ABSTRACT

As the increasing size of power grids, IR drop analysis has become more computationally challenging both in runtime and memory consumption. In this paper, we propose a linear complexity simulator named PowerRush, which consists of an efficient SPICE Parser, a robust circuit Builder and a linear solver. The proposed solver is a pure algebraic method which can provide an optimal convergence without geometric information. It is implemented by Algebraic Multigrid Preconditioned Conjugate Gradient method, in which an aggregation based algebraic multigrid with K-Cycle acceleration is adopted as a preconditioner to improve the robustness of conjugate gradient iterative method. In multigrid scheme, double pairwise aggregation technique is applied to the matrix graph in coarsening procedure to ensure low setup cost and memory requirement. Further, a K-Cycle multigrid scheme is adopted to provide Krylov subspace acceleration at each level to guarantee optimal or near optimal convergence. Experimental results on real power grids have shown that PowerRush has a linear complexity in runtime cost and memory consumption. The DC analysis of a 60 Million nodes power grid can be solved by PowerRush for 0.01mV accuracy in 170 seconds with 21.89GB memory used.

Categories and Subject Descriptors
B.7.2 [Integrated Circuits]: Design Aids—Simulation

Keywords
PowerRush, Power Grid, Algebraic Multigrid, Aggregation, K-Cycle

1. INTRODUCTION

The design and analysis of extremely large scale power grids is a very computationally challenging task for VLSI design. Many contributions have been developed, such as Krylov-subspace method [1], Hierarchical and Macro-modeling methods [2], Random Walk [3], Domain Decomposition [4], SPAI [5], H-Matrix [6]. Geometric based multigrid like technique was exploited to efficient power grid analysis [7]. And in order to further handle irregular power grids, algebraic multigrid (AMG) based techniques [8][9] were also developed.

For algebraic multigrid approaches, there exists a trade-off between setup cost and efficiency of error components reduction. The grid reduction methods in some AMG-like multigrid approaches [8] are also somewhat geometrically based which may degrade the efficiency due to the irregularity. A pure algebraic multigrid method [10] was proposed to improve the efficiency whose grid reduction was based on matrix graph. But its denser prolongation matrices increase setup costs and memory requirements. To overcome the bottleneck of limited convergence by controlling of the number of coarsening levels, smoothed aggregation method [11] is developed recently which is robust and efficient over a wide variety of problems. Aggregation based algebraic multigrid is also introduced to power grid analysis [12]. In [13], Fourier analysis has been explored for several aggregation based two-grid schemes for a model anisotropic problem. But in practice it is too difficult to hope for optimal order convergence with the V-Cycle or even with the standard W-Cycle.

Aiming to obtain an optimal convergence with low setup cost, in this work, we propose a pure algebraic multigrid solver for irregular power grid simulation. The contributions of our work are: (i) In our multigrid approach, double pairwise aggregation strategy [14] is introduced to reduce the grids effectively to small size by multilevel matrices representation which requires no explicit knowledge of the problem geometry. Further, the restriction and prolongation matrices are easy to construct with very low memory consumption. (ii) In order to increase the robustness of standard multigrid approaches, AMG is adopted as an implicit precondition for conjugate gradient iteration routine instead of stand-alone solver [15]. In this scheme, the main iteration routines smooth out high frequency errors rapidly and AMG uses a projection of the fine-grid problem on a coarser grid to remove the low frequency error components. (iii) Compared with previous aggregation based multigrid methods [12] in power grid analysis, the scalability is enhanced by K-Cycle multigrid scheme to provide Krylov subspace acceleration at each level. Related works have shown that K-Cycle multigrid can provide optimal or near optimal convergence under mild assumptions on the two-grid scheme [16].

The reminder of this paper is organized as follows. Section 2 gives a brief background of power grid analysis. In Section 3 we present a simulation flow of our simulator PowerRush. Following that, AMG-PCG solver and its practical implementation use is discussed in Section 4. Experimental results are provided and numerically illustrated in Section 5. Concluding remarks are given in Section 6.

*This work is supported by National Natural Science Foundation of China (NSFC) No.60976035.
†This paper is invited by Special Session of 2011 TAU Power Grid Simulation Contest in ICCAD 2011.
2. BACKGROUND

2.1 Power Grid Analysis

In this section, we give the basic modeling and analysis techniques for efficient and accurate analysis. Typically, power grid on die is designed from top-level metal layer, which is connected to the package, down through inter-layer vias and finally to the active devices. For DC simulation, power grid can be modeled as linear resistive network system. By using Modified Nodal Analysis (MNA) method, an n-node circuit network can be formulated as a linear system equations [1]. After reformulation by Newton’s Law this system equations can result in a symmetric, positive definite problem whose system matrix is a non-singular \( \mathcal{M} \)-matrix [8]:

\[
G u = I
\]

where \( u \in \mathbb{R}^{n \times 1} \) is an unknown vector of node voltages, \( G \in \mathbb{R}^{n \times n} \) is the conductance matrix, \( I \in \mathbb{R}^{n \times 1} \) is a vector of node current sources. The diagonal entries of matrix \( G \) are defined by \( g_{ii} = \sum_{j \in N_i} g_{ij} \), where \( N_i = \{ j \mid g_{ij} \neq 0 \} \) is the set of neighbors of node \( i \). \( g_{ij} \) defines the conductance between the two neighboring nodes \( i \) and \( j \), thus \( g_{ij} = g_{ji} \) which results in the \( G \) matrix being symmetric.

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 million(s). 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 stage 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.

2.2 Algebraic Multigrid Method

Multigrid methods [17] are called scalable and optimal which are expected to solve a linear system within linear complexity. Multigrid methods achieve optimality through the effect of a smoother and a coarse grid correction. In multigrid scheme, the smoother is fixed and generally based on a simple iterative relaxation method. The coarse grid correction involves transferring information to a coarse grid through restriction and computing an approximate solution to the residual equation on a coarser grid, which is to say, solving a linear system of smaller size. This solution is then transferred back to the original grid by means of an appropriate interpolation which is also called prolongation. In the classical multigrid setting, smoothing reduces high frequency error, whereas coarse grid correction eliminates low frequency error.

Due to the irregularity of real power grid designs [18], AMG methods have been well developed in power grid simulation area. AMG method [19] determines coarse grids, inter-grid transfer operators, and coarse-grid equations based solely on the matrix entries. Although the classical AMG methods work remarkably well for a wide variety of problems, some of the assumptions make it in derivation limit its applicability. The smoothed aggregation method [11] is a highly successful AMG method that is robust and efficient over a wide variety of problems. The most interesting aspect of aggregation based AMG is its approach to define interpolation. The aggregation algorithm first partitions the grid by aggregation grid points into small disjoint sets and then builds a preliminary interpolation operator to coarsen power grids by system matrix level. Thus, the prolongation matrices with at most one nonzero entry per row are much sparser than the ones obtained by the classical AMG approach. Numeric analysis has shown that for two-dimensional anisotropic model problem aggregation based two-grid methods [13] may have optimal order convergence properties.

3. SIMULATION FLOW

This work presents a friendly fast simulator with linear complexity for power grid network. The simulator consists of a smart SPICE Parser, a robust circuit Builder and a linear complexity Solver. The detailed simulation flow of our simulator PowerRush is shown in Figure 1.

![Figure 1: Simulation Flow of PowerRush.](image)

In parser procedure, it hashes each node name to index. The hash function we used in our approach ensures the average search length of each node is about 1.16 and the maximum search length is no more than 6. As we have known that the complexity of hash method to find an element in a table is expected to \( \mathcal{O}(1) \). So, the total complexity of our parser is expected to \( \mathcal{O}(N) \) which \( N \) is the size of grid nodes. After parser process for the SPICE file, the circuit is stored as nodes list and wires map which are linked to present their topologies.

The most important part of building circuit is to create a graph to handle all wires map and nodes list. The most frequently appearance in this procedure is short path between two nodes which must be merged as one equivalent node. The number of unknowns can be reduced by nodes merging for short path, such as vias with very small resistance value or zero resistance value. The disjoint data structure is used to find and union nodes sets whose complexity is \( \mathcal{O}(\log N) \). Further, it is noticeable that there exist several separated nets for each benchmark in real power grid designs [18]. So our builder identifies all separated nets by DFS (Depth-First Search) algorithm whose complexity is \( \mathcal{O}(N) \) and then solve each net independently. Thus, the total solving cost can be added linearly by each net.

After building the graph of power grid, we build the conductance matrix \( G \) by MNA method and the corresponding vector of current sources. Then the linear system is solved by AMG-PCG method which stands for Algebraic MultiGrid Preconditioned Conjugate Gradient method. Among AMG-PCG solver, aggregation based algebraic multi-
K-Cycle Multigrid Preconditioner

Setup

Aggregation

Coarsening

K-Cycle Multigrid Preconditioner

CG Iteration Method

Figure 2: AMG-PCG Solver.

As shown in Figure 2, the application of AMG-PCG to a given problem is a three part process. The first part, which is a fully automatic setup phase, consists of recursively choosing the coarser levels and defining the transfer and coarse-grid operators. The second part, which is the preconditioning phase, just uses the resulting components in order to perform acceleration recursively on all levels by the use of K-Cycle scheme. Thirdly, aggregation based algebraic multigrid with K-Cycle accelerating is adopted as an implicit preconditioner for CG iteration method to solve the power grid. The details of this linear solver are demonstrated in Section 4.

4. AMG-PCG SOLVER

In this work, we use aggregation based algebraic multigrid method with K-Cycle acceleration to improve the convergence. This procedure is fully algebraic, that is, it works with the information present in the system matrix only.

4.1 Aggregation Coarsening

As demonstrated in Figure 2, the aggregation scheme contains two steps. First, we need to partition the unknowns into disjoint subsets to generate the prolongation matrices. Then, the prolongation matrices are used to formulate the coarse matrix.

As described in [14], we focus on schemes that use coarsening by aggregation where the strongest connection is favored in forming pairs. With aggregation scheme, the fine grid unknowns are grouped into disjoint subsets, and each such subset is associated to a unique coarse level unknown. Prolongation from coarse level to fine level is a vector defined such subset is associated to a unique coarse level unknown. Grid unknowns are grouped into disjoint subsets, and each subset is associated to a unique coarse level unknown. Prolongation from coarse level to fine level is a vector defined such subset is associated to a unique coarse level unknown.

Algorithm 1: AMG as preconditioner at level $k$, $AMGprecond(r_k,k)$

<table>
<thead>
<tr>
<th>Input:</th>
<th>The residual $r_k$ of level $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output:</td>
<td>The preconditioned vector $z_k$</td>
</tr>
</tbody>
</table>

1. Pre-smoothing: relax the input residual $r_k$ to $v_k$ by using smoother of several iterations on $A_kv_k = r_k$;
2. Compute a new residual $\tilde{r}_k = r_k - A_kv_k$ and restrict it to coarser grid by $r_{k-1} = P_k^T \tilde{r}_k$;
3. Compute an (approximate) solution $y_{k-1}$ on coarser grid by $A_{k-1}y_{k-1} = r_{k-1}$ which is recursively obtained by $y_{k-1} = AMGprecond(r_{k-1},k-1)$ until the coarsest level with $k = 1$;
4. Interpolate coarse grid correction by $y_k = P_ky_{k-1}$ and compute new residual $\tilde{r}_k = r_k - A_ky_k$;
5. Post-smoothing: relax the new residual $\tilde{r}_k$ to $w_k$ by using smoother of several iterations on $A_kw_k = \tilde{r}_k$;
6. Obtain preconditioned vector $z_k = v_k + y_k + w_k$;

The multigrid preconditioning is called by the main iteration routine at the top level, and it recursively calls itself in step 3 with a smaller index until the coarsest level. We stop the coarsening when the coarse grid matrix has 400 rows, allowing fast direct solver with sparse Cholesky factorization. For $k > 1$, the AMG preconditioner at level $k$ computes $y_{k-1}$ approximately by using AMG preconditioner at level $k - 1$. The way this is done defines the cycling strategy which will be detailed discussed in next subsection.

4.2 Multilevel Preconditioning

Although AMG strategy has been tried to capture all relevant influences by accurate coarsening and interpolation, its interpolation will hardly ever be optimal. Multilevel error reduction technique can smooth the majority of the error components very quickly but inefficiency for just a few exceptional error components. Thus, its total convergence property is degraded by this non-ideal phenomenon. In order to improve the robustness of our approaches, AMG is used as an implicit preconditioner for conjugate gradient method instead of stand-alone solver [15].

The AMG preconditioner at each level on a given vector is computed according to the following algorithm [20]:

4.3 K-Cycle Multigrid

This paper adopts a K-Cycle multigrid [16], with which Krylov subspace acceleration is applied at every level to enhance robustness and scalability.

As shown in Figure 3, it illustrates Krylov subspace ac-
ceration recursively on each level. The Krylov-subspace smoothing is performed at the end of every recursive cycle on each level to reduce the residual, in which it is called inner iteration.

The runtime in simulation flow is measured by system clock counter. The peak memory usage is measured by a friendly tool `Memtime` whose memory usage is fetched from a PID information of `/proc/[PID]/stat`. In our simulator, the tolerance of linear solver is set as $10^{-6}$ on the relative residual norm to guarantee that all voltage solution error is less than $0.01mV$.

### 5.1 Simulation on Industrial Power Grid

Firstly we carried out various experiments on industrial power grids which are drawn from real designs [18] to validate the promising performance of the proposed simulator PowerRush. As shown in Table 1, the second column is the original grid size of each benchmark, and the second column is equivalent grid size. The max voltage errors of most benchmarks are less than $0.1mV$ and the average voltage errors of most benchmarks are less than $0.02mV$. The runtime and memory usage are relative small and increase slowly with the grid size increasing. Experimental results have shown that PowerRush is effective and robust with high accuracy for real power grids analysis.

### 5.2 Simulation on More Larger Power Grid

Aiming to present the promising linear complexity of PowerRush, a serious of larger scale power grid benchmarks are generated by our power grid planner, in which the grid size is from 5 Million to 60 Million as shown in Table 2. Among them, there exists only one single net in each benchmark, which means that there are no separated nets in a same benchmark. That is to say, it cannot take the advantage of divide and conquer strategy in circuit building process as described in Section 3.

In Table 2, it illustrates the behavior of our proposed simulator PowerRush on our generated power grids. The complexity character of each step in PowerRush is shown in Figure 4. The parsing time, building time, solving time and peak memory usage are all linear complexity with the increasing size of power grids. We also compare the performance of PowerRush and Cholmod solver [21] which is included in SuiteSparse packages. The last two columns list the solving time and memory usage of Cholmod which not include the parser and builder cost. As shown in Table 2, AMG-PCG solver has obtained many speedups and very low memory consumptions. For the largest two benchmark-s, Cholmod can not handle such huge scale grids because of much more memory required.

#### 5.3 Grid Reduction Efficiency

Also in Table 4, it presents the grid reduction efficiency of double pairwise aggregation strategy. For all benchmarks,
Accelerating is adopted as an implicit preconditioner for CG Rush explore a widely practical use in industrial world. PowerRush of other simulators. Excitedly, respectively compared with the second place performance PowerRush extremely low memory consumption. As shown in Table 3, other teams, are also illustrated in [24][25]. Comparing with the TicTac runtime and memory chart of top three teams is shown in Figure 5. Among them, the other two simulators SEVA and PowerRush has shown to be an effective and robust simulator, PowerRush on TAU Contest

The first annual Power Grid Simulation Contest has been successfully organized in TAU workshop by IBM Austin Research Lab to boost the academic research in power grid verification [22]. PowerRush is also evaluated with other nine simulators on the same HW/SW platforms by a set of real life industrial benchmarks in this contest [23]. The runtime and memory chart of top three teams is shown in Figure 5. Among them, the other two simulators SEVA and TicTac are also illustrated in [24][25]. Comparing with the other teams, PowerRush obtained a best convergence and memory usage. The K-Cycle scheme is a recursively accelerating W-Cycle where acceleration is performed by finding the optimal linear combination of two iterations. Thus a linear complexity solver is realized fully with algebraic information in the system matrix only. Also by taking the advantage of effective SPICE Parser and robust circuit Builder, PowerRush has shown to be an effective and robust simulator with a linear complexity for real power grid analysis.

5.4 PowerRush on TAU Contest

The first annual Power Grid Simulation Contest has been successfully organized in TAU workshop by IBM Austin Research Lab to boost the academic research in power grid verification [22]. PowerRush is also evaluated with other nine simulators on the same HW/SW platforms by a set of real life industrial benchmarks in this contest [23]. The runtime and memory chart of top three teams is shown in Figure 5. Among them, the other two simulators SEVA and TicTac are also illustrated in [24][25]. Comparing with the other teams, PowerRush obtained a best convergence and extremely low memory consumption. As shown in Table 3, PowerRush uses only about 40% runtime and 75% memory respectively compared with the second place performance of other simulators. Excitedly, PowerRush is expected to explore a widely practical use in industrial world.

6. CONCLUSIONS

We have presented the detail implementation of PowerRush. Aggregation based algebraic multigrid with K-Cycle accelerating is adopted as an implicit preconditioner for CG iteration method to solve the power grid. The double pairwise aggregation scheme uses two passes of a pairwise matching algorithm to ensure low setup cost both in runtime and memory usage. The K-Cycle scheme is a recursively accelerated W-Cycle where acceleration is performed by finding the optimal linear combination of two iterations. Thus a linear complexity solver is realized fully with algebraic information in the system matrix only. Also by taking the advantage of effective SPICE Parser and robust circuit Builder, PowerRush has shown to be an effective and robust simulator with a linear complexity for real power grid analysis.

7. ACKNOWLEDGEMENTS

The authors would like to thank Zhuo Li for his careful organizing effort on the contest, and to thank Sani R. Nassif for introducing this contest for us.

8. REFERENCES


Table 4: Grid reduction efficiency of Double Pairwise Aggregation scheme

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Initial Size</th>
<th>Reduction Level</th>
<th>Reduction Rate*</th>
</tr>
</thead>
<tbody>
<tr>
<td>ibmpg3/GND</td>
<td>151148</td>
<td>1 2 3 4 5 6 7 8 9</td>
<td>0.2650</td>
</tr>
<tr>
<td>ibmpg4/GND</td>
<td>303712</td>
<td></td>
<td>0.2711</td>
</tr>
<tr>
<td>ibmpg5/GND</td>
<td>291559</td>
<td></td>
<td>0.2691</td>
</tr>
<tr>
<td>ibmpg6/GND</td>
<td>430586</td>
<td></td>
<td>0.2720</td>
</tr>
<tr>
<td>ibmpgnew/GND</td>
<td>272655</td>
<td></td>
<td>0.2711</td>
</tr>
<tr>
<td>thupg1</td>
<td>3261850</td>
<td></td>
<td>0.2715</td>
</tr>
<tr>
<td>thupg2</td>
<td>6014504</td>
<td></td>
<td>0.2694</td>
</tr>
<tr>
<td>thupg3</td>
<td>7856200</td>
<td></td>
<td>0.2676</td>
</tr>
<tr>
<td>thupg4</td>
<td>10259771</td>
<td></td>
<td>0.2674</td>
</tr>
<tr>
<td>thupg5</td>
<td>13627506</td>
<td></td>
<td>0.2645</td>
</tr>
<tr>
<td>thupg6</td>
<td>17216192</td>
<td></td>
<td>0.2674</td>
</tr>
<tr>
<td>thupg7</td>
<td>18090184</td>
<td></td>
<td>0.2675</td>
</tr>
<tr>
<td>thupg8</td>
<td>27661623</td>
<td></td>
<td>0.2674</td>
</tr>
<tr>
<td>thupg9</td>
<td>31518068</td>
<td></td>
<td>0.2674</td>
</tr>
<tr>
<td>thupg10</td>
<td>38415188</td>
<td></td>
<td>0.2677</td>
</tr>
</tbody>
</table>

1 GND means that only the single net of ground network is used.

Average reduction rate of all reduction levels


