is extremely fast and the solution quality only depends on the models precision [1]. Naturally, this approach finds its applications restricted to small circuits or to more complex circuits but using simplified equations with the goal of achieving the first cut design.

2.2.2 Optimization-Based Approach

The optimization-based approach uses an optimization engine instead of a design plan to perform the design task. The optimization process is an iterative procedure where design variables are updated at each iteration until they achieve an equilibrium point.

The optimization-based approaches illustrated in Fig. 2.6, consist of an iterative loop, including an optimization engine or kernel together with an evaluation engine.

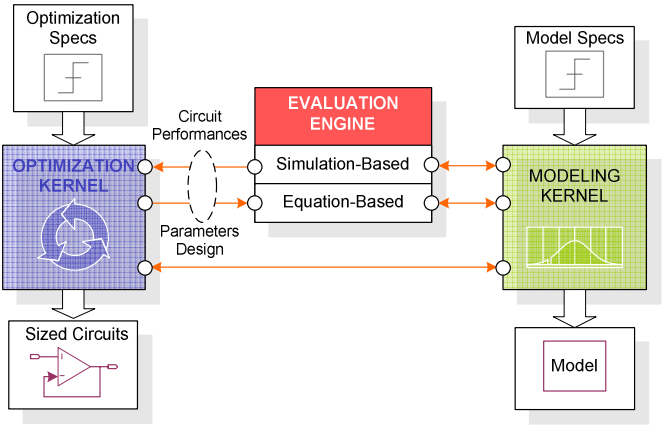

Fig. 2.6 Optimization-based approach

The optimization algorithm searches through the design space for values for each circuit component, whereas the performance evaluation tool verifies if the erformance constraints are met. If the system requirements are satisfied, then a solution is found and the component sizes are associated to the selected topology. The optimization engine should apply the appropriate techniques to efficiently guide the search mechanism in order to minimize the number of iterations required for the optimization process.

Different approaches can be described depending on the type of performance evaluation and the optimization technique employed. Concerning performance, the evaluation engine is typically implemented using an equation-based optimization, a simulation-based optimization or modeling-based optimization approach.

2.2.2.1 Equation-Based Methods

The equation-based methods use analytic design equations to evaluate the circuit performance. These equations can be derived manually or automatically by symbolic analysis tools. Then, the problem can be formulated as an optimization problem and normally solved using a numerical algorithm. Some of the most relevant approaches are OPASYN [9], STAIC [46], MAULIK [47], ASTRX/OBLX [48], AMGIE [11], GPCAD [49][50], SD-OPT [51]. This approach presents the advantage of allowing a performance evaluation speed-up (short evaluation time). The main drawback is that analytical models have to be used to derive the design

equations for each new topology and, despite recent advances in symbolic circuit analysis [52]-[53], not all design characteristics can be easily captured by analytic equations. The approximations introduced in the analytic equations yields low accuracy designs especially in complex circuit's designs.

A promising methodology that has received much attention is related to circuit problems formulated in *posynomial form* (expression 2.1) and seen in tools like GPCAD and [21], [54]. This methodology solves the convex formulated problem by *geometric programming* techniques in a very short time. These techniques take advantage of the development of extremely powerful interior-point methods for general convex optimization problems [21],[50]. Besides the extreme efficiency of these methods they have another great advantage, as the global solution is always found, regardless of the starting point. However, a significant drawback still exists due to the difficulty to reformulate high-accuracy device models as posynomials equations, "performance specifications, and objectives that can be handled are far more restricted than any of the methods described above" [50]. Despite the progress presented in [54] the lack of an automated scheme to generate these equations limit the usage of this tool to a few, predefined, circuit structures.

$$f(x_1,\ldots,x_n) = \sum_{k=1}^{t} c_k x_1^{\alpha_{1k}} x_2^{\alpha_{2k}} \cdots x_n^{\alpha_{nk}} \qquad (2.1)$$
$$where, c_j \geq 0 \text{ and } \alpha_{ij} \in \Re$$

2.2.2.2 Simulation-Based Methods

The simulation-based approaches such as DELIGHT.SPICE [43], FRIDGE [55], FASY [10], ANACONDA [56], MAELSTROM [57] and DARWIN [12] consist of using some form of simulation to evaluate the circuit's performance. In general, these types of tools for analog circuits design employ a circuit analysis tool in the inner loop of the optimization cycle to determine the circuit's performance. This is pointed out as a very flexible solution when compared with other methodologies (equation-based, knowledge-based) once it accommodates to any type of circuit topology and yields superior accuracy (depends on simulator models). Presently, the use of SPICE-like simulators are almost generalized and essential to support the optimization engine with all the feedback related to an accurate circuit evaluation, involving different performance characteristics, technological parameters and worst case corners analysis. Moreover, within this approach the same circuit can be optimized several times for different specs as long as the goal function is adapted, therefore, with this approach virtually all types of circuits can be sized and optimized with low setup time.

Despite these advantages, automated circuit sizing is not as commonly used as for example, circuit simulation, since it is computationally too expensive to evaluate electrical simulations. However, with the exponential increase of computer

power and efficient use of optimization algorithms it has become increasingly favorable. Nevertheless, a key difficulty is that the analog design problem, with all the involved design knowledge and heuristics, has to be formulated as an optimization problem, which often presents a high threshold for using a circuit-sizing tool.

2.2.2.3 Learning-Based Methods

A step forward to enhance the efficiency of optimization based methods corresponds to the introduction of modeling techniques [58] based in *learning strategies*, which are clearly more time-efficient, during the optimization cycle. In this class of methods, the behavior of the circuit to be optimized is modeled by a learning mechanism based on the distribution of variation parameters, thus allowing a quick evaluation of the performance for a specific set of design parameters. Nevertheless, these methods require a set of training samples in order to build the model in the target region. Generally, a high accuracy evaluation engine is used, such as a circuit simulator to evaluate the performance of the training sample. The amount of the training data will influence the accuracy of the performance predictions made by the learning machine. However, an increase on the training data means that the evaluation of the performance will take more time. Like in equation-based methods, there will always be a trade-off between accuracy and efficiency.

Some of the most significant behavioral-based methodologies are described by Rutenbar [58], Alpaydin [59], Vincentelli [60] and Vemuri [61]. In the basis of Alpaydin tool is a neural-fuzzy model approach combined with an evolutionary optimization strategy and simulated annealing where some of the AC performance metrics are computed using an equation-based approach.

In [60] Sangiovanni-Vincentelli and al. use a learning tool based in support vectors machines (SVM) to represent the performance space of analog circuits. Based on the knowledge acquired from a training set, the performance space is modeled as mathematical relations translating the analog functionality. In this work two classes of SVM are confronted in an optimization-less strategy where additionally two improvements of the basic one-class SVM performances, conformal mapping and active learning, are proposed by enhancing the resolution in the support region boundaries. SVMs are trained with simulation data, and false positives are controlled based on a randomized testing procedure.

The Vemuri approach [61] presents a performance macro-model for use in the synthesis of analog circuits based in a neural network approach. On the basis of this mathematical model is a neural network model approach that, once constructed, may be used as substitute for full SPICE simulation, in order to obtain an efficient computation of performance parameter estimates. The training and validation data set is constructed with discrete points sampling over the design space. The work explores several sampling methodologies to adaptively improve model

quality and applies a sizing rules methodology in order to reduce the design space and ensure the correct operation of analog circuits.

2.2.3 Commercial Tools

Besides the efforts introduced above some commercial EDA tools for circuit sizing have emerged in the past few years, such as the ADA's [63] Genius product line now integrated in Synopsis, Barcelona Design [49] which employ convex optimization techniques and recently the NeoCircuit from Neolinear Inc. [62], which implements a simulation-based approach.

The ADA (Analog Design Automation) Genius line of optimization tools, including Creative Genius, which automates device sizing and biasing to optimize circuit performance, and IP Explorer, which graphically provides N-dimensional circuit performance tradeoffs, were recently acquired [63] and integrated within the analog design environment from Synopsys [62], Mentor Graphics and other EDA vendors. The Genius tool builds its database of circuit from a transistor-level netlist, testbenches, objectives, process and environmental variations and variables. This system is comparable to NeoCircuit once it implements a simulation-based approach and interfaces with several industrial circuit simulators using parallel computation architecture.

The now extinct Barcelona Design was founded in 1999 by Stanford University researchers that apply advanced optimization techniques based in convex optimization to develop optimization solutions for a broad spectrum of circuit design problems including analog, RF and digital circuits. The final product introduces the synthesizable IP (intellectual property) block, which contains the required design equations written as posynomial expressions. The particularity of these products is that in opposition to standard IP blocks, which meet the given specifications, these blocks, may be synthesized to meet a range of different specifications. This implementation was reported to be able to increase design speed by 100 times and reduce total design costs by up to 50%.

The Neolinear package, now acquired by CADENCE [65], is composed by the NeoCircuit package, a simulation-based analog circuit sizing engine and the NeoCell module to automate the layout generation process. Both design packages together with the optimization engine based on a "genetic annealing" scheme creates a complete analog design flow. The integration of Neolinear's products in the Virtuoso design environment takes advantages of Cadence's multi-mode simulation and extensive layout design capabilities.

2.3 Design Automation Tools: Comparative Analysis

The existing design automation approaches are here compared, taking into account some qualitative and quantitative measures described in subsection 2.3.1. Table 2.3 presents the analog sizing tools used in this study and the conclusions are presented in the following subsections.

Table 2.3 Overview of analog sizing tools (1/4)

Features / Tools	Date	Evaluation Class	Algorithm Techniques	Equation / Design Plan	Implemen. Language	Robust Design	Circuit Complexity	Computation Time Effort	Setup Time	Interactive Design	Bookkeeping	Encapsulation	Advantages and Particular Properties
IDAC [3]	1987	Knowledge Based (KB) Simplified Equations	Design plan + Post Optimization	Symbolic analyzer + manually	☒	☒	5-20 devices	A few seconds (VAX780)	Long time to add a new circuit (months). Employs worst-case design.	☒	☒	☒	Able to size a library of analog schematics as a function of technology and building-block specs.
DELIGHT SPICE [43]	1988	Circuit Simulator with SPICE	Feasible Directions	☒	☒	✓	20 Par. 28 devices	18h (Masscomp MC 500)	Moderate setup	☒	☒	☒	Uses a large library of opt. algorithms. Problems with simulator convergence.
OASYS [7]	1989	KB Simplified Equations	Design Plan + Backtracking	Manually and specific to each class	☒	✓	19 Par. 17 devices OpAmp	3 sec. (VAXstation II/ GPX)	Long setup time (approx. 6 months), including circuit analysis	☒	☒	☒	Apply task decomposition, simplifies design reuse. Find out feasible and infeasible surfaces
BLADES [8]	1989	KB Lookup Tables and Rules	Artificial Intelligence. Uses divide and conquer strategy.	Manually	☒	☒	34 devices OpAmp	A few seconds to design and 20 min to complete simulation	Long time to create and maintain look-up tables.	☒	☒	☒	Uses a divide and conquer strategy to partition blocks written by “if-then clauses”. Ability to perform circuit design, generate test files and invoke circuit simulator automatically

Table 2.3 Overview of analog sizing tools (*continued* 2/4)

Features / Tools	Date	Evaluation Class	Algorithm Techniques	Equation / Design Plan	Implemen. Language	Robust Design	Circuit Complexity	Computation Time Effort	Setup Time	Interactive Design	Bookkeeping	Encapsulation	Advantages and Particular Properties
OPASYN [9]	1990	Simplified Equations	Grid + Steepest Descent	Manually. Simple analytical model	C and Lisp	✓	7 Par., 60 devices OpAmp	5 min. (VAX 8800)	2 weeks for known circuits and a few days for simpler circuits.	☐	☐	☐	Interface with Berkeley CAD environment. Includes layout simul.
MAULIK [47]	1991	Simplified Equations + BSIM	Branch and Bound	Manually	☐	☐	39 Pararm., 19 devices	1 min. DEC 3100	6 months, including circuit analysis	☐	☐	☐	Topology selection + device sizing
STAIC [46]	1992	Simplified Equations	Two Step Optimization	Manually	C++	☐	22 device OpAmp	3 min. MIPS 2000	Entering new descriptions take long time.	☐	☐	☐	Presents an analog description language with 4 levels
FRIDGE [55]	1994	Circuit Simulator	SA with SPICE +local Method	☐	☐	☒	10 Pararam OpAmp	45 min. Limited to a few thousand evaluations.	Effort to add a circuit in about one hour.	☐	☐	☐	Has only been demonstrated for problems of a small number of opt. var.
SD-OPT [51]	1995	Equations + Behavioral Simulations	SA	Manually	☐	☒	4th-order sigma-delta modulator	1.5 weeks	Exhaustive analysis Required. High cost to implement new structure	☒	☒	☒	Design for switched-capacitor delta-sigma modulators
FASY [10]	1995	Simulation Based	Fuzzy + SA + Gradient+ NN	SPICE as the evaluation engine.	☐	☒	9 devices OpAmp	6 hours (96 MIPs workstation)	☐	☒	☒	☐	Hybrid SA for coarse and gradient for fine simulations. Fuzzy logic for topology select.

Legend: ☐=There was no mention of it ☒=Means "not implemented in tool" ✓= Means "implemented in tool"

Table 2.3 Overview of analog sizing tools (*continued* 3/4)

Features / Tools	Date	Evaluation Class	Algorithm Techniques	Equation / Design Plan	Implemen. Language	Robust Design	Circuit Complexity	Computation Time Effort	Setup Time	Interactive Design	Bookkeeping	Encapsulation	Advantages and Particular Properties
ISAID [66]-[68]	1995	Simplified Equations, Qualitative Reasoning	Qualitative Reasoning + Post Optimization	Manually	☑	☑	8 Par., 13 devices OpAmp	☑	☑	☑	☑	☑	Replace exact performance relations with quantitative relations.
ASTRX-OBLX [48]	1996	AWE + Equations	SA	Automated. Setup highly automated	C	☒	33 devices	11.8h (IBM RS/6000-550)	A few days to add new circuits.	☒	☒	✓	Simulation + equation Based on waveform evaluation.
GPCAD [48],[55]	1998	Simplified Equations + Geometric Progr. (GP)	Simple Primal Barrier Method	Equations for Opamps are generated manually.	MATLAB	☒	10 devices OpAmp	Very fast run time, in the order of a few seconds	Doesn't specify setup time. Doesn't include automatic generations of equations.	☒	☒	☒	Achieves short execution times by the use of Geometric programming.
MEALSTROM [57]	1999	Circuit Simulator	GA+SA	☑	C++	☒	27 Par., 32 devices	3.6 h (15) SunSparc - 1)	Uses Cadence GUI. Long times are associated with simulations run.	✓	✓	✓	Uses simulator in distributed environment.
ANACONDA [56]	2000	Circuit Simulator	Stochastic Pattern Search	☑	C++	☒	20 Par., 37 devices	10 h (24 Sun Ultra 10)	Encapsulate commercial simulators (TISpice).	✓	✓	✓	Add constraints on devices to ensure a safety margin from manuf. or envir. variations. Parallelism.

Table 2.3 Overview of analog sizing tools (*continued* 4/4)

Features / Tools	Date	Evaluation Class	Algorithm Techniques	Equation / Design Plan	Implemen. Language	Robust Design	Circuit Complexity	Computation Time Effort	Setup Time	Interactive Design	Bookkeeping	Encapsulation	Advantages and Particular Properties
AMGIE [11]	2001	Simplified Equations	SA + several Local Methods	Symbolic analyzer + manual	☐	☒	14 Par 9 devices	5 min	8 hours	☒	☒	☒	Complete design flow. Device sizing is a compilation of tools
ALPAYDIN [59]	2003	Fuzzy Neural Network	Evolutionary Strategies + SA	Training points + circuit simulator	☐	✓	31 devices OpAmp	45 min. (124 min. with mismatch model)	Model accuracy depends on training points. Cannot handle mismatch.	☒	☒	☒	The N-F model uses a set of training data from SPICE.
VINCENTELLI [60]	2003	SVM	Use LIBSVM Package.	Performance model of analog circuits	☐	☐	4 opt. variables	10's min 50.000 samples	☐	☐	☐	☐	Improve accuracy of estimator using an Active Learning strategy
VEMURI [61]	2004	NN	MIT GAlib	performance parameter macro-models	MATLAB	☐	5-33 devices. Up 10 opt. var.	51.9 µs of execution time	It takes 1h47min in a SunBlade 100 to generate the model training samples(3125).	☐	☐	☐	Explores sampling methodologies to improve model quality.
GENOM [69]-[70]	2006	Circuit Simulator +SVM	GA / SVM	SPICE/ HSPICE engines + "feasibility models"	C	✓	31 Par. 21 devices 41 const.	20 min. with robust design (Intel Core2 CPU @ 2.40GHz PC)	Encapsulate in-house environment (AIDA). Can use dynamic model generation.	✓	✓	✓	Apply learning strategies. Distributed processing and robust design.

Legend: ☐=There was no mention of it ☒=Means "not implemented in tool" ✓= Means "implemented in tool"

2.3.1 Specific Characteristics

The tools described in Table 2.3 can be evaluated by several metrics that measure the final solution quality. The first column "date" is performance independent. There is no correlation between the availability of the design tool and its efficiency or accuracy. On the contrary, the next three characteristics columns "Evaluation Class", "Algorithm Techniques" and "Equation/Design Plan", which are often used for classification purposes, will have an important influence in the performance and accuracy as will be shown later in this chapter.

Particularly, the following metrics were considered in order to compare the characteristics of the presented applications.

(a) *Robust Design*: As far as sizing is concerned, robust design has to do with the accuracy and robustness of the solution. Accuracy is a measure of the quality that shows the difference between the synthesis tool's performance prediction mechanisms and the real performance of the obtained solutions, possibly including the layout-induced degradation. Robustness can be described as the capacity of the sizing tool to build and test circuits tolerant to manufacturing faults and operating point variations.

(b) *Automation Level*: It can be described as the ratio of time needed to accomplish the task of designing a circuit manually to the time spent on designing the same circuit with the help of a synthesis tool. In this metric two aspects must be considered:

- Run time response: The period of time taken by the optimization tool to give the first solution to the problem.
- Setup time: The setup time is a measure of the time spent by the designer to adequate the problem to the synthesis tool. This time is often longer than the execution of the synthesis tool. This feature is particularly important because it is strongly correlated with the success and acceptability of the tool. What is the advantage of a design tool which has the remarkable prodigy to output some results in seconds, if it is necessary two months to setup the complete algorithm of a hypothetic circuit when it is known it could be designed by hand in one month? Excluding a reused-based scenario, the answer is obvious "None".

(c) *Scope of the tool*: It can be described as a group of analog design problems, which can be solved by this tool. This is an important feature for analog design, because these problems usually require several types of optimization techniques. An analog synthesis tool which aims at solving a wide range of design problems will be successful in the long run, whereas tools planned to solve a narrow range of problems will soon be out of date. Although, it is not shown in Table 2.3, it will be used later for comparative analysis.

(d) *Design facilities*: It can be described as the set of additional features that can enrich a synthesis tool.

- **Multi-objective Optimization.** The DA tool presents the final solution in terms of a set of designs representing complementary tradeoffs of specific

objectives (for example, area versus consumption) instead of single design response.

— **Interactive Design.** The tool optionally produces intermediate performance reports (in the form of text or graphics) throughout the design execution time to inform the IC designer on the optimization progress. At the same time, the IC designer optionally has the possibility to interact with the tool in real time manner to tune up some parameters, e.g., the dimension of a transistor or the redefinition of some design bias.

— **Bookkeeping Facilities.** The tool should have additional capacities to help with the introduction and management of all the necessary data including the management of different technological files, different classes of circuits (e.g., operational amplifiers, phase-locked-loops, etc.), different performance measurements, different design parameters, different components, different topologies, and so on.

— **Encapsulate Details.** Some tools interact with external programs and so it makes sense that the interface with these additional tools can be made in an automatically way hiding unused options.

2.3.2 Performance Analysis

Performance results are intrinsically correlated with several factors, like the evaluation engine, the search mechanism, the technological model precision, the computer platform used to run the application, etc.

The computation time is highly correlated with the nature of the evaluation engine. All approaches leading with models derived either by numeric equations or by some artificial learning machine method are able to reach solutions quickly, however, the quality of results are always estimated approaches and the solution quality only depends on the models precision [1]. This important trade-off between accuracy and computation time can be observed in Table 2.4. By contrast, simulation based methods that play with a high accurate circuit simulator in each optimization loop cycle are able to produce good quality results, but at the expense of higher execution times.

In the knowledge-based approaches the execution speed is the highest of all methods, considering that, the design plan is already defined. In equation-based approach, this value is normally high and is directly related to precision of the designed equations that need to be additionally introduced. The use of automatic methods to generate equations, like symbolic analyzers, can significantly reduce the input overhead and increase automation levels.

The *setup time* in equation and knowledge-based approaches is normally high and is directly related to the precision of the designed equations that need to be considered. The use of tools to generate equations, like symbolic analyzers, can significantly reduce the input overhead and increase automation levels. In the simulation based approach the level of interaction is the lowest of all methods. Only a few configuration parameters are necessary to setup the data for the external evaluation tool and the optimization algorithm. The behavioral-based approach

is compared in performance with the equation based approach once both use fast evaluation models, but in what concerns to setup time, they behave like the simulation based one, requiring the configuration of only a small number of parameters. However, additional time will be required to configure the tool to produce the target samples. In this case the time to build the training points will augment the setup time slightly.

Table 2.4 Factors affecting tools performance

	Knowledge	Equation	Simulation	Behavioral
Computation Time	+	+	-	+
Setup Time	- -	- - M/-A	+	-
Accuracy	-	-	+	-
Robustness	-	-	+	-

M- means "by manual equation" and A-"automatic by symbolic methods"
Symbols ordered from the best to the worst: '+', '-', '- -'

With regard to robustness, the most promising classes of tools come from methodologies which are able to produce high accurate solutions like the simulation based approaches, although they require multiple simulations which adversely affect the run-time of the algorithm in a few orders of magnitude. Theoretically, all other approaches could reach the desired robustness in case they are able to produce efficiently accurate models. However, this solution would be impractical due to the large time spent in the preparatory phase to obtain those models. Besides that, the equation-based as well as behavioral-based approaches were explored in order to model the distribution of variation parameters in a form which can be efficiently optimized. However, the accuracy of these approaches is questionable.

2.3.3 Optimization Techniques

Analog circuit design is considered a hard optimization problem and has been used by researchers in classical artificial intelligence, classical optimization, and intelligent systems as a testbench for their methods. Some of the most significant approaches concerning the optimization-based techniques are presented in Table 2.5. However, both the classical AI approaches (tree search, expert systems, etc.) and the classical optimization approaches have some drawbacks. The former suffer from the lack of flexibility: a lot of effort is needed in order to handle new processes, topologies, etc., and even when those are in place, the tools tend to fail whenever slightly different problems are handled. The latter, tend to be gradient-based approaches, which can only be applied to local parameter optimization when the objective functions are differentiable and the design space is continuous.

Nevertheless, complex circuit problems tend to be non-differentiable and may have continuous or discrete design spaces making these approaches inefficient. The "Intelligent" systems-based approaches (EA+SA+Stochastic), on the other hand, offer the potential to meet the target required by an analog cell design in such complex search spaces. Through the observation of Table 2.5 the predominance of these methods for implementing the optimization engine is obvious.

Table 2.5 Optimization-based techniques

	EA	SA	Stochastic	SA+Local	AI/NN	Classical
Simulation Based	Maelstrom GENOM	Maelstrom	Delight.Spice Anaconda	FRIDGE	-	-
Equation based	-	SD-OPT ASTR/OBLX	Opasyn	AMGIE	-	Maulik GPCAD
Learning Based	Alpaydin	Alpaydin	GENOM	-	Alpaydin GENOM	-

A significant part of the tools initially employed simulated annealing but later SA was used more frequently as a complement to other techniques, i.e., ALPAYDIN [59], MAELSTROM [57]. Combinations of two or more different methods are named hybrid methods (sometimes the hybridization of EA with local search techniques is also known as Memetic algorithms (see Sect. 3.1.4) and were developed to take advantage from the potentials of each solution. The idea is to create a new algorithm with improved capacity to explore the promising regions of the search space. For example, in MAELSTROM [57] system, Krasnicki et. al. applied a "parallel recombinative simulated annealing (PRSA) method which combines multiple simulated annealing algorithms that run concurrently and share information via a genetic algorithm scheme. The same group of researchers developed another variation called ANACONDA [56] that introduces constraints variations in transistor devices which incorporate a genetic algorithm, coupled with a local "pattern search" technique. The FASY [10] system is a fuzzy-logic based synthesis tool with simulated annealing for coarse and gradient search for fine optimization. The fuzzy logic chooses a topology from a pre-defined library. The originality of this approach is the use of a NN model, built from data collected in optimization runs that is employed to update the fuzzy rules. The ALPAYDIN [59] system is an analog integrated circuit synthesis that computes the device sizing using neural-fuzzy performance models and user defined equations. The neural-fuzzy model is used to estimate some of the AC performance metrics. The remainder AC performance metrics are modeled by user specific equations. The performance model is built from a set of training data collected from SPICE simulations. This system incorporates the effect of process variations but has a drawback since new equations must be calculated by the user for each new topology.

In conclusion, the analog circuit synthesis is really a demanding task, which a unique optimization algorithm could hardly solve. The development of new methodologies and techniques should be explored to increase the efficiency of analog circuit design. The trends verified in this area show that the solution for some of the most important approaches lies on the integration of several methods to combine the best of each one, and on the employing of models to reduce computation times.

2.3.4 Other Characteristics

In what concerns to the tools scope (see Table 2.6), the simulation-based and artificial intelligent methods take the high scores because they can be normally applied to a broad range of analog circuits and, similarly, they modify the design capabilities of the system without too much overhead. However, their scope depends on the simulator model. The equation based-approach, when dealing with manual design equations, has short scope, however, if equations are derived by symbolic tools, a better incorporation of new design problems is possible, increasing the scope of the tool. Knowledge based-approaches are generally close tools, because they are limited to a reduced number of architecture topologies and design objectives.

Table 2.6. Scope characteristic

	Knowledge	**Equation**	**Simulation**	**AI/NN**
Scope of the tool	-	+/-	+	+

The “Encapsulate Details”, “Interactive design”, “Bookkeeping Facilities” and “Implementation language” issues were not subject to comparative analysis between methodologies once they do not depend on design methodology but result from the merit of each tool in particular.

2.3.5 Summary

The first efforts in the development of CAD tools started with low abstraction level implementations targeting primarily small systems. Large and complex systems were decomposed into small building blocks employing the expert knowledge. The variety of existing tools and techniques covering several aspects of analog design are summarized in Fig. 2.7. The first generation of design automation tools was driven to the optimization of design parameters, leaving to the designer the task of selecting an appropriate architecture.

Since then, different types of selection topologies evolved ranging from template approaches, to bottom-up and top-down topology generation approaches, executing simultaneously or independently from sizing activities.

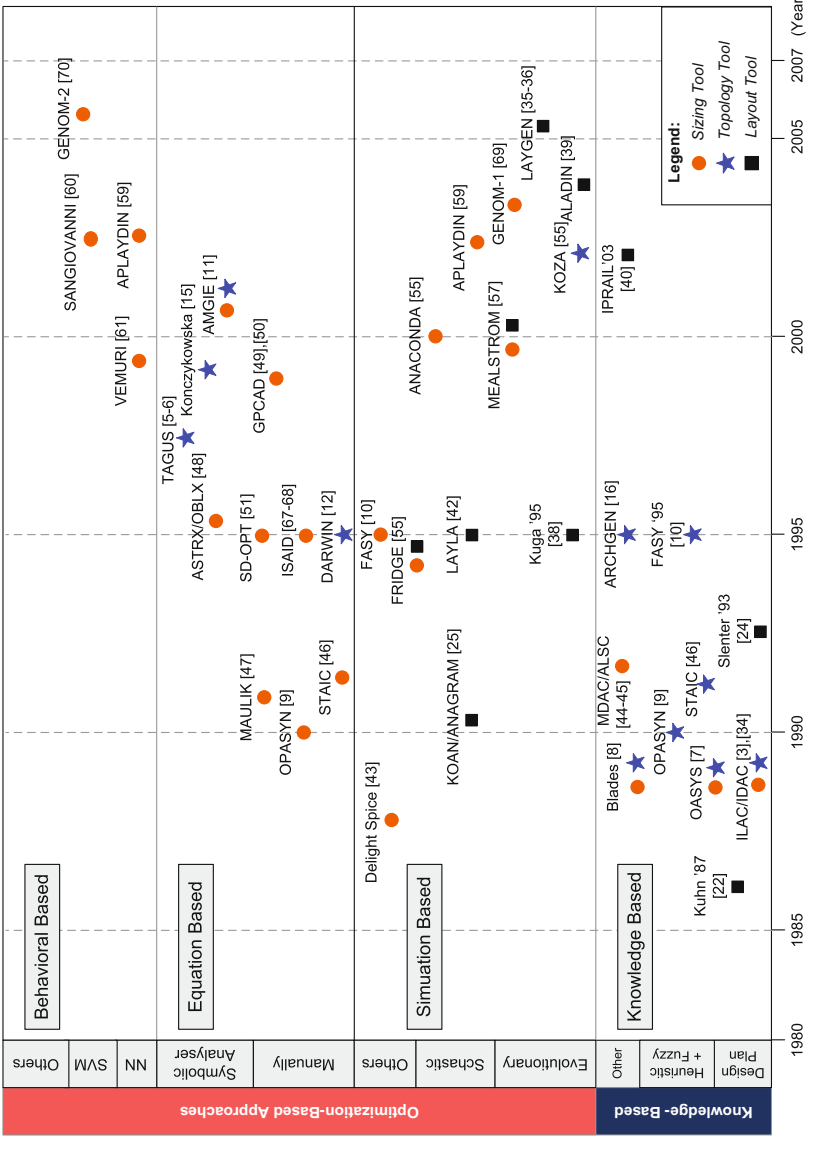

Fig. 2.7 Overview of analog synthesis tools

At the lower abstraction level, the knowledge methods based in special heuristic are out of date due to the long setup times involved, in the order of several weeks, which do not match the tight agenda of today's market pressure. In the equation based-approach, the run times are short and the setup can also be made short, if it uses automatic generation models, like symbolic analysis. The drawback is the limited accuracy of models, due to approximations or low-order design equations and limited flexibility in designs. The performance models based on posynomials and geometric programming foresee a great future if the time to produce these models is shortened or automatically generated without compromising accuracy. The simulation-based approach has high accuracy due to the use of circuit simulators. The generality is also high, allowing a large range of design problems to be addressed. However, the approach has longer execution times due to the use of a circuit simulator in the optimization loop. The model approach has short execution times and large generality. The model can be generated automatically and systematically. The drawback is, however, the large time spent in the preparatory phase as well as accuracy problems.

To conclude this topic, it must be said that effectively all presented methods have some points in favor and some against. Despite the broad spectrum of techniques and methodologies presented, there is not any "*defacto*" implementation for this area of applications. Despite the evolution verified in the high and low abstraction levels, both architecture selections, sizing and layout optimization remains the focus of research in analog EDA methodologies. The industrial commercial tools follow closely the main trends in academia and R&D workgroups. Their tools primarily focus the lower level of abstraction levels dealing with device sizing and layout description levels. All types of available frameworks assume the existence of a topology before the optimization run. Hence, no topology synthesis is available yet in any of the commercial analog EDA tools.

2.4 GENOM Optimization Tool: Implementation Goals

A tool aggregating all the best features, reviewed above, imposes hard challenges for the design of an automation tool. The set of all best covered features can be roughly interpreted as the main specifications of an ideal analog design tool. Naturally, only a subset of the ideal tool specifications is usually implemented in practice. Several important characteristics, however, can be appointed so that a tool can be accepted. They can be seen as the main specifications of a new design automation methodology. First, there is an undeniable trends in the use of optimization-based approaches, in order to, handle the challenges of the analog design. Second, the ideal tool should also deal with yield in order to take into account statistical fluctuations (process variations) inherent to the fabrication process and varying operating conditions (supply voltage or temperature variations), to make the design as robust as possible. Moreover, the design correctness and accuracy should be as close as possible to the industry electric validation tools. Furthermore, the overall optimization methodology should be as efficient as possible. Due to the existing trade-off between accuracy and computation time

(section 2.3.2), this important goal can not be treated individually. However, the performance achieved by the resulting tool should outperform the traditional methods or existing methodologies or tools. In order to have wide acceptance this tool should allow the designer to modify the design configuration in a short time. A graphical user interface has to be supplied in order to increase the productivity. The GUI interface adds reporting information, as the designer is able to evaluate some dynamic parameters of the optimization process and carry out some configuration steps (interactive design, and flexibility). As finally, the resulting application should be preferentially designed in an independent platform or integrated with current EDA design environments. The interaction with externals tools should be carried out with open standards, if possible, to make the application integration in industrial design easier.

Following the trends presented by several modern tools, GENOM combines state-of-the-art modeling and searching techniques to deal with the complexity of analog circuit design problem. Since it cannot be granted that derivatives of the objective functions are known for the generality of this multiobjective problem, we have to trust non-derivative optimization methods, hence this thesis assumes these methods as the best choice. To ensure the design correctness and accuracy, GENOM employs a standard simulation tool in the loop of a modified genetic algorithm kernel allowing the corner simulations. To increase the efficiency of the evolutionary algorithm, a machine learning algorithm based on SVM was introduced. The proposed approach results in a new GA-SVM learning scheme applied to analog circuit design composed by the interaction of two machine learning engines. GENOM is primarily designed to increase the automation level and so it encapsulates much of intrinsic algorithm parameters from normal users but it permits some algorithm parameter changes to restricted users, through a configuration file. To allow a better use of available resources GENOM allows the execution either in a single processor machine or in a multiprocessor distributed environment. For efficiency reasons the GENOM code was written in C, therefore, the default user interface is text file oriented despite it has built-in functions which allow it to integrate a graphical in-house design environment. The following chapters will explore the details of this new tool.

2.5 Conclusions

Automated design of analog circuits, also referred to as analog circuit synthesis, has been the subject of active scientific research for many years now. This chapter has covered some of the most significant design automation methodologies applied to analog IC design automation. Here, a set of general properties, that allow the characterization of each approach, was been identified and a better insight related to advantages and limitations has been presented.

The characterization of each different approach supports the definition and identification of the general specifications for a new design automation methodology to be implemented in GENOM, a tool that will be applied to the automation of mixed-analog ICs.

References

[1] Horta, N.C.: Analog and mixed-Signal IC design automation: Synthesis and optimization overview. In: Proc. 5th Conference on Telecommunications, Tomar, Portugal (2005)
[2] Martens, E., Gielen, G.: Classification of analog synthesis tools based on their architecture selection mechanisms. Integration, the VLSI Journal 41(2), 238–252 (2007)
[3] Degrauwe, M., et al.: IDAC: An interactive design tool for analog CMOS circuits. IEEE J. Solid-State Circuits 22, 1106–1115 (1987)
[4] Lourenço, N.: LAYGEN: Automatic layout generation of analog ICs, from a system to device level using both hierarchical template descriptions and intelligent computing techniques. Master thesis, Dept. Electrical and Computer Engineering, Instituto Superior Técnico, Portugal (2007)
[5] Horta, N.C., Franca, J.E.: High-Level data conversion synthesis by symbolic methods. In: Proc. IEEE Int. Symposium on Circuits and Systems, vol. 4, pp. 802–805 (1996)
[6] Horta, N.C.: Analogue and mixed-signal systems topologies exploration using symbolic methods. In: Proc. Analog Integrated Circuits and Signal Processing, vol. 31(2), pp. 161–176 (2002)
[7] Harjani, R., Rutenbar, R.A., Carley, L.R.: OASYS: A framework for analog circuit synthesis. IEEE Trans. Computer-Aided Design 8, 1247–1265 (1989)
[8] El-Turky, F., Perry, E.: BLADES: An artificial intelligence approach to analog circuit design. IEEE Trans. Computer-Aided Design 8, 680–692 (1989)
[9] Koh, H.Y., Sequin, C.H., Gray, P.R.: OPASYN: A compiler for CMOS operational amplifiers. IEEE Trans. Computer-Aided Design 9(2), 113–125 (1990)
[10] Torralba, A., Chávez, J., Franquelo, L.G.: FASY: A fuzzy-logic based tool for analog synthesis. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems 15(7), 705–715 (1996)
[11] Gielen, G., et al.: An analog module generator for mixed analog/digital ASIC design. Wiley Int. J. Circuit Theory Applications 23, 269–283 (1995)
[12] Kruiskamp, W., Leenaerts, D.: DARWIN: CMOS opamp synthesis by means of a genetic algorithm. In: Proc. ACM/IEEE Design Automation Conference, pp. 550–553 (1995)
[13] Stehr, G., Pronath, M., Schenkel, F., Graeb, H., Antreich, K.: Initial sizing of analog integrated circuits by centering within topology given implicit specifications. In: Proc. IEEE International Conference on Computer-Aided Design, pp. 241–246 (2003)
[14] Horta, N.C., Franca, J.E.: Algorithm-driven synthesis of data conversion architectures. IEEE Trans. Computer-Aided Design Integrated Circuits 16(10), 1116–1135 (1997)
[15] Konczykowska, A., Bon, M.: Structural synthesis and optimization of analog circuits symbolic analysis techniques. IEEE, Los Alamitos (1998)
[16] Antoa, B.A., Brodersen, A.J.: ARCHGEN: Automated synthesis of analog systems. IEEE Trans. VLSI Systems 3(2), 231–244 (1995)
[17] Lohn, J.D., Colombano, S.P.: A circuit representation technique for automated circuit design. IEEE Trans. Evolutionary Computation 3(3), 205–219 (1999)
[18] Koza, J.R., Bennett, F.H., Andre, D., Keane, M.A., Dunlap, F.: Automated synthesis of analog electrical circuits by means of genetic programming. IEEE Trans. Evolutionary Computation 1(2), 109–128 (1997)

[19] Sripramong, T., Toumazou, C.: The invention of CMOS amplifiers using genetic programming and current-flow analysis. IEEE Trans. Comput. Aided Design Integrated Circuits 21(11), 1237–1252 (2002)
[20] Leenaerts, D., Gielen, G., Rutenbar, R.A.: CAD solutions and outstanding challenges for mixed-signal and RF IC design. In: Proc. IEEE/ACM International Conference on Computer Aided Design, pp. 270–277 (2001)
[21] Aggarwal, V.: Analog circuit optimization using evolutionary algorithms and convex optimization. Master thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (2007) `http://web.mit.edu/varun_ag/www/msthesis.pdf` (Accessed March 2009)
[22] Kuhn, J.: Analog module generators for silicon compilation. In: Proc. VLSI System Design, pp. 75–80 (1987)
[23] Wolf, M., Kleine, U., Hosticka, B.J.: A novel analog module generator environment. In: Proc. European Conference on Design and Test, pp. 388–392 (1996)
[24] Beenker, G., Conway, J., Schrooten, G., Slenter, A.: Analog CAD for consumer ICs. In: Huijsing, J., Plassche, R., Sansen, W. (eds.) Analog Circuit Design, pp. 347–367. Kluwer Academic Publishers, Norwell (1993)
[25] Cohn, J., Garrod, D., Rutenbar, R.A., Carley, L.R.: KOAN/ANAGRAM II: New tools for device-level analog placement and routing. IEEE J. Solid-State Circuits 26, 330–342 (1991)
[26] Carley, L., Georges, G., Rutenbar, R.A., Sansen, W.: Synthesis tools for mixed-signal ICs: Progress on frontend and backend strategies. In: Proc. Design Automation Conference, vol. 33, pp. 298–303 (1996)
[27] Lampaert, K., Gielen, G., Sansen, W.: A performance driven placement tool for analog integrated circuits. IEEE Journal of Solid-State Circuits 30, 773–780 (1995)
[28] Khademsameni, P., Syrzycki, M.: A tool for automated analog CMOS layout module generation and placement. In: Proc. IEEE Canadian Conference on Electrical and Computer Engineering, pp. 416–421 (2002)
[29] Yılmaz, E., Dündar, G.: New Layout generator for analog CMOS circuits. In: Proc. 18th European Conference on Circuit Theory and Design, pp. 36–39 (2007)
[30] Hartono, R., Jangkrajarng, N., Bhattacharya, S., Shi, C.: Automatic device layout generation for analog layout retargeting. In: Proc. International Conference on VLSI Design, vol. 36, pp. 457–462 (2005)
[31] Lampaert, K., Gielen, G., Sansen, W.: Analog layout generation for performance and manufacturability. Kluwer Academic Publishers, Dordrecht (1999)
[32] Jingnan, X., Vital, J., Horta, N.C.: A SKILLTM-based library for retargetable embedded analog cores. In: Proc. Design Automation and Test in Europe Conference and Exhibition, pp. 768–769 (2001)
[33] Jingnan, X., Serras, J., Oliveira, M., Belo, R., Bugalho, M., Vital, J., Horta, N.C., Franca, J.: IC design automation from circuit level optimization to retargetable layout. In: Proc. 8th IEEE International Conference on Electronics, Circuits and Systems, vol. 1, pp. 95–98 (2001)
[34] Rijmenants, I., Schwarz, Y.R., Litsios, J.B., Zinszner, R.: ILAC: An automated layout tool for CMOS circuits. IEEE Journal of Solid-State Circuits 24(2), 417–425 (1989)
[35] Lourenço, N., Horta, N.C.: LAYGEN – An evolutionary approach to automatic analog IC layout generation. In: Proc. IEEE Conf. on Electronics, Circuits and System, Tunisia (2005)

[36] Lourenço, N., Vianello, M., Guilherme, J., Horta, N.C.: LAYGEN – Automatic layout generation of analog ICs from hierarchical template descriptions. In: Proc. IEEE Ph. D. Research in Microelectronics and Electronics, pp. 213–216 (2006)
[37] Zhang, L., Kleine, U.: A genetic approach to analog module placement with simulated annealing. In: Proc. IEEE Int. Symposium on Circuits and Systems, vol. 1, pp. 345–348 (2002)
[38] Handa, K., Kuga, S.: Polycell placement for analog LSI chip designs by genetic algorithms and tabu search. In: Proc. IEEE Conference on Evolutionary Computation, vol. 2, pp. 716–721 (1995)
[39] Zhang, L., Kleine, U.: A novel analog layout synthesis tool. In: Proc. IEEE Int. Symposium on Circuits and Systems, vol. 5, pp. 101–104 (2004)
[40] Jangkrajarng, N., Bhattacharya, S., Hartono, R., Shi, C.J.: IPRAIL: Intellectual property reuse based analog IC layout automation. Integration, the VLSI Journal 36(4), 237–262 (2003)
[41] Castro-Lopez, R., Fernandez, F.V., Guerra-Vinuesa, O., Vazquez, A.: Reuse Based Methodologies and Tools in the Design of Analog and Mixed-Signal Integrated Circuits. Springer, Heidelberg (2003)
[42] Cory, W.: Layla: A VLSI Layout Language. In: Proc. 22nd ACM/IEEE Conference on Design Automation, pp. 245–251 (1985)
[43] Nye, W., Riley, D.C., Sangiovanni-Vincentelli, A., Tits, A.L.: DELIGHT.SPICE: An optimization-based system for the design of integrated circuits. IEEE Trans. Computer-Aided Design 7(4), 501–519 (1998)
[44] Leme, C., Horta, N.C., Franca, J.E., Yufera, A., Rueda, A., Huertas, J.L., et al.: Flexible silicon compilation of charge redistribution data conversion systems. In: Proc. IEEE Midwest Symposium on Circuits and Systems, pp. 403–406 (1991)
[45] Horta, N.C., Franca, J.E., Leme, C.A.: Framework for architecture synthesis of data conversion systems employing binary-weighted capacitor arrays. In: Proc. IEEE Int. Symposium on Circuits and Systems, pp. 1789–1792 (1991)
[46] Harvey, J.P., Elmasry, M.I., Leung, B.: STAIC: An interactive framework for synthesizing CMOS and BICMOS analog circuits. IEEE Trans. Computer-Aided Design 11(11), 1402–1417 (1992)
[47] Maulik, P.C., Carley, L.R.: Automating analog circuit design using constrained optimization techniques. In: Proc. IEEE Int. Conf. Computer-Aided Design, pp. 390–393 (1991)
[48] Ochotta, E.S., Rutenbar, R.A., Carley, L.R.: Synthesis of high-performance analog circuits in ASTRX/OBLX. IEEE Trans. Computer- Aided Design 15(3), 273–294 (1996)
[49] Hershenson, M., Boyd, S., Lee, T.: GPCAD: A tool for CMOS op-amp synthesis. In: Proc. IEEE/ACM Int. Conf. Computer-Aided Design, pp. 296–303 (1998)
[50] Hershenson, M.M., Boyd, S.P., Lee, T.H.: Optimal design of a CMOS Op-Amp via geometric programming. IEEE Trans. Computer-Aided Design 20(1), 1–21 (2001)
[51] Medeiro, F., Verdu, B.P., Vazquez, A.R., Huertas, J.L.: A vertically integrated tool for automated design of modulators. IEEE Journal of Solid-State Circuits 30(7) (1995)
[52] Gielen, G., Wambacq, P., Sansen, W.: Symbolic analysis methods and applications for analog circuits: A tutorial overview. Proc. IEEE 82, 287–304 (1994)
[53] Wambacq, P., Fernandez, F.V., Gielen, G., Sansen, W., Rodriguez-Vazquez, A.: Efficient symbolic computation of approximated small signal characteristics. IEEE J. Solid-State Circuits 30, 327–330 (1995)

[54] Daems, W., Gielen, G., Sansen, W.: An efficient optimization–based technique to generate posynomial performance models for analog integrated circuits. In: Proc. 39th Design Automation Conference, pp. 431–436 (2002)
[55] Medeiro, F., et al.: A Statistical optimization-based approach for automated sizing of analog cells. In: Proc. ACM/IEEE Int. Conf. Computer-Aided Design, pp. 594–597 (1994)
[56] Phelps, R., Krasnicki, M., Rutenbar, R.A., Carley, L.R., Hellums, J.: ANACONDA: Simulation-based synthesis of analog circuits via stochastic pattern search. IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems 19(6), 703–717 (2000)
[57] Krasnicki, M., Phelps, R., Rutenbar, R.A., Carley, L.R.: MAELSTROM: Efficient simulation-based synthesis for custom analog cells. In: Proc. ACM/IEEE Design Automation Conference, pp. 945–950 (1999)
[58] Liu, H., Singhee, A., Rutenbar, R.A., Carley, L.: Remembrance of circuits past: Macromodeling by data mining in large analog design spaces. In: Proc. Design Automation Conference, pp. 437–442 (2002)
[59] Alpaydin, G., Balkir, S., Dundar, G.: An evolutionary approach to automatic synthesis of high-performance analog integrated circuits. IEEE Trans on Evol. Computation 7(3), 240–252 (2003)
[60] Bernardinis, F., Jordan, M.I., Sangiovanni-Vincentelli, A.: Support vector machines for analog circuit performance representation. In: Proc. Design Automation Conference, pp. 964–969 (2003)
[61] Wolfe, G.A.: Performance macro-modeling techniques for fast analog circuit synthesis. Ph.D. dissertation, Dept. of Electrical and Computer Engineering and Computer Science, College of Engineering, University of Cincinnati, USA (1999)
[62] Synopsys Inc.: Products and solutions-HSIM, PowerMill, NanoSim (2009), http://www.synopsys.com (Accessed March 2009)
[63] EEDesign, Synopsys acquires ADA for analog boost (2009), http://www.eetimes.com/ (Accessed March 2009)
[64] Synopsys Inc.: Circuit Explorer - analysis, optimization & trade-off (2009), http://www.synopsys.com (Accessed March 2009)
[65] Cadence Inc, Products: Composer, Virtuoso, DIVA, NeoCircuit, NeoCell, UltraSim, NcSim (2009), http://www.cadence.com (Accessed March 2009)
[66] Toumazou, C., Makris, C.: Analog IC design automation: Part I - Automated circuit generation: New concepts and methods. IEEE Trans. Computer-Aided Design 14, 218–238 (1995)
[67] Makris, C., Toumazou, C.: ISAID: Qualitative reasoning and trade-off analysis in analog IC design automation. In: Proc. IEEE Int. Symposium on Circuits and Systems, pp. 2364–2367 (1992)
[68] Toumazou, C., Makris, C.A., Berrah, C.M.: ISAID - a methodology for automated analog IC design. In: Proc. IEEE Int. Symposium on Circuits and Systems, vol. 1, pp. 531–535 (1990)
[69] Barros, M., Neves, G., Guilherme, J., Horta, N.C.: Enhanced genetic algorithm kernel applied to a circuit-level optimization E-Design environment. In: Proc. 10th IEEE International Conference on Electronics, Circuits and Systems, pp. 1046–1049 (2003)
[70] Barros, M., Neves, G., Guilherme, J., Horta, N.C.: Analog Circuits Optimization based on Evolutionary Computation Techniques. Integration, the VLSI Journal 43(1), 136–155 (2010)