A Multigrid-like Technique for Power Grid Analysis

Joseph N. Kozhaya, Sani R. Nassif, and Farid N. Najm

Abstract— **Modern sub-micron VLSI designs include huge power grids that are required to distribute large amounts of current, at increasingly lower voltages. The resulting voltage drop on the grid reduces noise margin and increases gate delay, resulting in a serious performance impact. Checking the integrity of the supply voltage using traditional circuit simulation is not practical, for reasons of time and memory complexity. We propose a novel multigrid-like technique for the analysis of power grids. The grid is reduced to a coarser structure, and the solution is mapped back to the original grid. Experimental results show that the proposed method is very *efficient* as well as suitable for both DC and transient analysis of power grids.**

I. Introduction

In recent years, there has been an increased demand for *high performance* and *low power* VLSI designs. High performance is achieved by technology scaling, increased functionality and competitive designs. On the other hand, a common technique used to obtain low power designs is to scale down the supply voltage. This stands to reason since the chip power P is proportional to the square of the supply voltage V_{DD}. Thus, the demand for *high performance* and *low power* has led to modern VLSI designs being characterized by reduced feature size, increased functionality and lower supply voltage.

Increased chip functionality results in the need for *huge* power distribution networks, also referred to as power grids since they typically have a grid structure. Lower supply voltage, on the other hand, makes the voltage variation across the power grids very critical since it may lead to chip failures. Voltage drops on the power grid reduce the supply voltage at logic gates and transistor cells to less than the ideal reference. This leads to reduced noise margins, higher logic gate delays, and overall slower circuits. Reduced noise margins may lead to false switching at certain logic gates and latches. Higher logic gate delays, on the other hand, may slow down the circuit enough so that timing requirements can not be met. Consequently, once voltage drops exceed certain designer-specified thresholds, there is no guarantee that the circuit will operate properly [1], [2], [3].

Thus, it is clear that in modern VLSI circuits, power grids are becoming performance limiting factors. Consequently, *efficient* analysis of power grids [4], [5] is necessary for both (1) predicting the performance and (2) improving the performance if necessary. Because of the large dimensions of power grids in modern VLSI circuits, existing analysis methods are falling behind. Thus, there is a need for new *efficient*, in terms of both execution time and memory, techniques for the analysis of power grids.

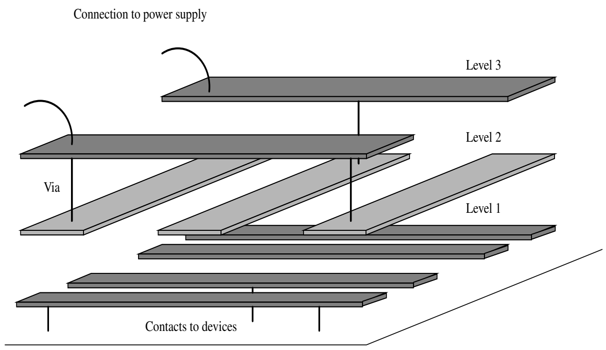

Fig. 1. Power grid components.

In this paper, we propose an *efficient* analysis technique that follows the lines of thought of multigrid methods which are commonly used for the solution of smooth partial differential equations (PDEs). Specifically, our method is inspired by the algebraic multigrid method, AMG, which is one variation of the multigrid approach. Thus, section III describes the multigrid approach with a specific emphasis on the algebraic multigrid variation. After discussing the multigrid technique, we present our proposed multigrid-like approach for the *efficient* analysis of power grids in section IV. The efficiency of our proposed technique is verified by the experimental results given in section IV-D. Finally, conclusions are provided in section V.

II. Modeling and Analysis of Power Grids

In this section, we discuss the basic modeling and analysis techniques of power grids. Specifically, section II-A discusses how to model typical power grids, sources, and drains, for *efficient* and *accurate* analysis. Section II-B, on the other hand, presents the basic analysis techniques and discusses some tricks to speedup the analysis.

A. Modeling of Power Grids

The connections from the power grid to the external supply voltage, V_{DD}, are called the power sources since the current is supplied from the supply to the grid through these connections. The power drains, on the other hand, are those modules which draw current from the power grid. For instance, transistors, logic gates, latches, clock buffers, memory units, register arrays, and I/O buffers are all considered power drains.

The first step in power grid analysis involves modeling the grids as well as the power *sources* and *drains* [6]. Modeling of the grids, sources, and drains involves a tradeoff between accuracy and speed. The more complex the model

J. N. Kozhaya is with IBM Corp., 1000 River St., Essex Junction, VT 05452. He was with the ECE Department, University of Illinois, Urbana, IL, kozhaya@us.ibm.com

S. R. Nassif is with IBM Austin Research Laboratory, Austin, TX, nassif@austin.ibm.com

F. N. Najm is with the ECE Department, University of Toronto, Ontario, Canada, f.najm@utoronto.ca

is, the more accurate the results of the analysis are but the more expensive the solution is (in terms of memory and CPU time).

Typically, power distribution within an integrated circuit is done from the top-level metal layer, which is connected to the package, down through inter-layer *vias* and finally to the active devices, as illustrated in Figure 1. We follow the same modeling approach as in [6], where the metal wires and vias are modeled as a linear, time-invariant and passive network consisting of resistive, capacitive and -rarely-inductive elements. For modern integrated circuits such as microprocessors, such a network can easily include millions of nodes and tens of millions of elements.

As for the power *sources* and *drains*, their models can be quite complex. The models for the power sources can be involved enough to include sophisticated package and board models. On the other hand, the models for the power drains can account for the complex interaction between the power grid, the underlying non-linear circuit, and the time-varying signals propagating across the chip. However, the huge size of the power grid makes it infeasible to include any but the simplest models for the power sources and drains. Hence, power sources are modeled as simple constant voltage sources and power drains are modeled as independent time-varying current sources.

Given the above, the complete power grid model is composed of a linear network of RLC elements excited by constant voltage sources and time varying current sources. Note that none of the network RLC elements are connected to ground, all the voltage sources (sources) are between certain grid nodes and ground, and all the current sources (drains) are between grid nodes and ground. The behavior of such a system can be expressed following the modified nodal analysis (MNA) [7] formulation as the following ordinary differential equation:

$$Gx + C\dot{x} = u(t) \tag{1}$$

where x is a vector of node voltages, and source and inductor currents; G is the conductance matrix; C includes the capacitance and inductance terms, and $u(t)$ includes the contributions from the sources and the drains. In fact, $u(t)$ has three kinds of rows: *i)* rows with a positive V_{DD} value, which correspond to nodes that are connected to power sources, *ii)* rows with a negative value of current (or sum of currents), which correspond to nodes that are connected to a power drain (or more than one), and *iii)* rows with 0, which correspond to all other nodes.

In all what follows, we ignore the effects of on-chip inductance and assume that the power grid is modeled as an RC network only. This is motivated by the fact that in today's technology, on-chip inductance in the power grid is too small to significantly affect the analysis results. Thus, in summary, the power grid is modeled as an RC network, the power sources are modeled as constant voltage sources, and the power drains are modeled as time-varying current sources.

Figure 2 shows a 3×3 grid with one voltage source at node 1 (indicated by an X) and two current sources at nodes 5 and 7 (indicated by a dot). This example shows the model for a two layer power grid with three wires in each layer, one voltage source at node 1, and two power drains, one drawing current from node 5 and the other from node 7. An example of a current source associated with an inverter power drain is shown in Fig. 3 and expressed mathematically as follows where t is the time in nanoseconds and I is the current in milliamps:

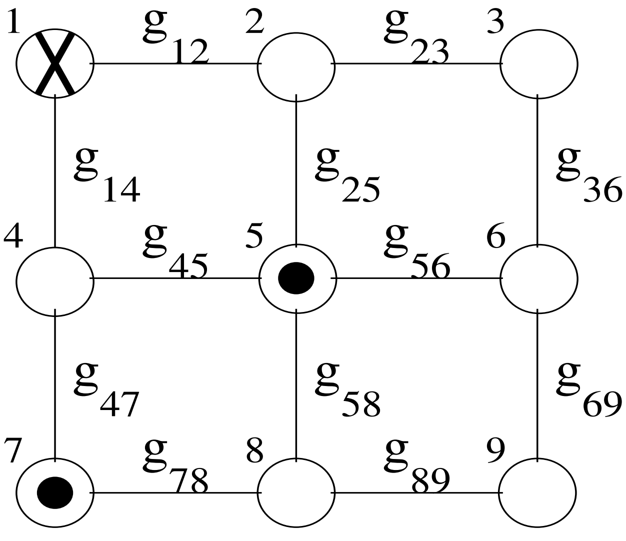

Fig. 2. Example of 3x3 grid.

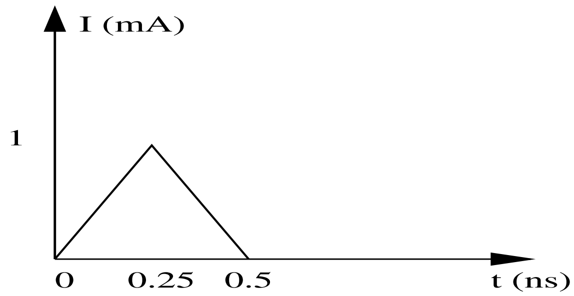

Fig. 3. Current drawn by an inverter.

$$I = \begin{cases} 410^6 t & 0 \le t \le 0.2510^{-9} \\ -410^6 t + 210^{-3} & 0.2510^{-9} \le t \le 0.510^{-9} \end{cases}$$

B. Analysis of Power Grids

Due to the large size of typical power grids, general circuit simulators such as Spice [8] are not adequate for power grid analysis because of CPU time and memory limitation. The inefficiency of standard simulators comes about because (a) they require a lumped element approximation of the circuit which requires the translation of a regular geometrical structure to an expansive set of equivalent circuit elements, and (b) they use general purpose solution methods meant to be robust in the face of stiff systems of equations. By contrast, power grids are well behaved spatially (nearly regular) and temporally (damped). This motivates a special-purpose simulator for power grids which can make use of these properties.

Solving (1) requires the use of some numerical integration formula. Typically, Backward Euler (BE) integration formula is the formula of choice mostly due to its stability

$$
\underbrace{\left[\begin{array}{ccccccccc|c}
G_1 & -g_{12} & & -g_{14} & & & & & & 1 \\
-g_{21} & G_2 & -g_{23} & & -g_{25} & & & & & \\
 & -g_{32} & G_3 & & & -g_{36} & & & & \\
-g_{41} & & & G_4 & -g_{45} & & -g_{47} & & & \\
 & -g_{52} & & -g_{54} & G_5 & -g_{56} & & -g_{58} & & \\
 & & -g_{63} & & -g_{65} & G_6 & & & -g_{69} & \\
 & & & -g_{74} & & & G_7 & -g_{78} & & \\
 & & & & -g_{85} & & -g_{87} & G_8 & -g_{89} & \\
 & & & & & -g_{96} & & -g_{98} & G_9 & \\
\hline
1 & & & & & & & & & 0
\end{array}\right]}_{A'}
\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix}}_{x}
=
\underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -I_5 \\ 0 \\ -I_7 \\ 0 \\ 0 \\ V_{DD} \end{bmatrix}}_{b'}
$$

Fig. 4. Linear system resulting from MNA.

$$
\underbrace{\begin{bmatrix}
1 & & & & & & & & \\
 & G_2 & -g_{23} & & -g_{25} & & & & \\
 & -g_{32} & G_3 & & & -g_{36} & & & \\
 & & & G_4 & -g_{45} & & -g_{47} & & \\
 & -g_{52} & & -g_{54} & G_5 & -g_{56} & & -g_{58} & \\
 & & -g_{63} & & -g_{65} & G_6 & & & -g_{69} \\
 & & & -g_{74} & & & G_7 & -g_{78} & \\
 & & & & -g_{85} & & -g_{87} & G_8 & -g_{89} \\
 & & & & & -g_{96} & & -g_{98} & G_9
\end{bmatrix}}_{A}
\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix}}_{x}
=
\underbrace{\begin{bmatrix} V_{DD} \\ g_{21}V_{DD} \\ 0 \\ g_{41}V_{DD} \\ -I_5 \\ 0 \\ -I_7 \\ 0 \\ 0 \end{bmatrix}}_{b}
$$

Fig. 5. Linear system after reformulation.

properties [7]. Applying Backward Euler to (1) results in a set of linear equations:

$$(G + C/h)x(t + h) = u(t + h) + x(t)C/h \qquad (2)$$

which can be readily simplified to $A'x(t + h) = b'$ with $A' = G + C/h$ and $b' = u(t + h) + x(t)C/h$.

The solution of (2) requires the inversion (factorization) of the matrix $A' = G + C/h$ which is independent of x, time-invariant, large and sparse. We note, however, that if we hold the time step h constant, then only one initial factorization is required, with a forward/backward solve at each time step. Since, for large matrices, a factorization is significantly more expensive than a forward/backward solve [9], the use of a constant time step results in large savings. The time step needs to be kept small enough to insure the accuracy of the solution. For application in the analysis of power grids of digital circuits, we find that using 100 steps per clock cycle (i.e. $h = 0.01{*}T_{period}$) is sufficient.

To illustrate, we simulate a simple grid of 33 wires in each of the horizontal and vertical directions, connected to a single voltage source at one of the corners, and loaded with 100 time-varying current sources at random locations. The resulting electrical model has 1089 nodes and a total of 1090 equations. We perform the simulation for 100 time steps. Both, Spice and our simulator, produce the same results. However, Spice [8] takes 13.3 sec. of CPU time, whereas our simulator implementing the method above takes 0.73 sec. for a net speedup of about 18x. Due to the superlinear dependence of solve time on matrix size, the speedup will be even more dramatic for the much larger systems normally encountered when simulating realistic power grids.

To better explain the process of power grid analysis, modified nodal analysis is applied to the example given in Figure 2. This results in the linear system shown in Figure 4.

Let N be the number of nodes of the power grid. Then, $N = 9$ in the example grid shown in Figure 2 since the grid is 3×3. Furthermore, there is one voltage source which means that the A' matrix is a 10×10 matrix where the first 9 equations are the KCL equations at all 9 nodes and the last equation is the KVL equation at node 1 where the voltage source is located. As for the current sources at nodes 5 and 7, they appear in the right hand side vector b'. On the other hand, g_{ij} defines the conductance between the two neighboring nodes i and j. Thus, $g_{ij} = g_{ji}$ which results in the A' matrix being symmetric. Furthermore, the diagonal entries of the A matrix are defined as follows:

$$G_i = \sum_{j \in N_i} |g_{ij}| + C_i/h \qquad (3)$$

where C_i is the capacitance at node i, h is the time step, and $N_i = \{j \mid g_{ij} \neq 0\}$ is the set of neighbors of node i.

In general, the system matrix, A', of the linear system $A'x = b'$ is *symmetric* but not necessarily positive definite [9]. However, the problem can be reformulated to result in a *symmetric, positive definite* matrix, A. Basically, the MNA formulation builds the linear system $A'x = b'$ by asserting both the KCL and KVL equations at the power source nodes as well as the KCL equations at all the remaining grid nodes [7]. Then, the solution, x, of the resulting linear system gives the voltages at all the grid nodes as well as the currents supplied by the different voltage sources. However, if we are only interested in the voltages at all the grid nodes, then we can ignore the KCL equations corresponding to the power source nodes. Furthermore, the voltage at a power source node is known to be exactly the supply voltage, V_{DD}. Thus, we can reformulate the problem by substituting the value of the supply, V_{DD}, for the voltage of all the power source nodes and ignoring the KCL equations for the power source nodes.

We illustrate by applying this reformulation to the 3×3 grid example given above. Node 1 has a voltage source, so we replace the KCL equation at node 1 (equation 1) with the KVL equation at node 1 (equation 10). Furthermore, we substitute the value $x_1 = V_{DD}$ in the equations which depend on x_1. The resulting system is shown in Figure 5.

Observe that the right hand side is also changed to result in a new vector, b, where two extra terms $g_{21}V_{DD}$ and $g_{41}V_{DD}$ are added at indices 2 and 4 respectively. It is clear that the resulting matrix A is still symmetric. It can also be shown to be positive definite. For that, note first that $\forall\ i,\ \sum_{j \neq i} |g_{ij}| = \sum_{j \in N_i} |g_{ij}|$ since $g_{ij} = 0$ for $j \notin N_i$. This, together with (3) results in the following:

$$|G_i| = \sum_{j \in N_i} |g_{ij}| + C_i/h = \sum_{j \neq i} |g_{ij}| + C_i/h \geq \sum_{j \neq i} |g_{ij}| \quad \forall\ i \qquad (4)$$

Equation (4) holds for every node i or equivalently for every row of the system matrix A. This shows that the modified matrix A is diagonally dominant and it was already pointed out that A is also symmetric. Consequently, A is a symmetric and positive definite matrix [9]. As a matter of fact, A can be shown to be an $\mathcal{M}$-matrix [10] since it satisfies the following:

1. $a_{ii} > 0 \ \ \forall \ i$
2. $a_{ij} \leq 0 \ \ \forall \ i \neq j$
3. $a_{ii} \geq \sum_{j \neq i} |a_{ij}| \ \ \forall \ i$
4. $a_{ii} > \sum_{j \neq i} |a_{ij}| \ \ \textit{for at least one } i$

where a_{ij} is the entry of the A matrix at row i and column j. Thus, in the rest of the paper, we will use the fact that the A matrix is a *non-singular* $\mathcal{M}$-matrix.

III. Multigrid Method

In a well designed power grid, the grid resistance is much smaller than the equivalent sink resistance since the power grid is required to deliver as constant a voltage as possible to all sinks. This causes local power disturbances, as would be caused by a large localized sink, to be *spread* across an area much larger than that of the sink causing the disturbance. This spreading leads to voltage distributions which are spatially smooth, and motivates solution methods which can make use of this smoothness to speed up the solution process.

Furthermore, we note that the analysis of power grids results in a system of linear equations structurally identical to that of a finite element discretization of a two-dimensional parabolic partial differential equation (PDE). This motivates us to consider the power grid problem as a discretization of a continuous PDE where the solution is needed at a spatially fixed set of points. Consequently, *efficient* methods for solving PDEs are worth considering as potential competitive solvers for the power grid problem.

Recently, the multigrid method (MG) has become the standard for solving smooth PDEs [11], [12], [13]. Multigrid involves two complementary steps: i) relaxation and ii) coarse grid correction. Relaxation involves running a few iterations of an iterative solver in order to smooth the error components; that is, reduce the high frequency error components. Coarse grid correction, on the other hand, involves mapping the problem to a coarser grid, solving the problem at the coarser grid, and then mapping the solution back to the original grid. These two complementary steps work together to provide an efficient technique for solving PDEs. Thus, in this paper, we argue the suitability/efficiency of the *multigrid* technique for power grid analysis.

Initial interest in multigrid resulted from a detailed analysis of *iterative* methods and the reasons for their slow convergence. Historically, multigrid methods faced slow acceptance during their early stages until their *practical efficiency* was demonstrated by Brandt in 1973 [12], [14]. Then in 1975/1976 Hackbusch developed the fundamental principles of multigrid without the knowledge of the existing literature. In his work, Hackbusch discussed a lot of theoretical and practical issues. He also presented a general convergence theory of multigrid methods [13].

While classical iterative methods suffer from slow convergence as the grid dimension increases (equivalently grid spacing decreases), multigrid performance doesn't deteriorate. As a matter of fact, multigrid has been shown to have optimal performance in that a system of N equations can be solved with O(N) complexity. Not only is the multigrid technique of optimal complexity but also the constant involved is small enough to provide an advantage of multigrid over other methods [13]. We should point out here that the multigrid method falls under the category of *iterative* solvers like Jacobi and Gauss-Seidel as opposed to *direct* solvers like Gaussian elimination.

We have already mentioned that the analysis of classical iterative methods has led to interest in multigrid. So we will start with a brief analysis of classical iterative methods. Given a linear system $Ax = b$ and an initial guess $\hat{x}^0$, an iterative scheme involves the following:

$$\hat{x}^{k+1} = \hat{x}^k + M^{-1}(b - A\hat{x}^k) \tag{5}$$

where k is the iteration index and $\hat{x}^k$ is an approximation of the exact solution x obtained at iteration k. As for M, it should be an *easy-to-invert* matrix defined such that $M^{-1} \approx A^{-1}$.

Let $e = x - \hat{x}$ be the error defined as the difference between the exact solution x and the approximate solution, $\hat{x}$. It can be shown that the error can be expressed as a linear combination of *low frequency* and *high frequency* Fourier modes [11]. Furthermore, the analysis of classical iterative methods leads to the following observation [11], [15]:

Classical iterative methods efficiently reduce the high frequency error components but are inefficient in reducing the low frequency error components.

In order to avoid the limitations of classical methods, multigrid methods consist of two complementary components [11], [13]:

1. *Relaxation* (smoothing) which reduces the high frequency error components.
2. *Coarse grid correction* which reduces the low frequency error components.

Relaxation involves running a few iterations of a classical iterative solver. This follows from the observation that classical iterative methods act as good *smoothers* as illustrated in Figure 6 [15]. Coarse grid correction, on the other hand, involves mapping the problem to the coarser grid, solving the mapped problem, and mapping the solution back to the fine grid. The key tools needed for communication between the two grids (fine and coarse) are the *intergrid transfer operators* which are referred to as the restriction and prolongation operators. One intuitive motivation for coarse grid correction is that the solution at a coarse grid typically provides a *good* initial guess for the iterative solver at the fine grid and thus results in rapid convergence. Another motivation for this approach is that

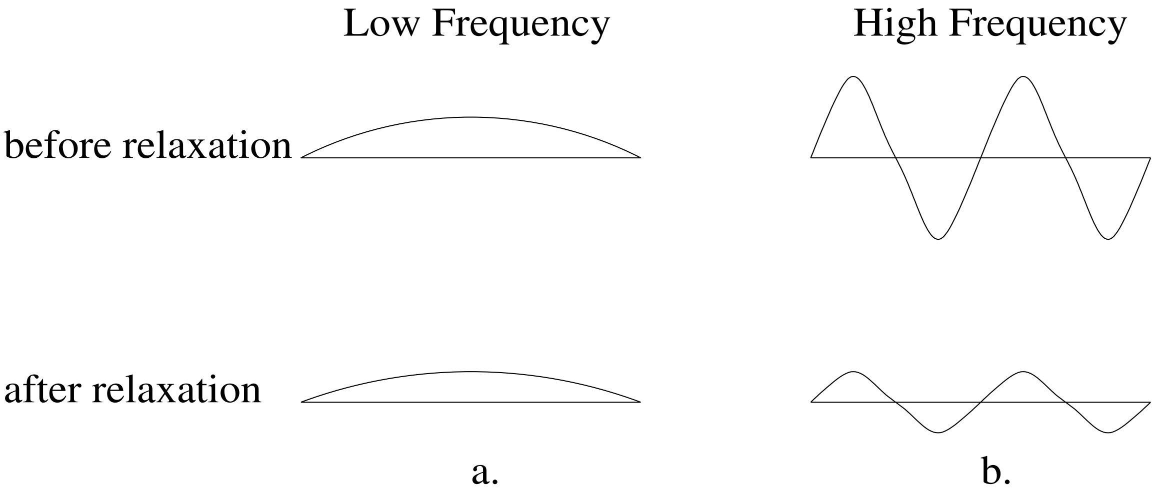

Fig. 6. Smoothing effect of iterative methods: a) Slight reduction of the amplitude of low frequency components. b) Significant reduction of the amplitude of high frequency components.

the low frequency error components at the fine grid Ω^h appear more oscillatory at the coarse grid Ω^{2h} as shown in Figure 7 [11]. Then, relaxation at the coarser grid reduces those components.

Multigrid techniques work as follows [11]. Starting at the fine grid, a few relaxation steps (iterations) are applied to reduce the high frequency modes of the error. Then the low frequency (smooth) modes of the error are well approximated by coarse grid correction. This leads to the *efficient* V-cycle multigrid method sketched by the following algorithm [11]. In the following, $\mathcal{R}_h^{2h}$ and $\mathcal{P}_{2h}^{h}$ correspond to the restriction and prolongation operators respectively. V^h, on the other hand, identifies the call to the $V-cycle$ and h defines the level at which the $V-cycle$ is called. As for ν_1 and ν_2, these are constants that define the number of iterations to be performed. These constants are chosen empirically and typically have values of 2 or 3.

$$\hat{x}^h \leftarrow V^h(\hat{x}^h, b^h)$$

1. Relax ν_1 times on $A^h x^h = b^h$ with a given initial guess $\hat{x}^h$.
2. If Ω^h = coarsest grid, then go to 4.
Else $b^{2h} \leftarrow \mathcal{R}_h^{2h}(b^h - A^h \hat{x}^h)$
$\quad \hat{x}^{2h} \leftarrow 0$
$\quad \hat{x}^{2h} \leftarrow V^{2h}(\hat{x}^{2h}, b^{2h})$
3. Correct $\hat{x}^h \leftarrow \hat{x}^h + \mathcal{P}_{2h}^{h} \hat{x}^{2h}$
4. Relax ν_2 times on $A^h x^h = b^h$ with initial guess $\hat{x}^h$.

We conclude this section by noting that each of the components of multigrid has its own advantages and disadvantages. However, the multigrid method derives its power by combining all these methods exploiting their advantages while avoiding their disadvantages.

A. Algebraic Multigrid

The standard multigrid methods, SMG, are focused on solving a continuous problem with a known underlying geometry. The process involves discretizing the operator on a sequence of increasingly refined grids and defining *proper* transfer operators between the grids. The coarsest grid is chosen so that the cost of solving the residual problem at that grid is negligible. On the other hand, the finest grid is chosen to provide a desired degree of accuracy. Typically, SMG methods involve *uniform* coarsening and *linear* interpolation to define the coarse grid and the grid transfer operators.

Note that for certain classes of problems, it may be hard or even impossible to apply the standard multigrid technique. One such interesting class of problems that relates directly to the power grid problem, is the class of *originally discrete* problems. For an originally discrete problem with unknown geometry, discretization at different resolution grids is impossible. Even if the geometry of the problem is known, discretization may still be hard especially if the geometry is *irregular*. This is because *uniform* coarsening and *linear* interpolation can't be applied to a discrete problem defined on an *irregular* grid. As a matter of fact, it is hard to define what uniform coarsening of an irregular grid means. For such problems, the algebraic multigrid, AMG, provides an alternative to standard multigrid, SMG, which attempts to solve a *general* system of equations using the multigrid principles. Of course, for algebraic multigrid to be a competitive alternative, it has to maintain the efficiency advantage of the standard multigrid methods.

Both, SMG and AMG, involve relaxation and coarse grid correction. Furthermore, the efficiency of either method relies mostly on the choice of the multigrid components: the relaxation operator and the inter-grid transfer operators. In SMG methods, *uniform* coarsening and *linear* interpolation define the coarse grid and the grid transfer operators. Thus, the efficiency of SMG methods is decided by the choice of the relaxation operator which is chosen to reduce those error components not well approximated by coarse grid correction [16]. AMG methods, on the other hand, work the opposite way. That is, the choice of the relaxation operator is first fixed and then, the coarsening procedure and interpolation technique are chosen to reduce those error components not well reduced by smoothing. Thus, the efficiency of AMG methods is decided by the choice of the

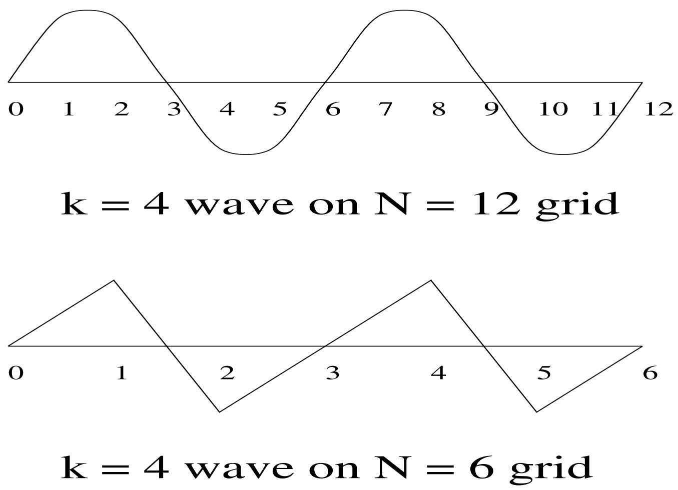

Fig. 7. Low frequency modes on fine grid Ω^h appear more oscialltory on coarse grid Ω^{2h}.

coarsening procedure and the interpolation method [16].

Consequently, most of the following analysis will target coarse grid correction. Specifically, the coarse grid correction operator at level m, C^m, is defined as follows [11], [16]:

$$C^m = I^m - \mathcal{P}^m_{m+1}(A^{m+1})^{-1}\mathcal{R}^{m+1}_m A^m \quad (6)$$

where I^m is the identity matrix at level m, A^m is the system matrix at level m, A^{m+1} is the system matrix at level $m+1$ (the coarser grid), and $\mathcal{R}^{m+1}_m$ and $\mathcal{P}^m_{m+1}$ are the inter-grid transfer operators. $\mathcal{R}^{m+1}_m$ is the restriction operator used to map the problem to the coarser grid: $b^{m+1} = \mathcal{R}^{m+1}_m b^m$. $\mathcal{P}^m_{m+1}$ is the prolongation operator used to map the solution back to the fine grid: $x^m = \mathcal{P}^m_{m+1}x^{m+1}$. In AMG, the intergrid transfer operators, as well as the coarse grid system matrix, A^{m+1}, are completely defined once the interpolation (prolongation) operator, $\mathcal{P}^m_{m+1}$ is defined [11], [16]:

$$\mathcal{R}^{m+1}_m = (\mathcal{P}^m_{m+1})^T \quad \text{and} \quad A^{m+1} = \mathcal{R}^{m+1}_m A^m \mathcal{P}^m_{m+1} \quad (7)$$

Analysis of AMG and the properties of the intergrid transfer operators lead to the following important result which indicates that the space of solution vectors at level m, $\mathbf{R}^{n_m}$, can be decomposed into two subspaces, the range space of $\mathcal{P}^m_{m+1}$ and the null space of $\mathcal{R}^{m+1}_m A^m$ as follows [11]:

$$\mathbf{R}^{n_m} = R(\mathcal{P}^m_{m+1}) \oplus N(\mathcal{R}^{m+1}_m A^m) \quad (8)$$

It follows then that the error $e^m \in \mathbf{R}^{n_m}$ can always be decomposed into two components $e^m = s^m + t^m$ where $s^m \in R(\mathcal{P}^m_{m+1})$ and $t^m \in N(\mathcal{R}^{m+1}_m A^m)$. The effect of coarse grid correction on each of the error components is given by the following [11]:

$$C^m s^m = 0 \quad (9)$$

$$C^m t^m = t^m \quad (10)$$

(9) clearly indicates that coarse grid correction perfectly approximates and thus completely nullifies those error components in the range space of interpolation. (10), on the other hand, shows that coarse grid correction has no effect on those error components in the null space of $\mathcal{R}^{m+1}_m A^m$.

The above analysis together with the definition of *algebraic smoothness* provide a mechanism for choosing the interpolation operator. So we define *algebraic smoothness* next. An error is defined to be algebraicly smooth if it is characterized by the fact that the residual, $r = Ae$, is small compared to the error, $r \ll e$ [16]. Furthermore, it is expected that on average $|r_i| \ll a_{ii}|e_i|$ [16]. This observation proves useful in providing a good approximation of the error in terms of its neighboring error values:

$$0 = a_{ii}e_i - r_i + \sum_{j \in N_i} a_{ij}e_j \approx a_{ii}e_i + \sum_{j \in N_i} a_{ij}e_j \quad (11)$$

where $N_i = \{j \neq i : a_{ij} \neq 0\}$ denotes the neighborhood of i. Geometrically, N_i denotes the grid nodes which are directly connected to node i.

Furthermore, since the A matrix is an $\mathcal{M}$-matrix, it can be shown that the error satisfies the following inequality [16]:

$$\sum_{j \neq i} \frac{|a_{ij}|}{a_{ii}} \frac{(e_i - e_j)^2}{e_i^2} \ll 2 \quad (12)$$

(12) states that the smooth error varies *slowly* in the direction of strong connections. That is, if $\frac{|a_{ij}|}{a_{ii}}$ is relatively large, then $(e_i - e_j)$ has to be small to satisfy (12) and thus, variation in the error values between nodes i and j is small.

(11) and (12) provide a mechanism for defining a *good* interpolation operator. In this context, *good* refers to an interpolation operator such that the error approximately lies in its range space. Furthermore, most AMG grid reduction algorithms are guided by (11) and/or (12) [16].

IV. Proposed Multigrid-like Power Grid Analysis Method

In this section, we present the details of a novel approach for power grid analysis. The technique follows the general lines of thought of the multigrid theory. However, it *uniquely* targets the specifics of the power grid problem to result in an *efficient* analysis method.

Power grid analysis involves three steps:

1. Modeling the power grids, the power sources, and the power drains.
2. Formulating the linear system $Ax = b$ using modified nodal analysis (MNA).
3. Solving the linear system $Ax = b$ to obtain the voltages at all nodes of the power grid.

The necessity of *efficient* modeling of power grids, sources, and drains for *efficient* analysis has already been discussed. Furthermore, the formulation of the linear system $Ax = b$ by applying modified nodal analysis (MNA) has been illustrated earlier. Thus, it remains to discuss how to *efficiently* solve the resulting linear system $Ax = b$.

As explained in section III-A, the power grid problem is an *originally discrete* problem thus motivating a solution using the algebraic multigrid technique. However, *successful* application of algebraic multigrid requires the definition of a *good* interpolation operator. This imposes a grid reduction mechanism that satisfies the following requirement [16]:

For each *removed* node, i, every node j which is strongly connected to i should be either *kept*, or strongly connected to at least one node k, where k is both *kept* and strongly connected to i.

Satisfying the above requirement may lead to inefficient grid reduction. Specifically, the resulting reduced grid may not be coarse enough; that is, the grid reduction removes only a small number of nodes. This translates to an expensive solution, in terms of CPU time and memory, of the reduced grid. Thus, to avoid the limitation of AMG, we propose a grid reduction algorithm similar to the reduction using standard multigrid. However, since typical power grids may be irregular, our algorithm is designed to *efficiently* handle the irregularities of the power grid and produce a significantly reduced coarser grid at every iteration. The details of the algorithm are given in section IV-A.

Once the grid is reduced, our proposed approach defines the interpolation operator so as to maintain the requirements of the algebraic multigrid method. That is, the interpolation operator is defined such that the error components which are not well-reduced by smoothing lie in its range space. Following (11), assume that the interpolation operator is defined such that:

$$e_i^h = e_i^H \quad \text{if } i \text{ is kept} \tag{13}$$

and

$$e_i^h = \sum_{j \in N_i} \frac{|a_{ij}|}{a_{ii}} e_j^H \quad \text{if } i \text{ is removed} \tag{14}$$

where h and H denote the fine and coarse grids respectively. This definition guarantees that the error lies in the range space of the interpolation operator since $e^h = P_H^h e^H$. Note that if $\frac{|a_{ij}|}{a_{ii}}$ is small, then the effect of node j is negligible and thus, can be ignored. That is, it is enough to interpolate the voltage at a removed node i from those nodes that are *strongly connected* to i. Based on this, we interpolate the voltage at a *removed* node, m, from the voltages of those *kept* nodes which are *strongly connected* to m. Since the geometry of the grid is available, the *kept* nodes which are *strongly connected* to m can be easily identified. Typically, these would be the geometric neighbors of m, and/or their corresponding neighbors. The exact definition of the interpolation operator is discussed and illustrated in section IV-B.

Thus, the proposed method combines the advantages of the multigrid techniques, the standard and the algebraic, while avoiding their limitations. Furthermore, there is one other significant advantage of the proposed approach over regular multigrid. As noted earlier, multigrid consists of two major components: *relaxation* (or smoothing) and *coarse grid correction*. Relaxation basically smoothes the error components and coarse grid correction approximates those smooth error components. In power grids, however, the error components are typically smooth since the power grids are designed so as to have smooth voltage variation over the grid. Consequently, in solving the power grid problem, it is possible to ignore the relaxation step of the multigrid and concentrate only on the coarse grid correction step. That is exactly what is done by our proposed approach and the results in section IV-D show that this assumption leads to significant speed-ups while incurring very small errors.

Assuming smooth voltage variation and ignoring the relaxation step, our proposed method falls under the category of *direct* solvers as opposed to the multigrid technique which is an *iterative* solver. This promises even more speedups when performing transient analysis. Given an initial grid Ω^h with the associated linear system $A^h x^h = b^h$, the proposed approach can be summarized as follows:
1. Apply the grid reduction algorithm described in section IV-A to produce a coarser grid Ω^{2h}.
2. Define the interpolation operator, $\mathcal{P}_{2h}^h$ as explained in section IV-B.
3. If Ω^{2h} is not coarse enough:
 (a) Copy Ω^{2h} to Ω^h.
 (b) Update the interpolation operator, $\mathcal{P}_{2h}^h$.
 (c) Go to step 1.

If, on the other hand, Ω^{2h} is coarse enough:
 (a) Map problem to coarse grid: $A^{2h} = \left(\mathcal{P}_{2h}^h\right)^T A^h \mathcal{P}_{2h}^h$ and $b^{2h} = \left(\mathcal{P}_{2h}^h\right)^T b^h$.
 (b) Solve problem at coarse grid, $A^{2h} x^{2h} = b^{2h}$.
 (c) Map solution back to fine grid: $x^h = \mathcal{P}_{2h}^h x^{2h}$ and exit.

Note that the criterion for the reduced grid to be coarse enough is user-specified. The criterion for a coarse-enough grid involves a tradeoff between accuracy and speed. The coarser the grid is, the faster the solution is but the less accurate. Typically, a coarse-enough grid is a grid where the size of the problem is small enough to be solved efficiently. In our implementation, four levels of grid reduction (chosen empirically) prove sufficient to result in a reduced system which can be solved efficiently.

In the rest of this section, we will be mostly concerned with the *coarse grid correction* process which is completely defined once the coarsening strategy is chosen and the interpolation operator defined. In section IV-A we define

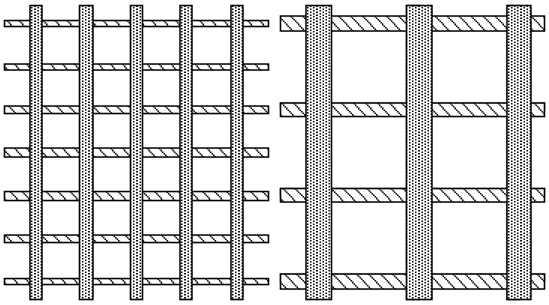

Fig. 8. Multiple resolution power grids.

StatusFlag	Indication
N	No flag (default)
K	Kept
H	Visited Horizontally
V	Visited Vertically
R	Removed

TABLE I
MEANING OF STATUS FLAGS.

and discuss the grid reduction algorithm. Then in section IV-B, we illustrate how the interpolation operator is defined. Finally, in section IV-C, the advantages of using the proposed multigrid-like technique for transient analysis are discussed.

A. Grid Reduction

A natural method for efficient grid reduction, inspired by SMG, is to skip every other wire, resulting in a situation as in Fig. 8. However, typical power grids may be *irregular*, i.e. different edges may have different lengths and different separation distances. Thus, the reduction algorithm should present a systematic mechanism for reducing any *general* grid. Furthermore, the algorithm should maintain the structure of the original grid so that it can be recursively applied until a coarse enough grid is obtained.

The major objective of our reduction algorithm is to remove as many nodes as possible while maintaining the ability to estimate voltages at the removed nodes by interpolation. The algorithm takes as input a fine grid Ω^h and a list of nodes to be *kept* and produces as output a reduced grid Ω^{2h} with a smaller number of nodes. The list of *kept* nodes should consist of specific nodes of interest to the user, but our technique automatically generates a default list containing the corner nodes and nodes where voltage sources are located.

The algorithm makes use of certain status flags, which are explained in Table I, to decide whether a node is kept or removed. Furthermore, these flags indicate how to interpolate the voltage at a removed node from its kept neighbors. The grid reduction algorithm makes repeated use of a so-called node *update* operation, which is defined as follows: *Starting from that node, go along a horizontal (vertical) direction and flag all visited nodes with H (V). Flag extremities as kept. A node which is visited both horizontally and vertically (flagged with both H and V), is flagged as kept.* The algorithm consists of three passes described as follows:

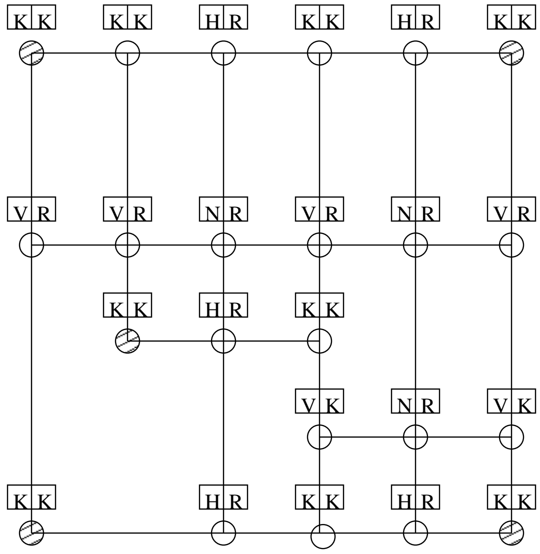

Fig. 9. Reduction of an irregular grid.

1. First Pass: *Update* every *kept* node.
2. Second Pass: For each H (V) node, flag it as removed (R). Flag its neighbors along the same row (column) as *kept* and *update* those neighbors. If a node is not flagged (N), flag it as removed (R), flag its *diagonal* neighbors as *kept*, and *update* those nodes.
3. Third Pass (defines interpolation): Voltage of a *kept* node is the same as that computed at the coarser grid. Voltage of an H (V) node which is then flagged as R is interpolated from its row (column) neighbors' voltages. Voltage of an N node which is then flagged as R is interpolated from its *diagonal* neighbors which are *kept*.

The *diagonal* neighbors of a node X are defined as those nodes reached by going 2 steps from X first horizontally and then vertically or first vertically and then horizontally. For example, if node Y is the upper neighbor of node X, then the left and right neighbors of Y are diagonal neighbors of X.

The algorithm is illustrated by the irregular grid Ω^h shown in Figure 9 which will be reduced to result in the grid Ω^{2h}. In our implementation of the grid reduction algorithm, grid nodes are ordered from top to bottom, left to right. However, note that this is not a limitation of the algorithm which is robust enough to handle any ordering of the nodes.

Initially, all nodes have the default status of N except for the nodes which should be kept. In this example, these would be all the corner nodes of the grid (dashed nodes in Fig. 9). A tag consisting of two fields is associated with every node of the grid. The left field indicates the status of the node after the first pass and the right field indicates the status of the node after the second pass.

As shown in Fig. 9, after the first pass, an edge (row or column) consisting of at least one *kept* node, has its extremities flagged as *kept*. The remaining nodes on that edge are flagged with H or V based on whether the edge is horizontal or vertical. Note that some nodes still have a status flag of N which indicates that these nodes have not

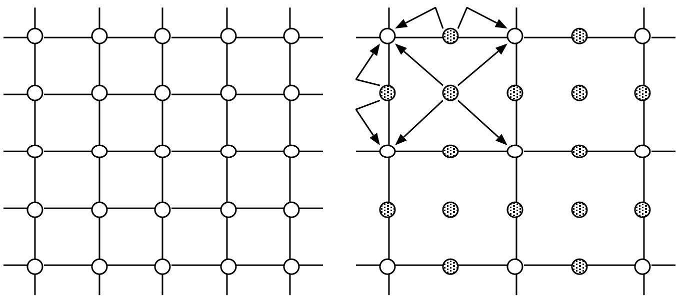

Fig. 10. Basic Multigrid operator.

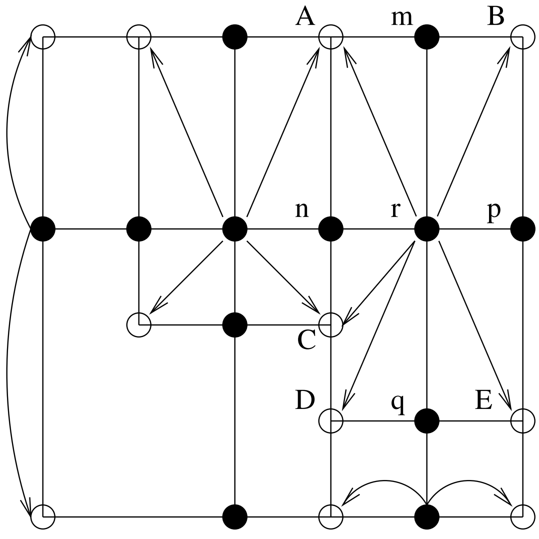

Fig. 11. Interpolation from reduced grid nodes.

been visited during the first pass. Then after the second pass, nodes with a K flag are kept while those with an R flag are removed thus resulting in the coarser grid Ω^{2h}.

Finally, we point out that if the original grid is *regular*, then the algorithm is optimal. That is, it results in maximal reduction in the number of nodes as illustrated in Figure 10. In that case, every grid reduction results in a linear system with approximately 4x fewer unknowns and consequently 8x smaller CPU time for solution by direct sparse matrix methods.

B. Interpolation

AMG interpolation is guided by (11) and (12). Thus, the interpolation operator should be chosen to relate the voltage of a *removed* node, i, to the voltages of those *kept* nodes which are *strongly connected* to i. Typically, AMG considers a connection between two nodes, i and j, to be strong when $|a_{ij}|/\max_{l\neq i}|a_{il}| \geq \theta$, where $0 \leq \theta \leq 1$ (θ is typically chosen to be 0.25 in practice [16]). With such a choice of the interpolation operator, the *coarse grid correction* would efficiently reduce the error.

In our reduction algorithm, the status flags indicate which neighbors of a removed node are to be used for interpolation, based on the fact that they are *kept* and *strongly connected* to a *removed* node m. As for the interpolation weights, these are obtained by considering the values of conductances between the nodes. Thus, if the voltage at a removed node m is interpolated from the voltages at nodes A and B, then the (linear) interpolation function $INT()$ is defined as:

$$V(m) = INT(V(A), V(B)) = a_0 V(A) + a_1 V(B) \qquad (15)$$

where $a_0 = \frac{g_{mA}}{g_{mA}+g_{mB}}$ and $a_1 = \frac{g_{mB}}{g_{mA}+g_{mB}}$. Here, g_{mA} it is the conductance between nodes m and A, and g_{mB} is the conductance between nodes m and B. Note that our technique for choosing the interpolation weights is inspired by the technique used in AMG. To illustrate, consider a *removed* node m whose voltage will be interpolated from the voltages at the *kept* nodes A and B. AMG uses the following interpolation scheme [16]:

$$V(m) = \frac{|a_{mA}|}{a_{mm}} V(A) + \frac{|a_{mB}|}{a_{mm}} V(B) \qquad (16)$$

where a_{mA} is the entry of the A matrix relating nodes m and A, and a_{mB} is the entry of the A matrix relating nodes m and B. As for a_{mm}, one AMG approach is to define it as the diagonal entry of the A matrix corresponding to node m. Another common AMG method defines a_{mm} as: $a_{mm} = |a_{mA}|+|a_{mB}|$. For the power grid problem, $|a_{mA}| = g_{mA}$ and $|a_{mB}| = g_{mB}$, which shows that our interpolation technique is motivated by AMG.

However, this is not the full story. Recall that our grid reduction algorithm differs from AMG grid reduction methods; it is actually based on SMG reduction - it uses only geometric information and removes as many nodes as possible. Hence, it is possible to come across cases where a removed node i has all the nodes that are *strongly connected* to it removed as well. To illustrate this, consider Fig. 11, where a filled node indicates a *removed* node and a blank node indicates a node that is *kept.* In this example, we assume that every horizontal or vertical link represents a *strong connection* but two nodes that are separated by two or more links are not strongly connected. This situation is typical of power grids. Thus, r is strongly connected to m and m is strongly connected to B, but r and B are not strongly connected. Nodes such as B that are separated by two strong links from r, but which are themselves not strongly connected to r, are said to be two-level strongly connected to r. Our reduction would remove node r, as well as all the nodes that are strongly connected to it, m, n, p, and q. However, it can be shown that our algorithm guarantees that, if a node i is removed, either some nodes that are strongly connected to i are kept or some nodes that are two-level strongly connected to i are kept. Therefore, in our interpolation technique, if all strongly connected neighbors of a node have been removed along with it, we use its two-level strongly *kept* neighbors for interpolation. This is clearly illustrated in Fig. 11 where the voltage at node r is interpolated from those nodes which are two-level strongly connected to r; specifically, nodes A, B, C, D, and E. Note that this approach maintains the advantage of efficient grid reduction as well as meets the requirement of a *good* interpolation operator.

C. Time Domain Analysis

In section II-B we pointed out that the fixed time step BE integration method offers large efficiency gains because it requires only one matrix inversion for all time steps. However, this efficiency comes at the cost of requiring the use of a *direct* solver. Since our proposed approach uses a *direct* solve, it is clear that it promises significant speed-ups when transient analysis is performed.

In addition to being advantageous over *iterative* solvers, the proposed multigrid-like technique offers significant advantages over *direct* solvers as well. Basically, given an $N \times N$ linear system $Ax = b$, the proposed technique solves this system by mapping the problem to a reduced $N_H \times N_H$ linear system $A_H x_H = b_H$ where $N_H \ll N$, solving the reduced system, and mapping the solution back to the original system.

Note that most of the cost for solving the original system using the proposed technique lies in solving the reduced system $A_H x_H = b_H$. Furthermore, both A and A_H are sparse matrices and thus a factorization of either matrix is significantly more expensive than a forward/backward solve [9]. However, since $N_H \ll N$, then factorizing A is significantly more expensive than factorizing A_H. The same observation holds true for performing a forward/backward solve. Consequently, the proposed technique promises more speed-ups when transient analysis is performed since transient analysis involves several forward/backward solves.

It is clear that *direct* solvers offer significant speed-ups over *iterative* solvers when transient analysis is performed. However, the major problem with *direct* solvers is their high memory demand which proves to be a limiting factor for many applications [1]. As a matter of fact, if the dimension of a linear system is very large, it may be impossible to solve such a system using a *direct* solver. In such cases, an *iterative* solver has to be used and the problem is seriously aggravated when performing transient analysis because an iterative solver has to be used at every time step thus losing the speed-up advantage of *direct* solvers.

However, our proposed technique offers an efficient solution to this problem because it uses a *direct* solver to solve a reduced system of a much smaller dimension. That is, our technique avoids the memory limitation of *direct* solvers while maintaining their speed-up advantage. It avoids the memory limitation by solving a reduced system of a much smaller dimension. On the other hand, it maintains the speed-up advantage because the reduced system matrix is factorized only once with several forward/backward solves performed for transient analysis. Of course, the advantages of the proposed technique come at a slight cost in the accuracy of the solution since the relaxation step is ignored. However, the results in the next section show that this error is small enough to maintain the efficiency and suitability of the proposed technique.

D. Experimental Results

The proposed multigrid method has been implemented and integrated into a linear simulator written in C++. All experimental results reported in this section were obtained by running the simulations on a 400MHz ULTRA 2 Sun workstation with 2GB of RAM and running the SunOS 5.7 operating system.

The practicality and efficiency of the proposed technique are illustrated by applying it for the analysis of the power grids of two *real* industrial ASIC designs. We will refer to these designs as C_1 and C_2. Both designs, C_1 and C_2 are 0.18μ CMOS designs and have a supply voltage of 1.8 V. Given the power grid, the technique requires as input the currents associated with the different power drains on the chip.

Different current measures can be used for the analysis depending on the application of interest. For instance, while peak current is a good representative measure for IR drop, average current is a better measure for electromigration analysis. On the other hand, a current waveform is the suitable current measure for transient analysis. A straight-forward technique for obtaining any current measure of interest is to simulate the power drains under *nominal* loads and *realistic* switching factors. This is how the current measures we use for our analysis are obtained. In all our experiments for DC analysis, we use the peak current drawn by the power drains as our current measure. As for transient analysis, the current measure used is the current waveform associated with the different power drains.

The irregular power grids of the two designs, C_1 and C_2, were simulated. Several grid reductions are applied and the problem accordingly mapped to the coarser grids (as explained earlier, the reduction is repeated until the grid is coarse enough as specified by the user). Specifying four levels of reduction, Table II shows the number of nodes of the grid at every level. Table II also shows the CPU times for solving the given linear system using both a regular direct solver (shown in column 4) as well as the proposed multigrid-like technique (shown in column 5). It is clear that the proposed technique is almost 16× to 20× faster than traditional simulation. Note that the same direct solver is used for solving both the original system as well as the reduced system which verifies that the speedup is not due to an advantage of one solver over another.

In order to verify the accuracy of the results provided by the proposed technique, the exact solution is compared to the estimated solution returned by the proposed technique. The histograms of percentage errors in the voltages of the different nodes of the power grids corresponding to the two designs, C_1 and C_2, are shown in Figures 12 and 13 respectively.

For the design C_1, the distribution of the node voltage errors has a mean of -0.0077% and a standard deviation of 0.0333%. As for the design C_2, the error distribution has a mean of -0.0026% and a standard deviation of 0.0167%. Furthermore, Figures 12 and 13 also show that the errors at all the power grid nodes of both designs lie in the -1.0% to 1.0% range. In fact, design C_1 has errors that range from -0.93% to 0.66% while design C_2 has errors that range from -0.30% to 1.0%. Thus, it is clear that the proposed technique provides an accurate solution to the power grid

Design name	Level	Number of nodes	Exact solve time (sec)	MG solve time (sec)
C_1	0	318074	456.79	21.6
	1	187630		
	2	128864		
	3	101209		
	4	86883		
C_2	0	671088	1114.13	69.28
	1	421460		
	2	310143		
	3	258591		
	4	231982		

TABLE II
GRID REDUCTION AND CPU TIMES USING EXACT SOLVE AS WELL AS MULTIGRID-LIKE TECHNIQUE.

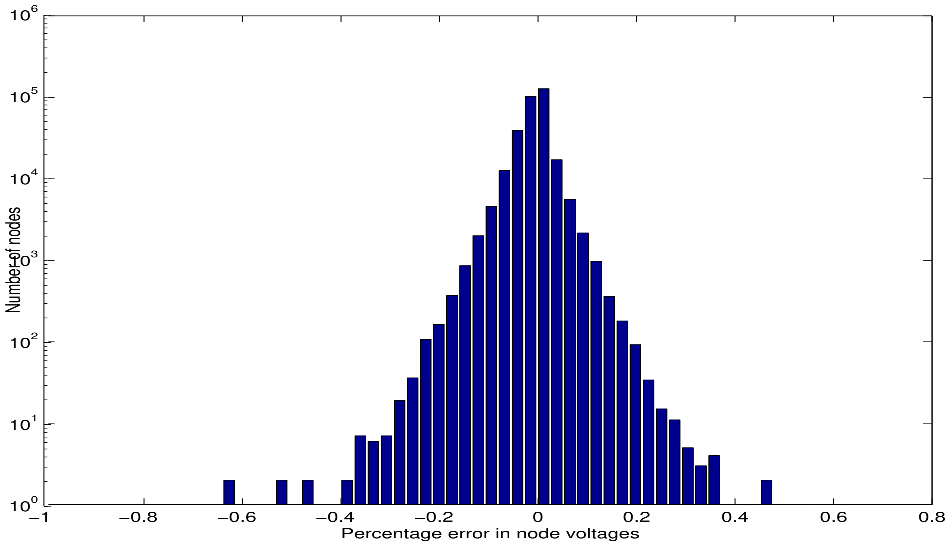

Fig. 12. Error in nodes voltages for the C_1 design.

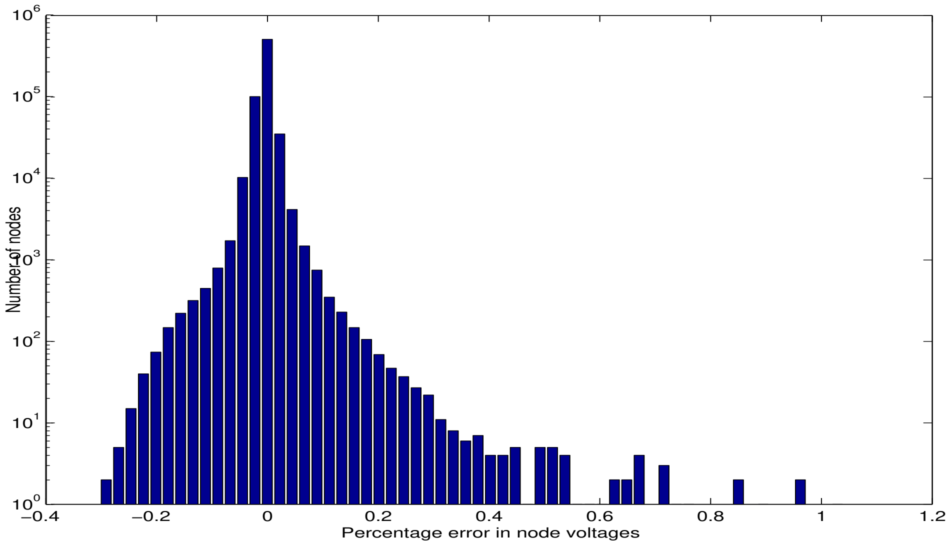

Fig. 13. Error in nodes voltages for the C_2 design.

Design name	Exact transient solve time	MG transient solve time
C_1	921.16 seconds	28.32 seconds
C_2	14.3 hours	86.36 seconds

TABLE III
Grid reduction and CPU times for transient analysis.

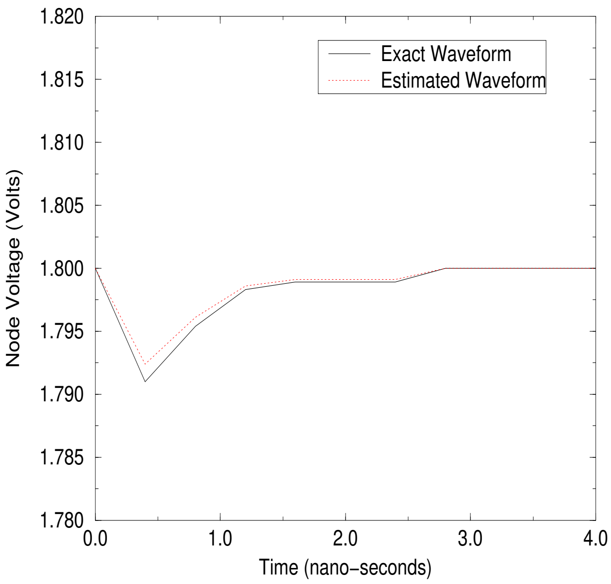

Fig. 14. Error in the voltage waveform at a power grid node in the C_1 design.

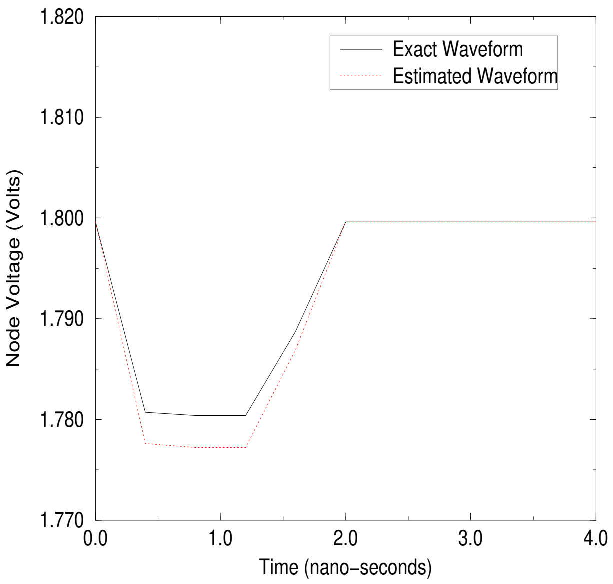

Fig. 15. Error in the voltage waveform at a power grid node in the C_2 design.

problem at a significant speed-up over regular solvers.

As explained earlier, the proposed multigrid-like technique is even more advantageous when applied for transient analysis. This is illustrated by Table III which shows the time required to run a transient simulation of the power grids of the two designs using a regular solver and the proposed technique. The power grids are simulated for a duration of $4ns$ with $0.4ns$ time steps. The speedup advantage is clear in both cases. However, it is more significant in the case of the C_2 design and the reason is that design C_2 is simulated using an iterative solver due to memory limitations. Thus, for each time step, the power grid is simulated using the iterative solver requiring a total of 14.3 hours. The proposed technique, on the other hand, uses a direct solver to solve the problem at the reduced grid. Thus, only one initial factorization is needed and only forward/backward solves are needed at the remaining time steps. The total time required for transient analysis using the proposed multigrid-like technique is 86.36 seconds, representing a speed-up of 600× for transient analysis.

Other methods to speed-up power grid analysis have been proposed [5]. In [5], a hierarchical power grid analysis technique is proposed which gives speed-ups between 2× and 5× for DC analysis. The authors also propose utilizing parallelism which increases the speed-ups to the range 10× to 23× [5]. However, their proposed method offers no speed-ups when transient analysis is applied in serial mode. Smaller speed-ups between 1.8× and 5.1× can still be observed when parallel execution is used for transient analysis. Note that the speed-up comparison is a function of the linear solvers being used as well as the size of the problems being solved. However, experimental results show that our method promises more significant speed-ups at a minimal cost in accuracy. Furthermore, these speed-ups are evident for both DC as well as transient analysis.

Finally, it remains to verify the accuracy of the resulting approximate solution. This is illustrated by Figures 14 and 15 which show the voltage waveform at some node of the power grids of designs C_1 and C_2 respectively. It is clear that the multigrid-like technique accurately tracks the exact voltage waveform at the given node. Figures 16 and 17 show the estimation error in the voltage *drop* (as opposed to voltage), showing a worst-case error of about 16%. Thus, the multigrid-like technique provides relatively good accuracy for both DC as well as transient analysis of the power grids with the added advantage of significant speed-up over regular analysis techniques.

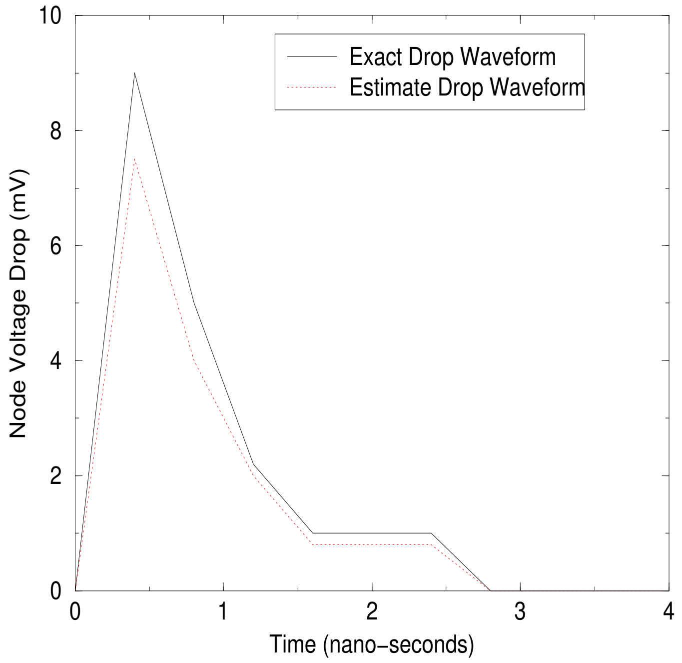

Fig. 16. Error over time in the voltage drop waveform at a power grid node in the C_1 design.

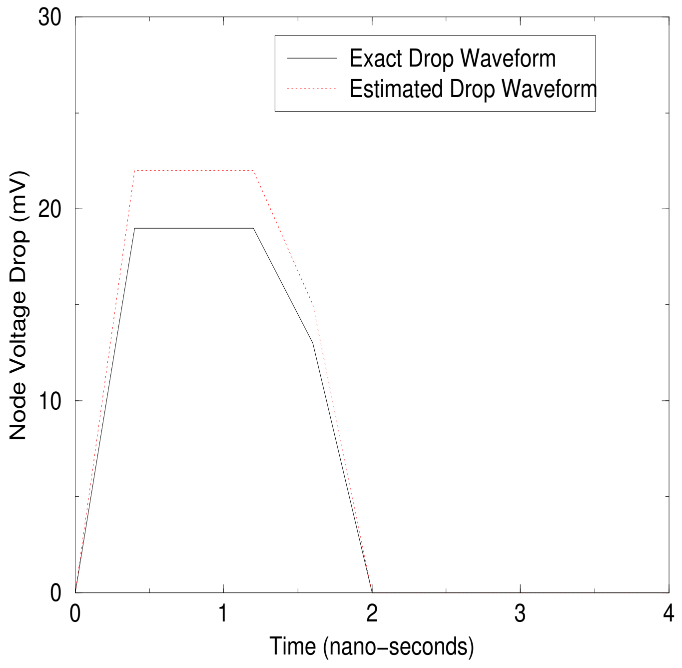

Fig. 17. Error over time in the voltage drop waveform at a power grid node in the C_2 design.

V. Conclusion

An efficient PDE-like method for power grid analysis is presented. It follows the basic lines of thought of the multigrid technique which is widely used for the solution of smooth PDE problems. However, the proposed technique falls under the category of *direct* solvers and thus, significantly differs from the regular multigrid method which falls under the category of *iterative* solvers. Experimental results on *real* designs show speed-ups of one to two orders of magnitude over current methods for both DC as well as transient analysis.

References

[1] A. Dharchoudhury et. al. Design and analysis of power distribution networks in powerpc^{TM} microprocessors. In *Proceedings of DAC*, 1998.

[2] S. O. Bae J. S. Yim and C. M. Kyung. A floorplan-based planning methodology for power and clock distribution in asics. In *Proceedings of DAC*, 1999.

[3] G. Steele et. al. Full-chip verification methods for dsm power distribution systems. In *Proceedings of DAC*, 1998.

[4] S. R. Nassif and J. N. Kozhaya. Fast power grid simulation. In *Proceedings of DAC*, 2000.

[5] M. Zhao et. al. Hierarchical analysis of power distribution networks. In *Proceedings of DAC*, 2000.

[6] H. H. Chen and J. S. Neely. Interconnect and circuit modeling techniques for full-chip power noise analysis. IEEE Transactions on Components and Packaging II, 1998.

[7] R. A. Rohrer L. T. Pillage and C. Visweswariah. *Electronic and System Simulation Methods*. McGraw-Hill, Inc., 1995.

[8] L. W. Nagel. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. PhD thesis, University of California, Berkeley, 1975.

[9] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[10] A. Berman and R. J. Plemmons. *Nonnegative Matrices in the Mathematical Science*. Academic Press, New York, 1996.

[11] W. L. Briggs. *A Multigrid Tutorial*. SIAM, 1987.

[12] A. Brandt. Multi-level adaptive technique (mlat) for fast numerical solution to boundary value problems. Proceedings Third International Conference on Numerical Methods in Fluid Mechanics, Paris 1972, 1973.

[13] W. Hackbusch. Multi-grid methods and applications. Springer-Verlag Berlin, 1985.

[14] A. Brandt. Multi-level adaptive techniques (mlat). i. the multigrid method. Research Report RC 6026.

[15] K. Stüben and U. Trottenberg. Multigrid methods: Fundamental algorithms, model problem analysis and applications. Multigrid Methods Proceedings, edited by W. Hackbusch and U. Trottenberg Köln-Porz, 1981, 1982.

[16] J. W. Ruge and K. Stüben. Algebraic multigrid. Multigrid Methods, edited by S. McCormick, 1987.