

ANALOG PERFORMANCE SPACE EXPLORATION BY FOURIER-MOTZKIN ELIMINATION WITH APPLICATION TO HIERARCHICAL SIZING

Guido Stehr, Helmut Graeb, Kurt Antreich

Institute for Electronic Design Automation, TU Munich, Arcisstrasse 21, 80333 Munich, Germany

ABSTRACT

Analog performance space exploration identifies the range of feasible performance values of a given circuit topology. It is an extremely challenging task of great importance to topology selection and hierarchical sizing. In this paper, a novel technique for the efficient simulation-based exploration of high-dimensional performance spaces is presented. To this end, fundamental circuit design knowledge is described by constraint functions. Based on a linearization of the latter and of the circuit performance functions, a description of the feasible performance range in the form of a polytope is derived. Moreover, the approach is integrated into a hierarchical sizing method, where it propagates topological and technological constraints bottom-up. Practical application results demonstrate the efficiency and usefulness of the new method.

1. INTRODUCTION

Analog design comprises three main steps: topology selection (or generation, respectively), component sizing, and layout generation. Especially in the topology selection phase, a notion of the achievable circuit performances of a given circuit topology is essential. One is interested in comparing different available topologies concerning their capability of satisfying the given performance specifications, and in selecting the most promising candidate for the expensive sizing process. Another extremely important application of performance space exploration is hierarchical circuit sizing. This design style relies on a top-down propagation of performance specifications. A key problem is to avoid unachievable specifications at the lower levels of abstraction. In order to guide the optimization engine at higher levels of abstraction, one urgently needs a representation of the capabilities of the lower-level building blocks.

Unfortunately, performance space exploration is extraordinarily challenging in terms of computational costs. A number of performance space exploration techniques have been published so far. To calculate the circuit performances, some of them [6, 11] rely on symbolic equations, which partially require a laborious manual setup. Others [13, 14] use detailed circuit simulation, which is flexible and accurate, but slow. As a common denominator, all these approaches are adapted for nonlinear circuit behavior in order to achieve a good degree of accuracy. Unfortunately, the nonlinear exploration of a high-dimensional performance space consumes a prohibitive amount of computational resources. Consequently, these techniques are forced to focus on a small number of performances, typically two or three. However, analog building blocks are often characterized by far more performances.

Therefore, we suggest a novel simulation-based performance space exploration technique, which calculates a polytope as approximation to the range of feasible performance values (*polytopal approximation*). Our algorithm is based on the idea that by its

operation, a circuit maps its feasible parameter space to its feasible performance space. Accordingly, a polytopal description of the feasible parameter space is determined first. Then, the map into the performance space is approximated by a linear relation of circuit parameters and circuit performances. Using Gaussian and Fourier-Motzkin elimination, we derive a polytopal approximation to the feasible performance space. Owing to the efficiency of this approach, it can deal with high-dimensional performance spaces, which are too complex for nonlinear exploration methods. This feature is crucial for hierarchical sizing, which, at higher levels of abstraction, requires a high-dimensional description of the achievable performance values of the available functional blocks. While this problem is frequently mentioned in the literature, e. g. [1, 11], hardly any general solution to it has been presented so far. Using the example of a filter design, we show how our performance space exploration technique is a key enabler for hierarchical sizing.

The remainder of the paper is structured as follows: Sec. 2 briefly reviews the fundamental concepts of performance space exploration. Sec. 3 then discusses the basic idea of our method. Sec. 4 explains the Fourier-Motzkin elimination method in detail, whereupon in Sec. 5, we give experimental results. Special emphasis is put on a hierarchical sizing approach, which relies on our new performance space exploration technique. Final conclusions are given in Sec. 6.

2. NONLINEAR PERFORMANCE SPACE DESCRIPTION

For a given topology and production technology, the performances of a circuit, such as gain and slew rate of an amplifier, are determined by the designable circuit parameters, mainly transistor widths and lengths. Hence, for the calculation of the feasible performance space, one needs to know which parameter values are feasible and how the topology relates parameters to performances.

2.1. Feasible Parameter Space

The feasible parameter space is defined by topological and technological constraints, which, collectively, are called *sizing constraints*. They always comprise lower and upper bounds for the individual designable parameters of the circuit, such as minimum and maximum transistor gate sizes. In [4, 7], for example, any circuit, that satisfies these box constraints and that can be simulated successfully, is considered feasible. Going beyond that, most authors agree on keeping transistors in saturation without going into detail [13, 16]. A few publications, especially on knowledge-based or symbolic approaches, discuss additional constraints like symmetry, matching or maximum currents [5, 10]. An *automatic*, even more comprehensive constraint setup is described in [9]: For example, the operating conditions of transistors in a current mirror must not differ significantly and their current ratios must not deviate too far from their channel width ratios.

Sizing constraints can be classified as follows:

- *Equality* constraints, which may result from matching requirements, require that parameters only differ by a constant factor at the most. These constraints are given as algebraic equations and can be used to reduce the dimension of the parameter space.
- *Inequality* constraints exclude portions of the parameter space without reducing its dimension. They can further be subdivided into two categories:
 - *Geometric* constraints *explicitly* refer to geometric parameters and might, for example, prescribe the minimum device area in order to reduce mismatch.
 - *Electric* constraints characterize the DC operating point and can, for example, ensure saturation. They result in *implicit* requirements for the circuit parameters and have to be evaluated using simulation.

The sizing constraints translate elementary design knowledge into a mathematical representation. After their setup for a given topology, the constraints describe the entirety of technically meaningful circuit realizations.

After the elimination of circuit parameters based on equality constraints, the remaining parameters denoted as \mathbf{p}^1 have to satisfy a number of inequality constraints, which are either explicitly given as algebraic expressions or have to be evaluated via DC simulation. Elementary algebraic transformations lead to a single nonlinear vector inequality $\mathbf{c}(\mathbf{p}) \geq \mathbf{0}$ ². The associated feasible parameter space \mathcal{P} can be written as

$$\mathcal{P} = \{\mathbf{p} \mid \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\} \quad \text{with} \quad \mathbf{c}(\mathbf{p}) \in \mathbb{R}^m, \quad \mathbf{p} \in \mathbb{R}^n. \quad (1)$$

Each constraint $c(\mathbf{p})$ defines a hypersurface in the performance space. It separates the region, where the particular constraint is satisfied ($c(\mathbf{p}) \geq 0$) from the region, where it is violated ($c(\mathbf{p}) < 0$). In $\mathcal{P} \subset \mathbb{R}^n$, all sizing constraints are fulfilled, cf. Fig. 1, left. Note that even for small circuits, there is a large number of sizing constraints ($m \gg n$), and that \mathcal{P} is certainly bounded since the sizing constraints always include upper and lower bounds for each parameter.

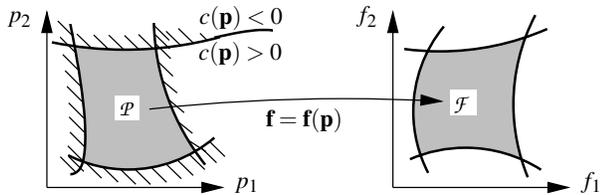


Figure 1: Nonlinear Relation of Feasible Parameter Space (left) and Feasible Performance Space (right)

2.2. Map from Parameter to Performance Space

By its operation, a circuit maps its parameters \mathbf{p} onto its performances \mathbf{f} :

$$\mathbf{f} = \mathbf{f}(\mathbf{p}) \quad \text{with} \quad \mathbf{f} \in \mathbb{R}^q \quad \text{and} \quad \mathbf{p} \in \mathbb{R}^n. \quad (2)$$

Practically, the map $\mathbf{f}(\mathbf{p})$ can only be evaluated pointwise, e.g. using simulation. Note that there are usually more parameters than performances, i.e. $n > q$.

¹In this paper, regular lower case letters denote scalars. Vectors are written in bold lower case. Matrices are bold capitals.

²Vector inequalities are interpreted element-wise.

2.3. Feasible Performance Space

As indicated in Fig. 1, the feasible parameter space $\mathcal{P} \subset \mathbb{R}^n$ has an image $\mathcal{F} \subset \mathbb{R}^q$ in the performance space:

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{f} = \mathbf{f}(\mathbf{p}) \wedge \mathbf{p} \in \mathcal{P}\}, \quad \mathbf{p} \in \mathcal{P} \Leftrightarrow \mathbf{c}(\mathbf{p}) \geq \mathbf{0}. \quad (3)$$

In this formulation, \mathcal{F} is still coupled to the parameter space. However, the goal of performance space exploration is a description of \mathcal{F} *entirely in the performance space* analogous to (1):

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{k}(\mathbf{f}) \geq \mathbf{0}\}. \quad (4)$$

Unfortunately, the nonlinear maps $\mathbf{f}(\mathbf{p})$ and $\mathbf{c}(\mathbf{p})$ can only be evaluated pointwise. Therefore, there is no straight way to map \mathcal{P} onto \mathcal{F} in its entirety. Combining (3) and (4), the task at hand can be described as follows: Find a vector inequality $\mathbf{k}(\mathbf{f})$ such that

$$\underbrace{\mathbf{c}(\mathbf{p}) \geq \mathbf{0}}_{\text{(I) feasible parameter space } \mathcal{P}} \quad \wedge \quad \underbrace{\mathbf{f} = \mathbf{f}(\mathbf{p})}_{\text{(II) map: parameter to performance space } \mathbf{p} \mapsto \mathbf{f}} \quad \Leftrightarrow \quad \underbrace{\mathbf{k}(\mathbf{f}) \geq \mathbf{0}}_{\text{(III) feasible performance space } \mathcal{F}} \quad (5)$$

3. POLYTOPAL PERFORMANCE SPACE DESCRIPTION

In most practical cases, the accurate calculation of a nonlinear description of \mathcal{F} according to (5)/(III) is impossible. Yet, this paper presents a method to calculate a polytopal approximation to (5)/(III): Based on a polytopal representation of the feasible parameter space (5)/(I) and on a linearized map from the parameter to the performance space (5)/(II), a polytopal description of the feasible performance space (5)/(III) can be calculated as described below.

With these computationally inexpensive polytopal approximations, we achieved good results in hierarchical sizing, cf. Sec. 5.5. Obviously, the choice of the linearization point is crucial to the accuracy of the final performance space description. In our experiments, we obtained good results with the linearization point located *in the center* of the feasible parameter space \mathcal{P} . An algorithm to reliably identify this point was presented in [15]. If more accurate performance space descriptions are wanted, they can be computed nonlinearly, if the computational resources allow it [13, 14].

3.1. Polytopal Feasible Parameter Space

In the neighborhood of a point $\hat{\mathbf{p}}$ in the parameter space, the constraint function $\mathbf{c}(\mathbf{p})$ can be approximated by a linear Taylor expansion:

$$\mathbf{c}(\mathbf{p}) \approx \mathbf{c}(\hat{\mathbf{p}}) + \left. \frac{\partial \mathbf{c}(\mathbf{p})}{\partial \mathbf{p}} \right|_{\hat{\mathbf{p}}} \cdot \Delta \mathbf{p}, \quad \Delta \mathbf{p} = \mathbf{p} - \hat{\mathbf{p}}. \quad (6)$$

From (5)/(I) and (6), we obtain the following polytopal description of the feasible parameter space:

$$\underbrace{\left. \frac{\partial \mathbf{c}(\mathbf{p})}{\partial \mathbf{p}} \right|_{\hat{\mathbf{p}}}}_{\mathbf{C}} \cdot \Delta \mathbf{p} \geq \underbrace{-\mathbf{c}(\hat{\mathbf{p}})}_{\mathbf{c}}. \quad (7)$$

The Jacobian matrix \mathbf{C} describes the linearized behavior of the sizing constraints with respect to the circuit parameters. Depending on the available circuit simulator, this can be simulated directly or by finite differences. Geometrically, the approximation to the feasible parameter space according to (7) represents a polytope [17], hence its name.

3.2. Linearized Map from Parameter to Performance Space

The performance function $\mathbf{f}(\mathbf{p})$ can be treated similarly to the constraints in the previous section:

$$\mathbf{f}(\mathbf{p}) \approx \mathbf{f}(\hat{\mathbf{p}}) + \left. \frac{\partial \mathbf{f}(\mathbf{p})}{\partial \mathbf{p}} \right|_{\hat{\mathbf{p}}} \cdot \Delta \mathbf{p}. \quad (8)$$

From (5)/(II) and (8) we obtain the wanted linearization

$$\underbrace{\left. \frac{\partial \mathbf{f}(\mathbf{p})}{\partial \mathbf{p}} \right|_{\hat{\mathbf{p}}}}_{\mathbf{F}} \cdot \Delta \mathbf{p} = \underbrace{\mathbf{f} - \mathbf{f}(\hat{\mathbf{p}})}_{\Delta \mathbf{f}}. \quad (9)$$

In (9), $\Delta \mathbf{f}$ can be seen as variations of the circuit performances, and the Jacobian matrix \mathbf{F} describes the linearized behavior of the performances \mathbf{f} with respect to the parameters \mathbf{p} .

3.3. Polytopal Feasible Performance Space

Based on the results from the previous sections, namely the polytopal feasible parameter space (7) and the linearized map from parameter to performance space (9), a polytopal approximation to the feasible performance space is sought:

$$\underbrace{\mathbf{C} \cdot \Delta \mathbf{p} \geq \mathbf{c}}_{(7)} \wedge \underbrace{\mathbf{F} \cdot \Delta \mathbf{p} = \Delta \mathbf{f}}_{(9)} \Leftrightarrow \underbrace{\mathbf{K} \cdot \Delta \mathbf{f} \geq \mathbf{k}}_{(10)}$$

If \mathbf{F} is nonsingular, then the parameter vector $\Delta \mathbf{p}$ in (9) can formally be obtained by matrix inversion:

$$\Delta \mathbf{p} = \mathbf{F}^{-1} \cdot \Delta \mathbf{f}. \quad (11)$$

Using this result in (7) yields

$$\underbrace{\mathbf{C} \cdot \mathbf{F}^{-1}}_{\mathbf{K}} \cdot \Delta \mathbf{f} \geq \underbrace{\mathbf{c}}_{\mathbf{k}}. \quad (12)$$

However, \mathbf{F} is usually not invertible. For this case, we developed a two-step method to calculate (10), as outlined below.

3.3.1. Equation-Based Parameter Elimination

For most circuits, the parameters outnumber the performances, i.e. $n > q$, and \mathbf{F} has full rank. Then, we can rewrite (9) as

$$(\mathbf{F}_{\square} \ \mathbf{F}_{\diamond}) \cdot \begin{pmatrix} \Delta \mathbf{p}_{\square} \\ \Delta \mathbf{p}_{\diamond} \end{pmatrix} = \Delta \mathbf{f}, \quad (13)$$

with a partitioning of \mathbf{F} into a regular quadratic part \mathbf{F}_{\square} and a remainder \mathbf{F}_{\diamond} :

$$\mathbf{F} = (\mathbf{F}_{\square} \ \mathbf{F}_{\diamond}); \quad \mathbf{F}_{\square} \in \mathbb{R}^q \times \mathbb{R}^q, \ \mathbf{F}_{\diamond} \in \mathbb{R}^q \times \mathbb{R}^{n-q}. \quad (14)$$

Then, $\Delta \mathbf{p}_{\square}$ can formally be obtained from

$$\Delta \mathbf{p}_{\square} = \mathbf{F}_{\square}^{-1} \cdot \Delta \mathbf{f} - \mathbf{F}_{\square}^{-1} \mathbf{F}_{\diamond} \cdot \Delta \mathbf{p}_{\diamond}. \quad (15)$$

Instead of determining $\mathbf{F}_{\square}^{-1}$ explicitly, Gaussian elimination or any other method to solve a system of linear equations can be used to numerically calculate the expressions including the inverse matrix. It is obvious that a good partitioning of \mathbf{F} is essential. Therefore, we combine those q columns from \mathbf{F} in \mathbf{F}_{\square} , which yield the best numerical condition.

In analogy to (13), (7) corresponds to

$$(\mathbf{C}_{\square} \ \mathbf{C}_{\diamond}) \cdot \begin{pmatrix} \Delta \mathbf{p}_{\square} \\ \Delta \mathbf{p}_{\diamond} \end{pmatrix} \geq \mathbf{c}. \quad (16)$$

The substitution of $\Delta \mathbf{p}_{\square}$ in (16) by the expression from (15) yields

$$(\mathbf{C}_{\square} \ \mathbf{C}_{\diamond}) \begin{pmatrix} \mathbf{F}_{\square}^{-1} & -\mathbf{F}_{\square}^{-1} \mathbf{F}_{\diamond} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \Delta \mathbf{f} \\ \Delta \mathbf{p}_{\diamond} \end{pmatrix} \geq \mathbf{c}, \quad (17)$$

where \mathbf{I} is an identity matrix of dimension $(n-q) \times (n-q)$. Note that in (17), the performances $\Delta \mathbf{f}$ replaced an equal number of parameters $\Delta \mathbf{p}_{\square}$. Geometrically, the feasible parameter polytope was mapped from the $\Delta \mathbf{p}$ space into the $\Delta \mathbf{f}/\Delta \mathbf{p}_{\diamond}$ space. To obtain the feasible performance polytope in the $\Delta \mathbf{f}$ space according to (10), the remaining parameters $\Delta \mathbf{p}_{\diamond}$ have to be eliminated as well. Geometrically, this corresponds to an orthogonal projection along the coordinate axes of the remaining parameters $\Delta \mathbf{p}_{\diamond}$.

3.3.2. Inequality-Based Parameter Elimination

To eliminate the remaining $(n-q)$ parameters $\Delta \mathbf{p}_{\diamond}$ from (17), *Fourier-Motzkin elimination* [3] can be used. This elimination technique for inequalities bears some resemblance to Gaussian elimination for equations. It is described in detail in Sec. 4. This second elimination step leads to

$$\mathbf{K} \cdot \Delta \mathbf{f} \geq \mathbf{k}, \quad (10)$$

which is the sought polytopal description of the feasible performance space. In contrast to this approach, interval methods can only describe *hyperboxes*, which are inferior in terms of accuracy.

Although Fourier-Motzkin elimination has been known for a long time, it has only recently gained increased attention in the area of combinatorial optimization and compiler optimization [2, 12]. However, these algorithms are inapplicable in this context because they are geared towards discrete problems. Therefore, we developed a dedicated Fourier-Motzkin elimination algorithm, which meets the demands of high-dimensional performance space exploration based on floating-point simulation data.

4. FOURIER-MOTZKIN ELIMINATION

As described above, Gaussian and Fourier-Motzkin elimination are used to map the polytopal approximation to the feasible parameter space into the performance space. Since Fourier-Motzkin elimination constitutes the core of our algorithm, it will be discussed in detail here. After a general overview, an example illustrates the method, whereupon practical implementation aspects follow.

4.1. Basic Algorithm

The elimination of a variable x_r , $1 \leq r \leq N$, from a system of M linear inequalities

$$\mathbf{A} \mathbf{x} \geq \mathbf{b}, \quad \mathbf{A} = \begin{pmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_M^T \end{pmatrix} = (a_{ij}), \quad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix},$$

$$\mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_M \end{pmatrix}, \quad M, N \geq 2, \quad (18)$$

corresponds to the calculation of a new vector inequality with $a_{ir} = 0, 1 \leq i \leq M$. To do this, Fourier-Motzkin elimination exploits two properties of inequalities:

$$s \geq t \wedge y \geq z \Rightarrow s + y \geq t + z, \quad (19)$$

$$s \geq t \wedge y \geq 0 \Rightarrow s \cdot y \geq t \cdot y. \quad (20)$$

The elimination of a variable x_r from (18) comprises two steps:

1. Sorting of inequalities

The individual inequalities $\mathbf{a}_i^T \mathbf{x} \geq b_i$ from (18) are partitioned into three sets:

$$\begin{aligned} (I) \quad I_{>} &= \{\mathbf{a}_g^T \mathbf{x} \geq b_g \mid a_{gr} > 0 \wedge g \in G\} \\ (II) \quad I_{<} &= \{\mathbf{a}_l^T \mathbf{x} \geq b_l \mid a_{lr} < 0 \wedge l \in L\} \\ (III) \quad I_{=} &= \{\mathbf{a}_e^T \mathbf{x} \geq b_e \mid a_{er} = 0 \wedge e \in E\} \end{aligned} \quad (21)$$

$$\begin{aligned} \text{with} \quad G \cap L &= G \cap E = L \cap E = \emptyset \\ &\wedge G \cup L \cup E = \{1, \dots, M\}. \end{aligned}$$

Note that the inequalities in $I_{=}$ are already independent of x_r .

2. Elimination of x_r by linear combination

The set $I_{=}$ is augmented by all pairwise linear combinations of inequalities from $I_{>}$ and $I_{<}$ according to

$$I_{=} \cup \{(a_{gr} \cdot \mathbf{a}_l^T - a_{lr} \cdot \mathbf{a}_g^T) \cdot \mathbf{x} \geq a_{gr} b_l - a_{lr} b_g \mid g \in G \wedge l \in L\}. \quad (22)$$

This set of inequalities can be written in vector/matrix notation:

$$\mathbf{A}^{\setminus r} \cdot \mathbf{x} \geq \mathbf{b}^{\setminus r}. \quad (23)$$

The entries in the r^{th} column of $\mathbf{A}^{\setminus r}$ are all zeros. Hence, the new system of inequalities is independent of x_r .

These two steps are repeated for all variables to be eliminated.

4.2. Example

To demonstrate the Fourier-Motzkin algorithm, a hypothetic example with three circuit parameters, p_1 to p_3 , and one circuit performance, $f_1 = f_1(p_1, p_2, p_3)$ is considered. Let the sizing constraints be linearized at a suitable operating point according to (7) and the circuit performance according to (9). Assume that, with Gaussian elimination, Δp_1 was replaced by Δf_1 :

$$\begin{pmatrix} 5 & 1 & 7 \\ 1 & 1 & -1 \\ 5 & -1 & -10 \\ 0 & -3 & -2 \\ -7 & 0 & -2 \end{pmatrix} \cdot \begin{pmatrix} \Delta f_1 \\ \Delta p_2 \\ \Delta p_3 \end{pmatrix} \geq \begin{pmatrix} 63 \\ 1 \\ -75 \\ -74 \\ -43 \end{pmatrix}. \quad (24)$$

Wanted is the value range of Δf_1 with the remaining circuit parameters varying within the allowed bounds. Consequently, Δp_2 and Δp_3 have to be eliminated. Let Δp_2 be removed first. To this end, the inequalities are sorted according to (21):

$$\begin{aligned} (I) \quad & \begin{pmatrix} 5 & 1 & 7 \\ 1 & 1 & -1 \end{pmatrix} \begin{vmatrix} 63 \\ 1 \end{vmatrix} & (a) \\ (II) \quad & \begin{pmatrix} 5 & -1 & -10 \\ 0 & -3 & -2 \\ -7 & 0 & -2 \end{pmatrix} \begin{vmatrix} -75 \\ -74 \\ -43 \end{vmatrix} & (b) \\ (III) \quad & & (c) \\ & & (d) \\ & & (e) \end{aligned} \quad (25)$$

For ease of notation, the resulting system of inequalities is written as an extended matrix with a double line separating the left-hand from the right-hand side. The horizontal lines separate the three sets of inequalities. The individual original inequalities are marked by lowercase letters for further reference.

Four new inequalities according to (22) can be created. Consequently, inequalities (a) through (d) are replaced by their linear combinations (a,d), (a,c), (b,c), and (b,d):

$$\begin{aligned} (I) \quad & \begin{pmatrix} 15 & 0 & 19 \\ 10 & 0 & -3 \\ 6 & 0 & -11 \\ 3 & 0 & -5 \\ -7 & 0 & -2 \end{pmatrix} \begin{vmatrix} 115 \\ -12 \\ -74 \\ -71 \\ -43 \end{vmatrix} & (a,d) \\ (II) \quad & & (a,c) \\ & & (b,c) \\ & & (b,d) \\ & & (e) \end{aligned} \quad (26)$$

In this new matrix, there are one positive, four negative and no zero coefficients of Δp_3 . Accordingly, the final elimination step yields a system of four inequalities:

$$\begin{aligned} (I) \quad & \begin{pmatrix} -103 & 0 & 0 \\ 235 & 0 & 0 \\ 279 & 0 & 0 \\ 132 & 0 & 0 \end{pmatrix} \begin{vmatrix} -587 \\ 117 \\ -141 \\ -774 \end{vmatrix} & (a,d,e) \\ (II) \quad & & (a,c,d) \\ & & (a,b,c,d) \\ & & (a,b,d) \end{aligned} \quad (27)$$

The second line, for example, was derived from (26),(a,d) and (26),(a,c). Therefore, it comprises the original inequalities (a), (c), and (d). With (27), the elimination is finished and the resulting inequalities in Δf_1 are

$$\begin{cases} \Delta f_1 \leq 587/103 \approx 5.699 & (a,d,e) \\ \Delta f_1 \geq 117/235 \approx 0.498 & (a,c,d) \\ \Delta f_1 \geq -141/279 \approx -0.505 & (a,b,c,d) \\ \Delta f_1 \geq -774/132 \approx -5.864 & (a,b,d) \end{cases} \quad (28)$$

Obviously, the feasible circuit performance range is only limited by inequalities (a,d,e) and (a,c,d):

$$0.498 \leq \Delta f_1 \leq 5.699. \quad (29)$$

Fourier-Motzkin elimination can be interpreted geometrically as an orthogonal projection along the coordinate axes, as depicted in Fig. 2. The original vector inequality (25) describes a polytope in the 3-dimensional space. The elimination of Δp_2 yields a polygon in the $\Delta p_3/\Delta f_1$ space. The final removal of Δp_3 leads to a line indicating the value range of Δf_1 .

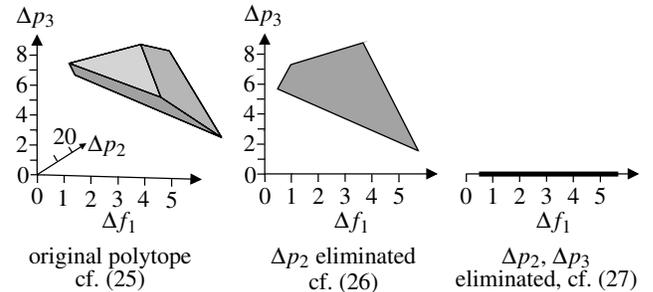


Figure 2: Fourier-Motzkin Elimination as Orthogonal Projection

4.3. Redundancy Detection

The simple example above does not immediately reveal complexity as the severest weakness of the Fourier-Motzkin elimination. It turned out in (28) that two inequalities were redundant. In fact, especially in high-dimensional spaces, and after several elimination steps, a huge number of inequalities can exhaust the computing resources easily if no special care is taken. In Fig. 3, the two-dimensional projection of a dodecahedron is shown, which was originally described by twelve inequalities. Note how many redundant inequalities occur.



with redundant inequalities without redundant inequalities

Figure 3: Generation of Redundant Inequalities

The elimination order severely influences the generation of redundant inequalities. Obviously, the most efficient way to cope with redundant inequalities is their avoidance. The number M of inequalities after an elimination step can be calculated from the powers of the three inequality sets before the elimination:

$$M = |I_{>}| \cdot |I_{<}| - |I_{>}| - |I_{<}| + |I_{=}|. \quad (30)$$

A *locally optimal* choice of the next parameter to be eliminated can be made by determining the inequality sets from (21) and calculating the associated number of inequalities according to (30).

Note, however, that each elimination step changes the properties of the remaining matrix. Especially geometric sizing constraints often only relate a few parameters, which results in zeros in the constraint matrix C from (7). This feature is highly desirable because it reduces the elimination effort. Therefore, in addition to a locally minimal number of new inequalities, the preservation of zero entries is a second criterion for the determination of an advantageous elimination order. Unfortunately, the determination of a globally optimal elimination order is still an unresolved problem.

Let M_i be the number of inequalities before elimination step i . In the worst case, an even number of inequalities splits up equally between $I_{>}$ and $I_{<}$ resulting in

$$M_{i+1} = \left(\frac{M_i}{2}\right)^2 - M_i. \quad (31)$$

Experience shows that typical application data cannot be handled without an exhaustive identification of redundancies. For efficiency, we apply a two-layered redundancy filter after each elimination step.

4.3.1. Fast Structural Redundancy Detection

An inequality is redundant if and only if it can be written as a positive combination of other inequalities in the system [17]. From this rule, Chernikov derived a criterion, which allows a fast and easy identification of redundant inequalities [17]:

An inequality is redundant after step i , if it is based on more than $i + 1$ original inequalities.

In (27), this criterion would have identified the third inequality, (a, b, c, d) , as redundant: After the second elimination step, it is based on more than three original inequalities. However, the redundancy of the fourth inequality, (a, b, d) , in (27) would not be detected. The reason is that this criterion only identifies *structural* redundancy and does not consider the numerical properties. In spite of this drawback, the Chernikov criterion has its value as a computationally cheap method to expunge obvious redundancies.

4.3.2. Definite Numerical Redundancy Detection

Experience shows that an additional test identifying *all* redundancies is required to keep the necessary computer resources within practical limits. After the first filtering step, we examine each of the remaining inequalities *numerically* using linear programming.

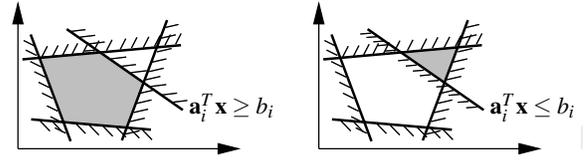


Figure 4: Numerical Detection of Redundant Inequalities

In a system of inequalities, the i^{th} inequality $\mathbf{a}_i^T \mathbf{x} \geq b_i$ is irredundant if and only if its removal extends the feasible space. As illustrated in Fig. 4, this extension is described by the original system of inequalities with the i^{th} inequality replaced by its inverse, $\mathbf{a}_i^T \mathbf{x} \leq b_i$. Therefore, inequality i is irredundant if and only if

$$\begin{pmatrix} \mathbf{a}_1^T \\ \vdots \\ -\mathbf{a}_i^T \\ \vdots \\ \mathbf{a}_M^T \end{pmatrix} \mathbf{x} \geq \begin{pmatrix} b_1 \\ \vdots \\ -b_i \\ \vdots \\ b_M \end{pmatrix} \quad (32)$$

has feasible solutions. This can be checked efficiently using the first phase of the simplex algorithm. Practical experience shows that more than 90% of the CPU time of the entire performance space exploration algorithm is required by this second filtering step. However, it is this very step that keeps the overall resource requirements in practical limits, cf. Sec. 5.4.

5. EXPERIMENTAL RESULTS

5.1. Polytopal vs. Nonlinear Performance Evaluation

We found that polytopal approximations to feasible performance spaces are reasonably accurate. The reason is that the sizing constraints only allow a small feasible parameter space \mathcal{P} [15]. In this region, the circuit behavior is usually not far from linear [9]. Accordingly, we achieved good results in hierarchical sizing using this computationally cheap method of performance space exploration, cf. Sec. 5.5.

For a folded cascode operational amplifier, Fig. 5 shows how well polytopal descriptions may approximate the feasible parameter space. The filled polygon represents the 2-dimensional projection of the polytopal feasible performance space, and the dots show the results of a large number of simulation runs. For this example, the center point of \mathcal{P} [15] was chosen as linearization point $\hat{\mathbf{p}}$.

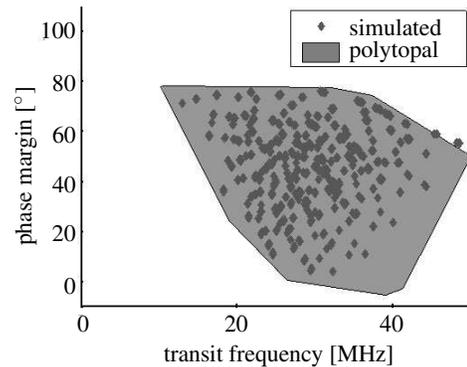


Figure 5: Polytopal vs. Sampled Feasible Performance Space

5.2. Topology Selection

Within flat as well as hierarchical circuit design, there are often several implementation alternatives for a certain building block. The presented performance space exploration technique allows a comparison of their capabilities and thus enables topology selection. Fig. 6 shows the polytopal feasible performance spaces of two different operational amplifiers. It is obvious that, for a high slew rate, the folded cascode is the architecture of choice while the Miller compensated amplifier is superior in terms of DC gain.

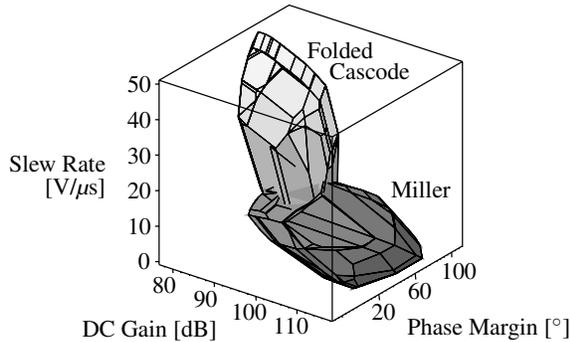


Figure 6: Performance Comparison of Two Operational Amplifiers

The computation time of this example, including simulation, performance space exploration and visualization, was less than three minutes on a cluster of Pentium 4 machines.

5.3. Determination of 9-Dimensional Performance Space

In the previous example, the performance space exploration algorithm was applied to a low-dimensional performance space, which could also be handled by nonlinear techniques, albeit with run-times of many hours. The true strength of our approach, however, lies in the examination of high-dimensional spaces, which cannot be handled practically by nonlinear approaches.

For the following example, the 9-dimensional performance space of a folded cascode amplifier was examined. This circuit consists of 22 transistors and has 11 designable parameters. An automatic topological analysis resulted in a total number of 204 sizing constraints. For performance space exploration, the following steps were done:

- Simulation-based determination of linearized circuit descriptions according to (7) and (9).
- Initial numerical redundancy check: 44 out of 204 inequalities identified as irredundant.
- Gaussian elimination of designable parameters 1, 4, 5, 6, 7, 8, 9, 10 and 11.
- Fourier-Motzkin elimination of remaining parameters 2 and 3:

	parameter 2	parameter 3	
$ I_{<}$	19	149	(I)
$ I_{>}$	· 24	· 176	(II)
$ I_{=}$	+ 1	+ 0	(III)
total # ineq.	= 457	= 26224	(IV)
# void	- 5	- 7	(V)
# red. ineq. Chernikov	- 0	- 24427	(VI)
# red. ineq. lin. prog.	- 127	- 542	(VII)
# irredundant ineq.	= 325	= 1248	(VIII)

Line (IV) lists the total numbers of inequalities after each parameter elimination before redundancy removal. The following lines give details on the redundancy detection: A few inequalities have only zero coefficients and can be discarded

(line (V)). The Chernikov criterion (line (VI)) and the linear-programming-based redundancy detection (line (VII)) leave a final number of irredundant inequalities as given in the last line. Note that the Chernikov criterion does not detect any structural redundancy after the elimination of parameter 2.

Excluding simulation, the calculations took 140 s on a 2.4 GHz Pentium 4 machine.

5.4. Effectiveness of Redundancy Detection

To illustrate the effectiveness of the two-step redundancy detection, we examined an operational transconductance amplifier with 11 parameters and 3 AC performances. In this case, 8 Fourier-Motzkin elimination steps were required. An initial complete redundancy check reduced the number of original inequalities from 120 to 47. The impact of different detection strategies is illustrated in Fig. 7. Note that only with the two-step redundancy detection presented in Sec. 4.3, the resource consumption remains within practical limits. The elimination with full redundancy detection took only 182 s on a 1.4 GHz Pentium M laptop computer and required just 7.3 MB of main memory. About 95% of the CPU time was used by the redundancy detection using linear programming.

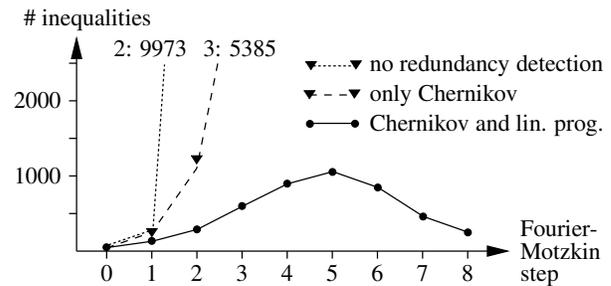


Figure 7: Effect of Different Redundancy Detection Strategies

5.5. Hierarchical Sizing of a Filter

While small analog blocks can be sized based on transistor models and simulators like SPICE, for complex electronic circuits, however, this approach results in impractical resource requirements. There is common agreement about the necessity of a hierarchical approach for the simulation-based sizing of larger analog circuits, e. g. [1, 11]. Yet, there is little material in the literature when it comes to concrete solutions. A core problem remained unsolved, namely the suitable formulation of physically motivated, possibly high-dimensional design constraints at higher levels of abstraction. This problem is a main target of our new performance space exploration method. In the following, we demonstrate a hierarchical sizing approach featuring two levels of abstraction.

In Fig. 8, an operational transconductance amplifier (OTA) is given in the form of a transistor netlist [8]. In the following, this level of abstraction is referred to as *block level*. Accordingly, the designable OTA parameters \mathbf{p} are called *block parameters*. They comprise the widths and lengths of the transistors and the bias current and voltages. Based on symmetry requirements and additional heuristics, nine essential block parameters were identified for this circuit. To adequately describe the electrical AC behavior of the OTA, the *block performances* \mathbf{f} do not only include its transconductance g , but also its input and output impedance, among others. In total, five important AC performances were identified. This OTA can be used as a building block for the biquad bandpass filter given in Fig. 9 [8].

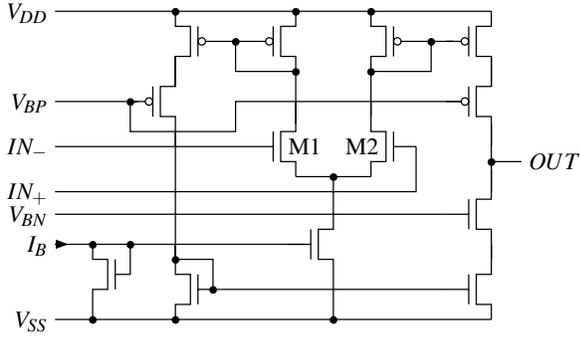


Figure 8: Operational Transconductance Amplifier Schematic

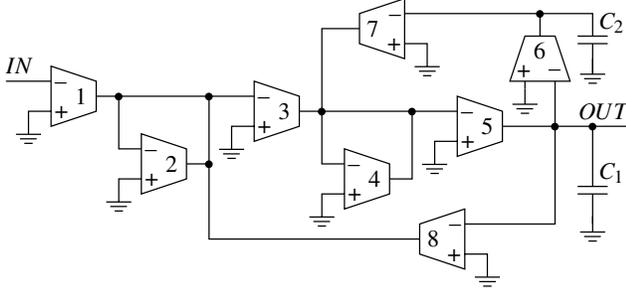


Figure 9: OTA-C Biquad Filter Schematic

For an efficient simulation of the entire filter, the OTAs were modeled behaviorally in VHDL-AMS based on [8]. In spite of the higher level of abstraction, they retain their electrical pins. Consequently, the filter is represented by a netlist of eight behavioral OTA models and two capacitors. This level of abstraction is called *system level* henceforth. Accordingly, the filter performances are termed *system performances* \mathbf{s} . In AC domain, they comprise the center frequency f_0 , the pass-band gain G_0 , and the quality factor Q . To make the behavioral models and the underlying transistor circuits behave alike, the OTA block performances serve as model parameters or *system parameters*, respectively. Due to symmetries in the filter, two pairs of OTAs, namely OTA₄/OTA₇ and OTA₅/OTA₆ can be sized identically. In contrast, the values of the two capacitors do not have a correspondence at block level and are genuine system parameters. Consequently, the biquad filter has $6 \cdot 5 + 2 = 32$ system parameters.

With $i \in I = \{1, 2, 3, (4/7), (5/6), 8\}$ and $\mathbf{f}_C = (C_1 C_2)$, the following table summarizes the features of the two abstraction levels.

	block level	system level
netlist elements	transistor models	VHDL-AMS models
parameters	$\mathbf{p}_i = (W_{M1} L_{M1} \dots I_B)$	$\mathbf{f} = (f_1 \dots f_8 \mathbf{f}_C)$
performances	$\mathbf{f}_i = (g R_{out} C_{out} \dots)$	$\mathbf{s} = (f_0 G_0 Q)$

Hierarchical Sizing Algorithm

The outline of a hierarchical sizing algorithm for the above filter is given in Fig. 10. It comprises the following procedural steps:

I) Determination of initial linearization point:

For a technically meaningful sizing, the OTAs have to satisfy all sizing constraints at block level. For symmetry reasons, for example, the drain-source voltages of the two transistors M1 and M2 must not differ too much. Therefore, it is advisable to choose an initial linearization point, which safely satisfies all block-level constraints $\mathbf{c}(\hat{\mathbf{p}}_i) \geq \mathbf{0}$. This is true for the center point of the feasible parameter space \mathcal{P} [15]. Even more, we

$\forall_{i \in I}$ determine $\hat{\mathbf{p}}_i$ with $\mathbf{c}(\hat{\mathbf{p}}_i) \geq \mathbf{0}$ according to [15]	(I)
$\forall_{i \in I}$ determine $\mathbf{C}_i, \mathbf{c}_i, \mathbf{F}_i, \mathbf{f}_i(\hat{\mathbf{p}}_i)$ via simulation at $\hat{\mathbf{p}}_i$	(II)
$\forall_{i \in I}$ calculate $\mathbf{K}_i, \mathbf{k}_i$, cf. (10)	(III)
$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmin}} \left\ (\mathbf{s}(\mathbf{f}) - \hat{\mathbf{s}}) / \hat{\mathbf{s}} \right\ $ s.t. $\forall_{i \in I} \mathbf{K}_i \cdot (\mathbf{f}_i - \mathbf{f}_i(\hat{\mathbf{p}}_i)) \geq \mathbf{k}_i$ $\wedge \mathbf{f}_C \geq \mathbf{f}_{C, \min} \wedge -\mathbf{f}_C \geq -\mathbf{f}_{C, \max}$	(IV)
$(\hat{C}_1 \hat{C}_2) = \hat{\mathbf{f}}_C$	(V)
$\forall_{i \in I} \left[\hat{\mathbf{p}}_i = \underset{\mathbf{p}_i}{\operatorname{argmin}} \left\ (\mathbf{f}_i(\mathbf{p}_i) - \hat{\mathbf{f}}_i) / \hat{\mathbf{f}}_i \right\ $ s.t. $\mathbf{c}_i(\mathbf{p}_i) \geq \mathbf{0}$	(VI)
$\left\ (\mathbf{s}(\hat{\mathbf{f}}) - \hat{\mathbf{s}}) / \hat{\mathbf{s}} \right\ _{\infty} < \hat{\epsilon}_s \wedge \left\ (\mathbf{f}_i(\hat{\mathbf{p}}_i) - \hat{\mathbf{f}}_i) / \hat{\mathbf{f}}_i \right\ _{\infty} < \hat{\epsilon}_f$ $\wedge \left\ (\mathbf{s}(\hat{\mathbf{p}}_1 \dots \hat{\mathbf{p}}_8 \hat{C}_1 \hat{C}_2) - \hat{\mathbf{s}}) / \hat{\mathbf{s}} \right\ _{\infty} < \hat{\epsilon}_v$	(VII)

Notes: $I = \{1, 2, 3, (4/7), (5/6), 8\}$

Element-wise vector division denoted as /
L-Infinity-Norm written as $\|\cdot\|_{\infty}$

Figure 10: Hierarchical Sizing Algorithm

found that this point yields good linearization results, since the circuit behavior is usually only weakly nonlinear in \mathcal{P} .

- II) *Determination of linearized circuit description via simulation:*
A polytopal approximation to the feasible parameter space \mathcal{P}_i of OTA i is given by \mathbf{C}_i and \mathbf{c}_i , cf. (7). The relation of block parameters \mathbf{p}_i and block performances \mathbf{f}_i is approximated by \mathbf{F}_i and $\mathbf{f}_i(\hat{\mathbf{p}}_i)$ according to (9).
- III) *Calculation of system sizing constraints:*
For each OTA, a polytopal representation of its feasible performance space \mathcal{F} according to (10) is determined using the method described in Sec. 3.3.
- IV) *System sizing:*
This sizing step is carried out entirely at system level. The associated sizing task can be interpreted mathematically as an optimization problem. The argmin operator yields the system parameter vector $\hat{\mathbf{f}}$ leading to the minimum relative deviation of the achieved performances from the specified values $\hat{\mathbf{s}}$ ³. The system sizing constraints limit the optimization engine to the feasible OTA performance spaces and to the upper and lower capacitance bounds.
- V) *Assignment of genuine system parameters:*
The determination of the genuine system parameters is trivial.
- VI) *Simultaneous block sizing:*
The block sizing step, the remaining system parameters $\hat{\mathbf{f}}_1 \dots \hat{\mathbf{f}}_8$ turn into the block specifications of the individual OTAs. One block-level sizing run per OTA yields the block parameter values.
- VII) *Check of exit conditions:*
If both the system sizing and the block sizing were successful, the filter performances can be simulated at block level to verify the sizing result⁴.

³The problem formulation above is deliberately simple for conceptual reasons. There are more sophisticated objective functions for solving vector optimization problems.

⁴Of course, further criteria are required to ensure reliable termination.

Exemplary Hierarchical Sizing Run

To illustrate the above algorithm, we trace a hierarchical sizing run of the above filter for the specifications

$$\hat{\mathbf{s}} = (\hat{f}_0 \hat{G}_0 \hat{Q}) = (1\text{MHz } 1.00 \ 50).$$

- First iteration

- The *system sizing* yields $\mathbf{s}(\hat{\mathbf{f}}) = (1.050\text{MHz } 1.020 \ 45.00)$. The maximum relative difference between the achieved system performances $\mathbf{s}(\hat{\mathbf{f}})$ and the system specifications $\hat{\mathbf{s}}$ is $\epsilon_s = 10\%$.
- The detailed results of the *block sizing* are omitted here for brevity. Instead, the table below gives the maximum relative difference between the achieved block performances and the block specifications for each OTA.

	OTA ₁	OTA ₂	OTA ₃	OTA _{4,7}	OTA _{5,6}	OTA ₈
ϵ_f [%]	28.6	2.0	1.9	2.0	2.1	2.1

- The *exit conditions* are not met: For $\hat{\epsilon}_s = \hat{\epsilon}_f = 2\%$, both the system sizing and the block sizing terminate with unacceptable errors. Consequently, another iteration is required.

- Second iteration

- *System sizing*:

$$\mathbf{s}(\hat{\mathbf{f}}) = (0.9974\text{MHz } 1.006 \ 49.87) \Rightarrow \epsilon_s = 0.6\%.$$

- *Block sizing*:

	OTA ₁	OTA ₂	OTA ₃	OTA _{4,7}	OTA _{5,6}	OTA ₈
ϵ_f [%]	0.23	0.14	0.12	0.13	0.13	0.11

- *Exit conditions*:

The sizing errors both at system and at block level meet the tolerances of $\hat{\epsilon}_s = \hat{\epsilon}_f = 1\%$. A final block-level simulation of the filter yields the system performances

$$\mathbf{s}(\hat{\mathbf{p}}_1 \dots \hat{\mathbf{p}}_8 \hat{C}_1 \hat{C}_2) = (0.9985\text{MHz } 1.002 \ 49.50)$$

$$\Rightarrow \epsilon_v = 1.0\% = \hat{\epsilon}_v$$

The final verification shows that the hierarchical sizing run of the filter has been finished successfully.

This example demonstrates how, in our hierarchical approach, iteration compensates for linearization inaccuracies. In the system sizing step in the first iteration, the polytopal system constraints do not allow the optimization engine to choose system parameter values, which would make the filter reach the specified goal. Furthermore, the specifications for OTA₁ turned out as overambitious.

However, new linearizations at the block-level sizing results $\hat{\mathbf{p}}_i$ yield an improved accuracy. In the second iteration, it turns out that the capabilities of the filter have been underestimated at system level previously. Furthermore, the system sizing step yields realistic block specifications for all OTAs.

6. CONCLUSIONS

We presented a simulation-based algorithm to calculate polytopal approximations to the feasible performance spaces of analog building blocks. It is especially suited for high-dimensional spaces, for which nonlinear techniques fail due to resource constraints. This performance space exploration technique is a key enabler for a successful hierarchical circuit sizing, because it propagates physically motivated constraints to the system level in a bottom-up fashion and thus prevents unachievable specifications for the block level. The algorithm relies on a polytopal approximation to the feasible parameter space and a linear description of the relation of parameters and performances. Using Gaussian and Fourier-Motzkin elimination, a polytopal representation of the feasible performance space is calculated. Special emphasis was put on a sophisticated redundancy detection, which compensates for

the complexity challenges of the Fourier-Motzkin algorithm. Experimental results demonstrate the efficiency of our algorithm and its applicability to topology selection and hierarchical sizing.

Acknowledgments

The authors would like to thank T. Theobald, TU Munich, for vital hints in the early research stages, G. Ziegler, TU Berlin, for helpful advice, and M. Oster, H. Caballero Figueroa and D. Mueller for their good work in the implementation phase at TU Munich.

This work was partially funded by edacentrum and by the German Federal Ministry of Education, Science, Research and Technology under grant 01M 3070C within the SAMS project.

7. REFERENCES

- [1] H. Chang, E. Charbon, U. Choudhury, A. Demir, E. Felt, E. Liu, E. Malavasi, A. Sangiovanni-Vincentelli, and I. Vasiliou. *A Top-Down, Constraint-Driven Design Methodology for Analog Integrated Circuits*. Kluwer Acad. Pub., 1997.
- [2] T. Christof and G. Reinelt. Combinatorial optimization and small polytopes. *Top (Spanish Statistical and Operations Research Society)*, 1996.
- [3] G. Dantzig and B. Eaves. Fourier-Motzkin elimination and its dual. *Journal of Combinatorial Theory (A)*, 1973.
- [4] F. De Bernardinis, M. I. Jordan, and A. Sangiovanni-Vincentelli. Support vector machines for analog circuit performance representation. In *DAC*, 2003.
- [5] M. del Mar Hershenson, S. P. Boyd, and T. H. Lee. GPCAD: A tool for CMOS op-amp synthesis. In *ICCAD*, 1998.
- [6] M. del Mar Hershenson, S. P. Boyd, and T. H. Lee. Optimal design of a CMOS Op-Amp via geometric programming. *IEEE Trans. CAD*, 2001.
- [7] A. Fuad Mas'ud, T. Ohtsuka, and H. Kunieda. Translation of specifications in hierarchical analog LSI design. In *IEEE ISCAS*, 1996.
- [8] G. J. Gómez, S. H. K. Embabi, E. Sánchez-Sinencio, and M. C. Lefebvre. A nonlinear macromodel for CMOS OTAs. In *IEEE ISCAS*, 1995.
- [9] H. Graeb, S. Zizala, J. Eckmueller, and K. Antreich. The sizing rules method for analog integrated circuit design. In *ICCAD*, 2001.
- [10] R. Harjani, R. Rutenbar, and L. Carley. OASYS: A framework for analog circuit synthesis. *IEEE Trans. CAD*, 1989.
- [11] R. Harjani and J. Shao. Feasibility and performance region modeling of analog and digital circuits. *Analog Integrated Circuits and Signal Processing*, 1996.
- [12] M. Jimenez, J. Llaberia, and A. Fernandez. Loop bounds computation for multilevel tiling. In *Sixth Euromicro Workshop on Parallel and Distributed Processing*, 1998.
- [13] B. D. Smedt and G. G. E. Gielen. WATSON: Design space boundary exploration and model generation for analog and RF IC design. *IEEE Trans. CAD*, 2003.
- [14] G. Stehr, H. Graeb, and K. Antreich. Performance trade-off analysis of analog circuits by normal-boundary intersection. In *DAC*, 2003.
- [15] G. Stehr, M. Pronath, F. Schenkel, H. Graeb, and K. Antreich. Initial sizing of analog integrated circuits by centering within topology-given implicit specifications. In *ICCAD*, 2003.
- [16] P. Veselinovic, D. Leenaerts, W. van Bokhoven, F. Leyn, G. Gielen, and W. Sansen. A flexible topology selection program as part of an analog synthesis system. In *European Design and Test Conference (ED&TC)*, 1995.
- [17] G. M. Ziegler. *Lectures on Polytopes*. Springer Verlag, New York, 1995.