Verification of the Power and Ground Grids Under General and Hierarchical Constraints

Mehmet Avci and Farid N. Najm, *Fellow, IEEE*

***Abstract*— As part of power distribution network verification, one should check if the voltage fluctuations exceed some critical threshold. The traditional simulation-based solution to this problem is intractable due to the large number of possible circuit behaviors. This approach also requires full knowledge of the details of the underlying circuitry, not allowing one to verify the power distribution network early in the design flow. Contrary to previous work on power distribution network verification, we consider the power and ground (P/G) grids together and describe an early verification approach under the framework of current constraints. Then, we present a solution technique in which tight lower and upper bounds on worst case voltage fluctuations are computed via linear programs. Experimental results indicate that the proposed technique results in errors in the range of a few millivolts. In addition to P/G grid verification techniques, we also provide very efficient solution technique to power (single) grid verification under hierarchical current constraints.**

***Index Terms*— Current constraints, hierarchical current constraints, linear programming (LP), power and ground (P/G) grid, sparse approximate inverse (SPAI) preconditioning, verification, voltage fluctuation.**

I. Introduction

THE feature size of modern integrated circuits (ICs) has been dramatically reduced in order to improve speed, power, and cost. The scaling of CMOS is expected to continue for at least another decade and future nanometer circuits will contain billions of transistors [1]. As CMOS technology is scaled, the power supply voltage will continue to decrease [1]. With reduced supply voltages and more functions integrated into ICs, the impact of voltage fluctuation is increasing and voltage integrity is becoming a big concern for chip designers.

There are many sources of on-chip voltage fluctuations, such as *IR*-drop, *Ldi/dt* drop, and the resonance between the on-chip grid and the package. Most available grid verification techniques use some form of circuit simulation to simulate the grid. Such an approach requires full knowledge of the current waveforms drawn by underlying transistor circuitry. These waveforms would then be used to simulate the grid and to find the voltage fluctuation at each node. However, since the number of possible circuit behaviors is very large, one needs to simulate the grid for a large number of vector sequences at each node, which is prohibitively expensive. Another disadvantage of the simulation-based approach is that it does not allow the designer to perform early grid verification, when grid modifications can be most easily done. To overcome these problems, we will adopt the notion of current constraints [2] to capture the uncertainty about the circuit details and behaviors. Under these constraints, grid verification becomes a problem of computing the worst case voltage fluctuations subject to current constraints.

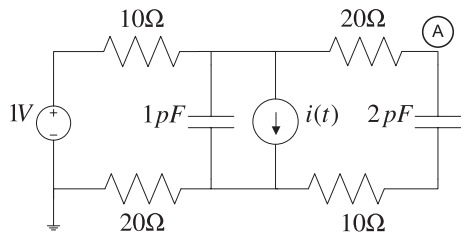

Fig. 1. Five-node grid.

In the literature, the ground grid has usually been assumed to be symmetric to the power grid. Popovich *et al.* [3] claim that the power and ground (P/G) grids have the same electrical requirements and therefore, the structures of these grids are often symmetric, particularly at the initial and intermediate phases of the design. They show that this symmetry can be exploited in a way to reduce the complexity of the power distribution network by an introduction of a virtual ground. The resulting circuit model contains two-independent symmetric grids, and therefore, the analysis of only one circuit is necessary. However, the assumption that the P/G grids are symmetric is not reliable, since even in initial stages of the design, some regions of the P/G grid are removed to make way for signal routing. This introduces nonsymmetry in the grid, which might lead to erroneous results if symmetry is assumed. We note in particular, that the presence of nonsymmetry can cause the voltage on a given node of the grid to fluctuate in both directions, i.e., voltage drop and overshoot, even for an *RC* grid (for an *RC* model of the power grid, voltage levels can normally only be below v_{dd}, under the assumption that the circuit does not inject current into the power grid). To see why, consider the simple unsymmetrical five-node grid shown in Fig. 1. Fig. 2 shows the current waveform assigned to the current source in the circuit and Fig. 3 shows the node

Manuscript received May 12, 2014; revised November 13, 2014 and January 22, 2015; accepted February 13, 2015. Date of publication March 31, 2015; date of current version January 19, 2016. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, in part by Semiconductor Research Corporation, and in part by Intel Corporation.

M. Avci was with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada. He is now with the Toronto Technology Center, Altera Corporation, Toronto, ON M5S 1S4, Canada (e-mail: mavci@altera.com).

F. N. Najm is with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mail: f.najm@utoronto.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2015.2413966

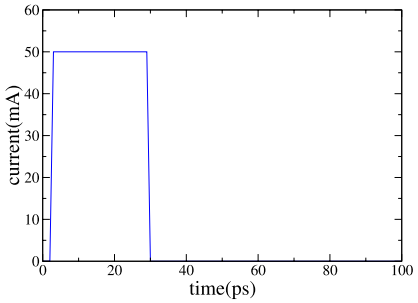

Fig. 2. Current configuration assigned to the current source.

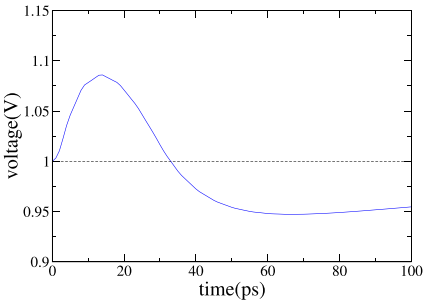

Fig. 3. Resulting voltage overshoot for node A.

voltage at node A as a result of an HSPICE simulation. The simulation shows an overshoot where the node voltage at node A goes above v_{dd}. Therefore, there is a necessity to verify the P/G grids together (i.e., a P/G grid verification), which is the main focus of this paper. In the remainder of this paper, we will refer to P/G networks as P/G grid, and to only power network with no ground network as power grid.

In addition to P/G grid verification techniques, we also show that a reduction in the complexity of power grid verification is possible under hierarchical current constraints. In the literature, there are proposed methods to tackle the complexity of the power grid verification problem under hierarchical constraints [4]–[6]. In this paper, we consider special hierarchical current constraints, whereby global constraints are required to be wholly contained in other global constraints, and we propose a solution based on sorting and deletion, which is significantly more efficient than standard linear program solvers.

The remainder of this paper is organized as follows. In Section II-A, we present the P/G grid model and formulate the problem under the notion of current constraints. Section II-B proposes an efficient solution for the lower and upper bounds on the worst case voltage fluctuations of the P/G grid. Implementation details are given in Section II-C, followed by the experimental results in Section II-D. In Section III, we propose a fast technique to solve power grid problem under hierarchical current constraints, and we also show experimental results of the proposed method. Finally, the conclusion is given in Section IV.

II. P/G Grid Voltage Integrity Verification

A. Problem Formulation

1) P/G Grid Model: We consider an *RC* model of the P/G grid where each branch is represented either by a resistor or by a capacitor. We define the nodes that are on the power grid as power grid nodes, and similarly, the nodes that are on the ground grid as ground grid nodes. Resistors are located between two power grid nodes or between two ground grid nodes, i.e., no resistor exists between a P/G grid node. In addition, the capacitors are located only between P/G grid nodes and, unlike previous work, we assume that a node can have multiple capacitors.

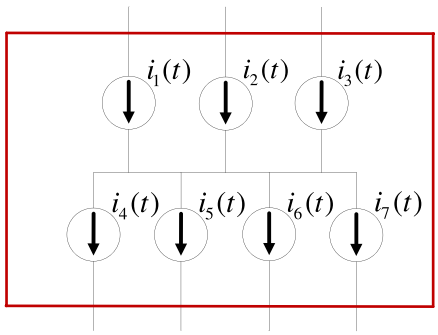

Fig. 4. Macroblock model.

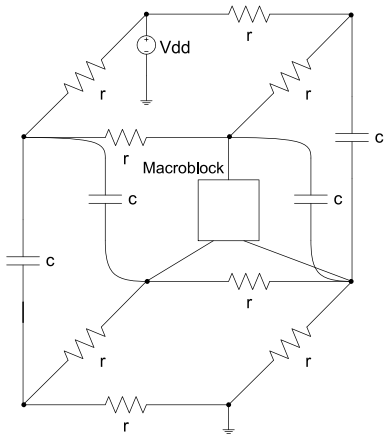

Fig. 5. P/G grid model.

One common simplification in the literature is to model the current drawn by the underlying transistor circuitry in a logic block as a single current source. We usually know the current drawn by a logic block, but that logic block is usually attached to multiple nodes in the grid. Therefore, modeling the current drawn by a logic block as a single current source is not valid, and yields pessimistic voltage fluctuations. In order to capture this notion, we introduce the model of a macroblock, which groups multiple current sources into a single block, as shown in Fig. 4.

The macroblock model in Fig. 4 captures the true behavior of a logic block, in the sense that it draws current from multiple nodes. Note that we do not require having the same number of current sources for P/G grid nodes. However, for each macroblock, we have to ensure that the current leaving power grid nodes must equal the current entering ground grid nodes. This will be an important equality constraint that defines the feasibility space of currents.

A simple P/G grid is shown in Fig. 5. Notice that the macroblock has multiple connections to the grid, and that some nodes have multiple capacitors attached.

2) System Equations: Let the P/G grid consist of $n+\alpha$ nodes, where nodes $1, 2, \ldots, n$ have no voltage sources attached, and nodes $(n+1), \ldots, (n+\alpha)$ are connected to voltage sources. Following the approach in [7], the Modified Nodal Analysis equation that describes the network can be written as:

$$\tilde{G}\tilde{u}(t) + \tilde{C}\dot{\tilde{u}}(t) = \tilde{i}(t) + \tilde{G}_0 v_{\mathrm{dd}(n)} \tag{1}$$

where $\tilde{G}$ and $\tilde{G}_0$ are $n \times n$ conductance matrices and $\tilde{C}$ is an $n \times n$ capacitance matrix. $\tilde{u}(t)$ is an $n \times 1$ vector of node voltages except the nodes that are connected to voltage sources. $\tilde{i}(t)$ is $n \times 1$ vector of current sources and $v_{\mathrm{dd}(n)}$ is an $n \times 1$ vector whose each entry is equal to the value of the supply voltage. The node voltages $\tilde{u}(t)$ are with respect to some reference (datum) node that is part of the ground grid.

Let h be the number of power grid nodes that are not connected directly to a voltage source, and l be the number of ground grid nodes that are not ground, so that $h + l = n$. We can rewrite (1) by partitioning the vector of node voltages with respect to P/G grid nodes, and reordering the rows and columns of $\tilde{G}$, $\tilde{C}$, and $\tilde{G}_0$ and the entries of $\tilde{i}(t)$ accordingly as follows:

$$\begin{bmatrix} G_p & 0 \\ 0 & G_g \end{bmatrix}\begin{bmatrix} u_p(t) \\ u_g(t) \end{bmatrix} + \begin{bmatrix} C_p & N \\ N^T & C_g \end{bmatrix}\begin{bmatrix} \dot{u}_p(t) \\ \dot{u}_g(t) \end{bmatrix} = \begin{bmatrix} -i_p(t) \\ i_g(t) \end{bmatrix} + G_0 v_{\mathrm{dd}(n)}. \tag{2}$$

Since no resistor exists between P/G grid nodes, $\tilde{G}$ can be partitioned into two submatrices G_p ($h \times h$ matrix) and G_g ($l \times l$ matrix). $i_p(t)$ ($h \times 1$ vector) and $i_g(t)$ ($l \times 1$ vector) are nonnegative vectors defining the current sources attached to P/G grid nodes, respectively. Since capacitors exist only between P/G grid nodes, C_p ($h \times h$ matrix) and C_g ($l \times l$ matrix) are nonnegative diagonal matrices, and N is an $h \times l$ nonpositive matrix.

If we set all current sources to 0, $\forall t$, then $u_p(t) = v_{\mathrm{dd}(h)}$ and $u_g(t) = 0$, $\forall t$, where $v_{\mathrm{dd}(h)}$ is an $h \times 1$ vector whose each entry is equal to the value of the supply voltage. For this case, the system of equations becomes

$$\begin{bmatrix} G_p & 0 \\ 0 & G_g \end{bmatrix}\begin{bmatrix} v_{\mathrm{dd}(h)} \\ 0 \end{bmatrix} = G_0 v_{\mathrm{dd}(n)}. \tag{3}$$

Substituting $G_0 v_{\mathrm{dd}(n)}$ from (3) into (2) and rearranging the terms, we obtain

$$\begin{bmatrix} G_p & 0 \\ 0 & G_g \end{bmatrix}\begin{bmatrix} u_p(t) - v_{\mathrm{dd}(h)} \\ u_g(t) \end{bmatrix} + \begin{bmatrix} C_p & N \\ N^T & C_g \end{bmatrix}\begin{bmatrix} \dot{u}_p(t) \\ \dot{u}_g(t) \end{bmatrix} = \begin{bmatrix} -i_p(t) \\ i_g(t) \end{bmatrix}. \tag{4}$$

Defining $v_p(t) = v_{\mathrm{dd}(h)} - u_p(t)$ to be the vector of voltage drops at power grid nodes, we can write (4) as

$$G_p v_p(t) + C_p \dot{v}_p(t) - N\dot{u}_g(t) = i_p(t) \tag{5}$$

$$G_g u_g(t) + C_g \dot{u}_g(t) - N^T \dot{v}_p(t) = i_g(t). \tag{6}$$

In matrix notation, (5) and (6) can be combined to yield

$$\begin{bmatrix} G_p & 0 \\ 0 & G_g \end{bmatrix}\begin{bmatrix} v_p(t) \\ u_g(t) \end{bmatrix} + \begin{bmatrix} C_p & -N \\ -N^T & C_g \end{bmatrix}\begin{bmatrix} \dot{v}_p(t) \\ \dot{u}_g(t) \end{bmatrix} = \begin{bmatrix} i_p(t) \\ i_g(t) \end{bmatrix}. \tag{7}$$

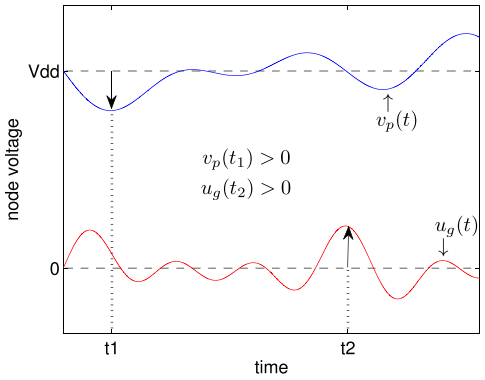

Fig. 6. Voltages on the P/G grid.

In this notation, $v_p(t)$ is positive when power grid nodes experience undershoots and $u_g(t)$ is positive when ground grid nodes experience overshoots, as shown in Fig. 6.

Define

$$\hat{G} = \begin{bmatrix} G_p & 0 \\ 0 & G_g \end{bmatrix}, \quad \hat{v}(t) = \begin{bmatrix} v_p(t) \\ u_g(t) \end{bmatrix}$$

$$\hat{C} = \begin{bmatrix} C_p & -N \\ -N^T & C_g \end{bmatrix}, \quad \hat{i}(t) = \begin{bmatrix} i_p(t) \\ i_g(t) \end{bmatrix}.$$

So that (7) becomes

$$\hat{G}\hat{v} + \hat{C}\dot{\hat{v}}(t) = \hat{i}(t). \tag{8}$$

Notice that the circuit described by (8) is the original P/G grid, but with all the voltage sources set to zero and the directions of current sources attached to the power grid nodes reversed. Furthermore, this equation is useful, because the current vector $\hat{i}(t)$ is a nonnegative vector and the matrix $\hat{C}$ consists of only nonnegative elements.

Assume that only m of n nodes of the P/G grid have current sources attached. Then, we can reorder the rows and columns of the matrices and the entries of the vectors in (8) to yield

$$Gv(t) + C\dot{v}(t) = \begin{bmatrix} i(t) \\ 0_{(n-m)} \end{bmatrix} = \bar{i}(t) \tag{9}$$

where G and C are matrices of size $n \times n$, which are simply reordered replicas of $\hat{G}$ and $\hat{C}$. $i(t)$ is the vector of size m representing the current loads, and $0_{(n-m)}$ is the zero vector of size $n - m$. Finally, using the backward Euler formula, (9) can be discretized in time as

$$Av(t) = Bv(t - \Delta t) + \bar{i}(t) \tag{10}$$

where $A = (G + (C/\Delta t))$ and $B = (C/\Delta t)$.

3) Current Constraints: We adopt the notion of current constraints in order to perform verification of the P/G grid. This approach [2] does not require complete information about the currents drawn by the underlying circuitry, and may be called a vectorless approach. The currents are typically hard to specify for at least two reasons. First, the number of combinations of possible current waveforms is very large, and simulation of a large set of waveforms is very time-consuming. Second, the simulation approach does not allow the designer to verify the grid early in the chip design. For the simulation, the details of the underlying circuitry must be already known,

but it might not be available or complete early in the design, when most of the major changes in grid characteristics can be most easily incorporated.

We use three types of constraints: 1) local constraints; 2) global constraints; and 3) equality constraints. Local constraints define upper bounds on individual current sources. They can be expressed mathematically as

$$0 \leq i(t) \leq i_L \tag{11}$$

where i_L is a vector of size m and stands for the peak value of currents that the current sources can draw. In this paper, we restrict our work to the case of dc constraints, i.e., the upper bound is fixed over time. However, note that it is only the constraints that are dc, the currents themselves are transient. An alternative is to use transient current constraints, which are more difficult to use in practice, both from the user's standpoint (supplying transient constraints) and the verification tools that would deal with them.

Local constraints do not completely capture the behavior of the grid, because it is never the case that all chip components draw their currents simultaneously. Therefore, we need global constraints, which are upper bounds on the sums of groups of current sources. They might represent the maximum current that the group of current sources in each macroblock can draw or the peak total power dissipation of a group of macroblocks. If we assume that we have μ global constraints, then they can be expressed as

$$0 \leq Ui(t) \leq i_G \tag{12}$$

where U is a $\mu \times n$ matrix that consists only of 0 and 1 s. If a 1 is present in a row of U, it indicates that the corresponding current source is included in that global constraint. Similar to the case of local constraints, i_G is a constant time-independent dc constraint, but the currents themselves are transient waveforms.

As previously mentioned, we need to ensure that the currents leaving the power grid are equal to currents entering the ground grid, which we will call an equality constraint. If we assume that we have γ macroblocks, then the equality constraints can be expressed as

$$Mi(t) = 0 \tag{13}$$

where M is a $\gamma \times n$ matrix that consists only of 1, -1, and 0 s. For each macroblock, 1 s corresponds to current sources that are attached to the power grid, and -1 s corresponds to current sources that are attached to the ground grid.

To simplify the notation, we use $\mathcal{F}$ to denote the feasible space of currents, so that $i(t) \in \mathcal{F}$ if and only if it satisfies (11)–(13) at all time.

4) Problem Definition: Our problem is to find, for every node, the worst case node voltage fluctuation over all possible currents in $\mathcal{F}$. To simplify the notation, let $E = A^{-1}$ and $D = A^{-1}B$, so that we can write (10) as

$$v(t) = Dv(t - \Delta t) + E\bar{i}(t). \tag{14}$$

We first write the matrix E as follows:

$$E = [e_1, e_2, \ldots, e_n] \tag{15}$$

where e_i is the ith column of E. Define

$$H = [e_1, e_2, \ldots, e_m] \tag{16}$$

where H is $n \times m$ matrix formed by the first m columns of E. Since we know that the last $n - m$ elements in $\bar{i}(t)$ are 0, we can write (14) as

$$v(t) = Dv(t - \Delta t) + Hi(t). \tag{17}$$

Now consider the case in which the grid had no stimulus for $t \leq 0$, which leads to $v_0 = v(0) = 0$. Then, writing at time Δt, $2\Delta t$, and $3\Delta t$, we obtain

$$v(\Delta t) = Dv_0 + Hi(\Delta t) = Hi(\Delta t) \tag{18}$$

$$v(2\Delta t) = Dv(\Delta t) + Hi(2\Delta t) = DHi(\Delta t) + Hi(2\Delta t) \tag{19}$$

$$\begin{aligned} v(3\Delta t) &= Dv(2\Delta t) + Hi(3\Delta t) \\ &= D^2Hi(\Delta t) + DHi(2\Delta t) + Hi(3\Delta t). \end{aligned} \tag{20}$$

Repeating this procedure for any future time $p\Delta t$, we have

$$v(p\Delta t) = \sum_{k=0}^{p-1} D^k Hi((p-k)\Delta t). \tag{21}$$

At every point in time $t \in [0, p\Delta t]$, the input vector $i(t)$ must be feasible, i.e., we must have $i(t) \in \mathcal{F}$. Under these conditions, we are interested in the worst case voltage fluctuations attained (separately) by each component of $v(p\Delta t)$. In order to capture this notion, we use the following notation, introduced in [8].

Suppose $f(c) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a vector function whose components are denoted $f_1(c), \ldots, f_n(c)$, and let $\mathcal{A} \subset \mathbb{R}^n$. Now, define a vector $x \in \mathbb{R}^n$, such that, with $i \in \{1, 2, \ldots, n\}$, and x_i is the maximum of $f_i(c)$ over all $c \in \mathcal{A}$. We denote this by the following operator:

$$x = \operatorname*{emax}_{\forall c \in \mathcal{A}}(f(c)). \tag{22}$$

Notice that each component $x_i, \forall i = 1, \ldots, n$ may be found separately by solving the following maximization problem:

$$\begin{aligned} &\text{maximize: } f_i(c) \\ &\text{s.t.: } c \in \mathcal{A}. \end{aligned} \tag{23}$$

Similarly, define a vector $y \in \mathbb{R}^n$, such that y_i is the minimum of $f_i(c)$ over all $c \in \mathcal{A}$. We denote this by the following operator:

$$y = \operatorname*{emin}_{\forall c \in \mathcal{A}}(f(c)) \tag{24}$$

and each component $y_i, \forall i = 1, \ldots, n$ may be found separately by solving the following minimization problem:

$$\begin{aligned} &\text{minimize: } f_i(c) \\ &\text{s.t.: } c \in \mathcal{A}. \end{aligned} \tag{25}$$

Using the emax(·) and emin(·) operators, we can express the worst case voltage fluctuation at all nodes at time $p\Delta t$ by

$$v^+(p\Delta t) = \operatorname*{emax}_{\forall i(t)\in\mathcal{F}} \left(\sum_{k=0}^{p-1} D^k H i((p-k)\Delta t) \right) \quad (26)$$

$$v^-(p\Delta t) = -\operatorname*{emin}_{\forall i(t)\in\mathcal{F}} \left(\sum_{k=0}^{p-1} D^k H i((p-k)\Delta t) \right) \quad (27)$$

where the notation $\forall i(t) \in \mathcal{F}$ means that, for every time point $t \in [0, p\Delta t]$, the current vector $i(t)$ satisfies all the (local, global, and equality) constraints. $v^+(t)$ is a nonnegative vector defining the worst case voltage drops on power grid nodes and the worst case voltage overshoots on ground grid nodes, and similarly, $v^-(t)$ is a nonnegative vector defining the worst case voltage overshoots on power grid nodes and the worst case voltage drops on ground grid nodes. We used a minus sign in front of emin(·) operator in (27) to avoid confusion about the notion of the lower and upper bounds in the rest of this paper. Using $v^+(t)$ and $v^-(t)$, $v(t)$ can be bounded as

$$-v^-(t) \le v(t) \le v^+(t). \quad (28)$$

Although the *RC* model is dynamic, i.e., its currents and voltages vary with time, the constraints are dc and do not depend on time. Hence, $\mathcal{F}$ is the same for each time step. With this, the components of (26) and (27) can be decoupled [8], leading to

$$v^+(p\Delta t) = \sum_{k=0}^{p-1} \operatorname*{emax}_{\forall i\in\mathcal{F}} [D^k H i] \quad (29)$$

$$v^-(p\Delta t) = -\sum_{k=0}^{p-1} \operatorname*{emin}_{\forall i\in\mathcal{F}} [D^k H i] \quad (30)$$

where i is simply an $m \times 1$ vector of variables that satisfies the (local, global, and equality) constraints, without reference to any particular point in time. This is an important simplification of the problem, as it has the advantage that the number of constraints for each optimization is fixed and does not span all previous time points. The advantage of using the matrix H instead of E is clear now, since at each time step, one needs to compute multiplication of an $n \times n$ matrix by an $n \times m$ matrix instead of two $n \times n$ matrices. Furthermore, the optimization variables do not include the redundant variables.

Claim 1: $\operatorname*{emax}_{\forall i\in\mathcal{F}} [D^k H i] \ge 0$ and $\operatorname*{emin}_{\forall i\in\mathcal{F}} [D^k H i] \le 0$, $\forall k$.

Proof: Let $x^* = \operatorname*{emax}_{\forall i\in\mathcal{F}} [D^k H i]$ and $y^* = \operatorname*{emin}_{\forall i\in\mathcal{F}} [D^k H i]$. Assume that the claim is not true, i.e., $x_j^* < 0$ and $y_j^* > 0$. Since $i^\dagger = 0$ is feasible, i.e., $i^\dagger \in \mathcal{F}$, we can choose $i^\dagger$, so that $(D^k H i^\dagger)_j = 0 > x_j^*$ and $(D^k H i^\dagger)_j = 0 < y_j^*$, which is a contradiction. This completes the proof. ∎

Using (29), (30), and Claim 1, and taking the difference in two consecutive time steps, we have

$$v^+(p\Delta t) - v^+((p-1)\Delta t) = \operatorname*{emax}_{\forall i\in\mathcal{F}} [D^{p-1} H i] \ge 0 \quad (31)$$

$$v^-(p\Delta t) - v^-((p-1)\Delta t) = -\operatorname*{emin}_{\forall i\in\mathcal{F}} [D^{p-1} H i] \ge 0 \quad (32)$$

meaning that $v^+(p\Delta t)$ and $v^-(p\Delta t)$ are monotone nondecreasing functions of the time point p, for any integer $p \ge 1$.

In practice, we are interested in the steady-state solution, where the system becomes independent of the initial condition $v(t) = 0, \forall t \le 0$. Since the *RC* grid model is a dynamical system with a limited memory of its past, the steady-state solution can be obtained by evaluating (29) and (30) at points far away from the initial condition, i.e., as $p \to \infty$. Thus, the general solution to the problem is

$$v^+(\infty) = \lim_{p\to\infty} \sum_{k=0}^{p-1} \operatorname*{emax}_{\forall i(t)\in\mathcal{F}} [D^k H i] \quad (33)$$

$$v^-(\infty) = -\lim_{p\to\infty} \sum_{k=0}^{p-1} \operatorname*{emin}_{\forall i(t)\in\mathcal{F}} [D^k H i]. \quad (34)$$

B. Proposed Solution

Using (33) and (34) is intractable, because they have to be evaluated for a large number of time steps until convergence is achieved and the emax(·) and emin(·) operators require linear programs proportional to the number of nodes in the grid, which for modern designs is in the order of millions. As an alternative, we propose an efficient solution to compute lower and upper bounds on the worst case voltage fluctuations of the P/G grid.

1) Vector of Lower Bounds: We show that, for specific initial conditions, both $v^+(t)$ and $v^-(t)$ will be monotone nondecreasing functions of time t. We have found that the dc verification solution of the grid is a good initial condition that satisfies the monotonicity property.

a) Nonzero initial conditions: We start by investigating the impact of starting the verification with different (nonzero) initial conditions on the worst case voltage fluctuations. If we have the initial condition v_0 at time $t = 0$, then the voltage on the grid at any future time $p\Delta t$ can be expressed as

$$v(p\Delta t) = D^p v_0 + \sum_{k=0}^{p-1} D^k H i((p-k)\Delta t). \quad (35)$$

Because the *RC* grid is a stable linear system and because the backward difference approximation used in (10) is absolutely stable [9], it follows that for $i(t) = 0, \forall t$ and any bounded initial condition v_0, (35) converges to 0 as $t \to \infty$. For $i(t) = 0, \forall t$, the voltage on the grid at any time $p\Delta t$ can be written as

$$v(p\Delta t) = D^p v_0. \quad (36)$$

Writing (36) as $p \to \infty$, we get

$$v(\infty) = \lim_{p\to\infty} v(p\Delta t) = \lim_{p\to\infty} D^p v_0 = 0. \quad (37)$$

Since (37) is valid for any bounded initial condition, it is clear that $D^p \to 0$ as $p \to \infty$. We use the following theorem [10] to conclude that this actually means $\rho(D) < 1$, where $\rho(D)$ is the magnitude of the largest eigenvalue of D, also called the spectral radius of D.

Theorem 1: Let D be a square matrix. Then, the sequence D^k, for $k = 0, 1, \ldots$, converges to zero if and only if $\rho(D) < 1$.

It is easy to see that at steady state, i.e., as $p \to \infty$, choosing a different initial condition other than $v_0 = 0$ does not have any impact on $v(\infty)$, because of the fact that D^p converges to zero as $p \to \infty$. Therefore, (21) and (35) become equivalent as $p \to \infty$. Using the notation in Section II-A-4 and using the fact that $\mathcal{F}$ is the same for each time step, we can express the worst case voltage fluctuations at time point $p\Delta t$ with the initial condition v_0 as

$$v^+(p\Delta t) = D^p v_0 + \sum_{k=0}^{p-1} \underset{\forall i \in \mathcal{F}}{\text{emax}}[D^k H i] \tag{38}$$

$$v^-(p\Delta t) = D^p v_0 - \sum_{k=0}^{p-1} \underset{\forall i \in \mathcal{F}}{\text{emin}}[D^k H i]. \tag{39}$$

b) Monotone nondecreasing $v^+(t)$: Under the dc model of the P/G grid, (9) becomes

$$Gv = \bar{i}. \tag{40}$$

Let $L = G^{-1}$ and let K be a $n \times m$ matrix, which is obtained as the first m columns of L, such that

$$K = [l_1, l_2, \ldots, l_m] \tag{41}$$

where l_i defines ith column of L. Using the matrix K, we can write $v = Ki$. Assume that we have the dc solution of the system as the initial condition, leading to

$$v_0 = Ki_0. \tag{42}$$

We define the voltage vector v_0 to be feasible, if it satisfies (42) for a current vector $i_0 \in \mathcal{F}$.

Claim 2: If v_0 is feasible, then $v^+(p\Delta t)$ given in (38) is a monotone nondecreasing function of the time point p, for any integer $p \geq 1$.

Proof: The claim is true if we can show that $v^+(p\Delta t) \geq v^+((p-1)\Delta t)$, for any integer $p \geq 1$. Substituting v_0 from (42) into (38), we obtain

$$v^+(p\Delta t) = D^p K i_0 + \sum_{k=0}^{p-1} \underset{\forall i \in \mathcal{F}}{\text{emax}}[D^k H i]. \tag{43}$$

Substituting $D^p K$ from (86) (Appendix) into (43), we obtain

$$v^+(p\Delta t) = \left(K - \sum_{k=0}^{p-1} D^k H \right) i_0 + \sum_{k=0}^{p-1} \underset{\forall i \in \mathcal{F}}{\text{emax}}[D^k H i]. \tag{44}$$

Taking the difference in two consecutive time steps, we have

$$v^+(p\Delta t) - v^+((p-1)\Delta t) = \underset{\forall i \in \mathcal{F}}{\text{emax}}[D^{p-1} H i] - D^{p-1} H i_0. \tag{45}$$

We see in (45) that the emax$(\cdot)$ operator assigns the maximum value to the first term on the right-hand side over all $i \in \mathcal{F}$, whereas the second term on the right-hand side has the variables $i_0 \in \mathcal{F}$, which may not result in the optimal solution. Therefore, we conclude that if v_0 is feasible, then $v^+(p\Delta t) \geq v^+((p-1)\Delta t)$, for any integer $p \geq 1$. This completes the proof. ■

c) DC initial condition: Using the notation $(X)_j$ to define jth row of a matrix X and incorporating (38), (39), and (42), we can express the worst case voltage fluctuation for the jth node at time $p\Delta t$ as

$$v_j^+(p\Delta t) = (D^p K)_j i_0 + \sum_{k=0}^{p-1} \max_{\forall i \in \mathcal{F}} (D^k H)_j i \tag{46}$$

$$v_j^-(p\Delta t) = (D^p K)_j i_0 - \sum_{k=0}^{p-1} \min_{\forall i \in \mathcal{F}} (D^k H)_j i. \tag{47}$$

A good choice of i_0 for a given node would be the current combination that leads to the worst case voltage fluctuation for that node under the dc model of the P/G grid. The worst case voltage fluctuation for the jth node under the dc model is given by

$$v_j^+ = \max_{\forall i \in \mathcal{F}} (K)_j i \tag{48}$$

$$v_j^- = -\min_{\forall i \in \mathcal{F}} (K)_j i. \tag{49}$$

Denote the optimal value of i of the maximization problem in (48) as i_j^+ ($m \times 1$ vector) and that of the minimization problem in (49) as i_j^- ($m \times 1$ vector). Since G is an M-matrix [8], its inverse consists only of nonnegative elements, which means that K is a nonnegative matrix. Therefore, the result of the minimization problem in (49) under nonnegative current constraints will be 0, which leads to $i_j^- = 0$. Using i_j^+ and i_j^- as the initial current at $t = 0$ for the jth node, (46) and (47) become

$$v_j^+(p\Delta t) = (D^p K)_j i_j^+ + \sum_{k=0}^{p-1} \max_{\forall i \in \mathcal{F}} (D^k H)_j i \tag{50}$$

$$v_j^-(p\Delta t) = -\sum_{k=0}^{p-1} \min_{\forall i \in \mathcal{F}} (D^k H)_j i. \tag{51}$$

d) Lower bound: Algorithm 1 describes the computation of the lower bound vector in detail, based on (50) and (51). Using Claim 1 and Claim 2, we can see that $v_j^+(p\Delta t)$ in (50) and $v_j^-(p\Delta t)$ in (51) are monotone nondecreasing functions of the time point p, for any integer $p \geq 1$. Since we stop the main loop of Algorithm 1 for a finite p, when $v^+(p\Delta t) - v^+((p-1)\Delta t)$ and $v^-(p\Delta t) - v^-((p-1)\Delta t)$ are less than a threshold ϵ, then $v_{lb}^+(\infty)$ and $v_{lb}^-(\infty)$ are clearly lower bounds on $v^+(\infty)$ and $v^-(\infty)$, respectively.

To see the advantage of starting the verification with a dc initial condition, consider Figs. 7 and 8. Fig. 7 shows $v_j^+(p\Delta t)$ for a particular node j on a P/G grid with the initial condition $v_0 = 0$, whereas Fig. 8 shows $v_j^+(p\Delta t)$ for the same node with the initial condition $v_0 = Ki_j^+$. Clearly, starting the verification with dc initial condition requires less number of times steps to converge to steady-state value than starting the verification with zero initial conditions.

2) Vector of Upper Bounds: Following an approach similar to [11], we compute an upper bound on the worst case voltage fluctuations of the P/G grid. Although the upper bound in [11] was derived for an RLC model of the power grid, it can be shown that it is also valid for the P/G grid model presented

Algorithm 1 Lower Bound Algorithm

Inputs: $K, D, H, \mathcal{F}$ and ϵ
Outputs: $v_{lb}^{+}(\infty)$ and $v_{lb}^{-}(\infty)$

Compute $v^{+} = \underset{\forall i \in \mathcal{F}}{\text{emax}}\,[Ki]$ and save i_j^{+} for each node
Set $p = 0$ and `stop_flag` = **false**
while `stop_flag` = **false do**
  $p = p + 1$
  for $j = 1, \ldots, n$ **do**
    Compute $v_j^{+}(p\Delta t)$ using (50) and $v_j^{-}(p\Delta t)$ using (51)
  end for
  if $v^{+}(p\Delta t) - v^{+}((p-1)\Delta t) < \epsilon$ **then**
    if $v^{-}(p\Delta t) - v^{-}((p-1)\Delta t) < \epsilon$ **then**
      Set `stop_flag` = **true**
      Set $v_{lb}^{+}(\infty) = v^{+}(p\Delta t)$ and $v_{lb}^{-}(\infty) = v^{-}(p\Delta t)$
    end if
  end if
end while

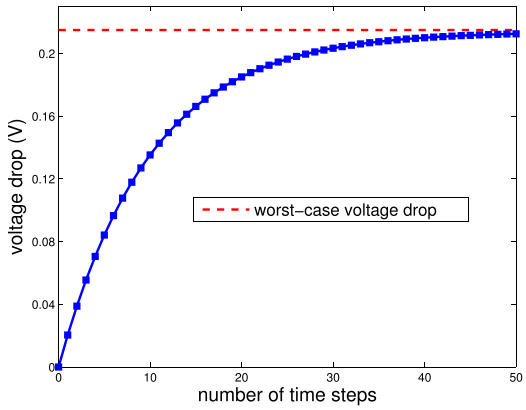

Fig. 7. $v_j^{+}(p\Delta t)$ with zero initial condition.

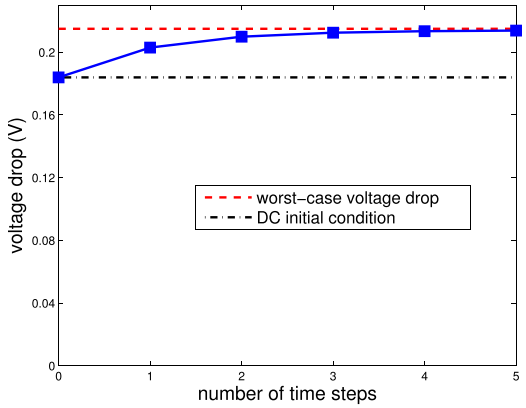

Fig. 8. $v_j^{+}(p\Delta t)$ with dc initial condition.

in this paper. Due to space considerations, and because it is mostly an adaptation of [11] to the P/G grid case, we will not show the background work associated with the derivation of the upper bound. Instead, we simply describe the computation of the upper bound on the worst case voltage fluctuations

Algorithm 2 Upper Bound Algorithm

Inputs: D, H and $\mathcal{F}$
Outputs: $v_{ub}^{+}(\infty)$ and $v_{ub}^{-}(\infty)$

Set $Q = I$, $s^{+} = 0$ and $s^{-} = 0$
while $\|Q\|_1 \geq 1$, $\|Q\|_\infty \geq 1$ and $\|Q\|_F \geq 1$ **do**
  $s^{+} := s^{+} + \underset{\forall i \in \mathcal{F}}{\text{emax}}\,[QHi]$
  $s^{-} := s^{-} - \underset{\forall i \in \mathcal{F}}{\text{emin}}\,[QHi]$
  $Q := QD$
end while
Set $R = \frac{1}{2}(|Q| - Q)$
Set $P = \begin{bmatrix} Q+R & R \\ R & Q+R \end{bmatrix}$
Compute $v_{ub}^{+}(\infty)$ and $v_{ub}^{-}(\infty)$ using:
  LU-factorize $(I - P)$: $(I - P) = L \cdot U$
  Forward solve: $Lw = \begin{bmatrix} s^{+} \\ s^{-} \end{bmatrix}$
  Backward solve: $U \begin{bmatrix} v_{ub}^{+}(\infty) \\ v_{ub}^{-}(\infty) \end{bmatrix} = w$

in Algorithm 2, in which we use the notation $|X|$ to denote the matrix of the element-wise absolute values of the entries of a matrix X. Our main contribution in Algorithm 2 is the addition of the convergence criterion $\|Q\|_F \geq 1$, which significantly reduces the number of execution of the main loop. Further details can be found in [11].

C. Implementation

1) Inverse Approximation Method: It is obvious from Section II-B that the inverse of the system matrix A is needed for the computation of the lower and upper bounds, because $D = A^{-1}B$. For the lower bound, we need to invert the conductance matrix G as well as A. We now explain how this can be efficiently done.

Power distribution networks have a mesh structure, where a node has a small number of neighboring nodes. Such a structure results in a matrix (i.e., A or G) that is sparse, symmetric, positive definite, and banded. In the literature, it is well known that the inverse of a nonsingular sparse matrix is dense. In particular, a matrix that results from a mesh structure has the inverse that is almost full. However, it is also well known in the literature that the inverse of a sparse, symmetric, positive definite, and banded matrix has entries whose values decay exponentially as one moves away from the diagonal [12]. This fact is the main idea of constructing sparse approximate inverse (SPAI) preconditioners to precondition large sparse linear systems when using an iterative method, such as conjugate gradient method. SPAI preconditioners try to find a good SPAI M, such that $MA \approx I$, where I denotes the identity matrix.

There are numerous SPAI techniques in the literature. One of them is SPAI [13] that is based on Frobenius norm minimization. SPAI starts with an arbitrary initial matrix M and iteratively refines its columns by minimizing the Frobenius norm $\|MA - I\|_F$. This technique has been applied to power

distribution network verification in [8]. Another technique is called AINV [14], which is based on the conjugate Gram–Schmidt (or A-orthogonalization) process. AINV has the advantage of using the fact that the matrix whose inverse is to be approximated is symmetric positive definite, while SPAI is a general sparse approximate preconditioner for unsymmetric matrices.

In what follows, we briefly explain the AINV method given in [14]. Assume that A is an $n \times n$ symmetric positive definite matrix. It is shown in [14] that the factorization of A^{-1} can be obtained from a set of conjugate directions $z_1, z_2, \ldots, z_n$ for A. By writing a set of conjugate directions in matrix form, we have

$$Z = [z_1, z_2, \ldots, z_n] \tag{52}$$

where Z is the matrix whose ith column is z_i. Knowing a set of conjugate directions for A, we can write

$$Z^T AZ = D = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{bmatrix} \tag{53}$$

where $d_i = z_i^T Az_i$. To obtain a factorization of A^{-1}, we write

$$A^{-1} = ZD^{-1}Z^T. \tag{54}$$

Notice in (54) that the inverse of A can be obtained easily if we know a set of conjugate directions $z_1, z_2, \ldots, z_n$. In [14], a set of conjugate directions is constructed by means of a conjugate Gram–Schmidt (or A-orthogonalization) process applied to any set of linearly independent vectors $v_1, v_2, \ldots, v_n$. The Gram–Schmidt process is a method for orthogonalizing a set of vectors in an inner product space of size n. It takes a finite, linearly independent set of vectors $v_1, v_2, \ldots, v_n$ and generates an orthogonal set $u_1, u_2, \ldots, u_n$ that spans the same inner product space n. For further details on the Gram–Schmidt process, the reader is referred to [15].

Since the Gram–Schmidt process can be applied to any set of linearly independent vectors, it is convenient to choose $v_i = e_i$, where e_i is the ith unit vector. From the Cholesky factorization of A, we have

$$A = LDL^T \tag{55}$$

where L is unit lower triangular that leads to $Z = L^{-T}$. Since the inverse of a unit lower triangular matrix is a unit lower triangular matrix, it follows that Z is unit upper triangular.

For more details about the algorithm, the reader is referred to [14]. For a dense matrix, this algorithm requires roughly twice as much work as Cholesky factorization [14]. Although for a sparse matrix the cost can be significantly reduced, the method is still expensive because the resulting matrix Z tends to be dense. However, the sparsity can be preserved by reducing the amount of fill-in occurring in the computation of z vectors. Reducing the amount of fill-in can be achieved by ignoring all fill-in outside selected positions in Z or by discarding fill-ins whose magnitude is less than a tolerance δ.

Zhang [16] proposes several strategies and special data structures to implement AINV algorithm efficiently. We have used many aspects of their implementation and we have made some modifications in their data structures to be able to access entries in a matrix that is in compressed column storage format. Since we do not know the sparsity pattern of the inverse upfront, we have used only the strategy in which we discarded fill-in occurring in the computation of z-vectors whose magnitude was less than $\delta = e^{-6}$.

Experimental results [17] show that AINV is more effective and faster than the other approximate inverse methods. Therefore, we have adopted it for our implementation.

2) Network Simplex Method: We show that the linear programs in our formulation can be efficiently solved with the help of a network simplex method. Using the notation given in Section II-A-3, we have the following linear program for each node:

$$\begin{aligned} &\text{maximize/minimize: } f(i) \\ &\text{s.t.: } 0 \le Ui \le i_G \\ &\qquad Mi = 0, \quad 0 \le i \le i_L \end{aligned} \tag{56}$$

where $f(i) : \mathbb{R}^n \to \mathbb{R}$ is the linear objective function of i. To simplify the notation, we augment the matrices U and M into the matrix T, and we augment the vectors i_G and the zero vector of size γ into the vector a

$$T = \begin{bmatrix} U \\ M \end{bmatrix}, \quad a = \begin{bmatrix} i_G \\ 0 \end{bmatrix}$$

where T is a matrix of size $(\mu + \gamma) \times n$ and a is a vector of size $(\mu + \gamma) \times 1$. With this notation, the linear program (56) can be rewritten as

$$\begin{aligned} &\text{maximize/minimize: } f(i) \\ &\text{s.t.: } 0 \le Ti \le a \\ &\qquad 0 \le i \le i_L. \end{aligned} \tag{57}$$

The constraint matrix T consists of entries that are 1, -1, or 0 s. It resembles the node-arc incidence matrix (NAIM) of a network, in the sense that NAIM also has entries that are 1, -1, or 0 s. This is the key observation that allows us to formulate the optimization problem (57) as a network flow problem. In our optimization problem, the equality constraints define flow conservation constraints of the network flow problem, whereas local constraints define capacity constraints on the flow along the edges in the network. Furthermore, global constraints define side constraints on the sum of the flows along the edges. For a more detailed discussion of network flow problems, the reader is referred to [18].

Network flow problems can be efficiently solved with the help of the network simplex method. Empirical results have shown that the method is significantly faster than the standard simplex method, when applied to the same network problem [19]. Furthermore, it is shown in [20] that significant computational speed up can be achieved if closely related instances of network flow problems are solved sequentially. This observation is quite beneficial for our problem in the sense that the feasibility space of currents remains the same at each instance of the optimization problem. Thus, we have

TABLE I
RUNTIME AND ACCURACY OF THE PROPOSED TECHNIQUE

grid	lower bound algorithm		upper bound algorithm		
nodes	runtime	time steps	runtime	time steps	maximum absolute error
65	2.25 sec.	7	4.63 sec.	3	0.87 mV
102	4.13 sec.	5	8.48 sec.	4	1.75 mV
437	39.33 sec.	4	57.49 sec.	3	2.52 mV
1,052	1.98 min.	4	2.43 min.	5	2.67 mV
4,285	14.66 min.	5	16.73 min.	4	2.59 mV
8,628	40.74 min.	4	46.35 min.	3	2.34 mV
18,472	1.57 h.	6	2.14 h.	3	2.65 mV
31,974	3.74 h.	7	4.73 h.	4	1.45 mV
76,753	9.45 h.	6	10.68 h.	4	2.72 mV
104,943	10.79 h.	4	12.45 h.	3	1.84 mV
192,974	18.55 h.	3	23.16 h.	3	1.39 mV

the same optimization problem for each node except different objective functions.

D. Experimental Results

To test our method, we have implemented Algorithms 1–3 in C++. We have set $\epsilon = e^{-5}$ to stop the main loop of Algorithm 1. To solve the required linear programs, Algorithms 1 and 2 use the network simplex method of the MOSEK optimization package [21]. We have used the hot-start option of the MOSEK network simplex solver for fast objective function switching. Several experiments were conducted on a set of test grids that were generated from user specifications, which include grid dimensions, metal layers (M1–M9), pitch and width per layer, and C4 and current source distribution. Minimum spacing, sheet resistance and via resistance were specified according to a 65-nm technology. A global constraint is specified for each macroblock, and additional global constraints were specified covering the entirety of the grid area. The computations were carried on a 64-bit Linux machine with 8-GB memory.

Table I shows the speed and the accuracy of our proposed solution technique for the computation of the vectors of lower and upper bounds on the worst case voltage fluctuations. The results are compared with each other and the maximum absolute difference between the upper and the lower bound vector is reported in column 6, where the absolute error is defined as $(v_{ub}(\infty) - v_{lb}(\infty))$. The results show that our solution technique resulted in a maximum absolute error of 2.72 mV across all nodes of all test grids. The number of time steps shown in columns 3 and 5 reports the number of time steps for which the lower and upper bound algorithms converge. The runtime for each one of two methods is also shown in Table I.

Fig. 9 shows a scatter plot with the lower bound on the worst case voltage fluctuation on one axis and the absolute error on the other axis, for a 4285-node grid. The figure shows that the absolute error between the upper and the lower bound is very small, meaning that the proposed method is very accurate. For the same grid, Fig. 10 shows a scatter plot with the lower bound on the worst case voltage fluctuation on one axis and the relative error on the other axis, where the relative error is defined as $(v_{ub}(\infty) - v_{lb}(\infty))/v_{lb}(\infty)$. The figure also

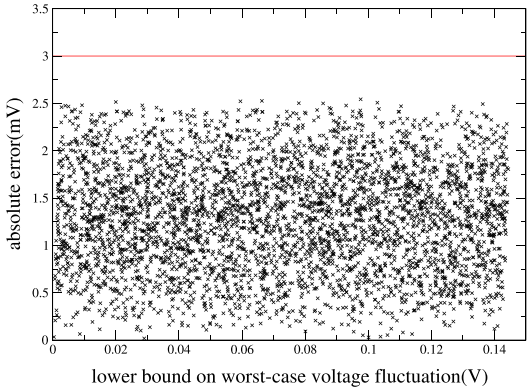

Fig. 9. Absolute error comparison for all nodes of a 4285-node grid.

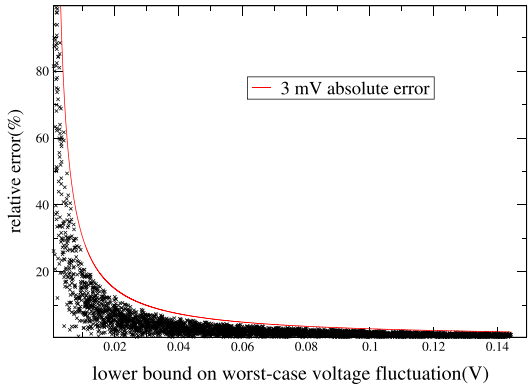

Fig. 10. Accuracy of the proposed technique.

shows the curve corresponding to 3 mV absolute error, where a point on the curve represents a node that has 3 mV difference between its upper and lower bound. Note that the relative error can be high, but only for small values of voltage fluctuations, and the absolute error does not exceed 3 mV.

Table II compares the worst case voltage fluctuations of (only) power grid verification and the (unsymmetrical) P/G grid verification, in which the P/G grid contains the same power grid as in the power grid verification and an unsymmetrical ground grid. The worst case voltage fluctuation for both scenarios is reported in columns 3 and 4, respectively, and the absolute difference between columns 3 and 4 is

TABLE II
COMPARISON BETWEEN (ONLY) POWER GRID VERIFICATION AND THE (UNSYMMETRICAL) P/G GRID VERIFICATION

number of nodes		worst-case voltage fluctuation		comparison	
power grid	p/g grid	power grid	p/g grid	absolute difference	relative difference
45	94	168.28 mV	175.49 mV	7.21 mV	4.28%
269	520	156.27 mV	148.93 mV	7.34 mV	-4.69%
1,043	2102	230.67 mV	256.95 mV	26.28 mV	11.39%
3,906	7932	190.84 mV	195.47 mV	4.63 mV	2.42%
8,297	16895	180.91 mV	214.72 mV	33.81 mV	18.68%
15,382	30675	170.68 mV	165.83 mV	4.85 mV	-2.84%
33,960	67488	286.39 mV	298.75 mV	12.36 mV	4.31%
50,231	102484	245.23 mV	210.47 mV	34.76 mV	-14.17%

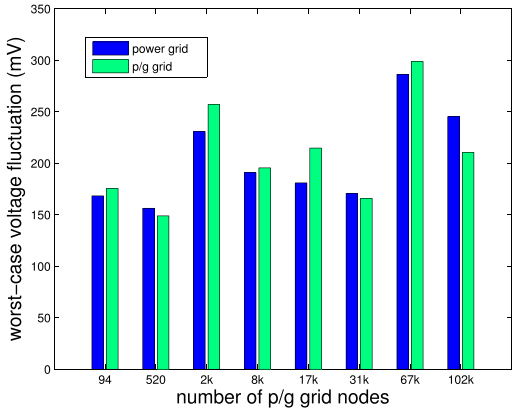

Fig. 11. Comparison between (only) power grid verification and the (unsymmetrical) P/G grid verification.

reported in column 5. The results show that the absolute difference can be as high as 35 mV. The results also show that in some cases (only) power grid verification results in larger worst case voltage fluctuation, whereas in some cases P/G grid verification results in larger worst case voltage fluctuation. This result is expected since the unsymmetry between the P/G grids can impact worst case voltage fluctuation. Fig. 11 shows a bar chart for the comparison between (only) power grid verification and the (unsymmetrical) P/G grid verification based on Table II.

Looking at the runtime results given in Table I, we notice that the computation of the lower and upper bound vectors of a 18 472-node grid takes ∼4 h (most of this runtime is dominated by approximate inverse computation). Such a grid is of course small compared with full-chip grids containing millions of nodes. However, the proposed method is applicable for early grid verification, where the size of the grids is normally not as large. The power of our approach is that it finds tight upper and lower bounds on the worst case voltage fluctuations under all feasible current combinations. It is a unique approach that offers this type of guarantee.

III. POWER GRID VERIFICATION UNDER HIERARCHICAL CURRENT CONSTRAINTS

In this section, we will first show that the linear programs in power grid verification can be efficiently solved, if we assume that the global constraint matrix U has no overlapping constraints, i.e., each column of U has only one nonzero element. We start with a base case, in which we assume that there is only one global constraint, and we present the solution to the problem in which there are more than one global constraint that are not overlapping. Then, we present the optimal solution to linear programs with tree-structured global constraint matrices (whereby global constraints are required to be wholly contained in other global constraints) based on [22]. Due to space considerations, we will not show the problem formulation of power grid verification problem in this paper. For background information on power grid verification with current constraint matrices, refer to [8]. The proposed methods in this section are only applicable to power grids, not to P/G grids. This is due to additional equality constraints in P/G grid problem formulation; these equality constraints are not part of the power grid problem formulation.

A. Verification Under Nonoverlapping Global Constraints

1) Problem Definition: Define x as a vector variable of size m representing the current sources, and $f(x) : R^m \rightarrow R$ as a vector function of x. Assuming that we have only one global constraint (and since we do not have any equality constraints in power grid model as apposed to P/G grid model), we have the following linear program:

$$\begin{aligned} &\text{maximize: } f(x) = a^T x \\ &\text{s.t.: } c^T x \leq b \\ &\qquad 0 \leq x \leq d \end{aligned} \tag{58}$$

where a is a vector of size m that defines the coefficients in the objective function and c is a vector of size m whose each entry is equal to 1. b is a positive number that is the upper bound defining the global constraint and d is the local constraint vector of size m. The linear program (58) resembles the fractional knapsack problem, except the fact that the coefficients in the constraint matrix of a fractional knapsack problem can be any nonnegative value [23]. Similar to the optimal solution of the fractional knapsack problem, we will show that the linear program (58) is solvable by a greedy strategy.

Assume without loss of generality that

$$a_1 \geq a_2 \geq \cdots \geq a_m \geq 0. \tag{59}$$

Thus a is sorted in a descending order and it consists of only nonnegative entries. Assume that $c^T d > b$, because otherwise

$x = d$ is optimal. Let k be the index satisfying

$$\sum_{i=1}^{k-1} d_i < b \text{ and } \sum_{i=1}^{k} d_i \geq b. \tag{60}$$

Define

$$\eta = b - \sum_{i=1}^{k-1} d_i. \tag{61}$$

We first state the following theorem [24], which will be useful for the proof of Claim 3.

Theorem 2 (Strong Duality): If x^* is a feasible solution to the primal problem $\max\{a^T x \mid Ax \leq b, x \geq 0\}$ and u^* is a feasible solution to the dual problem $\min\{b^T u \mid A^T u \geq a, u \geq 0\}$, then x^*, u^* are primal-dual optimal if and only if $a^T x^* = b^T u^*$.

Now define

$$J = \begin{bmatrix} c^T \\ I \end{bmatrix}, \quad \zeta = \begin{bmatrix} b \\ d \end{bmatrix} \tag{62}$$

where I is an $m \times m$ identity matrix. Notice that J is an $(m+1) \times m$ matrix, and ζ is a vector of size $m+1$. With the help of (62), we modify the linear program (58) to be in the format of the primal problem in Theorem 2 as

$$\begin{aligned} &\text{maximize: } f(x) = a^T x \\ &\text{s.t.: } Jx \leq \zeta \\ &\qquad x \geq 0. \end{aligned} \tag{63}$$

Claim 3: The optimal solution to the linear program (63) is given by

$$x_i^* = \begin{cases} d_i, & i = 1, \ldots, k-1 \\ \eta, & i = k \\ 0, & i = k+1, \ldots, m. \end{cases} \tag{64}$$

Proof: The dual of the linear program (63) is given by

$$\begin{aligned} &\text{minimize: } \zeta^T u \\ &\text{s.t.: } J^T u \geq a \\ &\qquad u \geq 0. \end{aligned} \tag{65}$$

Define $y = u_1$ and $z_i = u_{i+1}$ for $i = 1, \ldots, m$. Then, we can rewrite (65) as

$$\begin{aligned} &\text{minimize: } by + d^T z \\ &\text{s.t.: } y + z_i \geq a_i, \quad i = 1, \ldots, m \\ &\qquad y, z \geq 0. \end{aligned} \tag{66}$$

Now choose the variables y^* and z^* as

$$u_1^* = y^* = a_k \tag{67}$$

$$u_{i+1}^* = z_i^* = \begin{cases} a_i - a_k, & i = 1, \ldots, k-1 \\ 0, & i = k, \ldots, m. \end{cases} \tag{68}$$

Notice that the solution u^* given by (67) and (68) is a feasible solution to the dual problem (66) and that the solution x^* given by (64) is a feasible solution to the primal problem (63). If we can show that x^* and u^* satisfy the condition $a^T x^* = \zeta^T u^*$, then they are primal-dual optimal by Theorem 2. Using x^* given by (64), we write

$$\begin{aligned} a^T x^* &= \sum_{i=1}^{k-1} a_i d_i + a_k \eta \\ &= \sum_{i=1}^{k-1} a_i d_i + a_k \left(b - \sum_{i=1}^{k-1} d_i \right) \\ &= a_k b + \sum_{i=1}^{k-1} a_i d_i - a_k \sum_{i=1}^{k-1} d_i. \end{aligned} \tag{69}$$

Using u^* given by (67) and (68), we write

$$\begin{aligned} \zeta^T u^* = by^* + d^T z^* &= a_k b + \sum_{i=1}^{k-1} d_i (a_i - a_k) \\ &= a_k b + \sum_{i=1}^{k-1} a_i d_i - a_k \sum_{i=1}^{k-1} d_i. \end{aligned} \tag{70}$$

Thus, $a^T x^* = \zeta^T u^*$ for x^* given by (64) and for u^* given by (67) and (68). Therefore, they are primal-dual optimal by Theorem 2. This completes the proof. ■

In (63), we assumed that the coefficients of the objective function are nonnegative. However, the objective function can have negative coefficients under the RLC model of the power grid [11]. Now, we show that the optimization variables that correspond to the negative coefficients of the objective function in the linear program (63) will always be equal to 0 in the optimal solution. We first state the following theorem [24].

Theorem 3 (Complementary Slackness): If x^* is a feasible solution to the primal problem $\max\{a^T x \mid Ax \leq b, x \geq 0\}$, where A is an $l \times n$ matrix and u^* is a feasible solution to the dual problem $\min\{b^T u \mid A^T u \geq a, u \geq 0\}$, then x^*, u^* are primal-dual optimal if and only if the following hold.

1) For $i = 1, 2, \ldots, l$, if $u_i^* > 0$, then $A_{[i,:]} x^* = b_i$.
2) For $j = 1, 2, \ldots, m$, if $x_j^* > 0$, then $(u^*)^T A_{[:,j]} = a_j$.

Here, $A_{[i,:]}$ corresponds to ith row of A and $A_{[:,j]}$ corresponds to jth column of A.

Now assume without loss of generality that

$$a_1 \geq a_2 \geq \cdots \geq a_m. \tag{71}$$

Let h be the index satisfying

$$a_h \geq 0 \text{ and } a_{h+1} < 0. \tag{72}$$

Claim 4: The entries $x_{h+1}^*, \ldots, x_m^*$ of the optimal solution x^* to the linear program (63) are equal to 0.

Proof: Assume that the optimal solution to the dual problem (66) is given by u^*. Applying condition 2) of Theorem 3 to (63), we have

$$\text{For } j = 1, 2, \ldots, m, \text{ if } x_j^* > 0, \text{ then } u_1^* + u_j^* = a_j.$$

Since a_j is negative for $j > h$ and since $u \geq 0$, meaning $u_1^* + u_j^* \geq 0$, we conclude that $x_j^* \not> 0$ for $j > h$, leading to $x_j^* = 0$ for $j > h$. This completes the proof. ■

Thus, the optimization variables that have negative coefficients in the objective function can be simply ignored.

Let us have a look into the general problem in which we have more than one global constraint that is not overlapping. Suppose that there are p nonoverlapping global constraints. Denote the variables belonging to ith global constraint by $x^{(i)}$, their part of the objective function by $f_i(x^{(i)})$ and their feasible space due to ith global constraint and local constraints of $x^{(i)}$ by $\mathcal{F}_i$. It is shown in [25] that if the variables $x^{(i)}$ do not appear in common constraints, they are independent and the optimal value can be obtained by simply solving p-independent subproblems and summing the results of each p subproblem. Define

$$e_i = \max_{\forall x^{(i)} \in \mathcal{F}_i} f_i(x^{(i)}) \tag{73}$$

where e_i is the optimal solution to ith global constraint. The optimal solution e of the maximization problem due to p nonoverlapping global constraints can be written as the sum of the optimal solutions due to p independent maximization problems as follows:

$$e = \max_{\forall x \in \mathcal{F}} f(x) = e_1 + e_2 + \cdots + e_p. \tag{74}$$

Notice that the optimal solution to each subproblem in (74) is given by (64). Since every minimization problem can be converted into a maximization problem by simply negating the coefficients in the objective function, (74) is valid for minimization problems as well.

B. Verification Under Tree-Structured Global Constraints

Faaland [22] solved a simple class of linear programs with tree-structured constraint matrices. The special linear programming (LP) structure is in the following form:

$$\begin{aligned} &\text{maximize: } \sum_{j \in J} c_j x_j \\ &\text{s.t.: } \sum_{j \in J(i)} a_j x_j \le b_i, \quad i = 1, \ldots, m \\ &\qquad 0 \le x_j \le u_j, \quad j \in J \end{aligned} \tag{75}$$

where b_i, c_j, and a_j are positive scalars, $J = \cup_{i=1}^{m} J(i)$, and the sets $J(i)$ are nested meaning that if $i \neq k$, either $J(i)$ and $J(k)$ are disjoint, or one set is properly contained in the other. We may assume that $b_i < b_k$, whenever $J(i)$ is a proper subset of $J(k)$. When $m = 1$, (75) reduces to the fractional knapsack problem with upper bounds on variables.

In each column of the constraint matrix, all nonzero entries are equal and positive. This is also the case for the tree-structured linear programs in our formulation, because all nonzero entries in the global constraint matrix are 1 s. The only difference between (75) and the linear programs in our formulation is the lower bound for the global constraints. However, it is shown in [26] that the results obtained in [22] are valid for linear programs with nonnegative lower bounds for the constraint matrices.

The solution method shown in [22] separates the problem into a sequence of fractional knapsack problems. Each of these fractional knapsack problems requires linear time to solve, for total solution time no worse than proportional to the number of nonzero entries in the original constraint matrix.

TABLE III
EFFECTIVENESS OF THE SORTING METHOD COMPARED WITH LP SOLVER

power grid size	sorting method	LP solver	speed-up
1,836	6.46 sec.	1.19min	11.05x
7,361	30.24 sec.	8.37min	16.61x
30,393	2.35 min.	50.19min	21.35x
97,475	9.49 min.	3.9hr	24.65x

The optimal solution is obtained by giving each variable their minimum value from the list obtained by solving the sequence of fractional knapsack problems.

Tree-structured global constraints are more realistic than nonoverlapping global constraints in modern designs. We believe that modern designs are very hierarchical, and consist of functional blocks; IPs and subsystems, all of which can have their current constraints expressed as tree-structured global constraints. These constraints can come from high-level power constraints and functional block power/current constraints [4]. For more information on the application of this type of constraints, the reader is referred to [4].

C. Implementation and Experimental Results

We have implemented the solution method given in [22] in C++. To sort the coefficients of the objective function, we used quicksort algorithm.

To see the runtime behavior of our implementation, we have conducted several experiments to solve linear programs resulting from power grid verification with tree-structured hierarchical constraints. We compared the results of our implementation to the results obtained from solving the linear programs using MOSEK [21]. The computations were performed on the same machine as mentioned in Section II-D. The comparison of the results is presented in Table III. Column 1 shows the size of the power grid, columns 2 and 3 reports the runtime to solve the linear program for all the nodes using our implementation of the method given in [22] and using LP solver, respectively, and column 4 reports the achieved speed up. Notice that the runtime numbers do not include the amount of time needed to compute the matrix inversion. It can be seen from Table III that the proposed method is significantly fast, solving the large linear programs in significantly shorter time. In addition, note that both solutions are exact solutions to the LP problem, thus yielding the same results in both cases.

IV. CONCLUSION

Voltage fluctuations of the power distribution network are a key concern for modern chip design. In this paper, we presented an early grid verification technique that takes both P/G grids into account. We first formulated the grid verification problem as an optimization problem to compute the exact worst case voltage fluctuations at each node subject to current constraints, which is seen to be too expensive. As an alternative, we proposed a solution approach that formulates both upper and lower bounds on the worst case voltage fluctuations. Experimental results show that the proposed bounds have errors in the range of a few millivolts, and that the

verification of both P/G grids is required to capture realistic voltage fluctuations on the power distribution network. We also showed that, under hierarchical current constraints, power grid problem can be efficiently solved without any loss of accuracy, resulting in significant run-time savings.

The P/G grid model presented in this paper is an *RC* model of the grid. The grid inductance can contribute significantly to the voltage fluctuation. Our approach can be easily extended to take inductive effects into account for P/G grid verification. In addition, designers may be interested in the voltage difference between P/G grid nodes, rather than individual node voltages, and therefore, the future work should extend the mathematical formulation to calculate the worst case voltage difference. Besides, the current constraints used in this paper are dc constraints. The P/G grid verification problem should also be extended and solved under transient current constraints. The future work should also include the application of the improvements in power grid verification under hierarchical constraints to P/G grid verification.

APPENDIX

We will prove additional claims that are useful in the context of Section II-B1. We start with the following claim.

Claim 5: $I - D^p$ is invertible for any integer p.

Proof: Denoting the set of all eigenvalues of D by $\sigma(D)$ and using Theorem 1, we can write

$$\max_{\forall \lambda \in \sigma(D)} |\lambda| < 1. \tag{76}$$

From the spectral mapping theorem [27], we know that if k is an integer, we have the following relationship:

$$\sigma(D^k) = \{\lambda^k : \lambda \in \sigma(D)\}. \tag{77}$$

Thus, $\rho(D^p) < 1$, for an integer p. We also know that the series $\sum_{q=0}^{\infty} X^q$ for a square matrix X is known to converge [10] if and only if $\rho(X) < 1$, under which condition the series limit is $(I-X)^{-1}$, meaning that $I-D^p$ is invertible for an integer p. This completes the proof. ■

With the help of Claim 5, we will prove the following claim.

Claim 6: Suppose $W(p)$ is a function of p for $p \geq 1$ defined as

$$W(p) = (I - D^p)^{-1} \sum_{k=0}^{p-1} D^k H. \tag{78}$$

Then $W(p) = W(1), \forall p \geq 1$.

Proof: The case for $p = 1$ is satisfied trivially. The claim is true by induction if we prove the following, $\forall p \geq 2$:

$$W(p-1) = W(1) \Rightarrow W(p) = W(1). \tag{79}$$

Left multiplying both sides of (78) with $(I-D^p)$, and breaking the sum $\sum_{k=0}^{p-1} D^k H$ into $\sum_{k=0}^{p-2} D^k H$ and $D^{p-1}H$, we obtain

$$(I - D^p)W(p) = \sum_{k=0}^{p-2} D^k H + D^{p-1} H. \tag{80}$$

From Claim 5, we know that $I - D^p$ is invertible for an integer p. Left multiplying both sides of (80) with $(I - D^{p-1})^{-1}$, we obtain

$$(I - D^{p-1})^{-1}(I - D^p)W(p) = (I - D^{p-1})^{-1} \sum_{k=0}^{p-2} D^k H + (I - D^{p-1})^{-1} D^{p-1} H. \tag{81}$$

Since $(I - D^{p-1})^{-1} \sum_{k=0}^{p-2} D^k H = W(p-1)$, we can replace it with $W(1)$. Rearranging the terms of (81), we have

$$(I - D^{p-1})^{-1}((I - D^p)W(p) - D^{p-1}H) = W(1). \tag{82}$$

Left multiplying both sides of (82) with $(I - D^{p-1})$ leads to

$$W(p) - D^p W(p) - D^{p-1} = W(1) - D^{p-1} W(1). \tag{83}$$

Adding $D^{p-1}W(1)$ to both sides of (83) and rearranging the terms, we obtain

$$W(p) - W(1) = (I - D^p)^{-1} D^{p-1}(H - (I - D)W(1)). \tag{84}$$

Replacing $W(1)$ on the right-hand side of (84) with $(I - D)^{-1}H$ leads to

$$W(p) - W(1) = (I - D^p)^{-1} D^{p-1}(H - H) = 0. \tag{85}$$

Meaning that $W(p) = W(1)$. This completes the proof. ■

Finally, our main result is captured in the following claim.

Claim 7: For any integer $p \geq 1$

$$D^p K = K - \sum_{k=0}^{p-1} D^k H. \tag{86}$$

Proof: Since $W(p) = W(1)$ by Claim 6, meaning that

$$(I - D^p)^{-1} \sum_{k=0}^{p-1} D^k H = (I - D)^{-1} H. \tag{87}$$

Left multiplying both sides of (87) with $(I - D^p)$, we obtain

$$\sum_{k=0}^{p-1} D^k H = (I - D^p)(I - D)^{-1} H. \tag{88}$$

Since $D = A^{-1}B$ and $B = A - G$, we have $D = A^{-1}(A - G) = I - A^{-1}G$, leading to $I - D = A^{-1}G$. Since K and H are matrices obtained from the first m columns of G^{-1} and A^{-1}, respectively, we can write $K = (I-D)^{-1}H$, leading to

$$\sum_{k=0}^{p-1} D^k H = (I - D^p)K. \tag{89}$$

Rearranging the terms of (89), we get

$$D^p K = K - \sum_{k=0}^{p-1} D^k H \tag{90}$$

which completes the proof. ■

REFERENCES

[1] *International Technology Roadmap for Semiconductors*, Semiconductor Industry Association, Washington, DC, USA, 2012.

[2] D. Kouroussis and F. N. Najm, "A static pattern-independent technique for power grid voltage integrity verification," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, Anaheim, CA, USA, Jun. 2003, pp. 99–104.

[3] M. Popovich, A. V. Mezhiba, and E. G. Friedman, *Power Distribution Networks With On-Chip Decoupling Capacitors*. New York, NY, USA: Springer-Verlag, 2008.

[4] C.-K. Cheng, P. Du, A. B. Kahng, G. K. H. Pang, Y. Wang, and N. Wong, "More realistic power grid verification based on hierarchical current and power constraints," in *Proc. ACM/IEEE Int. Symp. Phys. Design (ISPD)*, Santa Barbara, CA, USA, Mar. 2011, pp. 159–166.

[5] Y. Wang, X. Hu, C.-K. Cheng, G. K. H. Pang, and N. Wong, "A realistic early-stage power grid verification algorithm based on hierarchical constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 1, pp. 109–120, Jan. 2012.

[6] X. Xiong and J. Wang, "Dual algorithms for vectorless power grid verification under linear current constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 10, pp. 1469–1482, Oct. 2011.

[7] J. N. Kozhaya, S. R. Nassif, and F. N. Najm, "A multigrid-like technique for power grid analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 10, pp. 1148–1160, Oct. 2002.

[8] N. H. Abdul Ghani and F. N. Najm, "Fast vectorless power grid verification using an approximate inverse technique," in *Proc. 46th ACM/IEEE Design Autom. Conf. (DAC)*, San Francisco, CA, USA, Jul. 2009, pp. 184–189.

[9] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*. Chichester, U.K.: Wiley, 1991.

[10] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, 2003.

[11] N. H. Abdul Ghani and F. N. Najm, "Fast vectorless power grid verification under an RLC model," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 5, pp. 691–703, May 2011.

[12] S. Demko, W. F. Moss, and P. W. Smith, "Decay rates for inverses of band matrices," *Math. Comput.*, vol. 43, no. 168, pp. 491–499, Oct. 1984.

[13] M. J. Grote and T. Huckle, "Parallel preconditioning with sparse approximate inverses," *SIAM J. Sci. Comput.*, vol. 18, no. 3, pp. 838–853, May 1997.

[14] M. Benzi, C. D. Meyer, and M. Tůma, "A sparse approximate inverse preconditioner for the conjugate gradient method," *SIAM J. Sci. Comput.*, vol. 17, no. 5, pp. 1135–1149, Sep. 1996.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1996.

[16] J. Zhang, "A sparse approximate inverse preconditioner for parallel preconditioning of general sparse matrices," *Appl. Math. Comput.*, vol. 130, no. 1, pp. 63–85, Jul. 2002.

[17] M. Benzi and M. Tůma, "A comparative study of sparse approximate inverse preconditioners," *Appl. Numer. Math.*, vol. 30, nos. 2–3, pp. 305–340, Jun. 1999.

[18] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[19] G. Sierksma, *Linear and Integer Programming: Theory and Practice*. New York, NY, USA: Marcel Dekker, 2001.

[20] A. Frangioni and A. Manca, "A computational study of cost reoptimization for min-cost flow problems," *INFORMS J. Comput.*, vol. 18, no. 1, pp. 61–70, 2003.

[21] *MOSEK Optimization Software*. [Online]. Available: http://www.mosek.com, accessed Jan. 19, 2014.

[22] B. Faaland, "A weighted selection algorithm for certain tree-structured linear programs," *Oper. Res. J.*, vol. 32, no. 2, pp. 405–422, Mar./Apr. 1984.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA, USA: MIT Press, 2001.

[24] R. K. Martin, *Large Scale Linear and Integer Optimization: A Unified Approach*. Boston, MA, USA: Kluwer, 1998.

[25] S. P. Bradley, A. C. Hax, and T. L. Magnanti, *Applied Mathematical Programming*. Menlo Park, CA, USA: Addison-Wesley, 1977.

[26] S. S. Erenguc, "An algorithm for solving a structured class of linear programming problems," *Oper. Res. Lett.*, vol. 4, no. 6, pp. 293–299, Apr. 1986.

[27] S. Lang, *Algebra*. Menlo Park, CA, USA: Addison-Wesley, 1993.

Mehmet Avci received the B.S. (Hons.) degree in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2008, and the M.A.Sc. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2010.

He is currently with the Toronto Technology Centre, Altera Corporation, Toronto, as a Senior Design Engineer, where he is involved in timing modeling and analysis. His current research interests include computer-aided design for integrated circuits, with a focus on timing modeling and analysis, power modeling, and grid integrity verification.

Farid N. Najm (S'85–M'89–SM'96–F'03) received the B.E. degree in electrical engineering from the American University of Beirut, Beirut, Lebanon, in 1983, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana—Champaign (UIUC), Champaign, IL, USA, in 1989.

He was with Texas Instruments, Dallas, TX, USA, from 1989 to 1992. He joined the Department of Electrical and Computer Engineering, UIUC, as an Assistant Professor, where he became an Associate Professor in 1997. He joined the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada, in 1999, where he is currently a Professor and the Chair. He is a Professor with the Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto. He has authored a book entitled *Circuit Simulation* (New York: John Wiley and Sons, 2010). His current research interests include computer-aided design for VLSI, with an emphasis on circuit level issues related to power, timing, variability, and reliability.

Dr. Najm is a fellow of the Canadian Academy of Engineering. He has received the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN Best Paper Award, the National Science Foundation (NSF) Research Initiation Award, the NSF CAREER Award, and the Design Automation Conference (DAC) Prolific Author Award. He was an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS from 2001 to 2009, and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS from 1997 to 2002. He served on the Executive Committee of the International Symposium on Low-Power Electronics and Design (ISLPED) from 1999 to 2013, and served as the Technical Program Committee Chair and General Chair of ISLPED. He has also served on the Technical Committees of various conferences, including the International Conference on Computer-Aided Design, DAC, the Custom Integrated Circuits Conference, the International Symposium on Quality Electronic Design, and ISLPED.