INTEGRATION, the VLSI journal 55 (2016) 67–84

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Γ (Gamma): A SaaS-enabled fast and accurate analog design System

Yishai Statter*, Tom Chen

Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523, USA

ARTICLE INFO

Article history:
Received 30 November 2015
Received in revised form
21 March 2016
Accepted 21 March 2016
Available online 31 March 2016

Keywords:
Analog circuit design automation
Table look-up
SaaS

ABSTRACT

With ever increasing demand for lower power consumption, lower cost, and higher performance, designing analog circuits to meet design specifications has become an increasing challenging task, Analog circuit designers must, on one hand, have intimate knowledge about the underlining silicon process technology's capability to achieve the desired specifications. They must, on the other hand, understand the impact of tweaking circuits to satisfy a given specification on all circuit performance parameters. Analog designers have traditionally learned to tackle design problems with numerous circuit simulations using accurate circuit simulators such as SPICE, and have increasingly relied on trial-and-error approaches to reach a converging point. However, the increased complexity with each generation of silicon technology and high dimensionality of searching for solutions, even for some simple analog circuits, have made the trial-and-error approach extremely inefficient, causing long design cycles and often missed deadlines. Novel rapid and accurate circuit evaluation methods that are tightly integrated with circuit search and optimization methods are needed to aid design productivity.

Furthermore, the current design environment with fully distributed licensing and supporting structures is cumbersome at best to allow efficient and up-to-date support for design engineers. With increasing support and licensing costs, fewer and fewer design centers can afford it. Cloud-based software as a service (SaaS) model provides new opportunities for CAD applications. It enables immediate software delivery and update to customers at very low cost. SaaS tools benefit from fast feedback and sharing channels between users and developers and run on hardware resources tailored and provided for them by the software vendor. On the downside, web-based tools are expected to perform in a very short turn-around schedule and be always responsive.

This paper presents a list of innovations that come together to a new class of analog design tools: 1). Lookup table-based approach (LUT) to model complex transistor behavior provides both the necessary accuracy and speed essential for repeated circuit evaluations. 2). The proposed system architecture tight integrate the novel LUT approach with novel system level functions to allow further significantly better accuracy/speed tradeoff and faster design convergence with designer's intent. 3). Incorporating use inputs at key junctures of the design process allows the tool to better capture designer's intent and improve design convergence. 4). The combination of high accuracy and faster evaluation time make it possible to incorporate SaaS features, such as short solution space navigation steps and crowdsourcing, into the tool. This allows sharing of server-side resources between many users. Instead of fully automating a signoff circuit optimization process, the proposed tool provides effective aid to analog circuit designers with a dash-board control of many important circuit parameters with several orders faster in computation time than SPICE simulations.

1. Introduction

Analog circuit design automation has been lagging significantly behind its digital counterpart. The main reason for this is the complexity of evaluating circuits against users' specifications and a large number of competing design goals in the optimization process. Since the 1980s, several software research projects attempted to improve design productivity and even provide fully automated synthesis of analog circuits [1–4]. Most did not mature beyond academic prototypes. Some made it into market and attracted customers [5], but did not scale up to complex topologies and failed to migrate along each generation of silicon technology [6]. Yet, demand for lower power consumption, lower cost, and higher performance for analog circuits is increasing. Consequently,

* Corresponding author.

http://dx.doi.org/10.1016/j.vlsi.2016.03.003
0167-9260/

designing analog circuits to meet tight specifications has become an increasing challenging task. The need for practical solutions to improve design productivity is ever present in the semiconductor industry, where time-to-market project constraints push the analog design efforts closer to a bottleneck position.

The majority of the existing tools have the following characteristics in common:

1. They start with a topology plus performance constraints and attempt to generate production-ready circuit autonomously without any designer feedback in the loop.
2. They use internal or enwrapped simulations to evaluate solutions during the optimization process.
3. They run “on premises”, as opposed to hosted remotely, requiring customers to provide the hardware and sometime complementary software.

To circuit designers, the existing tools and approaches can be viewed as black-box approaches. The black-box-type optimization has several disadvantages:

1. For approaches using the SPICE engine for their performance evaluations, it typically takes long time to completion, breaking the natural flow of users' decision making process.
2. For approaches using high-level algebraic formula for their performance evaluations, the results are typically far from desired optima due to tool's inabilities of capturing circuit's high order effects with even complex algebraic formula.
3. Users often have to constrain the design problem well and be vigilant about the tasks they hand-over to the machine, because of the high risk of starting a long run that ends with results that are outside the constraint region.
4. The internal optimization process in the flow does not allow designer feedback. Users complain that the pre-defined optimization flow is useful in solving high volume of problems in correct, consistent and repeatable process (e.g. ASIC place and route), but designers often reach better and faster solutions when presented with one at a time instances of more complex problem, such as analog circuit design.

In existing attempts, tool programmers make a common mistake of putting the machine at the center of the flow by prioritizing functional features [7] and failing to give users the control to run optimization steps and stages interactively and intuitively, which is crucial in analog circuit design process as the number of competing design goals are often too numerous for any algorithm to handle successfully. Instead, users are kept out of the loop once the optimization process has started and then are given the choice of accepting the final result or re-spin it, if those are not satisfactory. Conversely, web-based tools do not have the luxury of running hours-long processes of A–Z optimization flow. Commonly accepted expectations from internet responsiveness [8] dictate much faster turn-around, continuous controllability and intuitive, dynamic visualization of runs in progress. These challenging expectations can also be opportunities for a new class of analog design tools that put designers back at the helm of the design flow. Web application for sizing transistors and designing analog circuits must focus on the least computationally-ambitious atomic tasks and allow users to navigate between them and connect the flow. One aspect of design automation that can be a good fit for the web is mapping and visualizing solution spaces of topologies and specs to provide users with performance trade-off they need to make engineering decisions.

Stand-alone or tool-integrated SPICE simulations decouple the task of evaluating circuits from the optimization engine. The biggest benefit of using generic SPICE is that technology-specific parameters and models [9] can be integrated and modified with foundry characterized behavior with little tool vendor involvement. However, the silicon-accurate results provided by SPICE come at a cost. Evaluations of reasonably sized analog circuits include overheads, such as topology analysis, redundant calculations of transistor physical characteristics and simulations of elaborate test harnesses, necessary to adapt circuit property measurements to one of the generalized simulation types: DC, AC and Trans. The amount of overhead can make the repeated evaluations required during analog circuit design and tradeoffs extremely inefficient. This impact can be even worse for analog circuit design tools in the SaaS environment where a large number of circuits need to be evaluated during a typical HTTP transaction interval.

On-premises software is a common feature to most contemporary industrial EDA software. High demand in computational resources makes tool vendor focus on software alone, leaving the customer to provide their own hardware. The complexity and diversity of applications gave birth to the classic EDA support structure. Tools are coded by EDA vendors with a vast range of algorithms and configurations. The task of choosing between all the configurations is passed to the responsibility of customers, who are left to figure out best usage of input scripts, configuration files and GUI's. Vendors have to employ an army of field applications engineers (FAE) to help customers figure out the best configuration and keep the vendor itself up to date with the market needs. An FAE, like the software itself, is most often embedded on premises with the design team. This semi-automation structure is expensive and cumbersome. Effective SaaS systems attempt to implement crowd-sourced feature additions, peer networking and remote help forums to eliminate the need for human-based support structure. Direct communication channel can improve tool performance with automatically collected usage statistics. Certain results can be shared and reused between customers in a behind the scenes automated learning and archiving system. New features, conceived automatically from public-demand chats and surveys, are available instantly, without any user effort or need of IT support.

This paper presents the architecture and implementation of SaaS-enabled analog design tool that breaks away from the on-premises and all automated design approach to provide design assistance that is more aligned with analog workflow. The tool is designed to give a designer a mapping and navigating console in the solution space, based on user specification, topology and target technology. Once a solution is found, a SPICE netlist of the solution is produced for the sign-off phase.

The remaining part of this paper is organized in six sections. Section II provides a brief overview of past attempts to automate analog circuits design. The overview lists approaches and computational strategies that were leveraged on in this work. Section III presents the proposed SaaS-enabled architecture as a design assistant for analog circuit design. Section IV gives an overview of the usage model for proposed analog circuit design system, Γ (Gamma). Section V provides details about algorithmic and programming aspects of implementing the proposed Γ system. Section VI presents performance results of an implementation of the Γ system, in terms of accuracy, usability and speed. Section VII gives some discussions about the proposed system and provides concluding remarks.

2. Previous work

A. Overview

After the advent of the SPICE simulator [10], academic research in the field of analog design automation has focused mainly on automating the manual design flow [11]. This was done by

employing several optimization and synthesis strategies on one hand, while leaving circuit evaluation to SPICE on the other. Categorizing all those efforts can be done by 1) construction method: knowledge-based vs optimization-based, 2) usage model: e.g. interactive vs fully automated, and 3) scope: topology vs layout.

B. Symbolic analysis

Analytic models were demonstrated for circuit optimization by Gielen et. al. in their ISAAC/OPTIMAN system [3]. ISAAC was described as "symbolic simulator", in charge of calculating circuits' performance properties based on algebraic representation of the optimized topology. OPTIMAN was an optimization engine based on simulated annealing (SA). The objective it received was a weighted cost function that represented the user's priority. The system used simplified equation-based transistor models and ran on a mainframe computer. Algebraic representation of circuits (Modified Nodal Analysis [12], herein MNA) was used extensively when transistor equation models could be simplified reliably to be able to find a deterministic solution to circuit constraints. To improve the existing symbolic analysis methods, topology analysis can be limited to the initial steps of SPICE simulations and circuit-level equations play only internal role in it. Algebraic representation of circuit can be reused for an off-line compilation of the mapper's topology templates. Doing this offline enables optimized compilation of topology equations and thus shortens response time compared to SPICE significantly. Offline MNA and compilation stages come at the expense of flexibility, because the system requires a compiled module per analyzed topology. However, this inflexibility is expected and tolerated in a system that offers a library of topologies that can be sized in a short schedule. On a circuit-level, there is no difference in accuracy between SPICE and a symbolic evaluation engine. However, when it comes to transistor level modeling, equation models trade accuracy for speed [24]. This shortcoming can be avoided by using alternative models that are faster than SPICE, but do not sacrifice too much accuracy.

C. Alternative transistor modeling

Physics equation-based transistor models are used at the core of SPICE simulations [9]. Developed independently of foundries, they represent key physical phenomena that have significant influence on transistor behavior. The number of key phenomena grows with every technological generation and thus the number of equations and fitted coefficients that are supplied by foundries based on lab measurements [13]. Two advantages of equation models, accuracy and smooth waveforms, make them well suited for simulation. However, the overhead associated with the long stack of equations, many not even needed for the bottom-line analysis, makes these models too time-consuming for evaluating large sets of circuits. Trying to accelerate these models by simplifying the equation models made some commercial success.[2] However, speed advantages of simplified equation-based models were eventually negated by the resulting inaccuracy due to simplified treatment of device physics. As process generations progressed, those marginal phenomena took larger influence and made the models obsolete [6]. This weakness can be addressed by a modeling approach that's independent of the physics it is attempting to mimic: lookup tables (LUT). Yoon and Allen first suggested replacing equation models with lookup tables [14,15], with the goal of speeding up simulations. The result of their work was a mere 10% saving in run time, primarily because their method required run-time calculation of quadratic interpolation coefficients. This particularly expensive interpolation was chosen to ensure

1. Smoothness of curves to allow numeric derivation of I_{DS}.
2. Better-fit of the small tables they were using back with memory resources typical to the early 1990s.

Follow up simulation-level [16] models narrowed LUT role to correcting errors of compact equations-based models.

However, when no simulation is required, smoothness is redundant and computational cost can be saved by using first-order ad hoc data location and interpolation techniques. Furthermore, using contemporary hardware resources, larger tables can provide better resolution at low cost and the derivatives of I_{DS} can be pre-calculated and stored in separate tables. Latest experiments with a LUT-based model [17] showed that a 2MB table can generate a transistor's physical property (e.g. g_m) with $<1\%$ error compared to BSIM, in under ½ μs. The same hardware setup measured 140 μs per BSIM query.

D. Multi-objective optimization

Breadth analysis on a given circuit is not a new concept. Sweep-analysis and scripted simulations [18] are already standard in every analog design environment [19]. However, an automated sweep is often a simple loop, lacking search objectives and often limited in dimensionality. Scripted search can potentially perform any test algorithm [20], although the overheads associated with running an interpreted algorithm that launches costly SPICE runs make it less desirable for SaaS-enabled design tools. Still, there are strategies and algorithms explored in previous research that can be useful for mapping circuit-size spaces and can be re-implemented on much faster software environment. Ant colony optimization (ACO) [21], for instance, is a popular approach for populating multi-objective Pareto front database. Other parallel algorithms, such as particle-swarm (PSO) [22] and genetic algorithm (GA) [23] have been researched thoroughly. However, efficient implementation with regard to users' expectations was largely overlooked. Some of these approaches could power offline preparation steps of tool's database and some simpler ones (e.g. simulated annealing) could serve in online optimization.

E. Design-supporting analysis

Binkley et. al. presented a "MOS design tool" as a graphic calculator of key transistor performance properties [24]. Its novelty is not in promising an end to end automated design solution, but in providing immediate data to support design choices and calculations. The tool's engine did not offer automated search for optimal solution, but rather functioned as transistor-level behavior calculator to assist a design process. A similar tool was built by the authors [17] to demonstrate the effectiveness of LUT based models with better performance and accuracy than the tool proposed by Binkley.

F. Saas-enabled tools

Most analog design tools [25–27] found on the web are simply a web-adapted version of the on-premises software that's been part of the industry for three-plus decades. One can find "integrated design environment" for digital [19] and analog design disciplines, which are only different from their on-premises predecessors in one aspect: they include internet-collateral features that are easy to add and expected in every other creative website. Such features include user forums and blog outlets. The business model of online EDA tools is more web oriented: subscription fees and/or 3rd party advertising revenue. Easy updates and feedback also favor web-apps, such as the ones made famous by the Google Company. However, they make no computational utilization of web-specific capabilities, mainly because they only attempt to mimic the expected functionality of on-premises software that was designed without those capabilities available to begin with. There are no centrally-shared calculations, for instance, because the depth-first analyses they are designed to do not lend themselves to archiving and reusing results anonymously.

G. Layout-assisting generators

A class of analog design generators assumes the existence, ability, efficiency and accuracy of simulation engines, either commercial or freeware, and focus instead on facilitating the design flow in a higher programming or scripting level. Such design tools offer flow management, user intent record, replay and reuse framework and a suite of generators, especially for analog layout, through high level abstraction of circuits. Tools such as interactive design tool for analog CMOS circuits (IDAC) [2] and most recently the Berkeley Analog Generator (BAG) [28] belong to this category.

BAG is a Python-scripted framework designed for "closing the gap between designer and CAD communities" [28]. Both IDAC and BAG are knowledge-based, relying on cell-level libraries generated by PyCells, a Python-based layout-generator. It does not include its own circuit-evaluation software and therefore its ability to search for optimal sizing is bound by the commercial simulator it calls.

For sign-off quality of post layout circuits, on-premises, large, SPICE-accurate tools are still needed. However, for a preliminary analysis of topologies' fitness to perform a specified analog performance, a fast web-based application that requires neither setup nor dedicated hardware is more desirable. Such application can rapidly produce trade-off graphs and heat-map visualization of sizing to performance dependence with few clicks of buttons and can potentially be accessed from any web client.

H. Commercial attempts

Cadence provides an optimizer plug-in to their Virtuoso design environment, NeoCircuits [29], which serves as a top-level optimization loop on top of their simulators Specter [30] and UltraSim [31]. Its algorithmic infrastructure is based on the genetic algorithm (GA) and drives autonomously, rather than interactively, to a single solution that satisfies a given specification. This architecture of using a simulator as a black-boxed evaluation engine in an optimization loop was offered since the DELIGHT [1] project. The limited adoption [32] of NeoCircuits' in commercial designs can be attributed to its lack of user interaction during the optimization process.. Synopsys offers its own automation solution, Laker, which promises a "complete solution for analog, mixed-signal, and custom digital design and layout" [33]. However, Laker is geared mostly to layout and design-environment, rather than optimization of transistor sizes.

Barcelona Design made an attempt to break away from SPICE-based evaluation [5], but failed to convert its tool to the next technology and beyond its topology set. Its equations-based evaluation was not accurate enough for the next generation of technology and not fast enough for bigger circuits.[3] However, EDA industry did succeed in providing design environments, such as Cadence's ADE and websites that mimic its functionality. Those tools, while not automating the design process similar to those in the digital-domain, show that a preferred design flow is to keep the engineer at the center of the design process. A lesson learned from the failure of synthesizers is that designers want tools that provide them with accurate, useful and timely data they need to make decisions and navigate toward a solution by themselves. This type of user-centric automation also offers a fertile ground to be explored by SaaS, because of web's natural ability for interactivity and visualization.

3. Proposed architecture

A. Motivation

For a given circuit template, analog circuit designers try to find transistor sizes in the circuit to meet a list of performance specifications. A major challenge in that search is lack of reliable quantitative information about the tradeoff trend between two or more performance aspects, given a circuit topology and manufacturing technology. A designer often choses to run many single or sweep simulations to understand how circuit performance parameters are trending, relative to circuit physical parameters. This long process can be helped by providing a graphic mapping of a topology's performance Pareto front. A tool that can create graphical information for a designer about what a topology can and cannot do, what needs to be "given-up" by certain amount in the spec in order to find a solution and the expected costs of meeting the spec in terms of area and power consumption.

Furthermore, requirements from web-based tools are driven by users' expectation of short response time [7]. Therefore, programming decisions were taken to trade hardware resources and accuracy, where possible and within reason, to gain speed. The main construction guidelines were:

1. Prefer data look-up over calculations – Starting from a look-up table transistor modeling and continuing with archived Pareto containers, specifically designed for quick queries.
2. Precompile topology-specific calculations – Via generated C or tailored VM code, every analysis that can be done offline saves run time from online queries. The most time-consuming code optimization can be done offline to produce short as possible circuit-evaluating routines.
3. Pre-calculate topology-specific solutions – Use daemons, past user queries and retention guidelines to produce a general set of popular circuits that can be filtered as response to spec and be used as baseline for optimization.
4. Keeps optimization running interactively until session expires. Instead of A–Z algorithm runs, dynamically build up solution set in the database and let users run graphic analyses on update-to-date results. This way, even an atypical minutes-long online task remains interactive and avoids the risk of losing the user's interest or sense of control.
5. Make use of asynchronous web interface schemes (e.g. Ajax), leaving the user with a sense of using a search-engine, rather than a classic synthesizer. Similar to other web-based search applications used in popular sites, the quality of results should be developing dynamically over time and creating an animated picture of evolving solution. This way, users receive initial crude (yet useful) results and get closer to optimum results later on.
6. Curate optimized solutions for future queries in a background process – Crowdsourcing optimization results means that every spec is calculated only once for the entire user community.

B. System overview

We propose a SaaS-enabled analog design system. Γ (simply: "Gamma") is a circuit evaluating and sizing system. Its executable binary was derived from the RAMSpice [17] system's database structures and API, including lookup table transistor-level models and hierarchical data organization. Unlike its predecessor, a derivative of NGSPICE [10] used mainly to translate transistor models into table form,.

Fig. 1 shows the overall architecture of the proposed analog design system. The Γ system consists of four main building blocks:

1. Offline toolset for topology compilation and performance mapping that is designed to pre-calculate as much data as possible in relaxed schedule.
2. Circuit evaluating code designed for shortest possible response time.
3. Web front-end and session management scripts.

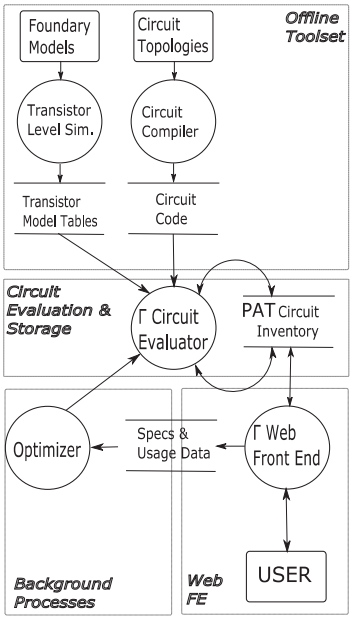

Fig. 1. High-level dataflow diagram of the Γ system.

4. Background daemon for optimizing and managing the circuit inventory.

The system operates in two modes. The offline mode generates sufficient amount of seed data to improve the responsiveness of the online mode. The latter is used only for web transactions during SaaS sessions.

Both modes of operation share the main building blocks of Γ which consists of two main parts of its own:

1. Γ circuit evaluator – Calculating a circuit's performance out of sizing parameters, using precompiled code and LUT-based transistor-level models.
2. Pareto Associative Table (PAT) – Stores pre-calculated sizing and performance figures for the existing circuit inventory.

The two operating modes are detailed in Figs. 2 and 3, for online and offline operations respectively.

C. Γ Circuit evaluator

Circuit evaluation is the operation of calculating a vector of a circuit's performance properties from a vector of transistor sizes and other design parameters. This is where Γ replaces the traditional role reserved for SPICE in simulating circuits' performance aspects. The evaluator is not fully pre-coded. Instead, the system contains a compiler that generates code per topology. This gives Γ its unique efficiency in producing a rapid performance evaluation per each point in the sizing space. Just like SPICE, the evaluator has to access transistor-level physical models. These, as discussed in the previous paper [17] are kept as LUTs for even more rapid execution time (Fig. 4).

The evaluator includes the following stages:

1. Operating-point loop – Calculate DC operating point of the circuit. The loop's starting point may be an MNA-derived rough estimate of nodes' voltages or previous OP calculated for a similarly sized circuit. User-defined dependences of design sizes, such as bias-circuit channel sizes to match required current and bias voltages to keep output DC level in mid-rail, are automatically tuned during this loop.
2. Performance parameter calculations – A set of performance parameters is calculated based on the converged DC operating point. Table 1 lists all the performance parameters being calculated during this stage. An output vector is created for each set of performance parameters.
3. PAT access – Add the evaluated circuit with all its performance parameters to the PAT, in accordance with Pareto domination rules.

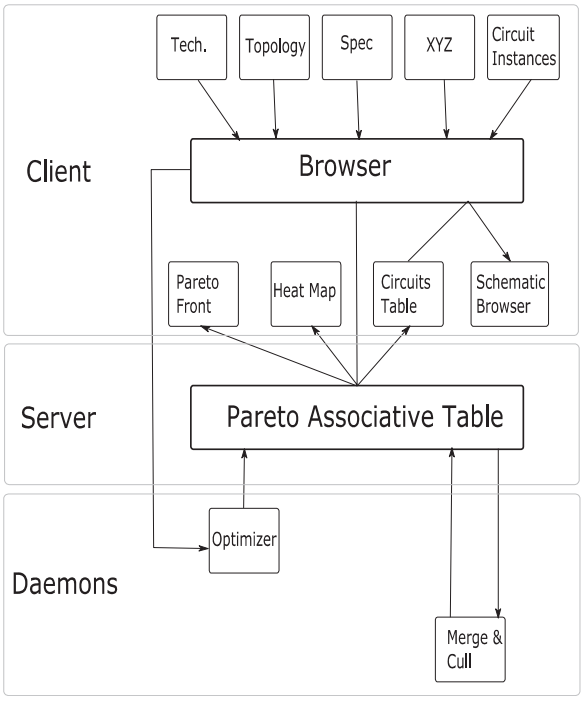

Fig. 2. Gamma online flow.

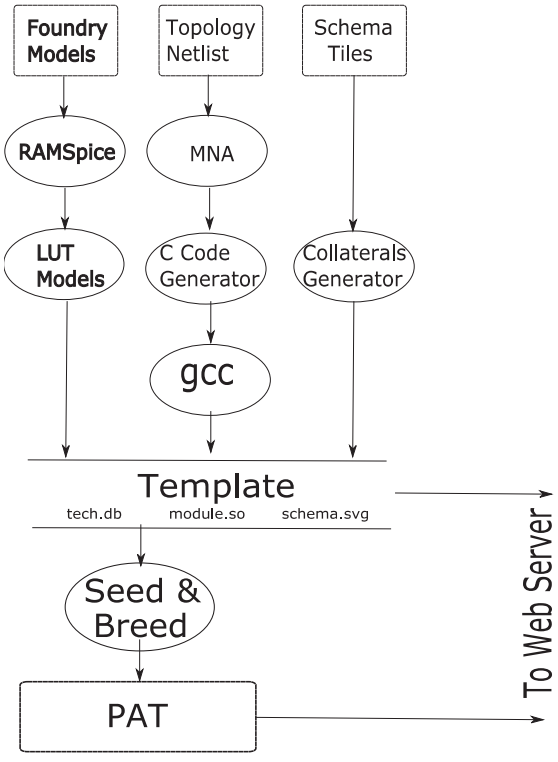

Fig. 3. Gamma offline flow.

The circuit evaluation process has several exit points. Since it is the system's gateway into the PAT, it includes conditions essential for protecting the database from accepting non-functional and

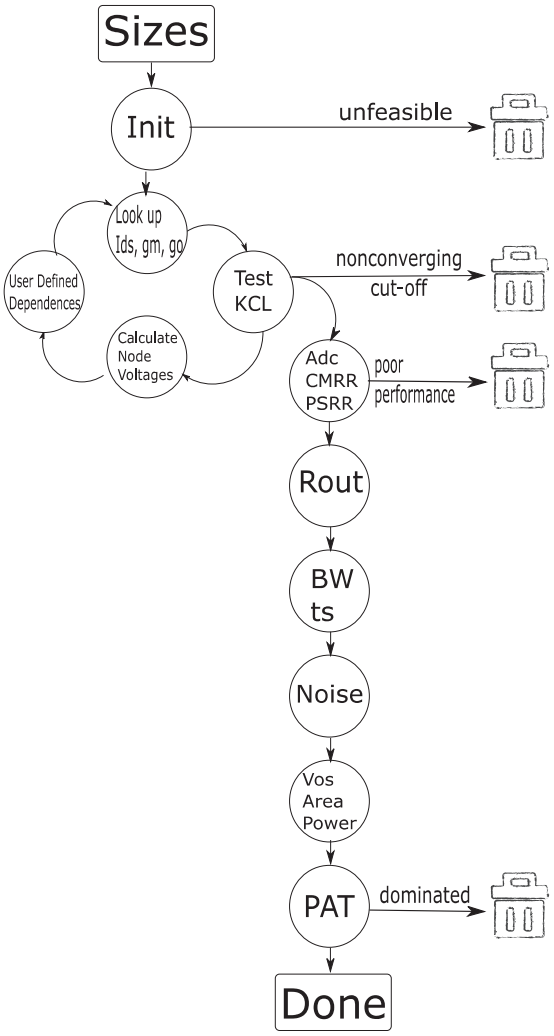

Fig. 4. Shows the dataflow in Γ's main engine ΓCE.

Table 1
Performance parameters calculated by ΓCE.

Parameter	Explanation
A_{DC}	DC Gain
CMRR	Common-mode rejection ratio
PSRR	Power-supply reject ratio
BW	Bandwidth
PM	Phase-margin
t_s	Output settling time to within 5% of steady state
N_T	Input-inferred thermal spot noise
N_f	Input-inferred flicker spot noise at 1 Hz
Total noise	Input-inferred noise, integrated on $2 \times$ BW
f_c	Noise corner frequency
V_{os}	Input offset voltage
Area	Total silicon area
Power	Total power consumption

non-optimal circuits. Non-functional and non-optimal circuits can be of any of the following categories:

1. Circuits violating geometrical rules – Based on manufacturing technology rules and constraints provided with the topology. Such circuit may be a result of random search and should be rejected before OP evaluation begins.
2. Circuits with non-converging DC OP.
3. Circuits containing transistors in either the triode or cut-off regions where they are expected to be in the saturation region. Circuits with extremely poor performance – e.g. negative dB (<1) gain at DC is not useful for amplifiers in any case and such circuit should not continue to be evaluated on other performance aspects.

When a circuit is eligible to be inserted into the PAT, its position regarding the topology's Pareto front is determined by the circuit evaluation process.

D. Circuit compiler

Circuit compiler provides topology analysis and produces the KCL equations and MNA-derived expression for the output as a small-signal function of all circuit input sources – signal inputs, power supplies and noise sources. However, this is the only resemblance between the proposed system and SPICE. The MNA-derived code in this system is precompiled either via an adapted virtual machine compiler for hardware-independence or as generated C code for hardware-efficiency (Fig. 5). This too gives the evaluator a significant advantage over SPICE, as the topology is loaded into the system in the form of pre-processed batch of instructions (be it VM or machine code) that's optimized by a compiler to perform all the needed calculations. SPICE is loaded with topologies as net lists, which are analyzed and inefficiently compiled in run time.

The evaluator is compiled automatically per topology. However, the compilation flow provides opportunity for user manual intervention in the process. The scenario for user manual intervention may include, but not limited to:

1. Identifying static nodes that can be eliminated from DC OP calculation.
2. Separating nodes of independent sub-circuits to a run in preliminary OP calculation loops (e.g. bias circuits).
3. Merging symmetric nodes (due to matching requirements) into a single stepping calculation.

Compiling shared-objects for symbolic simulation is one of the main novel features presented by the Γ system. SPICE loads the circuits it simulates as text files and proceeds to parse and convert them to pointer-linked structures in-memory. When operating online, Γ loads circuits in the form of machine code that already contains all the routines required for evaluating and managing the particular topology. This trade-off of longer offline circuit processing in exchange for much shorter online evaluation fits the response requirement of web-based tools.

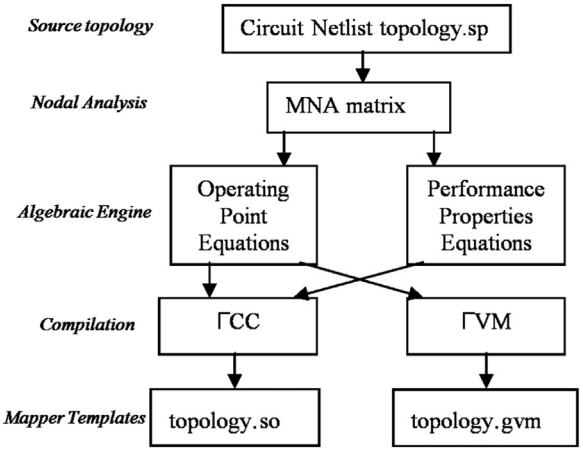

Fig. 5. Topology compilation flows: C and VM.

E. Circuit inventory container and manager – PAT

Pareto fronts of various topologies are kept in the PAT. The PAT has a vector structure, designed to keep the front up to date with every additional circuit. Every inserted circuit is checked against the previously accepted ones in the PAT main vector, to see if it is either dominated by any one of them or dominates some of them. Insertion can therefore result in one of the following:

1. The new circuit is dominated in terms of all its performance parameters by existing circuits in the PAT, and therefore, is rejected,
2. the performance parameters of the new circuit completely dominate some other circuits in the PAT, therefore, the new circuit is accepted and the other dominated circuits in the PAT are deleted from the PAT, and
3. no complete domination is found between the new circuit and the existing members in the PAT, and therefore, the new circuit is accepted into the PAT.

Other PAT operations include:

1. Selection of relevant subset of the existing circuits in the PAT based on a given user specification, and
2. automated culling of PAT entries to make sure they are unique and diverse.

The structure of the PAT is another novel feature of the Γ system. Its rapid circuit-domination checker accelerates extraction of specification-derived Pareto fronts, in comparison with scripted solutions typical to statistical post-processing of simulation results.

F. SaaS-enabled web infrastructure

The front-end of Γ is a website, implemented in Ajax architecture to accelerate transactions and match responsiveness with on-premises tools.

The site is organized in tabs. After the main Analysis tab, one can find other web services, such as forum and chat, report authoring utility and help. A Three-tier service model creates responsive interactivity (Fig. 6):

1. User-browser exchange – Γ Stores all the navigational data it can in the browser's JavaScript workspace. This makes selecting circuits from the presented map, populating the table and updating hover-events on the schematic pane to appear instantly, well under the 0.1 s time limit.
2. Browser-server queries – Γ Generate and updates design tradeoff maps as specification changes. The map generating script loads the PAT, applies the specification and generates the SVG and sometime bitmap graphics for the design tradeoff map in the form of heat-map.

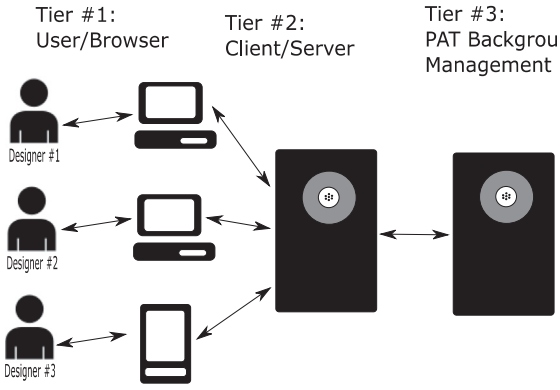

Fig. 6. Web service 3-tier model.

3. Background optimization – Γ Constantly captures user intent via user interactions from the GUI using a background daemon process. Any captured user intent is acted upon by Γ to improve on user-selected circuits. It inserts the improved ones back into a new copy of session-dedicated PAT and displayed in the design tradeoff map automatically.

4. Typical usage models

Γ's user-experience innovations are intended primarily for providing design aid to help users to better navigate complex design choices to meet specifications. Therefore, the primary usage model involves tight interactions between the user and the tool's evaluation engines.

A. Use the tool as a design aid

Fig. 7 shows Γ's main navigation page. The designer navigates the main page in the following order:

1. After login, select a target manufacturing technology and a topology to investigate.
2. A default main page appears, with no specific requirements.
3. The designer can now enter a specification by clicking on the tether icon (Fig. 8) of each relevant performance property and filling in a pop-up entry line.
4. Changing the graph panel's axes designation is done by toggling the XYZ markers from gray to black. When the Z axis is selected it is shown in form of “heat” false color (Fig. 10). Otherwise, a Pareto front is shown (Fig. 7).
5. Either 2D graph showing the corresponding Pareto front or a 2D heat-map showing performance dependence on selected parameters is displayed on the main page. Each marker corresponds to a circuit in the PAT. Intuitively, red ones show circuits that failed the specification and green ones that met. The markers are sensitive to mouse hover and click events. The designer can toggle each circuit in and out of the table pane by clicking on the marker. Chosen circuits are encircled and numbered to match with their line at the table.
6. Each line in the circuits table shown at the top of Fig. 7 corresponds to a circuit the designer found interesting and selected from the graph. Circuits can be removed by clicking on the X button on the left. One of the table circuits can become the pivot, on which heat-maps are centered and the schematic is updated accordingly, by clicking on the magnifying glass on the right.
7. The schematic of the target circuit topology (Fig. 9) is shown on the main page. Users can acquire transistor's size and node's DC voltage information of the chosen pivot circuit by hovering the mouse above nets and components.
8. The field of the heat-map is dotted with red and green cursors (Fig. 10), for spec-failing and meeting circuits, respectively. A designer may want to refocus to one of the other circuits by clicking on its marker. This adds a new line to the circuits' table, without eliminating the previous pivot circuit.
9. Out of the circuits that were added to the table, any circuit can be selected as pivot and navigation can continue via further heat-mapping performance vs selected sizes.
10. After all navigation through the solution space yields a circuit that meets the specification, a designer clicks on the cloud icon on the lower-left corner of the schematic pane. A “save as” pop-up appears and the system generates a ready to simulate SPICE netlist that can be further tweaked by external tools in a sign-off accuracy. The website generates the netlist, complete with the transistor models and stimulus needed to run a SPICE simulation.

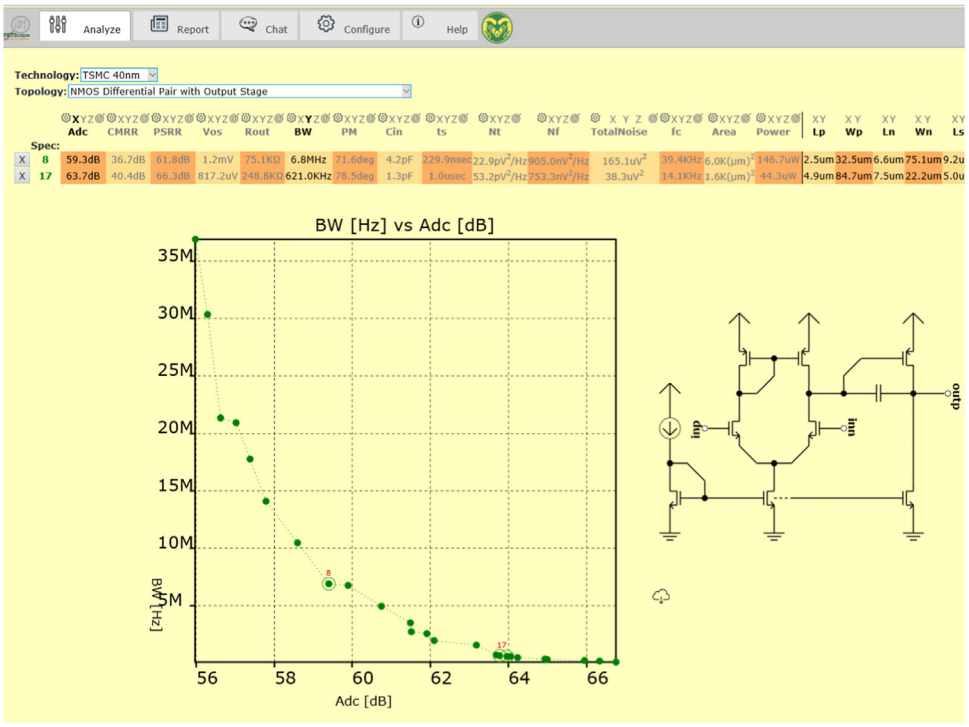

Fig. 7. Screen capture of the Γ-powered website.

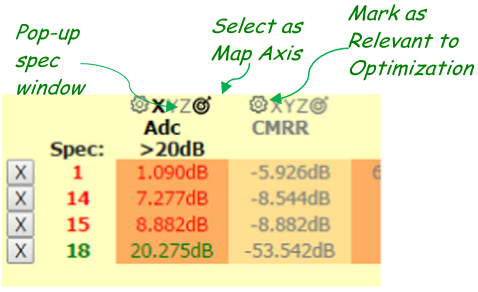

Fig. 8. Zoom-in on leftmost part of the circuits' table.

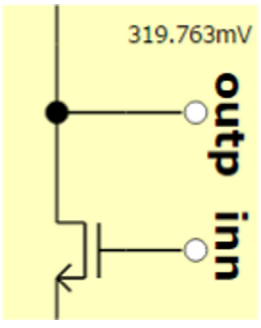

Fig. 9. Output DC level shown on mouse hover event.

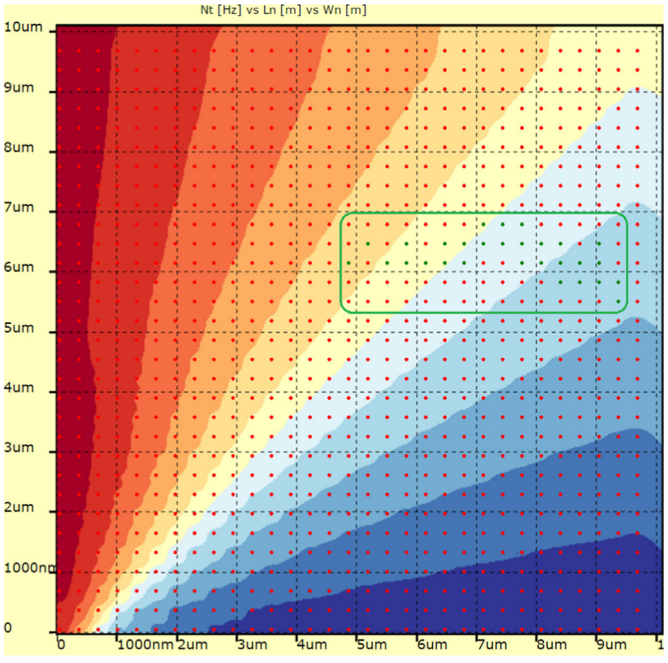

Fig. 10. Heat-map of thermal noise vs transistor size.

The exact channel dimensions and finger multiplier are included with the final netlist file.

5. System implementation

A. Initial compilation of circuit templates

Response time is highly prioritized for this system. Therefore, the system is designed to perform as many tasks as it can for all the circuit templates in offline processes and store the initial results in efficiently queried databases. Initial evaluations of circuit templates include topology analysis, compilation and Pareto front mapping (Fig. 11).

1. Nodal analysis

Γis first used to create topology templates. It contains an algebra engine, a netlist to Modified Nodal Analysis (MNA) matrix. The input is a SPICE-like netlist (See example topology in Fig. 12),

0	0	0	0	0	0	0	1	0	0	inn	0
0	0	0	0	0	0	0	0	0	1	inp	0
0	nin_1:gm	p_1:go +p_1:gm +nin_1:go	0	-nin_1:go -nin_1:gm	0	-p_1:go -p_1:gm	0	0	0	outm	-nin_1:Id -p_1:Id
nin_2:gm	0	p_2:gm	Gds_outp_ 0+p_2:go +nin_2:go	-nin_2:go -nin_2:gm	0	-p_2:go -p_2:gm	0	0	0	outp	-nin_2:Id -p_2:Id
-nin_2:gm	-nin_1:gm	-nin_1:go	-nin_2:go	nin_1:go +nin_1:gm +nin_2:go +nin_2:gm +n_tail:g o	n_tail:gm	0	0	0	0	tail	-tail:Id +nin_2:Id +nin_1:Id
0	0	0	0	0	n_ref:go +n_ref:gm	0	0	0	0	vbias	-n_ref:Id +iref
0	0	-p_1:go -p_1:gm -p_2:gm	-p_2:go	0	0	p_1:go +p_1:gm +p_2:go +p_2:gm	0	1	0	vdd	p_2:Id +p_1:Id -iref
1	0	0	0	0	0	0	0	0	0		inn
0	0	0	0	0	0	1	0	0	0		vdd
0	1	0	0	0	0	0	0	0	0		inp

Fig. 11. MNA matrix generated for the compiled topology in Fig. 12.

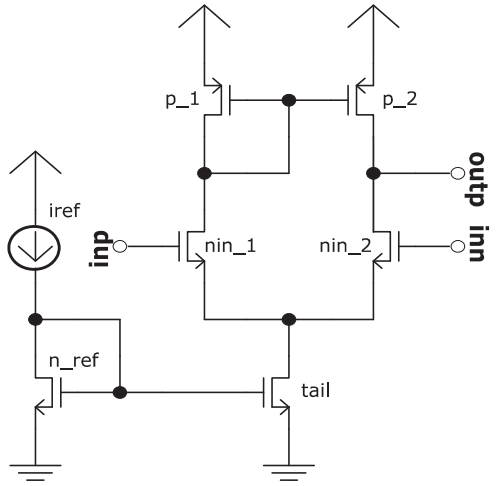

Fig. 12. Differential pair compiled to MNA matrix.

given with variable size parameters and sizing constraints (see example in Fig. 8).

Three MNA matrices are produced automatically and printed out for inspection in HTML format (Fig. 11):

1. DC operating point and small signal matrix.
2. R_{OUT} test matrix.
3. AC transfer function matrix.

2. Algebra solution engine

The matrices are solved symbolically, i.e. they each produce an algebraic equation for the output and other nodes' voltage. These equations can be derived automatically via abstract definitions to generate code for specific properties.

The three matrices produce a set of equations for the following performance properties:

1. Operating point voltages (*OP*): standard equations that are the symbolic $Gv=i$ solution of KCL. Since transistors are nonlinear in their behavior, those equations are put inside a loop for convergence.
2. DC gain (A_{DC}), Common-mode Reject Ratio (CMRR), and Power Supply Rejection Ratio (PSRR): equations based on derivatives of the output w.r.t. the inputs, common-mode and power respectively. Generated from the DC matrix.
3. Output impedance (R_{OUT}) Bandwidth (BW), output settling time (ts), input capacitance, and phase-margin (PM): equations generated from analysis of the transfer function. Generated from the AC matrix.
4. Noise parameters and corner frequency (fc): parameters generated by looking up noise contributions per transistor and converted to inferred-input noise.

In addition to the MNA-derived equations, other performance parameters are added:

5. Input offset voltage (V_{OS}): parameter estimated with parametrized equations based on foundry statistics.
6. Power consumption: estimated as $\Sigma I_{DS}V_{DS}$.
7. Total silicon area: estimated based on gates' lengths and widths and additional parameters taken from the foundry design rules.

Fig. 12 shows a basic topology that's compiled to an MNA matrix (Fig. 11) and then to C code. The circuit's 7 nodes and 3 sources (V_{DD} and two inputs) create a 10×10 sparse matrix and a 10 current-sources vector.

3. Internal circuit auto-tuning

The circuit evaluator is designed to process all sizing dimensions into performance metrics. However, for larger circuits, some of the input sizes cannot be left up to a random search, because they are meant to serve a specific performance target that is normally controlled manually.

For example: an op-amp has a bias voltage to keep the DC level of the output at $½V_{DD}$. Left to random search and filtering, this biasing voltage complicates the design optimization and adds a redundant dimension to an otherwise simple evaluation. To incorporate this constraint, the evaluator is programmed to set V_{OUT} to $½V_{DD}$ by default by specifying this constraint in the netlist. The evaluator tunes the bias voltages during the operating point convergence loop. The OP loop is designed not to stop until both KCL and auto-tune conditions are met, unless the number of iterations so far suggests non-convergence.

The designer need not worry about this bias voltage being correctly set, the output being any other DC voltage (non-converging circuits would not register in PAT anyway) and the circuit's design dimensionality stays in check. In a SPICE-based design flow, designers either search for a right bias manually or by a scripted binary search, because there is no way to intervene and tune sizes to fit constraints mid operating-point analysis. The time it takes to

achieve a circuit that is ready to be evaluated against the spec is therefore much longer than Γ's operating-point loop, even with the extra iterations taken to achieve both KCL and auto-tune equilibrium. Other dependences can be defined between transistors' widths and lengths and final performance properties.

Another type of internal dependences is defined directly between size parameters of the topology's netlist. A trivial example for that is matching pairs of transistors: differential pairs, current-mirrors, etc. are matched by simply sharing the same sizing parameters. The designer can also program some transistor sizes to be expressions that are automatically calculated from other transistors' width and lengths. This again reduces the number of search-space dimensions and improves Γ's ability to find valid solutions.

4. Final compilation

The algebraic representation of the topology can be converted to programs using one of the two compilers:

1. ΓVM – "Gamma Virtual Machine" an interpreter tailored for the Γ system, with flexibility and hardware independence for cross platform execution.
2. ΓCC – "Gamma Circuit Compiler", uses generic GCC to produce. so (alternatively,.dll on Windows) file.

The virtual machine is a fast byte code interpreter that mixes stack-machine and custom data structures, such as polynomials and rational functions. It has specific instructions for accessing Γ-specific data structures, such as LUT queries and PAT insertions. The main motivation behind that option is uncertainty about the target machine that runs the evaluator itself. It is possible that topology preparation may be done by a 3rd party organization, such as IP company, who would like to keep the internals of the circuit to itself, but still want the template produced to be available to run on a public Γ website. Using the virtual machine option, the evaluator is still running in reasonable speed and independently from the equipment that created it.

The second choice is the faster one and the one recommended for united site operation and topology preparation model.

It takes ~30 s to compile a differential-pair circuit from netlist to. so, which is done once and offline.

The first step divides the determinants of the minor corresponding to the output and the MNA matrix. "Ted" is the inverse of the main determinant, which is independent of any voltage source or inputs, and therefore can be calculated once and then be excluded from voltage derivations.

The equation is sent to GCC with all the necessary optimization flags. In the ΓVM flow, the equations go through standardized representation, where redundant operations are eliminated and some common sub expression elimination takes place.

Admittance values used in this equation are looked-up during the OP loop in the fast transistor-level models.

The final stage of compilation creates a shared-object module from C code (or the equivalent ΓVM assembly). The API it provides Γ is common to all topologies. However, initial values and sizing/performance vectors are topology-dependent. The compiler prepares all the collateral database binaries that go with the compiled shared object. These two binaries, together with the topology netlist and schematic representation, form a topology + technology template that can be used for the next step – Initial mapping of the Pareto front of the topology's performance envelope.

B. Pareto archive – PAT

The PAT plays a role in both offline and online operations. In the offline phase, it is pre-populated with a set of sized circuits that can be used as a starting point for charting tradeoff curves and jump-start optimization algorithms.

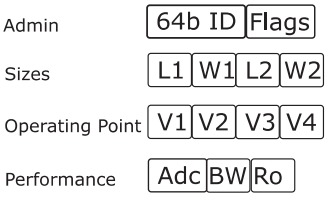

Fig. 13. PAT entry structure.

1. PAT entry

Each PAT entry (Fig. 13) keeps the following data:

1. Unique 64b identification number and scratch flags for marking dominated circuits.
2. Design choices in the circuit: geometric parameters, reference current, bias voltage and loads.
3. Operating point voltages for saving future run time.
4. Performance parameters associated with the circuit.

The PAT and the LUTs, are part of Γ's hierarchical database. This database is stored as raw binary sequence on file, which together with the shared-object output of the circuit compiler forms the topology template.

2. Offline PAT populating

Once a circuit template and its associated topology is compiled, Γ will populate a PAT with a general Pareto front with samples from the topology's sizing space. The goal of this stage is to find circuits that span a wide range of the performance parameters of the given circuit topology, which makes them markers of the Pareto front of size vs performance. The process of populating the PAT consists of two stages (Fig. 14):

1. Seeding, the PAT populating loop draws random sizers, based on specified min, max and distribution and creates a set of circuit samples. The circuits are then tested for viability. A viable circuit is one that meets all the criteria outlined in Section 3.C. Viable circuits are then inserted into the PAT. The retention rate of circuit samples by the PAT drops over time (Fig. 15), which is an indication that the front is getting saturated and is used as the stopping criteria for the first stage.
2. Breeding, the PAT is populated with circuits that are random augmentations of circuits that are already retained in the front. The retention rate at this point climbs up, because there are more chances of finding a non-dominated and non-dominant circuit in the neighborhood of a member of the front than elsewhere in the size space.

3. Extracting pareto fronts

The PAT does not pre-assign circuit performance properties to potential "constraints" or "objectives", thus allowing users to switch between dual problems, e.g. "what minimal area should be expected from $A_{DC} >= 20$ dB?" Or "What's' the maximal A_{DC} that can be expected for area $<= 10$ um^2?" The PAT does discriminate between "more is better" (e.g. BW) and "less is better" (e.g. thermal noise) properties.

The first step toward a solution is extracting archived solutions that are relevant to a given specification. This is done by collapsing the pre-calculated Pareto according to simple set of dominance

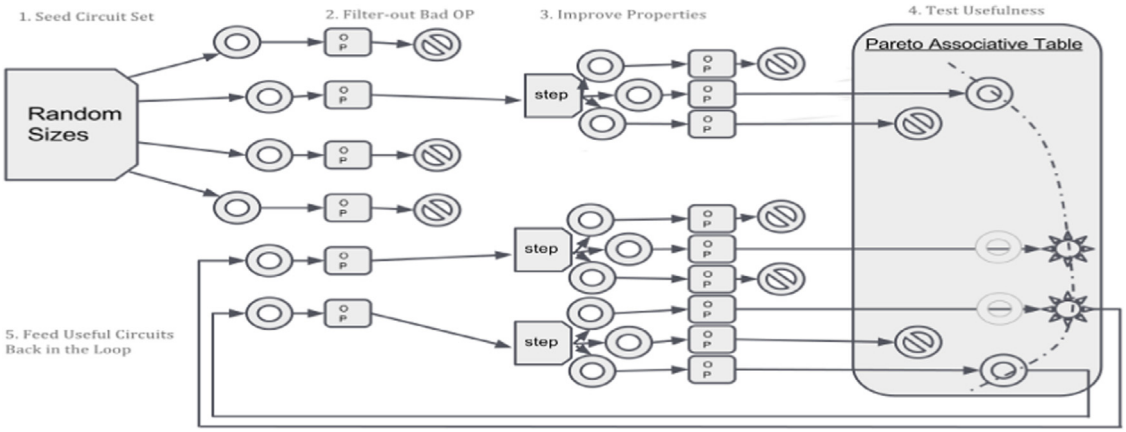

Fig. 14. PAT seed and breed phases.

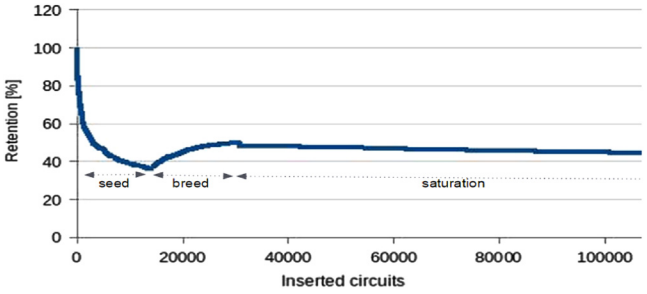

Fig. 15. PAT populating decreasing retention rate.

rules. A specification may assign one of the following types of values to each one of the circuit's performance parameters:

1. Donot care – Ignore this parameter for dominance consideration.
2. Best – Consider this parameter value for dominance.
3. Inequality – use this parameter value as-is, unless it is greater than the threshold in the spec, in which case take the threshold in its stead.
4. Equality – Actual value for dominance consideration is the distance between the parameter value and one in the specification. – only used in operating-point and constant sizer requirements.

When a specification is applied to a Pareto front, previously co-existing circuits can now be dominating/dominated, because ones advantage over the other may be assigned “do not care”. Another possibility of elimination is that they both exceed an inequality, which means they are equal under the specification, leaving them with only disadvantages.

Table 2 shows an example of applying a specification to the PAT, resulting in two circuits being eliminated:

1. Circuit 1 is better than 2 in general, although it pays for its improved gain, common-mode rejection and bandwidth in area. When the spec sets the requirement to 30 dB and 2 MHz, both circuits’ surplus gain and BW are ignored and circuit 2's lesser area makes it dominant over 1.
2. Circuit 4 has better common-mode rejection compared to circuit 3, which is the justification for its inclusion in the PAT in general. However, this performance aspect is outside of the spec. In other words: specifically irrelevant. All that's left are disadvantages compared to circuit 3, which cause circuit 4's elimination.

The specific-front is therefore a subset of the general front. It may contain circuits that do not meet the spec's inequality levels,

Table 2
Circuit grading and eliminating w.r.t. a specification.

	Adc (dB)	CMRR (dB)	BW (MHz)	Area (μm^2)	
Spec:	*> 30*	*do not care*	*> 2*	*best*	
	35	45	4	20	Dom by 2
Circ2	32	40	3	15	Met spec
Circ3	25	50	2	5	Trade-off
	20	60	1.5	15	Dom by 3

but are useful for showing how loosening the requirements on one property can yield better results on another. In the example in Table 2, DC gain can be traded for area. It is useful, because a small sizing change to it may bring it up to the specified gain at a still smaller area of circuit 2. In optimization theory terms: it may not be a feasible solution, yet closer to the optimum than a feasible one. In case there is more than one “Best” value, which translates to multi-objective optimization, there may be more than one circuit on the front that meets the thresholds. The algorithm for re-calculating a sub-front is $O(PN^2)$ P being the number of properties and N number of circuits, which can be slow for PAT's of N in the magnitude of 10^5 and P in the magnitude of 10^1. To overcome that, a loose fitness function is first applied to filter out all but $N=10^3$ circuits. This non-pure step actually assigns weights to the properties and thus contradicts the multi-objective concept behind Pareto. However, the circuits left by this filtration still form well populated fronts, because the circuits that are filtered out are the least fit to the spec. The filter's target size can be modified if it is too aggressive for some specs.

4. Culling

Another PAT operation, designed to keep the circuit set diversified, removes circuits with duplicate performance with a predefined tolerated distance. Culling (Fig. 16) is necessary for community-wide deployment to make sure the PAT's do not grow unchecked (Fig. 17).

The culling operation detailed in Fig. 16 has the time complexity of $O(PN^2)$. However, because this algorithm runs in the more relaxed offline phase, the impact of its run-time complexity on tool's on-line performance is negligible.

5. PAT online content adaptation

Since every online evaluation of a viable circuit ends with an insertion to the PAT, the presence of more visited circuits becomes more and more dominant with time. The least visited circuits have more chances of getting culled. Therefore, over

```
Pi,j - property i of circuit j
S - Similarity factor (smaller->aggressive)
# Init thresholds
Foreach i {
    Pi,min=FLT_MAX
    Pi,max=-FLT_MAX
    Foreach j {
        If (Pi,j<Pi,min) Pi,min=Pi,j
        If (Pi,j>Pi,max) Pi,max=Pi,j
    }
    Pi,th=( Pi,max- Pi,min)/S
}
# Detect and eliminate similarities
For (i=0;i<|circuits|;i++) {
    Similar=1;
    For (k=i+1;k<|circuits|;k++) {
       Foreach j: If (abs(Pi,j-Pk,j)>Pj,th) Similar=0;
       If (Similar) eliminate(i);
    }
}
```

Fig. 16. Pseudo code for culling cycle.

time, the PAT gives preference to the more popular specs and spends fewer resources on less needed requirements.

6. PAT size requirements and limitations

The pre-populated PAT is necessary to give an initial indication of a topology's performance limitations. Without a pre-populated PAT, online generation of Pareto fronts would take hours. The lower-limit of a PAT size was observed to be in the 10 K sized circuits, provided they are diverse enough to capture the performance surface evenly. The upper limit is determined by the number of circuits inserted to the PAT in a single transaction and storage limitations. At measured average insertion time of 300nsec per already-stored circuit (see. Experimental results D.2.ii), a transaction of 1 K circuits to take place in ~10 s is in the magnitude of 1 M circuits, which occupy 10 MG of disk space, a reasonable size per topology.

C. Transistor-level characterization and modeling

As described in the previous paper [17], transistor-level models are produced from BSIM with a modified NGSPICE simulator that populates LUTs. Table 3 lists the transistor parameters modeled by the LUTs, the size of the LUT for each parameter, and the expected access time for each LUT. Generally, the LUT includes a pointer to a lookup function that performs both entry (hypercube) location and specific interpolation operation. For evaluating a circuit's operating point, I_{DS} needs to be more accurate than the rest of the parameters, because small amount of errors in I_{DS} can translate to significant shifts in V_{DS} and thus nodes' DC levels. To achieve higher accuracy, a composite lookup algorithm was developed. This method looks up g_m and g_o of a transistor first and then uses their values as additional slope information of I_{DS} along V_{GS} and V_{DS} respectively. The result is both faster look up, since entry location is done once for all three tables, and better accuracy, since it adds a 2nd order approximation to the linear one. Fig. 17 shows a segment from the V_{DS}/I_{DS} curve as it is plotted from a SPICE simulation and a Lagrange lookup of the Γ model. The differences in I_{DS} and V_{DS} between SPICE and the lookup line are in the order of 100 nA and 700 μV, respectively. Fig. 18 zooms-in on the 50 mV region and shows how closer the composite interpolation lookup (green) gets to the SPICE curve (black), in comparison with the linear one (red). V_{DS} error drops from 700 μV to under 50 μV.

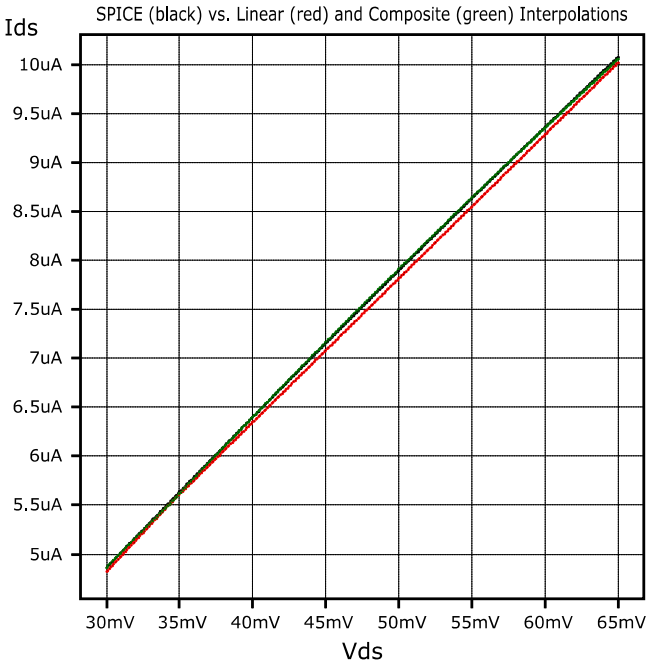

Fig. 17. SPICE vs LUT modeled current.

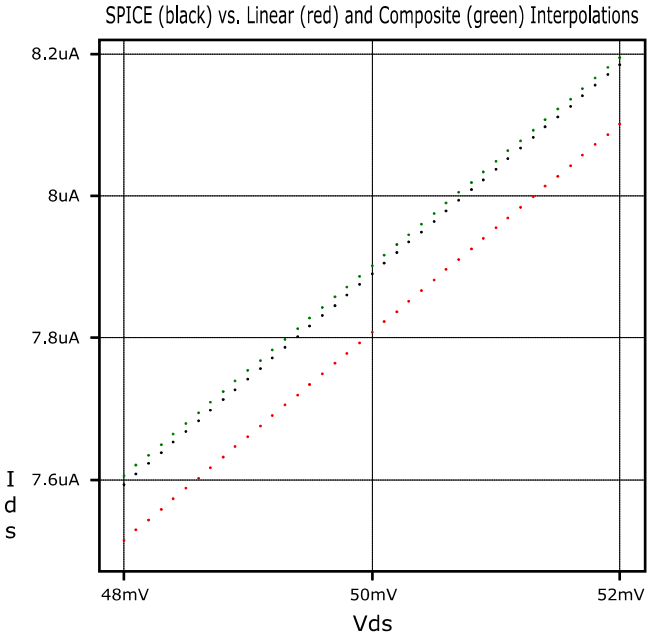

Fig. 18. Zoom-in on 50 mV region. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Table 3
Transistor-level modeled parameters.

Characterized parameter	Table resolution	Approximate access time
g_m, r_o, I_{DS}	2.5 M Samples	450 nsec
V_T, V_A	10 K Samples	300 nsec
N_T, N_f	6.5 K Samples	1.4 μsec
C_{GS}, C_{GD}	6.5 K Samples	1.4 μsec
I_{DSmis}, V_{OS}	257 Samples	120 nsec

6. Experimental result

A. Tested topologies

For the purpose of collecting experimental results, four topologies were compiled to Γ templates.

The topologies in Fig. 19 represent progressing complexity in circuit analysis and evaluation. A single transistor common-source amplifier (Fig. 19.1) can be analyzed by simply applying transistor-level models. However, Γ analyzes its behavior in a circuit-level context by going through MNA, code generation and compilation and finally, PAT population. The single transistor circuit provides an initial pipe-cleaning run to the offline flow. A first multi-transistor circuit with differential input (Fig. 19.2) tests Γ's ability to find the circuit's operating point voltages accurately. A differential-pair circuit has four independent nodes. The outp, outm and tail nodes are connected via transistor channels, an arrangement that proves Γ's ability to solve circuits beyond simple voltage dividers. This topology is limited in both performance and number of sizing parameters. To show Γ's scalability to a more useful circuit, with additional sizing parameters, an output stage is added to the differential pair (Fig. 19.3).

Finally, a folded 2-stage differential amplifier is used to test Γ's ability to optimize a more complex amplifier topology (Fig. 19.4).

Unless otherwise noted, the folded, 2-stage differential amplifier (Fig. 19.4) was chosen to illustrate Γ's performance.

B. Accuracy analysis

1. Procedure

The accuracy gold standard is NGSPICE result. Transistor models were translated from a commercial 40 nm CMOS process release to Cadence® Specter® and two simulators (Specter® and NGSPICE) were accuracy-matched in the previous effort to generate Γ's LUT models. Γ's website contains a "download from cloud" button, which converts the latest pivot circuit to NGSPICE-ready, downloadable netlist file. This conversion includes locating the exact model and bin of each participating transistor and automatically partitioning the channels to fingers. The netlist file is self-contained, with all the BSIM models needed for simulation and commented tags for automating post-processing of simulation results.

NGSPICE simulates 3 domains to be matched with Γ evaluations:

1. Operating point voltages (OP).
2. Low frequency small signal gains – Differential and Common mode gain are measured in.dc analysis and compared to the figures stored in the PAT. This stage gives an indication of impact of OP accuracy on equation-level analysis.
3. AC performance – The output amplitude and phase is simulated and matched against Γ's bandwidth analysis.

2. Results

Fig. 20 shows the histogram of difference between the SPICE DC simulation and Γ's operating point results on the amplifier's

EXPERIMENTAL RESULT

A. *Tested Topologies*
For the purpose of collecting experimental results, four topologies were compiled to Γ templates.

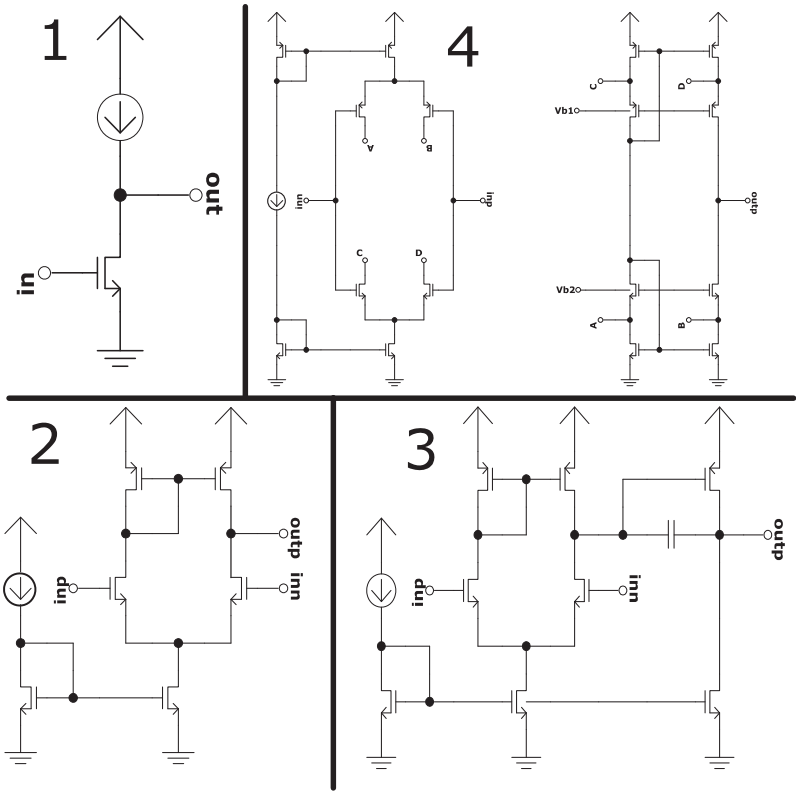

Fig. 19. Topologies implemented as Γ modules for testing the system's performance.

output node. Γ achieved an 3-σ error for operating point calculation within 200 μV of average.

Fig. 21 shows the correlation between the SPICE DC gain simulation results and Γ's results on the amplifier (Fig. 19.4). 95% of Γ's A_{DC} error is within ± 0.5 dB, which is $\sim$6% linear error.

Fig. 22 shows the correlation between the SPICE BW simulation results and Γ's results on the amplifier (Fig. 19.4). Γ follows SPICE closely with typical 5.5% under-estimation and up to 3% over-estimation of the BW.

Fig. 23 shows the correlation between the SPICE phase margin simulation results and Γ's results on the amplifier (Fig. 19.4). The results show around 5° error margin.

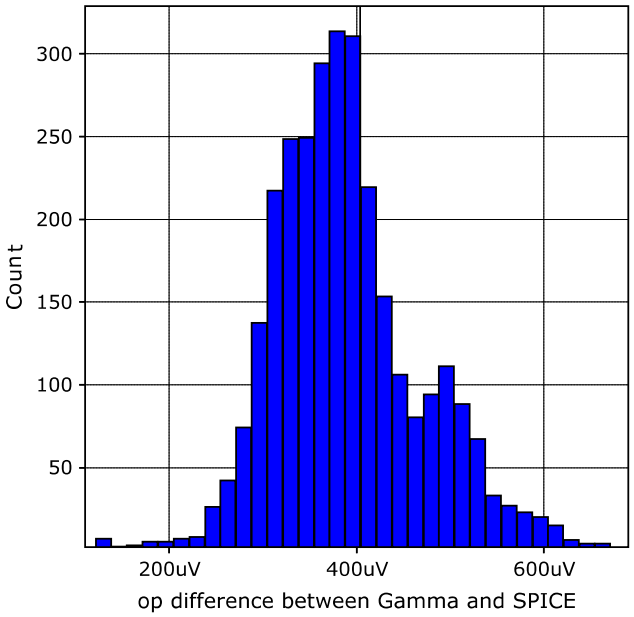

Fig. 20. Γ vs SPICE Op. amplifier output DC level.

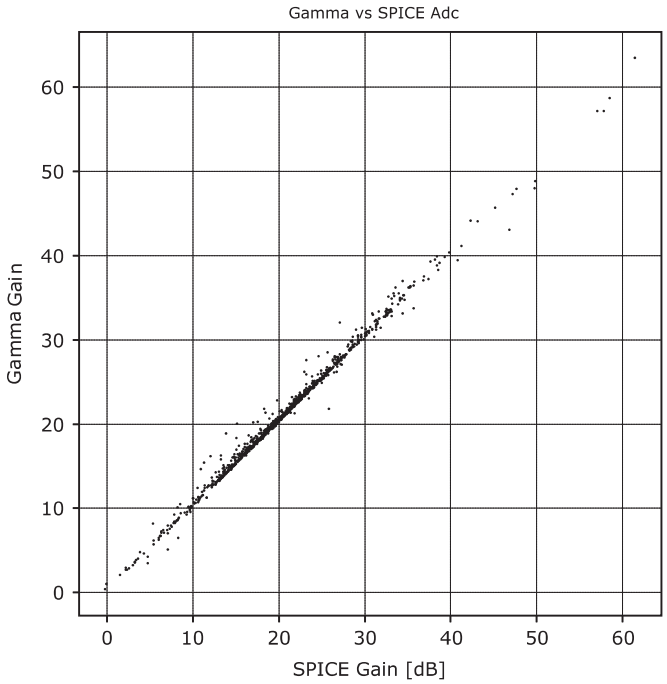

Fig. 21. Γ vs SPICE in estimating DC gain.

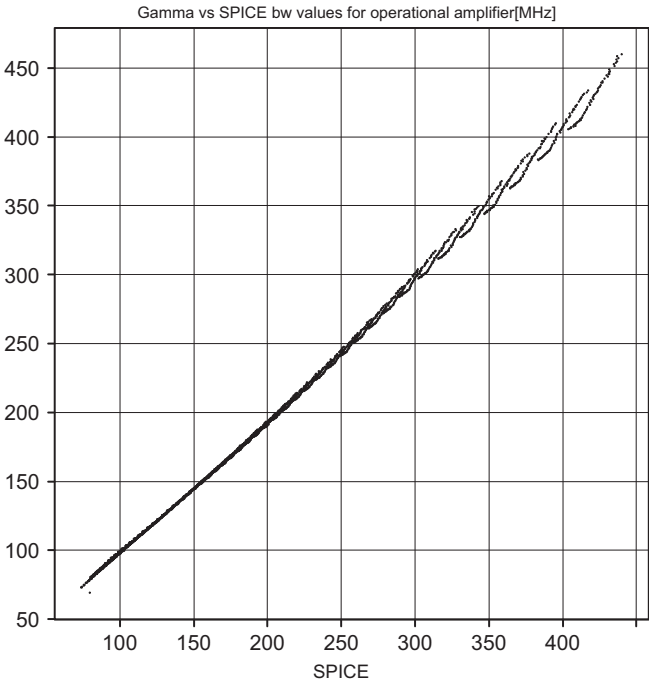

Fig. 22. Γ vs SPICE bandwidth analysis.

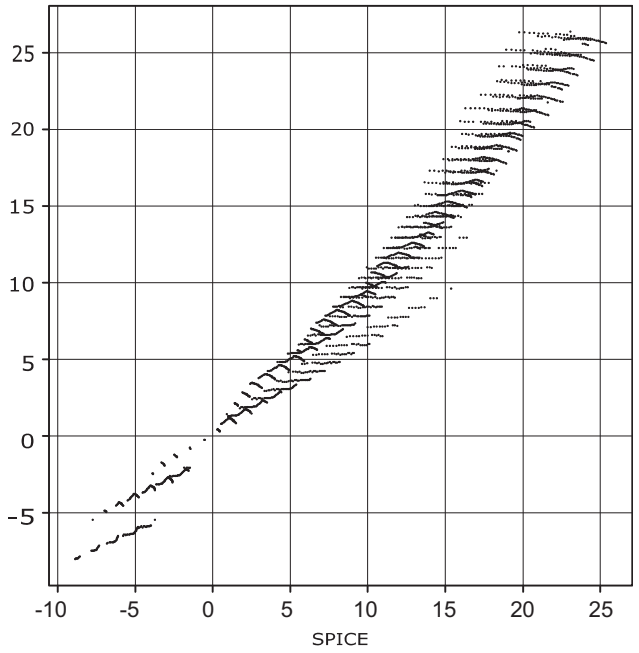

Fig. 23. Γ vs SPICE PM detection.

C. Execution time

1. Offline activities

The first offline task regarding a topology is compilation. This requires nodal-analysis, algebraic manipulation and generation of performance-property equations. Typical compilation time ranges from a few tens of seconds to a few hundreds of seconds depending on circuit complexity.

The next offline task is to populate an initial PAT with $\sim 10^5$ circuits that span the performance front of the topology.

Table 4
Run time of ΓCE and NGSPICE.op [ms].

Tool	Min	Max	Med	Average	σ
SPICE	23.8	387.8	52.8	108	104
Γ	4.5	14	4.9	5.7	1.8

Typically, populating PAT takes between a minute to tens of minutes depending on the population size.

Once a PAT is initially populated in the server's data area, the topology can be mapped and further resolution can be added to performance corners of interest. Overall offline activities per topology, including preparation of netlist and schematic representation, can be finished in about an hour and result in about 100MB of data per topology+technology pair.

2. Circuit evaluation time

One of the key innovations of the proposed design system is rapid evaluation of circuit performance without sacrifice in accuracy. Table 4 illustrates the run time difference between Γ and SPICE on determining the operating point for the folded cascod amplifier circuit collected over 1000 circuit evaluations. The "Min" and "Max" numbers show the shortest and longest evaluation times respectively. "Med" is the median evaluation time. It is clear that Γ is significantly faster than SPICE while maintaining the SPICE level accuracy as it is demonstrated by the results in Fig. 20.

SPICE does not have built-in commands for evaluating all listed performance metrics for a given circuit, but rather has a number of simulation commands that can lead to scripted or manual post-processing. Therefore, the.op command was chosen for comparison, as it has the closest parallel in Γ, the Circuit Evaluator. Γ_{CE} includes operating-point loop, but also calculations for all the performance parameters. It completes all the steps needed to evaluate a circuit in 1/10 of the typical time SPICE takes to just analyze the operating point.

Both execution times for SPICE and Γ were measured on the server-side machine, without any network effects. In the online SaaS environment, the actual response time may be affected by network traffic. However, the significant performance advantage by Γ will allow Γ's server to handle many client requests to improve the overall response time for remote users.

3. PAT insertion time

The time complexity for inserting a point into the PAT is O(N), with *N* being the number of circuits already in the PAT. This is because every inserted point can potentially be compared with all stored ones before it is retained. However, every insertion has a probability to be discarded as dominated before it visits all the stored points. Therefore a cone-shaped scatter-plot of insertion time vs size is expected. Fig. 24 shows the insertion time as a function of PAT population size.

Fig. 25 shows the histogram of PAT insertion time, after it is normalized per number of points retained in the PAT prior to the insertion. This gives the distribution of the slope per point in Fig. 25. The average insertion time is therefore ~300 nsec (marked by wide vertical line in Fig. 25) per pre-stored circuit. The bimodal distribution of the PAT insertion time can be attributed to two classes of operations: one without requiring prolonged evaluations for eligibility, and the other involving a series of computations for eligibility, including operations to delete other dominated entries in the PAT.

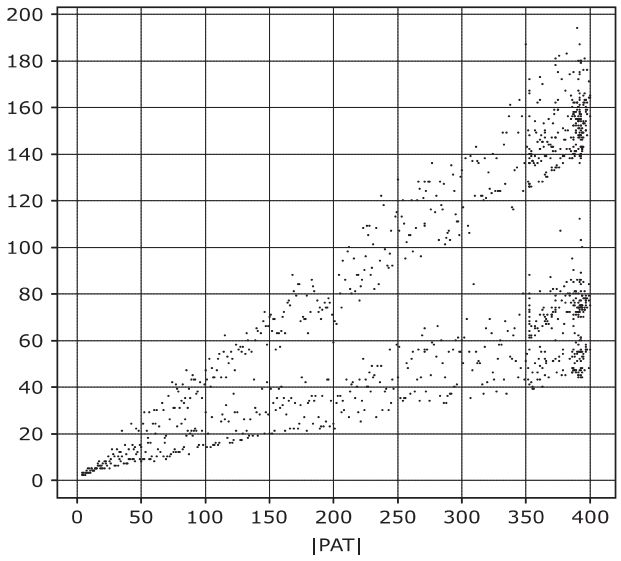

Fig. 24. PAT Insertion time per size.

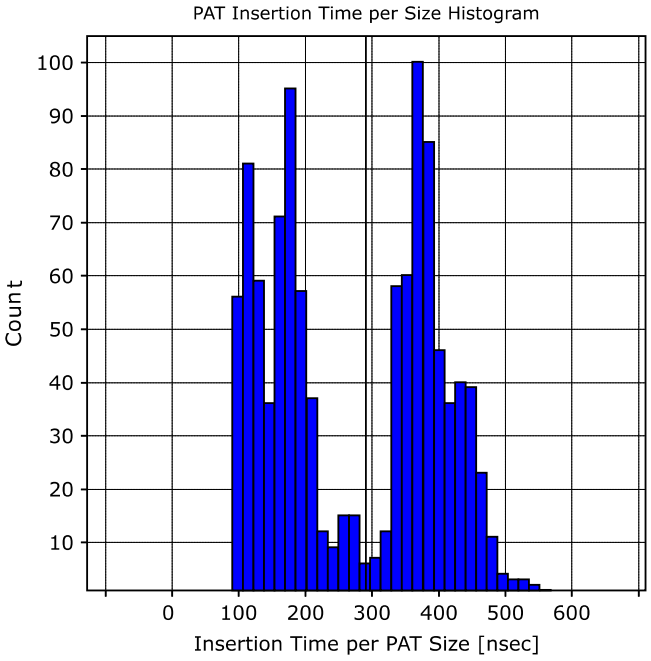

Fig. 25. Ratio between PAT Insertion time and size.

D. Example of using Γ' for circuit sizing

To demonstrate the tool's usage, a flow that may be used by designers to optimize the circuit size (Fig. 19.3) for a given specification is shown here. The specification for the circuit is:

Primary specification:

1. Technology: 40 nm
2. DC Gain >60 dB
3. Bandwidth $>$ 5 MHz
4. Total noise $<100\ \mu V^2$
5. Main optimization objective: Area

Secondary specification:

1. CMRR >30 dB
2. PSRR >50 dB
3. PM $>60^\circ$

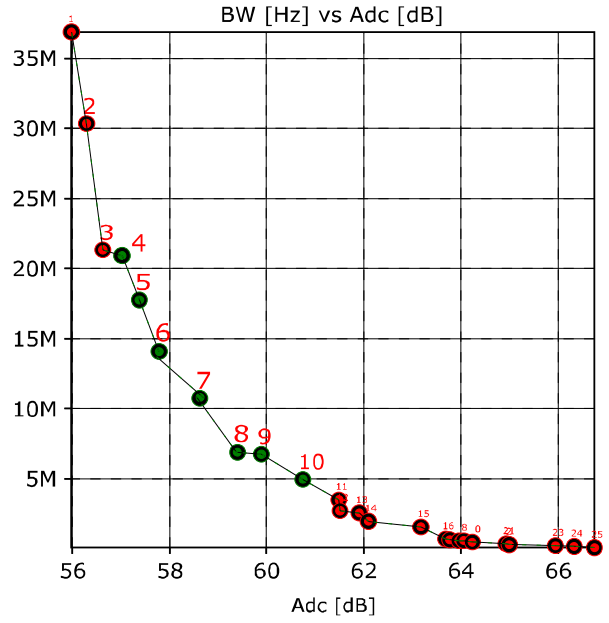

Fig. 26. PAT circuits that meet gain and BW.

4. Corner Frequency < 30 kHz

1. Select topology and enter specification

After login, the specified technology and topology are selected from the drop menus. A default A_{DC}/BW front appears and the thresholds are typed in. To allow trade-off, a slightly relaxed specification values are specified: 57 dB and 4.5 MHz for gain and BW respectively, while the noise parameters are waived for the moment, so we can start the search from a close enough circuit that can be improved to meet the whole specification. The updated perato front from Γ shows existing circuits already in the PAT that meet the relaxed specification (Fig. 26).

Circuits 9 and 10 of that front are selected. Each of them violates the original specification. Circuit 10 meets gain requirement, but fails on BW and vice-versa for Circuit 9. Both circuits violate the noise requirement.

However, circuit 9 has significant BW slack (6.7 MHz), which can be traded more easily for gain and noise and therefore it is chosen to be the pivot circuit for the next search step.

2. Pick an initial circuit

Circuit 9 has the following performance metrics:
DC Gain = 59.8 dB
BW = 6.7 MHz
Total noise = 186 μV^2

3. Search for gain-valid circuit

Increasing the gain of Circuit 9 can be done by upsizing the input transistors. The search for better gain is done by plotting a tradeoff map of the gain with respect to changes in Ln and Wn of the input transistors (Fig. 27).

To increase the gain and reduce noise, while keeping the BW, the next circuit is selected from the white dots[1], in the center of the deepest red area and as wide channel as possible (see marker 930 in Fig. 27). The BW only drops to 6.3 MHz and the total noise drops to 177 μV^2.

[1] The original heat maps' green markers were accented in white for print quality.

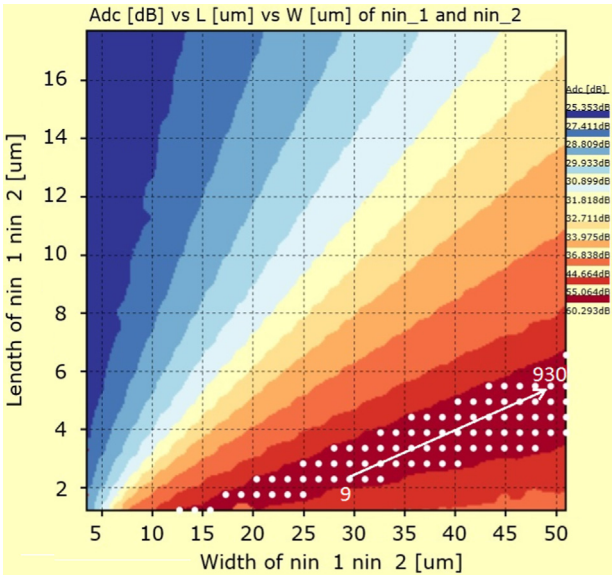

Fig. 27. DC Gain vs input transistor length and width.

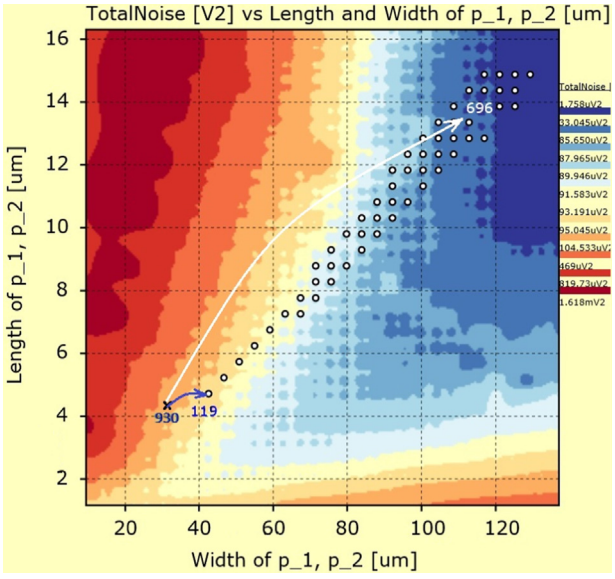

Fig. 28. Circuit 696 trades BW for less noise.

Table 5
Breakdown of noise contribution per transistor.

Transistors	% Noise contribution (each) (%)
p_1, p_2	19.7
nin_1, nin_2	15.1
n_tail, n_ref	2.2
n_out	1.1
p_out	24.9

4. Reduce noise to meet spec

The next step is to reduce the noise to the specified level of 100 μV^2. Another tradeoff map centered on Circuit 930 is generated using the current-mirror p-channel transistors' sizes to trade BW with noise (Fig. 28). The initial circuit showed a relative noise contribution of ~40% from the two p-channel transistors (Table 5), which makes them good candidates for noise reduction.

Table 6
Γ and virtuoso final performance metrics.

Property	Γ	Virtuoso	Rel. Err. (%)
DC Gain	60.2 dB	60.8 dB	1
BW	5.1 MHz	5.5 MHz	7.8
Total Noise	94.9 μV^2	104 μV^2	9.4
Corner Freq.	25 kHz	27 kHz	8
PM	83°	81°	3
CMRR	39.8 dB	40 dB	2.3
PSRR	59.1 dB	59.8 dB	8
Est. Area	8100 μm^2		

Table 7
Search path summary.

Circuit #	Gain (dB)	BW (MHz)	Noise (μV^2)
9	59.8	6.7	186
930	60.4	6.3	177
696	60.2	5	86
119	60.2	5.1	94.9

Circuit from the “cooler” parts of the heat map, represented by Circuit 696 in Fig. 28, lose all the BW slack and meets the noise requirements at 86μV^2. However, they have significant penalty on total area. The noise slack of 14 μV^2 means that the circuit is over designed and a better solution can be more area-optimal. Circuit 119 (in blue, Fig. 28) is a better solution, since it is in the color region corresponding to the required noise and closer to the origin, which means it has smaller area. Circuit 119 is the final circuit candidate, with 94.9 μV^2 total noise.

The final circuit (Circuit 119) was simulated on Cadence® Virtuoso® and the following results were collected and compared with Γ's (Table 6):

Through a set of steps (Table 7), the results here show that Γ provides designers with a powerful, and yet simple, process to converge to an optimal design solution for a given specification. Further fine tuning may be necessary after verifying Γ's final results with SPICE.

7. Conclusions and further discussion

This paper presented a tool, Γ, for analog circuit designs. Using the novel LUT modeling techniques described in this paper as well as our related paper [17], Γ combines the speed of symbolic evaluation of circuits in consideration and the accuracy of SPICE-class circuit level simulations. The LUT approach to model complex transistor behaviors played an essential role to achieve SPICE-like accuracy and evaluation speed that is orders of magnitude better than SPICE.

Architecturally, the proposed tool tightly integrates the LUT functions with foundation with novel system level functions to allow further significantly better accuracy/speed tradeoff and allows faster design convergence with designer's intent. More specifically, the novel system architectural features include:

- Careful division of computation efforts is made between offline and online flows. All data that can be made “ready to serve” is calculated in an offline schedule and the results are kept in an efficient database for direct online mining.
- Single executable tool implements all online and offline steps, from characterization of transistor models to circuit compilation and evaluations.
- Circuit representation in the form of symbolic analysis, compiled to machine code executables, instead of the common SPICE netlist/schematics, accelerates evaluation by providing the specific steps, such as direct calculation of gain from transistors' operating parameters, for evaluating a circuit.
- Management techniques for Pareto front accelerate storage and retrieval of pre-calculated circuits and provide an infrastructure for rapid extraction of circuit design parameters for users during the design convergence process.

With a detailed design example for design convergence, Γ successfully incorporates user inputs at key junctures of the design process to better capture designer's intent and to improve design convergence. This is in direct contrast to existing design approaches where tools attempted to take designs from their specs to the final realization without much designer interaction.

Γ's novel features in speed and accuracy also lend itself to providing the SaaS features through its web functions. The breadth analysis strategy through archived as well as online Pareto candidates of viable circuits is well suited for SaaS applications, because the entire design process can be broken into a series of short queries and much of the computation effort can be archived in a shared database that serves all present and future users and sessions. Enabling of SaaS features for Γ is a significant step towards providing faster and updated design tool delivery at very low cost. With increasing recognition of the power of cloud computing, Γ certainly moves the delivery, maintenance, quality, and usability of future analog design tools in the right direction.

Γ currently makes no attempt to reduce the amount of slacks among any performance parameters in the solutions, because the process of reducing slacks of any performance parameters often results in worsening of other performance parameters in often tightly constrained designs. Without knowing designer's intent, it is difficult for Γ to reach correct decisions during the search process. Rather, Γ collects and stores all relevant data and presents them to designers to facilitate useful designer manual interventions to incrementally capture design intent during the design process. This was illustrated by the design example in Section 5-D. Allowing designer manual interventions during the design process is one of the key features that distinguish Γ from the previous attempts in developing analog circuit design tools. It provides a better path for faster design convergence.

While some of the performance parameters implemented in Γ's circuit evaluator are natural for static, small-signal analysis (e.g. CMRR, BW and other parameters can be estimated from that analysis, such as settling time from pole/zero analysis), other parameters cannot be inferred directly from component dimensions and operating point. Such parameters, such as total harmonic distortion, can only be derived from classic simulation techniques. While the existing Γ features are capable of meeting the majority of design needs for analog circuits, further development of Γ may include alleviate these limitations, for example, by using piece-wise linear LUTs with predictive distortion models for output distortion calculation. Furthermore, additional analyses and visualization features may include:

- Combined objective through cost function – Multi-objective analysis keeps the data open for follow up application-specific choice. The final choice can be illustrated by using cost-function weights. Visualization of circuit choice per weight assignment can be implemented on the browser side using JavaScript.
- Load support – Re-evaluate circuits in real time with a given capacitive load for more realistic performance values.
- Integration of support circuits – Allow better integration of sizing algorithms to include support circuits, such as bias circuits, to achieve better global results.

- Multi-topology search – Instead of limiting the search to the selected topology template, Γ perform search across different topology templates to find better solutions.

Acknowledgments

The authors would like to thank the National Science Foundation (GDE-0841259) for their funding of this project. Special thanks to Tyler Gus, for his help in programming website graphics, to Devashish Vedre for contributing and advising on the test-case operational amplifier and to Don Bartlett for offering his feedback and perspective as analog circuit designer.

References

[1] William Nye, David C. Riley, Alberto Sangiovanni-Vincentelli, Andre L. Tits, DELIGHT. SPICE: an optimization-based system for the design of integrated circuits, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 7 (4) (1988) 501–519.
[2] Marc G.R. Degrauwe, et al., IDAC: an interactive design tool for analog CMOS circuits, IEEE J. Solid-State Circuits 22 (6) (1987) 1106–1116.
[3] Georges G.E. Gielen, Herman C.C. Walscharts, Willy M. Sansen, Analog circuit design optimization based on symbolic simulation and simulated annealing, Solid-State Circuits 25 (3) (1990) 252–255.
[4] Rutenbar, Rob, Analog design automation: Where are we? Where are we going? in: Proceedings of the Custom Integrated Circuits Conference, Proceedings of the IEEE 1993, IEEE, 1993.
[5] Barcelona Design Unveils Revolutionary Analog Circuit Solution, prnewswire.com/prn/11690X15251303, April 8, 2002.
[6] Costello's analog automation pioneer, Barcelona, to fold, EETIMES doc_id: 1217996, 3/4/2005.
[7] Carl Martin Allwood, Tomas Kalen, Usability in CAD – A psychological perspective, Int. J. Human Factors Manuf. 4 (2) (1994) 145–165.
[8] Jakob Nielsen, Usability Engineering Chapter 5.5, Morgan Kaufmann, San Francisco, 1993.
[9] Chenming Hu, BSIM model for circuit design using advanced technologies, in: Proceedings of the Symposium on VLSI Circuits, Digest of Technical Papers, 2001, pp. 5–10.
[10] Ngspice Users, Manual version 26 (describes ngspice-26 release version) Paolo Nenzi, Holger Vogt (2014).
[11] Georges G.E. Gielen, Rob Rutenbar, Computer-aided design of analog and mixed-signal integrated circuits, Proc. IEEE 88.12 (2000) 1825–1854.
[12] Chung-Wen Ho, Albert E. Ruehli, Pierce A. Brennan, The modified nodal approach to network analysis, IEEE Trans. Circuits Syst. 22 (6) (1975) 504–509.
[13] Borchers, Carsten, Lars Hedrich, Erich Barke, Equation-based behavioral model generation for nonlinear analog circuits, in: Proceedings of the 33rd Annual Design Automation Conference, ACM, 1996.
[14] Kwang S. Yoon, Phillip E. Allen, An adjustable accuracy model for VLSI analog circuits using lookup tables, Analog Integr. Circuits Signal Process. 1 (1) (1991) 45–63.
[15] A. Rofougaran, et al., A Table lookup FET model for accurate analog circuit simulation, IEEE Trans. CAD 12 (2) (1993) 324–335.
[16] V. Rajiv, N. Joshi, Kanj Rouwaida, Kim Keunwoo, Table lookup method for physics based models for SPICE-like simulators, U. S. Patent No. 606,557,10 Dec 2013.
[17] Yishai Statter, Tom Chen, A novel high-throughput method for table look-up based analog design automation, Integr. VLSI J. (2016) 168–181.
[18] T. Tuma, B. Árpád, Circuit simulation with SPICE OPUS: theory and practice, Springer Science & Business Media (2009).
[19] Litho Guide, Physical Analyzer User. "Cadence", San Jose, CA, USA, 2004.
[20] Kim Jang Dae, et al., Algorithmic reactive testbench for analog designs, U. S. Patent No. 7, 853,908, 14 Dec 2010.
[21] Benhala Bachir, Ahaitouf Ali, Mechaqrane Abdellah, Multiobjective optimization of an operational amplifier by the ant colony optimisation algorithm, Sci. Acad. Publ. Elec. Electron. Eng. 2 (2012) 230–235.
[22] R.A. Vural, T. Yildirim, Analog circuit sizing via swarm intelligence, AEU – Int. J. Electron. Commun. (2012) 732–740.
[23] J.R. Koza, R. Poli, Genetic programming, Chapter 5: Search Methodologies, Springer, US (2005), p. 127–164.
[24] D.M. Binkley, Tradeoffs and optimization in analog CMOS design. Mixed Design of Integrated Circuits and Systems, MIXDES'07. 14th International Conference on. IEEE, 2007.
[25] Circuit Lab, Online Circuit Editing and Simulating, ⟨https://www.circuitlab.com/docs/the-basics/⟩.
[26] Part Sim, An Online Simulator, ⟨https://www.circuitlab.com/docs/the-basics/⟩.
[27] Logic Lab, Online Tool for Simple Logic Simulations, ⟨http://www.neuroproductions.be/logic-lab/⟩.
[28] J. Crossley, A. Puggelli, H.-P. Le, et al., BAG: a designer-oriented integrated framework for the development of AMS circuit generators, in: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2013, pp. 74–81.
[29] Cadence, Virtuoso NeoCircuits Datasheet, vol. 2, 2007, Available online: ⟨http://www.cadence.com/products/custom_ic⟩.
[30] Spectre, Cadence Virtuoso, Cadence home page.
[31] Simulator, Cadence Virtuoso UltraSim Full Chip, Datasheet, pp. 1–4.
[32] Business Wire, Business Wire, 5 February 2004. (Online).
[33] Laker3 Custom Design Tools, Synopsys Solvenet.