Fast Poisson Solver Preconditioned Method for Robust Power Grid Analysis[*]

Jianlei Yang Yici Cai Qiang Zhou Jin Shi
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology
Tsinghua University, Beijing, China
yjl09@mails.tsinghua.edu.cn caiyc@mail.tsinghua.edu.cn
zhouqiang@tsinghua.edu.cn maplesj@gmail.com

ABSTRACT

Robust and efficient algorithms for power grid analysis are crucial for both VLSI design and optimization. Due to the increasing size of power grids IR drop analysis has become more computationally challenging both in runtime and memory consumption. This work presents a fast Poisson solver preconditioned method for unstructured power grid with unideal boundary conditions. In fact, by taking the advantage of analytical formulation of power grids this analytical preconditioner can be considered as sparse approximate inverse technique. By combining this analytical preconditioner with robust conjugate gradient method, we demonstrate that this approach is totally robust for extremely large scale power grid simulations. Experimental results have shown that iterations of our proposed method will hardly increase with grid size increasing once the pads density and the range of metal resistances value distribution have been decided. We demonstrated that this approach solves an unstructured power grid with 2.56M nodes in only 1/3 iterations of classical ICCG solver, and achieves almost 20X speedups over the classical ICCG solver on runtime.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids—*Simulation*

General Terms

Algorithms, Design, Performance, Verification

Keywords

Power Grid, Fast Poisson Solver, Preconditioning

1. INTRODUCTION

The design and analysis of extremely large scale power grids is a very challenging task for VLSI design. Many contributions have been developed to run power grid simulations including direct solvers and iterative solvers. Direct solvers are robust but not adequate for tremendous amount of power grid nodes because of CPU speed and memory limitation. Iterative solvers are more memory efficient but unstable because of performance limitation by preconditioner [1]. Especially there are some PDE-like solvers such as Random Walk [2], Multigrid Methods [3], Domain Decomposition Methods [4], Hierarchical Methods [5] and matrix techniques based solvers such as SPAI [6] and $\mathcal{H}$-Matrix [7], but all of them pose some weakness in either efficiency or robustness when addressing very large industrial designs.

A friendly fast Poisson solver [8] using GPU-based FFT acceleration was proposed as an analytical direct method for solving 2D structured power grids with the computation complexity of $\mathcal{O}(NlogN)$. But the fast Poisson solver above was mostly suitable for highly structured grids, which may limit its practical applications in general grids. Some GPU based Multigrid methods [9][10] were proposed to solve 3D irregular power grids. However, the GPU based Multigrid methods above ignored via resistances when mapping 3D irregular grids to 2D regular grids. Obviously it will result in considerable errors therefore slow convergence. Especially with the industrial CMOS process become more advanced, these approaches will converge very slowly for power grids with bad via, such as via with big resistance value.

In this paper, we propose a fast Poisson solver preconditioned iterative method for general grids with unideal boundary conditions. In this approach, multilayer power grid is modeled as several single layers by treating vias as current sources. Then each layer is transformed into a structured grid so that all layers can be modeled by analytical formulation. Further, by taking the advantage of analytical formulation of transformed grids we propose an analytical preconditioner which can be considered as sparse approximate inverse technique. At last, an efficient and robust Fast Poisson Solver Preconditioned Conjugate Gradient method (FPS-PCG) is introduced to solve the original unstructured grids. By theoretical analysis for sparse approximate inverse technique we proof that this analytical preconditioner is very close to exact inverse which is only influenced by the distribution rage of metal resistances value. Due to the certain regularity of real power grid designs this distribution rage is often so small that a good preconditioning performance can be guaranteed. Beneficially, the iterations will hardly increase with the grid size increasing once the distribution rage has been decided.

This paper is organized as follows. Section 2 presents the power grid analysis background and brief introduction of our proposed approach. Section 3 provides the analytical formulation for power grids. Section 4 is the efficient implementation of proposed FPS-PCG method. Experimental results on large scale power grids are shown in Section 5. Concluding remarks are given in Section 6.

2. BACKGROUNDS AND OVERVIEW

[*]This work is supported by National Natural Science Foundation of China (NSFC) No.60976035.

978-1-4577-1400-9/11/$26.00 ©2011 IEEE

2.1 Power Grid Analysis Background

For DC simulation, power grid can be modeled as linear resistive network. By using Modified Nodal Analysis (MNA) method, a n-node circuit network can be formulated as the following linear system equations [1]:

$$GV = I \tag{1}$$

where the conductance matrix $G \in \mathbb{R}^{n\times n}$ is a symmetric positive definite (*s.p.d.*) matrix which represents the interconnecting relationship and resistor values, $V \in \mathbb{R}^{n\times 1}$ is an unknown vector of node voltages, $I \in \mathbb{R}^{n\times 1}$ is an input vector of node current sources. As the VLSI technology scaling associated with significantly increasing device numbers in a die, the number of nodes in the power grid may easily exceed many millions. The most accurate and stable methods for solving such huge linear systems are sparse direct solvers such as *SuperLU* and *Cholmod*, but both of them are time expensive and memory inefficient. Another state of art approach is iterative methods especially preconditioned iterative methods which can be used to solve such linear systems with memory efficiently. However, preconditioned iterative methods are not stable for many cases because of either expensive cost or unsatisfactory performance of their preconditioners. Also there are some fast and robust solvers for special regular and structured power grids such as fast Poisson solver [8], but its regularization limits its practical applications in general power grids. To overcome this limitation we extend this fast solver to general power grids with little acceptable cost.

2.2 Prior Works and Proposed Approach

An initial planning power grid is highly regular. The metal resistances in each layer are located at a small range but totally distinct in different layers. Even though the grid is gradually modified, the typical distribution is affected so slightly that the regularity can still be exploited to improve the numerical characters. As we know, the elements distribution of system matrix corresponding to the metal segments variations dominates the eigenvalues distribution, consequently the condition number. It is obvious that dramatically conductance variations in different metal layers or vias will directly lead to slow convergence rate of iteration methods. With the industrial CMOS process become more advanced, via resistance value exceeds the sheet resistance by several orders of magnitude. Thus, the numerical character will be intensively affected by vias. All these unbalanced distributions badly affect the max to min ratio of eigenvalue to become larger.

In the other hand, aiming to take the advantage of geometry Multigrid method the MGPCG method [9] solves power grid by compressing 3D grid to 2D grid but ignoring via resistance. Obviously this strategy will result in considerable error and slow convergence later. Especially for bad via which means small conductance, this effect will be enlarged, consequently the performance of this approach will be degraded.

Since neither ICCG nor MGPCG is a smart choice we may draw some advantage from *divide and conquer* strategy. Here we use fast Poisson solver as a analytic preconditioner for conjugate gradient method to handle general unstructured grids with unideal boundary conditions. For clarity, the flow of the proposed approach is shown as Figure 1.

The original multilayer power grid is modeled as several single layers by treating vias as current sources. Because of grid regularity the vias resistance almost never change between two certain layers. For a certain layer, it calculates the width of all metal slices and uses the average value as the typical width of this layer. Then, each layer is transformed into a structured grid with typical width so that all layers can be modeled by analytical formulation. By analytic formulation the grids can be compressed as a Poisson Block which can be solved by fast Poisson solver. This fast Poisson solver is adopted as an analytical preconditioner which can be considered as sparse approximate inverse technique to accelerate the iteration method for original unstructured grids. The most advantage of this idea is that we just need to handle the single layer independently whose metal segment resistances are located at a small range. Thus, a good convergence rate is performed. In summary, the convergence and runtime efficiency of the proposed approach depend on the efficiency of FPS-PCG solver. We will demonstrated that our approach largely improves the robustness of power grid analysis.

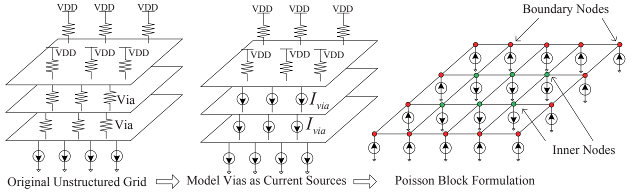

Figure 1: Overall Analysis Flow.

3. ANALYTIC FORMULATION

A concept of *Poisson Block* was introduced in [8] to explore the particularity of power grid. The *Poisson Block* corresponding to a finite difference discretization of a continuous Poisson problem on a 2D rectangular homogeneous domain with Dirichlet conditions can be solved by fast Poisson solver perfectly. For more general grids with unideal boundary conditions, new approach should be considered.

3.1 Fast Poisson Solver

Unlike the traditional formulation equation (1) for matrix G with sparse format, a dense matrix is adopted to describe the node voltages. With another two interesting matrix the *Kirchhoff's Law* can be satisfied correctly. Considering a two layer small 3×4 power grid, the nodes ordering is shown in Figure 2.

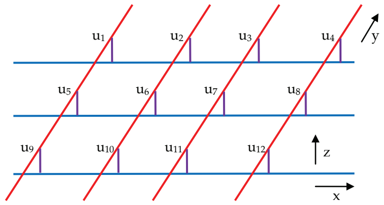

Figure 2: Topology of a two layer small power grid.

For the top layer in y-direction, voltage pads are attached to the nodes located at the boundaries. And for bottom layer in x-direction , current sources are attached to crossed nodes. Also the vertical metal segments in z-direction are vias. We suppose all metal stripes at bottom layer to have the same resistance r_1 and top layer to have the same resistances r_2 and vias to have the same resistances R because of the grid regularity. Then we can use a dense matrix U_1 to describe the node voltages at bottom layer and U_2 for top layer in which each element represents each node voltage.

The current source loadings can be formulated as matrix F. Particularly, grid vias are modeled as current loadings from top layer to bottom layer which can be formulated as matrix I.

With two tridiagonal matrix T_1 and T_2 the Kirchhoff Current Law for top layer and bottom layer can be represented by two matrix equations:

$$\frac{T_2 \cdot U_2}{r_2} = I, \qquad \frac{U_1 \cdot T_1}{r_1} = F - I \tag{2}$$

$$T_1 = \begin{pmatrix} -1 & 1 & & \\ 1 & -2 & 1 & \\ & 1 & -2 & 1 \\ & & 1 & -1 \end{pmatrix}, T_2 = \begin{pmatrix} -1 & 1 & \\ 1 & -2 & 1 \\ & 1 & -1 \end{pmatrix}$$

Also for vias there is an obvious current relationship:

$$\frac{U_2 - U_1}{R} = I \tag{3}$$

By eliminating the matrix I from (2) and (3) we can obtain:

$$U_1 = U_2 - \frac{R}{r_2} \cdot T_2 \cdot U_2 \tag{4}$$

From this equation we know once U_2 is solved we can get U_1 directly by substitution.

By eliminating the matrix U_1 from (2), (3) and (4), we can obtain:

$$\frac{U_2 \cdot T_1}{r_1} + \frac{T_2 \cdot U_2}{r_2} - \frac{R}{r_1 r_2} \cdot T_2 \cdot U_2 \cdot T_1 = F \tag{5}$$

If we define voltage drop matrix D_2 for nodes on top layer and Vcc for standard supply voltage, thus we can obtain $U_2 = Vcc \cdot E - D_2 = V - D_2$ (where E is unit matrix), the equation (5) can be represented as:

$$\frac{(V - D_2) \cdot T_1}{r_1} + \frac{T_2 \cdot (V - D_2)}{r_2} - \frac{R}{r_1 r_2} \cdot T_2 \cdot (V - D_2) \cdot T_1 = F$$

By sufficiently using the special character of T_1 and T_2 (where $V \cdot T_1 = 0$, $T_2 \cdot V = 0$ and $T_2 \cdot V \cdot T_1 = 0$) and the properties of *Poisson Block* (where the boundary voltage drop of a *Poisson Block* is equal to zero), the final matrix equation can be obtained:

$$\frac{D_2 \cdot P_1}{r_1} + \frac{P_2 \cdot D_2}{r_2} + \frac{R}{r_1 r_2} \cdot P_2 \cdot D_2 \cdot P_1 = F \tag{6}$$

$$P_1 = \begin{pmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 2 \end{pmatrix}, P_2 = \begin{pmatrix} 2 & -1 & \\ -1 & 2 & -1 \\ & -1 & 2 \end{pmatrix}$$

The matrix P is typical tridiagonal *Toeplitz* matrix which has analytical eigen decomposition $P = z \cdot \Delta \cdot z^T$, where z is a symmetry orthogonal dense matrix and Δ is a diagonal matrix given by

$$z(i,j) = \sqrt{\frac{2}{n+1}} \sin\left(\frac{i \cdot j \cdot \pi}{n+1}\right), \Delta(i,i) = 2\left(1 - \cos\frac{i \cdot \pi}{n+1}\right)$$

Let $X_2 = z_2^T \cdot D_2 \cdot z_1$, substitute this analytical eigen decomposition to (6), remembering that matrix z_1 and z_2 are symmetry orthogonal matrices. The system matrix equation can be expressed as an analytical form:

$$X_2 = \left(z_2^T \cdot F \cdot z_1\right) \odot W$$
$$W(i,j) = \left(\tfrac{\Delta_1(j,j)}{r_1} + \tfrac{\Delta_2(i,i)}{r_2} + \tfrac{R}{r_1 r_2} \cdot \Delta_1(j,j) \cdot \Delta_2(i,i)\right)^{-1}$$

In this equation, operator $\odot$ means that the result matrix in brackets performs *Hadamard* matrix multiplication with matrix W. Finally, we can get the analytical voltage drop solution as shown below:

$$D_2 = z_2 \cdot X_2 \cdot z_1^T = z_2 \cdot [(z_2 \cdot F \cdot z_1) \odot W] \cdot z_1 \tag{7}$$

So far, once we know the current loadings distribution matrix F we can get the voltage drop distribution D_2 of top layer. Later, the voltage drop distribution D_1 of bottom layer can be obtained by substitution.

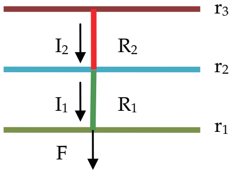

Figure 3: Demonstration of three layer power grid.

Similar analytical expression for multilayer power grid can be obtained by matrix equations. Take three layers as an example in Figure 3. As Figure 2 demonstrated, r_1, r_2 and r_3 is the typical resistance of each metal layer, matrix F is the current source loadings by transistors, R_1 and R_2 is the typical resistance of via and they are modeled as current source matrix I_1 and I_2. Also we can map all the electrical parameters to the top layer to solve the single layer, which we just need to reformulate the dense matrix W:

$$X_3 = \left(z_2^T \cdot F \cdot z_1\right) \odot W$$
$$W(i,j) = \begin{pmatrix} \frac{\Delta_1(j,j)}{r_1} + \frac{\Delta_2(i,i)}{r_2} + \frac{\Delta_2(j,j)}{r_3} \\ + \frac{R_1}{r_1 r_2} \cdot \Delta_1(j,j) \cdot \Delta_2(i,i) \\ + \frac{R_2}{r_2 r_3} \cdot \Delta_2(i,i) \cdot \Delta_3(j,j) \\ + \frac{R_1 + R_2}{r_1 r_3} \cdot \Delta_1(j,j) \cdot \Delta_3(j,j) \\ + \frac{R_1 R_2}{r_1 r_2 r_3} \cdot \Delta_1(j,j) \cdot \Delta_2(i,i) \cdot \Delta_3(j,j) \end{pmatrix}^{-1}$$

Once the voltage distribution of top layer is solved, the solution of middle and bottom layer can be obtained by substitution.

3.2 Special Consideration for Pads

As mentioned in equation (7) the current source matrix F contains some boundary elements which standard for pad nodes and their current drawn is unknown. Here we model the pad node as a voltage source Vcc with a resistance in series. Then this model can be transformed as an equivalent circuit of a current source with a resistance in parallel by *Norton's Theorem.* Once the current drawn by pads has been directly obtained we just need to update the conductance matrix G with resistance of pad and to update current source matrix F with equivalent independent current source. But for asymmetric pads distribution of Wire-Bound package or Flip-Chip package, ideal boundary conditions on *Poisson Block* will introduce corresponding errors. We will demonstrate that the main error of our approach is located near the pad nodes. Within several iterations, the error will be cut down to a satisfactory accuracy level.

4. FPS-PCG SOLVER

For large linear systems, the classical Krylov-subspace iterative methods lack for fast convergence so preconditioning technologies are developed. But it is difficult to get a good preconditioner because of neither accuracy lost nor badly memory cost. Some effective preconditioners are based on deep insight into the structure of the problem. For partial differential equations, where it is shown that certain discretized second-order elliptic problems on simple geometries can be very well preconditioned with fast Poisson solvers [11]. A domain-decomposed fast Poisson solver on a rectangle was developed for parallel implementation in [12]. Several preconditioners involved by fast Fourier transform were proposed in order to improve the convergence of iterative methods. The effectiveness of such a preconditioner has been analyzed in [13] and some of the many ways to implement the solver efficiently are discussed in [14].

4.1 FPS-Preconditioning Method

As demonstrated by analytical formulation, the basic idea behind FPS preconditioning method is to use the regularization method in *Poisson Block*. In a certain layer, within a block boundary, it calculates the width of all metal slices and uses the average value as the typical width of this layer. Then, the original irregular grid layer is transformed into a regular grid layer with typical width above. The main step is using the fast Poisson solver on this regular grid to precondition the irregular grid. The details of preconditioning algorithm are shown in Algorithm 1.

Algorithm 1: FPS preconditioning algorithm

```
Input: The average resistance value $r_{average}$, via
       resistance $R_{via}$, the residue $r$ of all grid nodes,
       the grid size $m \times n$, $z_1$, $z_2$, $\Delta_1$, $\Delta_2$, which the
       $reshape\,()$ function is a reordering method
       between matrix and vector from Matlab.
Output: The solution of $z$ for all grid nodes.
1 Formulate dense matrix W by $r_{average}$, $R_{via}$, $\Delta_1$, $\Delta_2$;
2 Formulate loadings matrix $F = reshape\,(r, n, m)$;
3 Obtain the voltage drop $D = z_2 \cdot ((z_2 \cdot F \cdot z_1) \otimes W) \cdot z_1$;
4 Return $z = reshape\left(D^T, m \cdot n, 1\right)$.
```

It should be emphasized that this analytical preconditioning method can be very efficient for unstructured grid. In another view, this analytical method is a direct solver for structured grid with ideal boundary conditions. But for unstructured grid or unideal boundary conditions, one or two preconditioning can give an approximate voltage distribution for global nodes which can be viewed as the error in low frequency has been reduced and then several preconditioning can handle the global error all around the grid which can be seemed as the error in high frequency has been smoothed. Obviously this error smoothing strategy is the essence of our fast Poisson solver preconditioning technique.

This analytic preconditioner can be viewed as an approximate inverse of conductance matrix G. We will analyze its preconditioning performance theoretically. For a linear system equation $Ax = b$, preconditioning technique is to apply iterative algorithm to $M^{-1}A$, where M is chosen so that $M^{-1}A$ is better conditioned and systems of form $Mz = y$ are easily solved. For structured power grid with metal sheet conductance g_x in x-direction and g_y in y-direction, define $\Sigma_1 = g_x \cdot I_1$, $\Sigma_2 = g_y \cdot I_2$, where Σ_1, Σ_2 are diagonal and I_1, I_2 are identity matrix, the conductance matrix G which is denoted as A can be formulated as $A = \Sigma_1 \otimes P_2 + P_1 \otimes \Sigma_2$, where the operator $\otimes$ means *Kronecker product*, P_1 and P_2 are typical tridiagonal *Toeplitz* matrix which has analytical eigen decomposition. Using properties of the *Kronecker product* and remembering that matrix z_1 and z_2 are symmetry orthogonal matrices, A can be reformulated as:

$$\begin{aligned} A &= \Sigma_1 \otimes P_2 + P_1 \otimes \Sigma_2 \\ &= (z_1 \cdot \Sigma_1 \cdot z_1) \otimes (z_2 \cdot \Delta_2 \cdot z_2) + (z_1 \cdot \Delta_1 \cdot z_1) \otimes (z_2 \cdot \Sigma_2 \cdot z_2) \\ &= (z_1 \otimes z_2) \cdot (\Sigma_1 \otimes \Delta_2) \cdot (z_1 \otimes z_2) \\ &\qquad\qquad + (z_1 \otimes z_2) \cdot (\Delta_1 \otimes \Sigma_2) \cdot (z_1 \otimes z_2) \\ &= (z_1 \otimes z_2) \cdot (\Sigma_1 \otimes \Delta_2 + \Delta_1 \otimes \Sigma_2) \cdot (z_1 \otimes z_2) \end{aligned}$$

where $(\Sigma_1 \otimes \Delta_2 + \Delta_1 \otimes \Sigma_2)$ is diagonal. If we define $\Lambda = (\Sigma_1 \otimes \Delta_2 + \Delta_1 \otimes \Sigma_2)$, the inverse of matrix A can be obtained directly by:

$$\begin{aligned} A^{-1} &= \left[(z_1 \otimes z_2) \cdot (\Sigma_1 \otimes \Delta_2 + \Delta_1 \otimes \Sigma_2) \cdot (z_1 \otimes z_2)\right]^{-1} \\ &= (z_1 \otimes z_2)^{-1} \cdot (\Sigma_1 \otimes \Delta_2 + \Delta_1 \otimes \Sigma_2)^{-1} \cdot (z_1 \otimes z_2)^{-1} \\ &= (z_1 \otimes z_2) \cdot \Lambda^{-1} \cdot (z_1 \otimes z_2) \end{aligned} \tag{8}$$

For structured power grid, we can get analytic solution by (8). And for unstructured case we adopt this analytic expression as a preconditioner. It's easy to find that the closed formulation (8) is equivalent to the step 3 of algorithm 1. Because we use the transformed structured power grid as an analytic preconditioner the difference between original unstructured grid and transformed structured grid will affect the preconditioning performance. But once the resistances value range of metal sheet is decided, the preconditioning performance will hardly affected by grid size increasing. Thus, a robust preconditioning performance is guaranteed by this analytic preconditioner with computation complexity of $\mathcal{O}(NlogN)$ which has analyzed in [8].

Algorithm 2: FPS-PCG algorithm

```
Input: The sparse conductance matrix $A \in {}^{n \times n}$, the
       FPS preconditioner $z = FPSPrecond\,(r)$, the
       rhs vector $b \in \mathbb{R}^{n \times 1}$, the residual tolerance $tol$.
Output: The solution for all grid nodes $x \in \mathbb{R}^{n \times 1}$.
$x_0 = FPSPrecond\,(b)$;
$r_0 = b - Ax_0$;
$z_0 = FPSPrecond\,(r_0)$;
$p_0 = z_0$;
for $k = 0, 1, \ldots,$ until converge to tol do
    $\alpha_k = \frac{r_k^T z_k}{p_k^T A p_k}$;
    $x_{k+1} = x_k + \alpha_k p_k$;
    $r_{k+1} = r_k - \alpha_k A p_k$;
    $z_{k+1} = FPSPrecond\,(r_{k+1})$;
    $\beta_k = \frac{r_{k+1}^T z_{k+1}}{r_k^T z_k}$;
    $p_{k+1} = z_{k+1} + \beta_k p_k$;
end
Return the solution $x_{k+1}$.
```

4.2 FPS-PCG Algorithm

The most classical preconditioner is incomplete Cholesky factorization which is neither too expensive nor inefficient for larger grids because of the tradeoff between the elements fill in and the performance of preconditioning. Especially for unbalance metal width distribution which locates at a broad range the condition number of conductance matrix G is too large so ICCG solver is hard to converge. By taking the advantage of analytical preconditioner of FPS a highly robust precondition strategy is proposed for fast and stable power grid analysis.

As shown in our experiments, the number of iterations required by FPS-PCG method is much smaller than the traditional conjugate gradient method especially for larger grid size. The fast Poisson solver preconditioned conjugate gradient algorithm has been described in Algorithm 2 with more details. The run time cost of each FPS-PCG iteration mainly comes from FPS preconditioning step. More specifically, the main cost of each FPS preconditioning process lies in matrix multiplication.

4.3 Simulation Flow for Multilayer

As demonstrated above the numerical characters are intensely sensitive to distinction of metal resistance value among different layers and vias. By analytical formulation of multilayer grids we just need to handle the single layer independently whose metal segment resistances are located at a small range. We describe a simulation flow for multilayer power grid in Algorithm 3.

Algorithm 3: Simulation flow for multilayer power grid

```
  Input: Grid parameters for each layer and via.
  Output: Node voltage distribution of each layer.
1 Map the electrical parameters of all grids to top layer
  and form the dense matrix $W$;
2 Solve the top layer by FPS-PCG method to get the
  node voltage distribution;
3 Substitute the voltage distribution on top layer to
  obtain the distribution on other layers.
```

5. EXPERIMENTAL RESULTS

Various experiments are carried out to validate the promising performance of the proposed FPS-PCG algorithm. The regular topology of power grid is adopted in this paper because it is really common in use for early design stage. All power grids are formed by unstructured metal stripes whose resistance values are randomly generated in a certain range. This range is among 0.01Ω and 1Ω which is close to real industrial designs. Also the pads distribution is unideal for boundary conditions which are closer to real designs. There are 10% grid nodes on boundaries connected to pads both for Wire-Bound and Flip-Chip package. The resistance value of pad contact is set to 5Ω. The current loadings of power grid are also generated reasonably and randomly.

All algorithms have been implemented using Matlab language and the performance of different solving techniques are compared under the same environment. The simulation platform is a Linux Server with 2 Quad-Core Intel Xeon E5620 CPU@2.4GHz and 12GB RAM. All running times are measured in seconds. All solvers are terminated when the max residue of node voltage reaches below $10^{-6}V$.

5.1 Comparing FPS-PCG with ICCG

The comprehensive results of ICCG solver and FPS-PCG solver for unstructured power grids with ideal boundary conditions are shown in Table 1. The ICCG solver is based on incomplete Cholesky factorization with zero fill-in by the threshold of 10^{-3} for drop tolerance. The preconditioning step of FPS-PCG solver is implemented by matrix multiplication directly as shown in Algorithm 1. Experiments show that the main error of FPS-PCG methods is located near the pads. But the ICCG method is different. This phenomenon can be explained by the essence of these two preconditioners. The incomplete Cholesky factorization preconditioner is limited by its threshold of elements fill-in so that the performance of preconditioning is badly weakened all around the grid. But for analytical FPS preconditioner, one or two preconditioning can give an approximate voltage distribution for global nodes and then each precondition step can handle the global error all around the grid especially for pad nodes.

Table 1: Comparison between ICCG and FPS-PCG

Grid Size	ICCG			FPS-PCG		
	Iter	Time	E_{max}	Iter	Time	E_{max}
$10K$	19	0.25	$8e-7$	59	0.08	$9e-7$
$40K$	34	1.73	$4e-7$	54	0.25	$7e-7$
$90K$	48	5.60	$7e-7$	59	0.99	$8e-7$
$160K$	57	12.41	$6e-7$	63	1.63	$7e-7$
$250K$	70	23.96	$7e-7$	57	2.29	$9e-7$
$360K$	81	40.26	$6e-7$	57	3.58	$9e-7$
$490K$	92	63.30	$9e-7$	64	5.69	$8e-7$
$640K$	103	92.89	$9e-7$	59	6.97	$7e-7$
$810K$	118	132.76	$9e-7$	62	10.02	$9e-7$
$1M$	120	175.21	$9e-7$	65	13.89	$7e-7$
$1.21M$	130	233.39	$7e-7$	64	17.98	$9e-7$
$1.44M$	145	305.02	$7e-7$	69	20.00	$7e-7$
$1.69M$	154	393.10	$9e-7$	62	26.25	$8e-7$
$1.96M$	165	500.68	$8e-7$	67	35.83	$7e-7$
$2.25M$	177	633.65	$8e-7$	64	40.17	$7e-7$
$2.56M$	185	781.93	$8e-7$	64	47.32	$7e-7$

As can be observed in Table 1, the iterations of ICCG solver increase with the grid size and the memory consumption becomes impractical because of the incomplete Cholesky factorization. A significant reduction in the runtimes is observed on all power grid designs when using fast Poisson solver preconditioning technique. As shown in Figure 4, FPS-PCG solver is more robust than ICCG solver which the iterations of FPS-PCG are surprisingly almost constant, not varying significantly with grid size increasing.

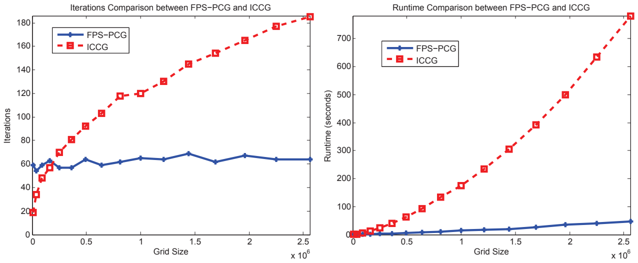

Figure 4: Convergence ratio and run time comparison between ICCG and FPS-PCG.

The computational complexity of the ICCG is near $\mathcal{O}(N^{1.5})$ where N is the number of nodes. The computational complexity of our proposed method can be found in algorithm 2. As we have proofed that the iterations are robust to grid size we just need to analyze the complexity of each iteration. For each iteration process, a FPS preconditioning step and a sparse matrix vector multiplication are implemented whose complexity is $\mathcal{O}(NlogN)$ and $\mathcal{O}(N)$ respectively. Accordingly the total computational complexity of our approach is between $\mathcal{O}(N)$ and $\mathcal{O}(NlogN)$ which is a considerable improvement. This verifies that our proposed analytical preconditioner is a good sparse approximate inverse technique for original power grid systems, and consequently performs an effective and robust preconditioner for iterative methods.

5.2 Comparing FPS-PCG with FPS

By taking the advantage of *Norton's Theorem* we don't need to carry out the boundary iteration process. Also we compare run efficiency between fast Poisson solver with boundary iteration process [8] and FPS-PCG solver both on structured grids. It should be noticed that the fast Poisson solver in [8] was implemented by C++ language and our method is implemented by Matlab. Even though there is an obvious disadvantage of programming language our method still reduces the runtime on all power grids. As shown in Table 2, FPS-PCG solver eventually achieves approximately $5X$ speedup than FPS with boundary iteration on structured grids. The following insightful experiments are also conducted. According to comparing the FPS-PCG solver on structured grids and unstructured grids we can observe that the additional iterations of unstructured grids are resulted by irregular metal resistances distribution.

At last we test our algorithm on a variety of larger scale power grids the max size of which is increased to $11.56M$ while ICCG solver can not handle this because of suffering from memory overflow and excessive runtime. Details of all experiments are demonstrated in Table 3. As observed, the proposed preconditioning technique which takes the fully advantages of analytical formulation can handle more general and larger grids with extremely high accuracy and robustness.

Table 2: Comparison between FPS and FPS-PCG

Grid Size	T_{FPS}[8]†	FPS-PCG†		FPS-PCG*	
		Iter	Time	Iter	Time
$1M$	15.64	16	3.60	65	13.68
$1.2M$	20.93	16	4.83	62	16.89
$1.4M$	27.16	16	5.81	67	22.82
$1.6M$	36.33	16	6.90	68	27.88
$1.9M$	44.87	16	8.97	63	33.22
$2.3M$	54.90	16	11.49	61	40.75
$2.5M$	66.27	16	12.00	65	48.48
$2.9M$	79.16	16	14.81	66	58.20
$3.2M$	94.30	16	16.93	60	60.43
$3.6M$	110.51	16	19.40	66	75.69
$4M$	125.63	16	22.65	67	89.90

† On Structured Power Grids
* On Unstructured Power Grids

Table 3: Analysis for larger power grids

Grid Size	FPS-PCG†			FPS-PCG*		
	Iter	Time	E_{max}	Iter	Time	E_{max}
$4.84M$	16	29.37	$4e-7$	61	105.54	$7e-7$
$6.25M$	16	40.97	$4e-7$	61	147.60	$5e-7$
$7.84M$	16	55.32	$4e-7$	61	199.73	$9e-7$
$9.61M$	16	75.11	$4e-7$	65	287.60	$6e-7$
$11.56M$	16	95.36	$4e-7$	66	369.74	$8e-7$

† On Structured Power Grids
* On Unstructured Power Grids

6. CONCLUSION AND FUTURE WORK

In this work, we have proposed an efficient and robust fast Poisson solver preconditioned conjugate gradient method for large scale power grid analysis. By analytical formulation of multilayer power grids we have improved the numerical characters for multilayer power grids with dramatically conductance variations in different metal layers and vias. To properly tackle the unstructured grids with unideal boundary conditions a friendly stable analytical preconditioner is adopted to get the approximate voltage distribution and then the residual is smoothed very fast to a satisfactory level. The most benefit of this approach is that iterations are insensitive to grid size increasing. Moreover, due to the analytical formulation which is related to FFT, this approach can also be accelerated on GPU platforms.

7. REFERENCES

[1] Tsung-Hao Chen and Charlie Chung-Ping Chen. Efficient large-scale power grid analysis based on preconditioned Krylov-subspace iterative methods. In *Proc. of DAC*, pages 559–562, 2001.

[2] Haifeng Qian, Sani R. Nassif, and Sachin S. Sapatnekar. Power grid analysis using random walks. *ACM Trans. on CAD*, 24(8):1204–1224, 2005.

[3] Joseph N. Kozhaya, Sani R. Nassif, and Farid N. Najm. A multigrid-like technique for power grid analysis. In *Proc. of ICCAD*, pages 480–487, 2001.

[4] Kai Sun, Quming Zhou, Kartik Mohanram, and Danny C. Sorensen. Parallel domain decomposition for simulation of large-scale power grids. In *Proc. of ICCAD*, pages 54–59, 2007.

[5] Min Zhao, Rajendran V. Panda, Sachin S. Sapatnekar, Tim Edwards, Rajat Chaudhry, and David Blaauw. Hierarchical analysis of power distribution networks. In *Proc. of DAC*, pages 150–155, 2000.

[6] Stephen Cauley, Venkataramanan Balakrishnan, and Cheng-Kok Koh. A parallel direct solver for the simulation of large-scale powerground networks. *ACM Trans. on CAD*, 29(4):636–641, April 2010.

[7] J. M. S. Silva, Joel R. Phillips, and L. Miguel Silveira. Efficient simulation of power grids. *ACM Trans. on CAD*, 29(10):1523–1532, October 2010.

[8] Jin Shi, Yici Cai, Wenting Hou, Liwei Ma, Sheldon X.-D. Tan, and et al. GPU friendly fast Poisson solver for structured power grid network analysis. In *Proc. of DAC*, pages 178–183, 2009.

[9] Zhuo Feng and Peng Li. Multigrid on GPU: Tackling power grid analysis on parallel SIMT platforms. In *Proc. of ICCAD*, pages 647–654, 2007.

[10] Zhuo Feng and Zhiyu Zeng. Parallel multigrid preconditioning on Graphics Processing Units (GPUs) for robust power grid analysis. In *Proc. of DAC*, pages 661–666, 2010.

[11] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia, 1995.

[12] Tony F. Chan and Diana C. Resasco. A domain-decomposed fast Poisson solver on a rectangle. *SIAM J. Sci. Stat. Comput.*, 8(1):14–26, January 1987.

[13] Thomas A. Manteuffel and et al. Preconditioning and boundary conditions. *SIAM Journal on Numerical Analysis*, 27(3):656–694, June 1990.

[14] Paul N. Swarztrauber. The methods of cyclic reduction, fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle. *SIAM Review*, 19(3):490–501, July 1977.