Random Walks in a Supply Network

Haifeng Qian
University of Minnesota
200 Union Street SE
Minneapolis, MN 55455
qianhf@ece.umn.edu

Sani R. Nassif
IBM Austin Research Labs
11400 Burnet Rd. MS/9460
Austin, TX 78758
nassif@us.ibm.com

Sachin S. Sapatnekar
University of Minnesota
200 Union Street SE
Minneapolis, MN 55455
sachin@ece.umn.edu

ABSTRACT
This paper presents a power grid analyzer based on a random walk technique. A linear-time algorithm is first demonstrated for DC analysis, and is then extended to perform transient analysis. The method has the desirable property of localizing computation, so that it shows massive benefits over conventional methods when only a small part of the grid is to be analyzed (for example, when the effects of small changes to the grid are to be examined). Even for the full analysis of the grid, experimental results show that the method is faster than existing approaches and has an acceptable error margin. This method has been applied to test circuits of up to 2.3M nodes. For example, for a circuit with 70K nodes, the solution time for a single node was 0.42 sec and the complete solution was obtained in 17.6 sec.

Categories and Subject Descriptors
B.7.2 [**Integrated Circuits**]: Design Aids – *simulation.*

General Terms
Algorithms, Reliability, Verification.

Keywords
Random walk, power grid.

1. INTRODUCTION
Ensuring the reliability of the power grid is an indispensable part of high-performance design and there is a strong need for efficient supply grid analyzers. The problem of IR drops on supply grids becomes worse in successive technology generations as wire resistances increase due to the reduced interconnect widths, and the currents through the grid increase. These effects are magnified with the decreasing noise margins that accompany V_{DD} reductions from one technology node to the next.

The problem is often solved at two levels of detail. The steady-state case corresponds to DC analysis of the network, while the transient analysis problem includes the effects of capacitors/ inductors and time-varying current waveform patterns. We will first focus our attention on the DC analysis problem, and later, we will extend our algorithm to perform transient analysis.

This research was supported in part by the SRC under contract 99-TJ-692, by the NSF under award CCR-0205227, and by an IBM Faculty

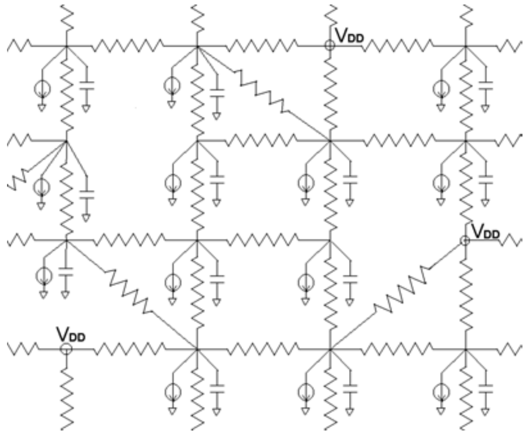

Figure 1. A typical circuit model for a part of a power grid.

The power grid model is illustrated in Figure 1. It consists of wire resistances, wire capacitances, decoupling capacitors, V_{DD} pads, and current sources that represent the currents drawn by logic gates or functional blocks. For DC analysis, all capacitors are open-circuited and inductors are short-circuited, so that the DC model contains no capacitors and has ideal voltage sources at the pads. The DC analysis problem is formulated as:

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

where G is the conductance matrix for the interconnected resistors, $\mathbf{X}$ is the vector of node voltages, and $\mathbf{E}$ is a vector of independent sources. Traditional approaches to solving the DC analysis problem often exploit the sparse and positive definite nature of G to solve this system of linear equations for $\mathbf{X}$. However, the cost of doing so can become prohibitive for a modern-day power grid with tens of millions of nodes, with the growing from one technology generation to the next.

Different methods have been proposed to address this issue. For example, [10] 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. These 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 the problem of power grid analysis, and use test results to show that it reaches a good accuracy-runtime tradeoff, compared with other methods. A significant advantage of this method over prior methods is that it is particularly useful and efficient when only a small fraction of the nodes in the grid are to be analyzed, as may be the case when a small part of the grid is modified.

This paper is organized as follows. Section 2 presents the theoretical basis of the proposed algorithm, followed by a simple illustrative example in Section 3. Experimental results for DC analysis on larger circuits are provided in Section 4. The proposed algorithm is extended to RC-network transient analysis in Section 5, and Section 6 presents some concluding remarks.

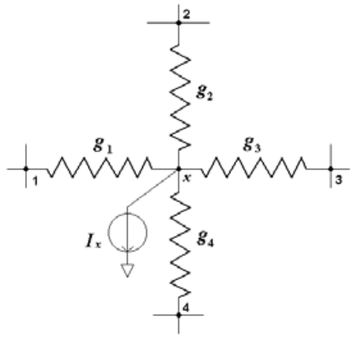

Figure 2. A representative node in the power grid.

2. RANDOM WALK PRINCIPLES

The random walk is a classical problem in statistics that has had numerous applications 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][9]. The work in this paper is inspired by the results due to Doyle and Snell [1], which 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. We apply and extend this method to the problem of DC analysis of a power grid, and develop practical implementational strategies that allow the solution to be applied to large instances.

We will focus our discussion on the description of a V_{DD} grid, pointing out the difference for a ground grid where applicable. For the DC analysis of a power grid, let us look at a single node x in the circuit, as illustrated in Figure 2. Applying Kirchoff's Current Law, Kirchoff's Voltage Law and the device equations for the conductances, we can write down the following equation:

$$\Sigma_{i=1 \text{ to degree}(x)}\ g_i\,(V_i - V_x) = I_x \qquad (2)$$

where the nodes adjacent to 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 as follows:

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

We can see that this implies that the voltage at any node is a linear function of the voltages at its neighbors. We also observe that the sum of the linear coefficients associated with the V_i's is 1. 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.

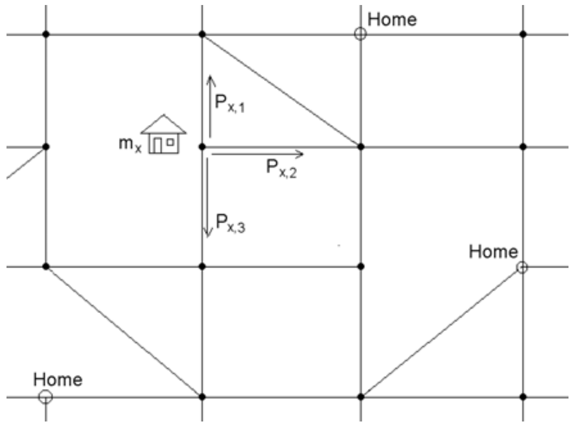

Figure 3. An instance of a random walk "game."

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 relationship:

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

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 accumulated 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

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

It is obvious that

$$f(\text{one of the homes}) = m_0 \qquad (6)$$

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

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

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 these two problems. Equation (7) becomes identical to Equation (3), and Equation (6) reduces to the condition of perfect V_{DD} nodes if we set

$$p_{x,i} = g_i \Big/ \sum_{i=1}^{\text{degree}(x)} g_i \qquad i = 1, 2, \cdots, \text{degree}(x)$$
$$m_x = I_x \Big/ \sum_{i=1}^{\text{degree}(x)} g_i \quad , \quad m_0 = V_{DD} \quad , \quad f(x) = V_x \qquad (8)$$

The calculations for ground net analysis are 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 by the same set of equations. It can be proven, easily 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 uses 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 [8], an acceptably accurate solution can be obtained. This is the idea behind our proposed algorithm.

According to the Central Limit Theorem [8], 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 stopping criterion driven by a user-specified error margin, Δ:

$$P[-\Delta < V_e - V < \Delta] > 99\% \quad (9)$$

where V_e is the estimated voltage from M experiments. If Var is the variance of these results, the above criterion can be written as

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

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 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 demonstrate the usefulness of this method for the solution at all nodes in the grid.

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.

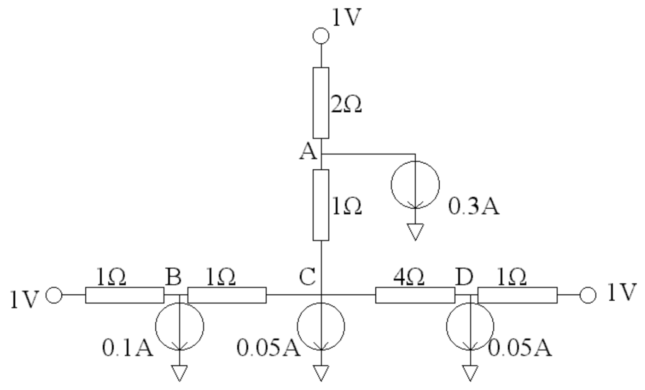

Figure 4. A simple circuit example.

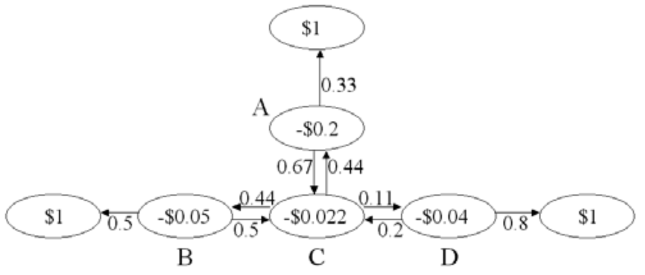

Figure 5. The random walk game corresponding to the circuit in Figure 4.

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

Let us 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. The Columns in the table represent five different runs of the proposed algorithm.

4. RESULTS OF DC ANALYSIS

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.

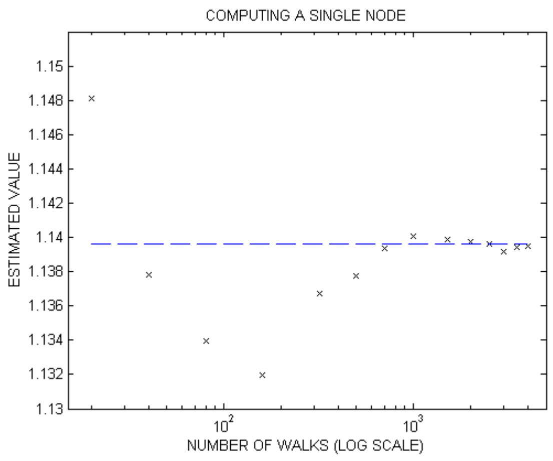

Figure 6. Estimated voltages at a single node for various values of M.

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 multiple node voltages, 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 needs 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.

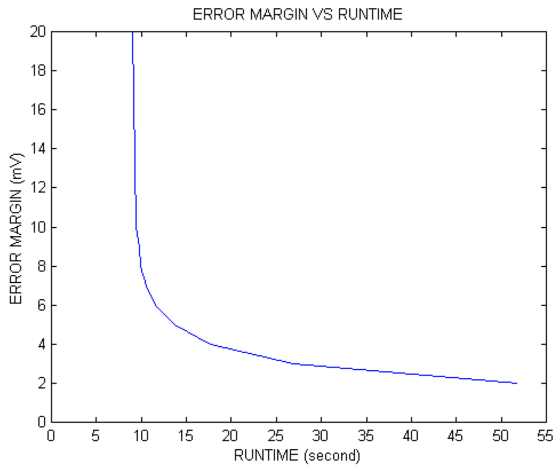

Figure 7. CPU time-accuracy tradeoff for the computation of all nodes in the test circuit.

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 runtime with other algorithms, we use the runtimes reported in [10] as the baseline. [10] 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.

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 [10], according to SPEC benchmarks [7]. The six circuits in [10] 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 [10] due to its superlinear time complexity. Since our algorithm has linear time complexity, as power grid size increases, it will outperform [10] 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.

Table 2. Runtime comparison.

Method		Runtime per thousand nodes (sec)
Random walk		0.25
Method in [10]	Chip-1	0.66
	Chip-2	0.55
	Chip-3	1.09
	Chip-4	1.33
	Chip-5	1.44
	Chip-6	1.70

5. TRANSIENT ANALYSIS: EXTENSION

The proposed method can be extended to transient analysis of RC supply grids, where the transient effects of capacitances and time-varying current waveforms are considered. In this case the equations to be solved may be written as follows [3]:

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

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

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

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 have:

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

where V_x, V_i, I_x and g_i are as defined in Equation (2), and C_x is the capacitance between node x and ground.

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

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

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 $V_x(t\text{-}h)$ with probability

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

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

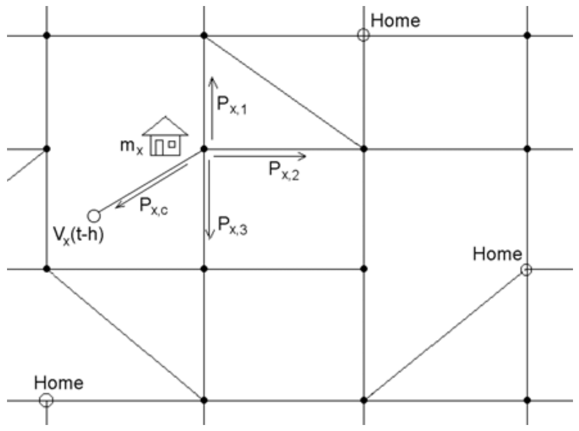

Figure 8. Rules for the transient analysis "game."

For transient analysis, traditional direct linear equation solvers 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. N is the circuit size, S is the number of timesteps, Δ is the voltage range, T is CPU time per timestep for subsequent timesteps, E1 is the average error, and E2 is the max error.

Ckt	N	S	Δ (V)	T (sec)	E1 (mV)	E2 (mV)
#1	2500	30	0.9297–1.1950	0.7m	1.0	8.6
#2	400	300	1.0886–1.1731	0.06m	1.4	5.9
#3	3700	500	1.0881–1.1774	2.6m	1.6	11.9
#4	2.3M	1000	N/A	0.65	N/A	N/A

In order to evaluate the transient analysis, since we were unable to obtain real-life RC power grid circuits, we generated four random circuits with realistic parameters. The results of our approach are shown in Table 3. The CPU times correspond to the runtimes for the time steps that follow the initial DC analysis and the first transient step. The solutions for circuits 1, 2 and 3 are compared with HSPICE, while circuit 4 is too large to be simulated in

HSPICE. The runtimes are several times faster than traditional direct solver runtimes reported in [10], even after normalization by the speed factor of 3 (estimated in Section 4).

The efficiency of transient analysis can be improved by taking advantage of the property of localization for random walks. Because the voltage value at the previous time step, *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 ignore all the capacitors in the faraway region, and thus do not update their *V(t-h)* values.

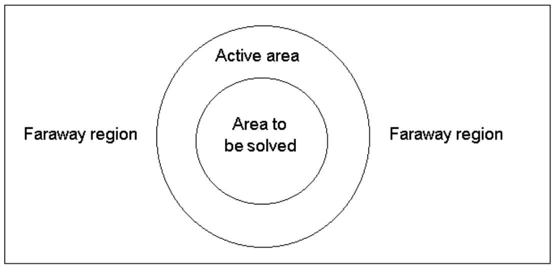

Figure 9. A scheme for localizing transient analysis.

The border is defined such that any walk from a node in "area to be solved" to a node in "faraway region" needs at least *K* steps. Because error is induced if and only if a walk from the center region reaches outside the border, and that error can be viewed as an incorrect estimation of the border voltage, the induced error is upper-bounded as follows.

$$\text{Error} \le P_{escape} * \Delta \qquad (15)$$

where Δ is a upper bound of the voltage swing of this power grid, and P_{escape} is the probability that a walk reaches outside the border. The value of Δ can be estimated by the initial DC analysis and that of P_{escape} by experiments. Thus, we can achieve a tremendous speed-up by localizing computation, and yet choose *K* to control the induced error within a user-specified margin.

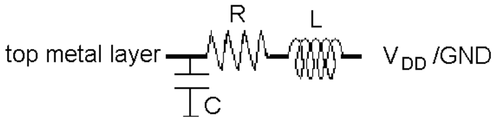

Figure 10. A pad model with self-inductance.

Incorporating pad inductances: A realistic power grid model uses RLC models at the pads, as shown in Figure 10, where L corresponds to a self-inductance. It is easy to extend our transient analysis method to handle these self-inductances. The application of the Backward Euler method at each time point creates a companion model for an inductor that consists of a voltage source in series with a resistor. We have already shown in our discussion on capacitances that such a structure can easily be handled within our formulation.

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 has the meaningful feature of localizing computation, making it especially useful when only a part of the supply grid is to be solved.

7. ACKNOWLEDGMENTS

Our thanks go to Haihua Su for help with the benchmark circuits, and to Narendra Shenoy for introducing the last author to the Doyle and Snell monograph many years ago.

8. REFERENCES

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

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

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

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

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

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

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

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

[9] J. Zagajac, "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.

[10] M. Zhao, R. V. Panda, S. S. Sapatnekar, and D. Blaauw, "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.