RC22722 (W0302-050) February 12, 2003
Electrical Engineering

IBM Research Report

Power Grid Analysis using Random Walks

Haifeng Qian
University of Minnesota
200 Union St SE
Minneapolis, MN 55455

Sani R. Nassif
IBM Austin Research Lab
11501 Burnet Rd.
Austin, TX 78758

Sachin S. Sapatnekar
University of Minnesota
200 Union St SE
Minneapolis, MN 55455

Research Division
Almaden - Austin - Beijing - Delhi - Haifa - India - T. J. Watson - Tokyo - Zurich

Power Grid Analysis using Random Walks

Haifeng Qian
University of Minnesota
qianhf@ece.umn.edu

Sani R. Nassif
IBM Austin Research Lab
nassif@us.ibm.com

Sachin S. Sapatnekar
University of Minnesota
sachin@ece.umn.edu

IBM Technical Contact: Charles J. Alpert, IBM Austin Research Laboratory

Abstract

This paper presents a linear-time algorithm for the DC analysis of a power grid, based on a random walk technique. Experimental results show that the proposed method is faster than existing approaches and has an acceptable error margin. It also has a desirable property of localizing computation, and can be extended to RC-network transient analysis. This method has been applied to circuits of up to 70K nodes, for which the solution time for a single node was 0.42 sec and the complete solution was obtained in 17.6 sec.

1. Introduction

A reliable power grid is an indispensable part of a VLSI design. A drop in the supply voltage causes increases in gate delays, and, when it exceeds a certain limit, causes logic failure. As technology proceeds from one generation to the next, interconnect resistance increases due to the reduced wire width, and the amount of current flowing through a chip increases. These effects lead to potentially larger IR voltage drops. At the same time, the VDD voltage decreases from one technology node to the next and demands a narrower noise margin. Therefore, power grid analysis and optimization is becoming a critical issue.

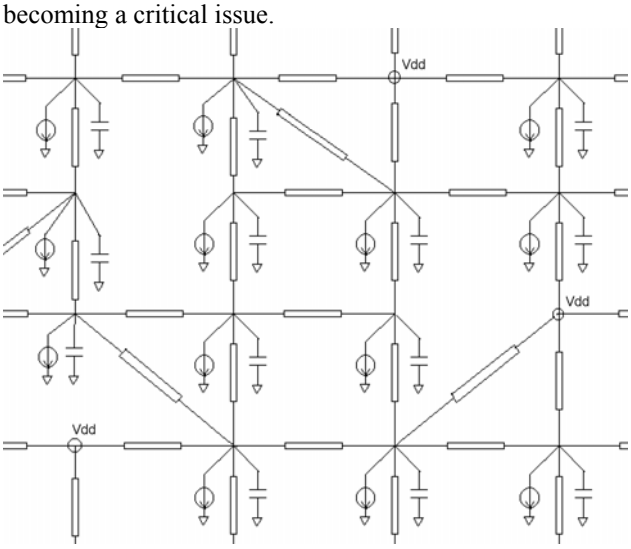

Figure 1. Circuit model for a power grid.

An important subproblem in power grid analysis is related to the DC analysis, where a power grid is modeled as a resistive network, as shown in Figure 1. Currents drawn by logic gates are represented by current sources connected to the nodes in the bottom-most layer of the grid, and for modeling purposes, perfect voltage sources are distributed at nodes in the top-most layer. The problem of finding the voltage value at each node is formulated as:

$$G\mathbf{X}=\mathbf{E} \quad (1)$$

where G is a conductance matrix, $\mathbf{X}$ is the vector of node voltages, and $\mathbf{E}$ is a vector of independent sources. By exploiting the sparse and positive definite nature of G, $\mathbf{X}$ can be solved efficiently. However, it is still very expensive to solve a power grid with tens of millions of nodes, and with the circuit-size growing, it will eventually become prohibitive.

Different methods have been proposed to address this issue. For example, [9] utilizes the hierarchical structure of a power grid, divides it into a global grid and multiple local grids, and solves them separately. The approach in [4] proposes a grid-reduction scheme to coarsen the circuit recursively, solves a coarsened circuit, and then maps back to find the solution to the original circuit. The existing methods sacrifice a certain degree of accuracy for a lower time and space computational complexity.

In this paper, we apply a statistical approach based on the relationship between random walks and electrical networks to solve this problem, and use test results to show that it reaches a good accuracy-runtime tradeoff, compared with other methods. This method is particularly useful and efficient when only a small fraction of the nodes in the grid are to be analyzed.

This paper is organized as follows. Section 2 presents the theoretical basis of the proposed algorithm, Section 3 uses a simple example to illustrate how to apply this theory in circuit analysis, Section 4 gives experimental results and compares with other methods, Section 5 extends the proposed algorithm to RC-network transient analysis, and Section 6 provides a conclusion.

2. Random walk principles

The random walk is a classical problem in statistics, and has been found useful in engineering. A prominent

example of the use of random walks in CAD is [6], which applies this idea to capacitance extraction. Other applications can be found in [2][5][8]. An earlier approach in [1], which inspires the work in this paper, interprets the relationship between resistive networks and probabilities. Specifically, the solution to any network with resistors and voltage sources can be interpreted as being equivalent to an equivalent probabilistic problem. In our work, we apply and extend this method to handle the problem of DC analysis of a power grid. For purposes of illustration, we will focus our discussion on the description of a V_{DD} grid, pointing out the difference for a ground grid where applicable.

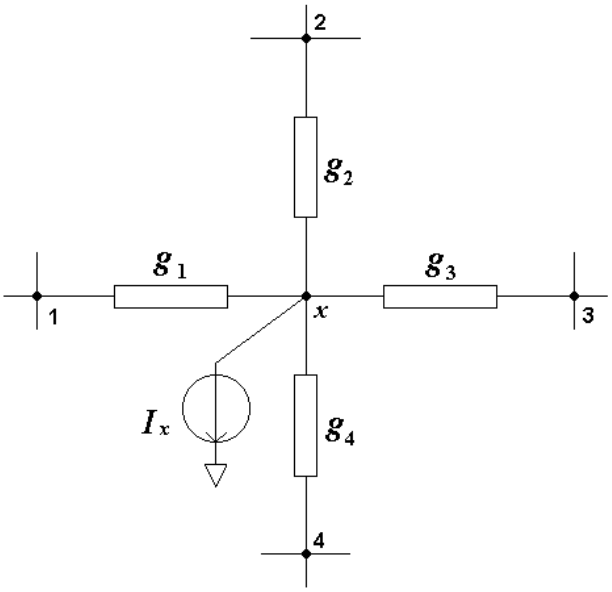

Figure 2. A representative node in a power grid.

To apply the method to the DC analysis of a power grid such as that shown in Figure 1, let us look at a single node in the circuit, node x, which is shown in Figure 2. For convenience, we will represent the resistors in terms of conductances. Applying Kirchoff's Current Law, Kirchoff's Voltage Law and the device equations for the conductances, we can write down the following equation

$$\sum_{i=1}^{\text{degree}(x)} g_i \left(V_i - V_x\right) = I_x \tag{2}$$

where adjacent nodes of x are labeled 1, 2, … degree(x), V_x is the voltage at node x, V_i is the voltage at node i, g_i is the conductance between node i and node x, and I_x is the current load connected to node x. Equation (2) can be reformulated into the following form:

$$V_x = \sum_{i=1}^{\text{degree}(x)} \frac{g_i}{\sum_{i=1}^{\text{degree}(x)} g_i} V_i - \frac{I_x}{\sum_{i=1}^{\text{degree}(x)} g_i} \tag{3}$$

For a power grid problem with N non-V_{DD} nodes, we have N linear equations similar to the one above, one for each node. Solving this set of equations will give us the exact solution.

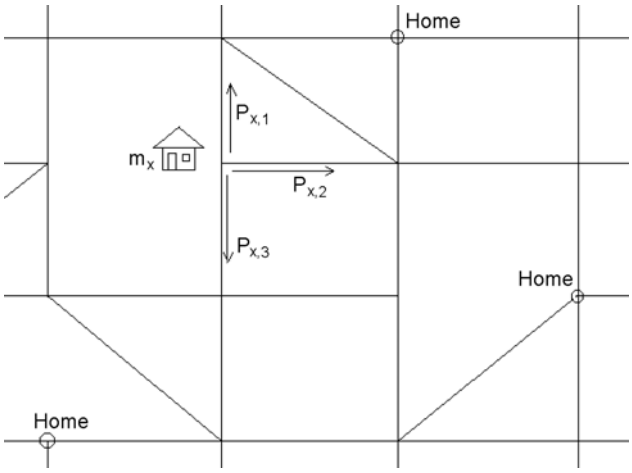

Figure 3. The random walk model.

Now let us look at a random walk "game." Given a finite undirected connected graph (for example, Figure 3) representing a street map. A walker starts from one of the nodes, and goes to an adjacent node k every day with probability $p_{x,k}$ for k= 1,2,…, degree(x), where x is the current node, and degree(x) is the number of edges connected to node x. These probabilities satisfy the following relationships:

$$p_{x,1} + p_{x,2} + \cdots + p_{x,\text{degree}(x)} = 1 \tag{4}$$

The walker pays an amount m_x to a motel for lodging everyday, until he/she reaches one of the homes, which are a subset of the nodes. If the walker reaches home, he/she will stay there and be awarded a certain amount of money, m_0. We will consider the problem of calculating the expected amount of money that the walker has at the end of the walk, as a function of the starting node, assuming he/she starts with nothing.

The gain function for the walk is therefore defined as

$$f(x) = E[\textit{total money earned} \mid \textit{walk starts at node } x] \tag{5}$$

It is obvious that

$$f(\textit{one of the homes}) = m_0 \tag{6}$$

For a non-home node x, assuming that the adjacent nodes of x are labeled 1, 2, … degree(x), we can write down the following equation

$$f(x) = p_{x,1} f(1) + p_{x,2} f(2) + \cdots + p_{x,\text{degree}(x)} f(\text{degree}(x)) - m_x \tag{7}$$

For a random-walk problem with N non-home nodes, we have N linear equations similar to the one above, and the solution to this set of equation will give us the exact values of f at all nodes.

It is easy to draw a parallel between this problem and that of analyzing supply nets. Equation (7) becomes identical to Equation (3), and Equation (6) reduces to the condition of perfect V_{DD} nodes if we let

$$p_{x,i} = \frac{g_i}{\sum_{i=1}^{\text{degree}(x)} g_i} \qquad i = 1, 2, \cdots, \text{degree}(x)$$

$$m_x = \frac{I_x}{\sum_{i=1}^{\text{degree}(x)} g_i} \tag{8}$$

$$m_0 = V_{DD}$$

$$f(x) = V_x$$

The analysis technique for GND nets is analogous; the major differences are that (i) the I_x's have negative values, (ii) V_{DD} is replaced by zero. As a result, the walker earns money in each step, but gets no award at home.

In other words, for any power grid problem, we can construct a random walk problem that is mathematically equivalent, i.e., characterized be the same set of equations. And, it can be proven, easily by contradiction, that such an equation set has and only has one unique solution [1]. It is both the solution to the random walk problem, and the solution to the power grid problem. Therefore, if we find an approximated solution for the random walk, it is also an approximated solution for the power grid. A natural way to approach the random walk problem is to perform a certain number of experiments and use the average money left in those experiments as the approximated solution. If this amount is averaged over a sufficiently large number of walks by playing the game a sufficiently large number of times, by the law of large numbers [7], an acceptably accurate solution can be obtained. This is the idea behind our proposed algorithm.

According to the Central Limit Theorem [7], the error is a 0-mean Gaussian variable with variance inversely proportional to M, where M is the number of experiments. Thus we have an accuracy-runtime tradeoff. Instead of fixing M, we employ a user-specified stopping criterion:

$$P[-\Delta < V_e - V < \Delta] > 99\% \tag{9}$$

where V_e is the estimated voltage by M experiments, and Δ is a user-specified error margin. The above criterion can be written as

$$Q\left(\frac{\Delta}{\sqrt{Var / M}}\right) < 0.005$$

$$\frac{Var}{M} < \left(\frac{\Delta}{Q^{-1}(0.005)}\right)^2 \tag{10}$$

where Var is the variance of the results of the M experiments. In a normal power grid, each node is in a similar environment, with similar-value-range devices around it and similar distances to perfect-voltage nodes. Therefore, different nodes have similar Var, and M is roughly a constant.

In the implementation, we will impose a limit, L, on the number of steps in a walk; details are provided in Section 4. Thus, for a power grid with N non-V_{DD} nodes, we can estimate worst-case time complexity as $O(LMN)$, where each unit corresponds to one random-number generation, a few logic operations and one addition. Practically, since both L and M are upper-bounded by constants, we have worst-case time complexity that is linear in the number of nodes. For average case, since each node is in a similar environment, the M experiments take similar CPU times for processing each node. Therefore, the average-case runtime is also linear in the number of nodes.

A desirable feature of the proposed algorithm is that it localizes the computation, i.e., it can calculate a single node voltage without having to solve the whole circuit. This is especially meaningful when the designer knows which part of his power grid is problematic, or when the designer makes a minor change in the design and want to see the impact.

For example, if the objective of the analysis is to find the voltage at a single node, then this approach is very useful since it can perform a number of random walks starting from that node. In a typical supply net that has a sufficiently large number of pads that are reasonably close to any node, such a walk is likely to reach home soon. As compared to a conventional approach that must solve the full set of matrix equations to find the voltage at any one node, the computational advantage of this method could be tremendous, and we validate this in Section 4. We will also study the applicability of this method, as against direct solution, for finding the voltages at a larger number of nodes in Section 4, and study its extension to transient analysis in Section 5.

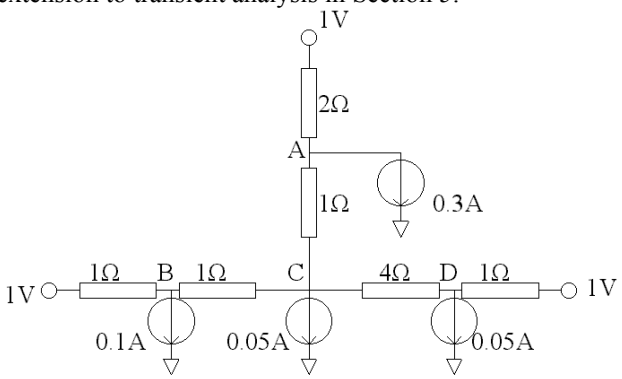

Figure 4. A simple example circuit.

3. A simple example

In order to show how the proposed algorithm works, let us look at a simple circuit, as shown in Figure 4. The true voltage values at node A, B, C and D are 0.6, 0.8, 0.7 and 0.9, respectively.

Applying Equation (8) to this circuit, we construct an equivalent random walk game, as shown in Figure 5, where numbers inside circles represent motel prices and

home awards, and numbers beside the arrows represent the transition probabilities from each node to a neighboring node.

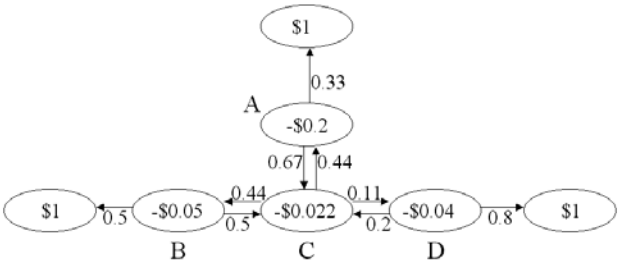

Figure 5. The random walk game corresponding to the circuit in Figure 4.

Let's say we want to find out the voltage of node A, we start the walker at node A. He/she pays the motel price of $0.2, then either goes up with probability 0.33 to the terminal and end this walk, or goes down with probability 0.67 to node C, then pays 0.022, and continues from there. Such a walk could be very short: for example, the walker may directly goes up and ends up with $0.8. Alternatively, the walk could be very long, if it keeps going back and forth between A, B, C and D, so that the walker ends up with very little money; however, the probability of such a walk can easily be verified to be low. We perform M such experiments and take the average of the M results as the estimated money earned during the walk, and change the units from dollars to volts to obtain the estimated voltage of node A.

Table 1 shows how the estimated voltage of node A converges to the true value of 0.6V, as the number of experiments increases. Columns in the table represent 5 different runs of the proposed algorithm.

Table 1. Convergence of the simple example.

M	Exp #1	Exp #2	Exp #3	Exp #4	Exp #5
100	0.6108	0.6316	0.6456	0.6250	0.6001
1000	0.5955	0.6090	0.5898	0.5861	0.5887
5000	0.6033	0.5998	0.6043	0.6049	0.5978

4. Experimental results.

We now apply the proposed algorithm to a real-life power grid model. Our benchmark is a 70729-node industrial circuit, and we solve for the 15876 bottom-layer V_{DD} nodes and 15625 bottom-layer GND nodes, as they are the voltages of interest. The V_{DD} value is 1.2V. Because we need HSPICE to provide the correct answer to evaluate the accuracy of the method, the circuit size is limited by the maximum size that HSPICE can handle in a reasonable amount of time and within the memory constraints of the machines available to us. However, we can be assured that the proposed algorithm will have the same accuracy for a larger circuit, and the runtime will be proportional to the number of nodes, as it is a linear-time algorithm. Our computations are carried out on a Linux workstation with 2.8GHz CPU frequency.

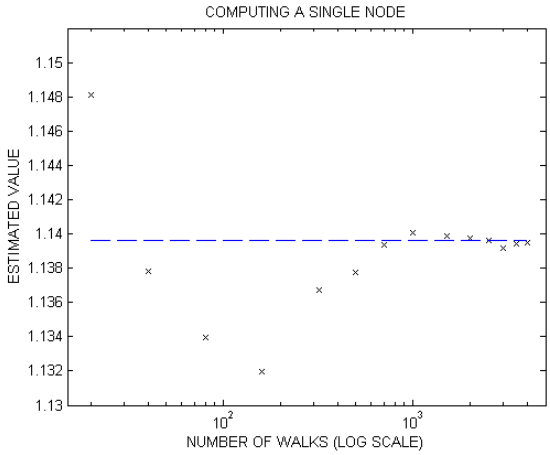

Figure 6. Voltage waveform for a single node.

Figure 6 shows the result of computing the solution for only one node, where the markers are estimated values of the voltage for different M, and the dashed line is the true voltage. The ultra-accurate right-most point, for which M=4000, only takes 0.42 second runtime, and thus shows the efficiency of using our algorithm to solve individual nodes without solving the whole circuit.

When solving for the voltage multiple nodes, several efficiency-enhancing techniques can be used. Since the voltage at each already calculated node is known, it becomes a new home in the game with an award amount equal to its calculated voltage. This operation speeds up the algorithm dramatically, as there are more terminals to end a walk, and therefore the average number of steps in each walk is reduced. At the same time, this operation improves accuracy without increasing M, because each experiment that ends at such a node is equivalent to multiple experiments.

As indicated in Section 2, another implementation issue is that, in order to avoid any possible deadlock, we need to set a limit, L, on the number of steps in a walk. Any walk that fails to end within L steps will be forced to end, and be awarded V_{DD} if inside the V_{DD} net, be awarded 0 if inside the GND net. This operation is optimistic and will results in a bias in the estimated voltage; however, if the limit is chosen appropriately, the error will be very small as the probability of a walk of this length is minute. Thus a new degree of accuracy-runtime tradeoff is introduced, and we empirically set this limit to be 10000 steps as a good tradeoff point, where the bias error is acceptable and not much runtime is wasted.

The above tradeoff only affects runtime indirectly, while the error margin Δ in Equation (9) decides M, which is directly proportional to runtime and need careful investigation. Figure 7 plots the relation between Δ and runtime for the industrial circuit. The runtime is always

larger than 8 seconds because the minimum value of M is set to be 40.

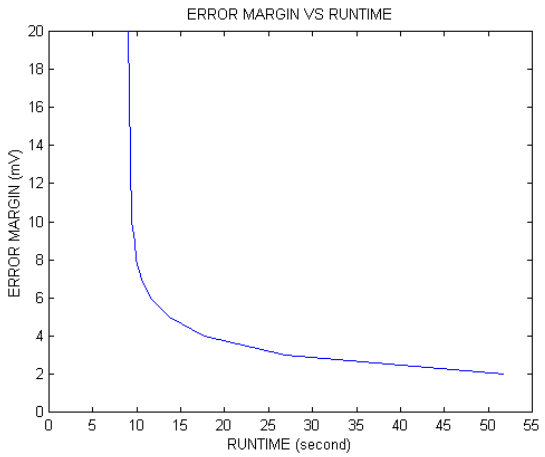

Figure 7. Runtime-accuracy tradeoff.

In practice, the user decides the tradeoff point by choosing Δ according to the needs of the analysis. Here we choose Δ=4mV as a good tradeoff point. By definition, 99% nodes have an estimation error less than 4mV. In fact, among the 15876 bottom-layer nodes in the V_{DD} net, the average error is 1.5mV, and the maximum error is 7.4mV. Considering the true voltage range 1.1324—1.1917, this accuracy is sufficient. The corresponding runtime is 17.60 seconds.

To compare the runtime of our approach with other algorithms, we use the runtimes reported in [9] as the baseline. [9] reports both serial runtime and parallel runtime. Since the random-walk algorithm is inherently compatible with parallel computing, and it is likely that it could beat any other method in that case, we only compare serial mode runtime.

Table 2. Runtime comparison with [9].

Method		Runtime per thousand nodes (sec)
Random walk		0.25
Method in [9]	Chip-1	0.66
	Chip-2	0.55
	Chip-3	1.09
	Chip-4	1.33
	Chip-5	1.44
	Chip-6	1.70

The runtime comparison for a *complete* analysis of all nodes in the supply grid is shown in Table 2. In viewing these numbers, it is important to note that our computer is approximately 3 times faster than the computer used by [9], according to SPEC benchmarks [10]. The six circuits in [9] have much larger sizes than our benchmark, Chip-2 is the smallest, and Chip-6 is the largest. The runtime per thousand nodes increases with circuit size for [9] due to its superlinear time complexity. Since our proposed algorithm has linear time complexity, as power grid size increases, it will outperform [9] even further. Additionally, as mentioned earlier, if the objective is to analyze a small subset of the grid, the random walk approach has major speed advantages.

5. Transient analysis: extension

The proposed method can be extended to transient analysis of RC supply grids. In this case the equations to be solved may be written as follows [3]:

$$G\,\mathbf{y}(t) + C\,\mathbf{y}'(t) = \mathbf{b}(t) \qquad (11)$$

where G is a conductance matrix, C is the matrix introduced by capacitors, $\mathbf{y}(t)$ is the vector of node voltages, and $\mathbf{b}(t)$ is the vector of independent sources. Applying the backward Euler formula with a time step of h, the equations become

$$(G + C/h)\ \mathbf{y}(t) = \mathbf{b}(t) + C/h\ \mathbf{y}(t\text{-}h) \qquad (12)$$

This transformation translates the problem to that of solving a circuit with resistors and capacitors, as before, and considering node x at one time step at time t, we can write down the following equation

$$\sum_{i=1}^{\text{degree}(x)} g_i\left(V_i(t) - V_x(t)\right) = \frac{C_x}{h}\left(V_x(t) - V_x(t-h)\right) + I_x(t) \qquad (13)$$

where V_x is the voltage at node x, $V_i(t)$ is the voltage at node i = 1, 2, … degree(x), g_i is the conductance between node i and node x, C_x is the capacitance between node x and ground, and $I_x(t)$ is the current source connected to node x.

For RC-network with capacitors between nodes, those capacitors can be replaced by resistors and voltage-controlled current sources, while a current source between two nodes can be replaced by two current sources between the two nodes and ground. The following algorithm is still applicable. Here we only discuss the case described in Equation (13).

Equation (13) can be converted to the following form

$$V_x(t) = \sum_{i=1}^{\text{degree}(x)} \frac{g_i}{\sum_{i=1}^{\text{degree}(x)} g_i + \frac{C_x}{h}} V_i(t) + \frac{\frac{C_x}{h}}{\sum_{i=1}^{\text{degree}(x)} g_i + \frac{C_x}{h}} V_x(t-h) - \frac{I_x(t)}{\sum_{i=1}^{\text{degree}(x)} g_i + \frac{C_x}{h}} \qquad (14)$$

The rules of the random walk game are changed to accommodate the changes in the above equation. As shown in Figure 8, each node x has an additional connection, and the walker could end the walk and be awarded the amount V_x(t-h) with probability

$$\frac{C_x / h}{\sum_{i=1}^{\deg ree(x)} g_i + C_x / h}$$

Intuitively, this rule is equivalent to replacing each capacitor by a resistor and a voltage source.

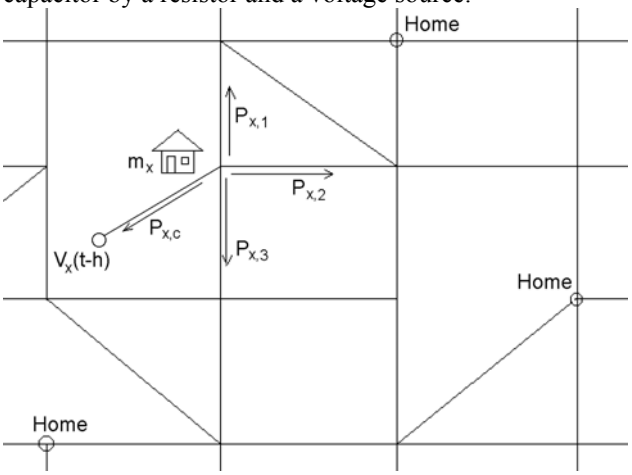

Figure 8. Altered random walk model for transient analysis.

In transient analysis, traditional methods, i.e., those that solve Equation (12) directly, are efficient in computing solutions for succeeding time steps after initial matrix factorization since only a forward/backward substitution step is required for each additional time step. Analogously, our random walk algorithm employs a speed-up mechanism. We first perform a DC analysis that is used as the initial condition. Next, when computing the first transient timestep, we keep a record for each node. This record keeps a count of, in these *M* walks, how many times the walker ends at V_{DD}, how many times the walker ends at some V(t-h), how many times the walker pays for a motel at some node, and so on. Then, in the follow-up timesteps, we do not need to walk any more, simply use these records recursively and assume that the walker gets awards at same locations, pays for same motels, and only the award amounts and motel prices have changed. Thus new voltages can be computed by some multiplications and additions efficiently. The space complexity demanded by this bookkeeping is approximately linear in the number of nodes, and is not worse than the space complexity of a traditional direct solver.

Table 3. Transient analysis results.

	Circuit 1	Circuit 2	Circuit 3
Circuit size	2500	400	10000
Number of time steps	30	300	1000
Voltage range	0.9297V to 1.1950V	1.0886V to 1.1731V	*(cannot be simulated by HSPICE)*
CPU time per timestep for sub-sequent timesteps	0.7msec	0.06msec	1.6msec
Mean error	0.0010	0.0014	*(unavailable)*
Maximum error	0.0086	0.0059	*(unavailable)*

In order to evaluate the transient analysis, since we were unable to obtain real-life RC power grid circuits, we generated 3 circuits with random resistances and capacitances. Perfect voltage sources are connected to 1% of their nodes at random locations, and current sources have a random-size peak at a random time range. The results of our approach are shown in Table 3. The CPU times shown correspond to the runtimes for the time steps that follow the initial DC analysis and the first transient step. The solutions for circuits 1 and 2 are compared with HSPICE, while circuit 3 is too large to be simulated in HSPICE for an accuracy evaluation. The runtimes are several times faster than traditional direct solver runtimes reported in [9], even after normalization by the speed factor of 3.

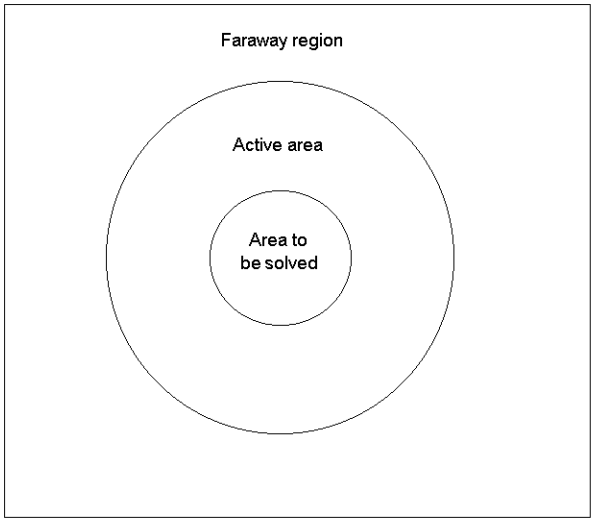

Figure 9. Localizing transient analysis.

The efficiency of transient analysis can be improved by taking advantage of the property of localization for random walks. Because the value of V(t-h) must be updated for every node for each timestep, we cannot restrict the computation to only a single node any more. However, the computation can still be limited to a small region because of the inherent locality of the problem. As

shown in Figure 9, the smallest circle is the region that we are interested to solve, and we define a larger area around this, called the "active area," which consists of nodes whose voltages are likely to affect nodes in the area to be solved. Since faraway nodes are unlikely to influence the solution significantly, we do not solve for their values with a great deal of accuracy. Specifically, we perform some approximations in the outside faraway region: for example, we can either ignore all the capacitors, or perform a grid coarsening by combining sets of nodes. Since the likelihood that a random walk from the smallest circle will go out of the active region is very small, this approximation has minor impact on accuracy, yet there is a tremendous speed-up.

6. Conclusion

An efficient power grid DC analysis algorithm has been proposed based on random walk technique, and can be extended to RC transient analysis. It has linear time complexity, and is shown to reach a good accuracy-runtime tradeoff. It also has the meaningful feature of localizing computation, making it especially useful when only a part of the supply grid is to be solved.

Acknowledgements

Our thanks to Haihua Su at IBM Austin Research Laboratory for help with the benchmark circuit.

References

[1] Doyle, P.G. and Snell, J. L., "Random walks and electric networks", 1984. Available at http://front.math.ucdavis.edu/math.PR/0001057

[2] Hagen, L. and Kahng, A.B., "A new approach to effective circuit clustering," *Digest of Technical Papers, IEEE/ACM International Conference on Computer-Aided Design*, pp. 422-427, 1992.

[3] Ho, C., Ruehli, A. and Brennan, P., "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, vol. CAS-22, no. 6, pp.504-509, 1975.

[4] Kozhaya, J., Nassif, S.R. and Najm, F.N., "A multigrid-like technique for power grid analysis," *IEEE Transactions on Computer-Aided Design*, vol. 21, no. 10, pp.1148-1160, 2002.

[5] Kuehlmann, A., McMillan, K.L. and Brayton, R.K., "Probabilistic state space search," *Digest of Technical Papers, IEEE/ACM International Conference on Computer-Aided Design*, pp. 574 –579, 1999.

[6] Le Coz, Y. L. and Iverson R. B., "A stochastic algorithm for high speed capacitance extraction in integrated circuits," *Solid-State Electronics*, vol.35, no. 7, pp. 1005-1012, 1992.

[7] Yates, R. D. and Goodman, D. J., *Probability and Stochastic Processes: A Friendly Introduction for Electrical and Computer Engineers*, John Wiley & Sons, New York, 1999.

[8] Zagajac, J., "A fast method for estimating discrete field values in early engineering design," *IEEE Transactions on Visualization and Computer Graphics*, vol. 2, no. 1, pp. 35-43, 1996.

[9] Zhao, M., Panda, R.V., Sapatnekar, S.S. and Blaauw, D., "Hierarchical analysis of power distribution networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 21, no. 2, pp. 159 –168, Feb. 2002.

[10] SPEC CPU2000 Results, available at http://www.specbench.org/cpu2000/results/cpu2000.html