A Hierarchical Floating Random Walk Algorithm for Fabric-Aware 3D Capacitance Extraction

Tarek A. El-Moselhy
Department of Electrical Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139
tmoselhy@mit.edu

Ibrahim M. Elfadel
Systems & Technology Group
IBM Corporation
Yorktown Heights, NY 10598
elfadel@us.ibm.com

Luca Daniel
Department of Electrical Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139
luca@mit.edu

ABSTRACT

With the adoption of ultra regular fabric paradigms for controlling design printability at the 22nm node and beyond, there is an emerging need for a layout-driven, pattern-based parasitic extraction of alternative fabric layouts. In this paper, we propose a hierarchical floating random walk (HFRW) algorithm for computing the 3D capacitances of a large number of topologically different layout configurations that are all composed of the same layout motifs. Our algorithm is not a standard hierarchical domain decomposition extension of the well established floating random walk technique, but rather a novel algorithm that employs Markov Transition Matrices. Specifically, unlike the fast-multipole boundary element method and hierarchical domain decomposition (which use a far-field approximation to gain computational efficiency), our proposed algorithm is exact and does not rely on any tradeoff between accuracy and computational efficiency. Instead, it relies on a tradeoff between memory and computational efficiency. Since floating random walk type of algorithms have generally minimal memory requirements, such a tradeoff does not result in any practical limitations. The main practical advantage of the proposed algorithm is its ability to handle a set of layout configurations in a complexity that is basically independent of the set size. For instance, in a large 3D layout example, the capacitance calculation of 120 different configurations made of similar motifs is accomplished in the time required to solve independently just 2 configurations, i.e. a 60× speedup.

1. INTRODUCTION

The fundamental objective of litho-friendly physical synthesis methodologies that use restricted design rules (RDR) [1], or regular fabrics [2] is the maximization of layout printability. One crucial step in such methodologies is the enumeration of alternative layouts that implement a given boolean mapping, while satisfying a set of geometric constraints imposed by lithographic ground rules. The layout enumeration is driven by the two requirements of micro and macro regularity. "Micro" regularity is achieved by restricting shape edges to lie on a restricted design grid that also imposes stringent directionality on shape orientation. "Macro" regularity is achieved by using a very restricted set of litho-friendly logic cells. The enumeration can happen either upfront as a step toward finding the set of feasible layouts [1], or as part of an iterative process aimed at finding a feasible layout that is optimal not just in terms of printability, but also in terms of design requirements such as area or timing. An interesting instance of the iterative enumeration approach is given in [2], where a sequence of decompositions and recompositions are used to find the optimal set of litho-friendly patterns implementing a boolean mapping.

The objective of this paper is to show that the decomposition/recomposition enumeration and optimization paradigm can be moved down the litho-aware design methodology to the layout parasitic extraction step. Specifically, we focus on the 3D capacitance extraction of a large set of enumerated configurations composed of the same library of motifs. In our terminology, a "motif" is the fundamental building block seen by the extraction program, and it consists of a set of shapes along with their bounding box. In the extraction world, the motif plays the same role that the standard cell or the brick plays in the logic synthesis world.

To better convey the novel contributions of this paper, the reader is referred to Figures 1 and 2. Figure 1 represents a layout that has been decomposed into 6 motifs. The arrangement of these motifs constitutes one "configuration". Figure 2 is a different arrangement of the *same* motifs present in Figure 1. This is a second configuration. Figures 1 and 2 constitute a typical sequence of layout decomposition/recomposition steps in the new fabric oriented design methodology. The number of configurations resulting from such steps is potentially very large, i.e. $O(N_M!)$, where N_M is the number of motifs in the layout composition. The main question addressed in this paper is whether one can compute the capacitances of *all* these configurations in a highly efficient manner. Such a question is of course motivated by the obvious need for making litho-friendly physical synthesis methodologies as electrically-aware as possible [3]. To do so would require the fast electrical modeling of a large number of alternative layouts. It is important to notice that in this layout enumeration context, fast sensitivity or incremental extraction methodologies of the type advocated

ICCAD '09, November 2-5, 2009, San Jose, California, USA.

in [4] are not appropriate. This is because they rely on the fact that the layout variability is solely edge-based. Our approach addresses the more fabric relevant case, where the layout variations are motif-based. Such case is also the one that is more likely to affect the electrical performance of the design.

In this paper we introduce a novel algorithm for the efficient calculation of 3D capacitances of layout configurations composed of the same motifs. The efficiency of the algorithm is near-optimal in the sense that it is practically independent of the number of configurations. The algorithm is based on a novel combination of domain-decomposition techniques with floating random walk (FRW) techniques. The algorithm is hierarchical in the sense that it allows a hierarchy of motifs to be defined in the decomposition/recomposition sequence. This paper is focused on a detailed description of one level of the hierarchy, namely that of a single domain decomposition and how it can be used to gain orders of magnitudes of efficiency in the FRW calculation of capacitances of a large number of configurations.

The domain-decomposition step in our algorithm does not result in any loss of accuracy as is the case in the far-field approximation used in fast multipole [7], or pre-corrected FFT [6] techniques. Nor does our domain decomposition hierarchy result in the typical increase of complexity observed when solving the subdomain problems within hierarchical Boundary Element Methods [5]. Specifically, we employ Markov Transition Matrices (MTMs) to integrate the different subdomains (motifs) without any need for approximations. Furthermore, such MTMs can be easily precomputed and stored. Such additional storage is a small cost to pay for the orders of magnitude gain in efficiency. Finally, our hierarchical FRW algorithm inherits the "embarrassing" parallelism of FRW, and in a GPU-based parallel implementation, the MTMs can easily fit in the local storage associated with each GPU.

This paper is organized as follows. In Section 2 we summarize the traditional FRW algorithm for capacitance extraction. In section 3 we develop our hierarchical floating random walk (HFRW) algorithm using domain decomposition and Markov Transition Matrices (MTM). In section 4 we develop a fabric-aware HFRW algorithm for 3D capacitance extraction of a large number of configurations constructed by different recompositions of a set of motifs. In section 5 we present a theoretical analysis of our algorithms. Finally, in section 6 we show a variety of examples validating our algorithm.

2. BACKGROUND

2.1 Standard Floating Random Walk

The FRW algorithm [8] is based on expressing the capacitance C_{IJ} between conductor I and conductor J as a multidimensional (possibly infinite dimensional) integral

$$C_{IJ} = -\lim_{n\to\infty} \int_{S_0} d\eta_0 \hat{n}\cdot\nabla \int_{S_1} d\eta_1 G_1(\eta_0,\eta_1) \int_{S_2} d\eta_2 G_2(\eta_1,\eta_2) \times\cdots\times \int_{S_n} d\eta_n G_n(\eta_{n-1},\eta_n)\phi(\eta_n). \quad (1)$$

where S_0 is a Gaussian surface surrounding conductor I, $\hat{n}$ is the corresponding normal, ϕ is the electrostatic potential.

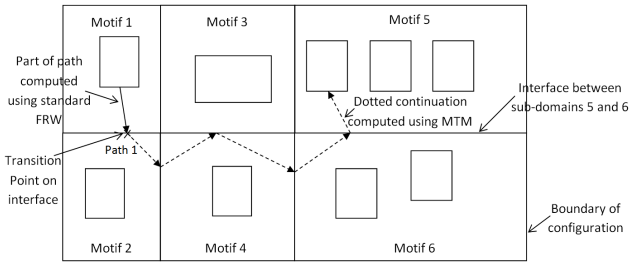

Figure 1: Domain decomposition of a hypothetical layout pattern. Each subdomain with its enclosed shapes constitute a layout motif. The arrangement of these motifs constitutes one configuration.

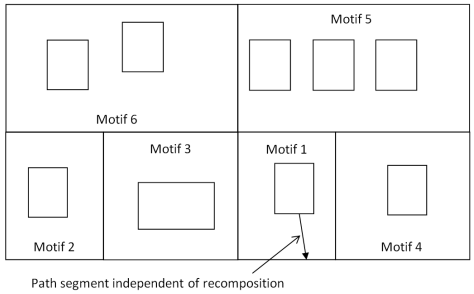

Figure 2: An alternative layout configuration made of the same set of motifs defined in Fig. 1.

$G_i(\eta_{i-1},\eta_i)$ is the Green's function used to write the potential at η_{i-1} as a function of the potential at the boundary of surface S_i, constructed such that it is centered around η_i and extends to the nearest conductor without including any metal structures (Fig. 3). Within the FRW algorithm, the Green's function $G_i(\eta_{i-1},\eta_i)$ of a given domain has a probabilistic interpretation, namely, it identifies a transition probability that measures the likelihood of a point η_{i-1} inside S_i to be connected with a point η_i on the boundary of S_i. In the rest of the paper, we will refer to the region inside such boundary as "transition domain" and we will refer to its associated Green's function as "transition probability".

The multidimensional integral (1) is computed by the FRW using Monte Carlo integration, where only one quadrature point is selected for each integral over a transition domain boundary. The sequence of quadrature points on the boundary of different transition domains can be interpreted as a "random walk (or path)" from a transition domain to another, whose stopping criterion is achieved when a step of the walk falls within a small distance ϵ from a surface with *known* potential belonging to a conductor.

Note that the first point of each path is randomly selected using a random variable p_0 uniformly distributed over the close surface S_0 surrounding conductor I. The capacitance C_{IJ} is then computed by averaging the contributions of all the M paths, $C_{IJ} = \frac{1}{M}\sum_{m=1}^{M}\gamma_m$ where

$$\gamma_m = \frac{\nabla_n G_1(\eta_0,\eta_1)}{p_0 G_1(\eta_0,\eta_1)}\Phi_m, \quad (2)$$

and the potential Φ_m at the end of the m^{th} path is 1 if the path terminates on conductor J and 0 otherwise.

One can further observe that the FRW paths used to cal-

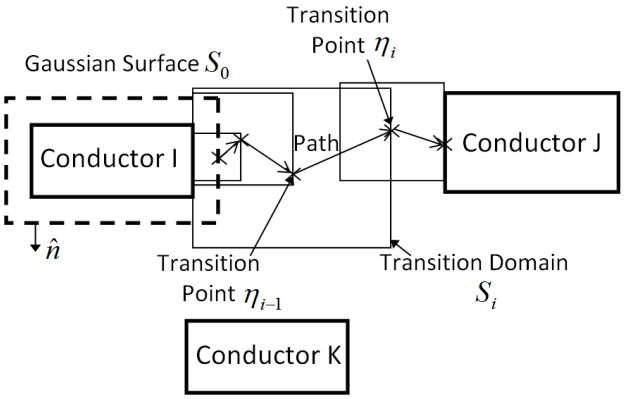

Figure 3: Typical random walk path from conductor I to conductor J.

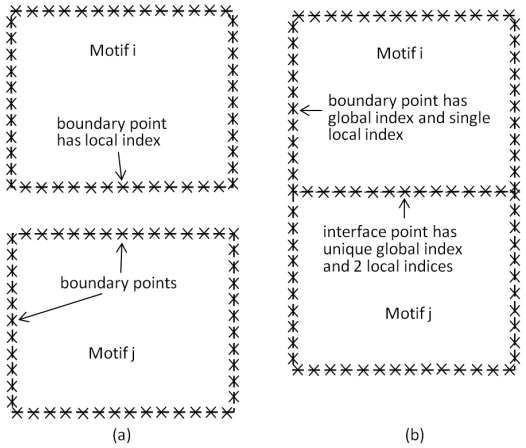

Figure 4: Geometry partitioned into two different motifs. The boundaries and interfaces are divided into segments. The center point of each segment is given both local and global indices.

culate C_{IJ} are not affected by the numerical value of the conductor potentials. Therefore, one single run of the FRW can provide all the entries for column I.

3. HIERARCHICAL FLOATING RANDOM WALK (HFRW)

The key observation enabling the development of our hierarchical floating random walk (HFRW) algorithm is that the paths are memoryless. That is, a FRW path starting from any point in the geometry can fully proceed without any knowledge of how this particular starting point has been reached. This observation appears to be new, as we are not aware of any reference to it in the existing literature or in available FRW implementations.

The first step of our HFRW is a domain decomposition, in which we partition the domain into motifs (see Fig. 1). The way in which the geometry is partitioned is arbitrary. However, one should try as much as possible to generate motifs that are small, contain dense metal configurations, are relatively independent, and have regular interfaces that can be easily recomposed together to construct different configurations.

The second step of HFRW is to generate *independently* for every motif a complete Markov transition matrix (MTM), representing the probability of moving from any point on the boundary of the motif to either any other point on the boundary of the motif, or any point on a conductor surface inside of the motif. In order to compute the MTM, the boundary of the motif is first discretized into N_B smaller segments. The MTM is of size $N_B \times (N_B + N_C)$, where N_C is the number of conductors inside the motif. The MTM is stochastically generated by initiating a large number of paths from each of the center points of the N_B segments. Such paths are allowed to proceed within the motif and only stop when they reach one of the motif boundary points, or a conductor surface inside the motif. Every time a path starting from point i on the motif boundary reaches another point j on the boundary or on a conductor surface (all the points on a single conductor are given the same index), the matrix entry (i, j) is incremented by 1. The final step in the computation of MTM is to normalize the rows of the matrix dividing each row by its total sum. The generation of the MTM is very simple since the typical motif is small and dense, and therefore the paths are typically short. Furthermore, since the number of stopping points is large, the average path length is small. Moreover, since every motif can be handled independently, and even every point on the boundary of any motif can be handled independently, the generation of the MTMs is "embarrassingly" parallelizable. Algorithm 1 summarizes the computation of the MTM for a given set of N_M motifs.

Clearly, the proposed algorithm requires storing the MTM of each motif. A large amount of memory would be required if the number of motifs is large and if each MTM is fully dense. None of the two conditions are likely to occur in practical layouts. In particular, if the motif boundary points are numbered judiciously, the Markov transition matrix will be sparse and structured (see for instance Fig. 8) and often banded with small bandwidth. Such banded structure is expected because distant points are likely to have very small transition probabilities. The MTM storage requirement will therefore be minimal. In fact, even if the MTM is dense, it can easily fit in the local storage of a GPU for a GPU-based parallel implementation of HFRW.

The third step is the recomposition step, in which the different motifs are combined together to construct different configurations of interest (see Fig. 2). Notice that when motifs are combined together some of the boundaries of the motifs become interfaces in the constructed configuration. To keep track of the arrangement of the motifs each boundary and interface in the configuration is divided into segments, and each center point of every segment is given a global index (see Fig. 4). We use simple maps to relate the global index of every interface/boundary point to its local index within every motif it belongs to. Note that boundary points belong to a single motif, while interface points belong to at least two motifs.

Algorithm 2 describes how the capacitance vector of a particular conductor I, is extracted. According to this algorithm, the complete HFRW path consists of a standard FRW path inside the first motif and a sequence of Markov transitions between points on the motif interfaces. The Markov transition part of the HFRW (walk on interfaces) does not require any geometric manipulations, such as tran-

Algorithm 1 Generation of MTM $\mathcal{T}_k$ for motif $\mathcal{M}_k$ within HFRW

for each motif $\mathcal{M}_k$ **do**
  $\mathcal{T}_k \leftarrow 0$
  for each boundary point i of motif $\mathcal{M}_k$ **do**
    repeat
      generate a FRW path starting from point i and directed inside of motif $\mathcal{M}_k$
      if path terminates at point j on the boundary **then**
        $\mathcal{T}_k(i,j) \leftarrow \mathcal{T}_k(i,j) + 1$
        break
      else if path terminates at a conductor l inside motif $\mathcal{M}_k$ **then**
        $\mathcal{T}_k(i, N_B + l) \leftarrow \mathcal{T}_k(i, N_B + l) + 1$
        break
      end if
    until convergence is achieved
    $S = sum(\mathcal{T}_k(i,:))$
    $\mathcal{T}_k(i,:) \leftarrow \frac{\mathcal{T}_k(i,:)}{S}$
  end for
end for

sition domain determination, and is therefore extremely efficient. The HFRW path terminates when it reaches either a conductor or a configuration boundary. In the former the HFRW path value is added to the value of the capacitance between conductor I and the conductor at which the path terminates. In the latter the value of the HFRW path is added to the self capacitance of conductor I.

Algorithm 2 HFRW for a given configuration

repeat
  generate a FRW path from conductor I fully contained in its motif $\mathcal{M}$ reaching either a conductor inside $\mathcal{M}$ or a point on the boundary of $\mathcal{M}$
  if path reached a point on the interface between motifs **then**
    repeat
      choose one of the motifs to which the point belongs
      Use MTM of new motif to make a transition
    until transition reaches a conductor or a configuration boundary
  end if
  if transition terminated on a conductor **then**
    add value of path (2) to capacitance $C(I, L)$, where L is the index of the terminating conductor
  else {transition terminated on configuration boundary}
    add value of path (2) to self capacitance $C(I, 0)$
  end if
until convergence achieved

3.1 Comparison between HFRW and standard hierarchical domain decomposition

Our HFRW algorithm exploits the locality of FRW and couples it with domain decomposition to realize a very fast, parallel solver. Beside the fact that HFRW is based on FRW while domain decomposition is typically used for BEM and FEM, there are two other important differences between HFRW and domain-decomposition algorithms:

1. First, domain decomposition algorithms are based on neglecting the interactions between elements in different subdomains. No single interaction needs to be neglected in our HFRW since the algorithm is inherently local.

2. Second, domain decomposition algorithms require solving a complex problem within each motif, as opposed to the generally simpler problem solved by the standard version of the algorithm within the entire configuration. For instance, the hierarchical BEM algorithm presented in [5] requires inverting a potential matrix to compute a capacitance matrix for each motif, as opposed to just solving a linear system to compute the charges. On the other hand, the FRW extracts the capacitance directly, and therefore the procedure required by HFRW to solve the individual motifs is exactly the same as the one required by standard FRW to solve the entire configuration. In addition, computing the MTM within each motif is extremely efficient, since FRW is at its best when dealing with small dense structures.

Based on the above two observations, one can conclude that, compared with standard FRW, HFRW is computationally more efficient, preserves the same accuracy, and requires slightly more memory. Compared with domain-decomposition methods based on FEM or BEM, HFRW is more accurate since it is exact and does not require any approximations.

4. FABRIC-AWARE 3D HFRW ALGORITHM

In this section we present an algorithm for 3D capacitance extraction of a large number of configurations constructed by different recompositions of a set of motifs. Recall that if we have N_M motifs then there are $O(N_M!)$ possible different configurations. Algorithm 3 summarizes the steps of our proposed approach.

Algorithm 3 HFRW for all configurations

for each motif $\mathcal{M}_k$ **do**
  use Algorithm 1 to compute MTM $\mathcal{T}_k$
end for
for each desired capacitance matrix column $C(I,:)$ **do**
  repeat
    generate a FRW path from conductor I reaching either a conductor inside the same motif or a point on the boundary of the motif
    for each configuration **do**
      use MTMs to walk on interfaces and terminate on a conductor or configuration boundary
      add value of HFRW path to appropriate capacitance
    end for
  until convergence achieved
end for

Since all configurations are constructed from the same set of motifs, the MTMs for each motif can be precomputed separately as shown in Step 2. The complexity of this part of the algorithm depends linearly on the number of motifs

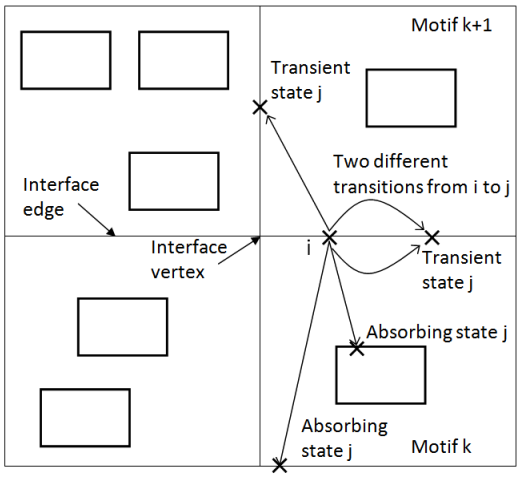

Figure 5: Transition to interfaces (transient states) contribute to the Q matrix and transition to conductors and configuration boundaries (absorbing states) contribute to R.

$O(N_M)$, and does not depend on the total number of configurations.

Step 6 (standard FRW) in Algorithm 3 is independent of the configuration structure, and therefore it is implemented once per motif and reused for all configurations. This step is very efficient since the standard FRW requires short paths when the boundary of the domain is close to the conductor as in the case of a motif.

The remaining part of each HFRW path depends on the particular recomposition of the motifs, therefore it must be implemented separately for each configuration. Since this part of the algorithm does not involve any geometrical manipulations it is extremely cheap. Consequently, the bottleneck of the algorithm are Steps 2 and 6, and the complexity of our algorithm is almost completely independent of the total number of configurations.

5. THEORETICAL ANALYSIS OF HFRW

Many standard "Absorbing Markov Chain" theorems and well known results can be exploited to certify properties of our HFRW algorithm once we show how it is possible to appropriately construct a large Markov Transition Matrix $\mathcal{T}$ for the entire configuration:

$$\mathcal{T} = \begin{bmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3}$$

where $\mathbf{Q}$ is the transition probability matrix between "transient states" (in our case any point on the interface between motifs as shown in Fig. 5); $\mathbf{R}$ is the transition probability matrix from the transient states to the "absorbing states" (in our case all the conductors and any point on the external configuration boundary). Matrices $\mathbf{0}$ and $\mathbf{I}$ simply define the behavior of the absorbing states: once an absorbing state is reached the probability to remain in it is one, and consequently the probability to transition to any other state is zero. The upper part $[\ \mathbf{Q}\quad \mathbf{R}\ ]$ of the Markov Transition Matrix $\mathcal{T}$ can be related to the individual Markov Transition Matrices of each motif using the Law of Total Probability

$$[\ \mathbf{Q}\quad \mathbf{R}\ ](i,j) \quad = \sum_{\text{each motif } k} P[i \in \mathcal{M}_k] \quad P[i \to j \mid i \in \mathcal{M}_k]$$

or in other words, the sum of the probabilities of choosing motif $\mathcal{M}_k$ in step 5 of Algorithm 2 multiplied by the conditional probabilities of transitioning from point i to point j, given the choice of motif $\mathcal{M}_k$. Notice that

$$P[i \to j \mid i \in \mathcal{M}_k] = \begin{cases} \mathcal{T}_k(i,j) & j \in \mathcal{M}_k \\ 0 & \text{otherwise} \end{cases}$$

where $\mathcal{T}_k(i,j)$ is the MTM of motif $\mathcal{M}_k$ constructed by Algorithm 1. Also notice that in the simple 2D uniform media case:

$$P[i \in \mathcal{M}_k] = \begin{cases} 1/2 & i \text{ on interface edge of 2D } \mathcal{M}_k \\ 1/4 & i \text{ on interface vertex of 2D } \mathcal{M}_k \\ 0 & \text{otherwise} \end{cases}$$

and in the simple 3D uniform media case:

$$P[i \in \mathcal{M}_k] = \begin{cases} 1/2 & i \text{ on interface surface of 3D } \mathcal{M}_k \\ 1/4 & i \text{ on interface edge of 3D } \mathcal{M}_k \\ 1/8 & i \text{ on interface vertex of 3D } \mathcal{M}_k \\ 0 & \text{otherwise} \end{cases}$$

Having cast our HFRW algorithm as an Absorbing Markov Chain problem it is easy to rigorously answer many legitimate questions using the literature available on that topic [9]. For instance, the following theorem can be used to prove the termination of each HFRW "path" in a finite number of transitions, and to even provide a precise estimate on the average number of transitions before termination.

THEOREM 5.1. *Assume that HFRW starts at a point i on an interface between motifs (i.e. a transient state), then the average length of the walk on interfaces, or expected number of transitions before reaching a conductor or the configuration boundary (i.e. an absorbing state) is finite and is given by the row sum of the i-th row in the "fundamental matrix"* $\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1}$.

PROOF. The proof follows directly from Theorem 3.3.5 in [9]. □

6. RESULTS

6.1 Accuracy Validation

The first example validates the accuracy of the proposed algorithm. The 2D geometry for this example is shown in Fig. 6, and is composed of 12 conductors. To implement our HFRW, the geometry is divided into 4 different motifs. In addition, empty motifs are used at the boundary (shown only partially in Fig. 6) to mimic the infinity boundary condition.

The time required to compute the MTM for all motifs is half the time required to simulate all 12 conductors using the standard FRW. We extracted the capacitance between a target conductor in motif 1 (see Fig. 6) and all the other 11 conductors using both our HFRW and the standard FRW. The time required for our HFRW is approximately half that required for the standard FRW for the same accuracy. Therefore the time required for our complete algorithm is about the same time (1.01×) required for the standard FRW. When compared to a standard FRW with

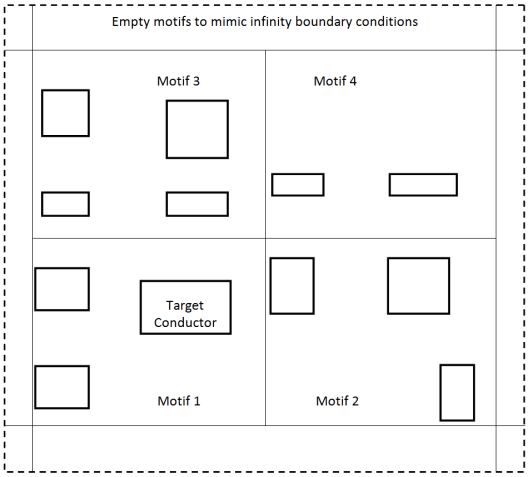

Figure 6: Geometry partitioned into different motifs. Empty motifs at the boundary (shown only partially) are used to mimic the infinity boundary condition.

2×10^6 different random paths to guarantee convergence, our approach obtained for all extracted capacitances a 1% accuracy.

6.2 A 2D Fabric-Aware Extraction Example

In this example we use the same motifs used in Fig. 6 (from the previous example 6.1) to construct a total of $4! = 24$ different configurations, corresponding to all possible different recompositions of the four internal motifs. The capacitance matrices of each of the configurations are computed using both our HFRW and the standard FRW. All the values of the computed capacitances for each configuration are within 2% of the values computed using the standard FRW. The total time to compute *all* 24 capacitance matrices using our HFRW is about equal (1.1×) to the time required to compute the capacitance matrix of just *one* configuration using the standard FRW. This corresponds to a 21× speedup.

6.3 A Large 3D Example

In this subsection we demonstrate that the HFRW can also be used to treat 3D structures very efficiently. The example under consideration is a 5 layer structure (Fig. 7). Two of such layers each contain a total of 100 cubic shaped conductors arranged on a 10×10 grid. The size of each conductor is 100nm. These small conductors represent for instance "metal fill", i.e. small floating conductors inserted in empty regions of the layout to facilitate the planarization. The other three layers each contain 3 parallel long wires of dimensions 100nm×1400nm×100nm. The wires are separated by 100nm. Each of the five layers is 300nm thick. Each layer is treated as a motif. We recompose such motifs to construct a total of 120 different configurations. Note that each configuration will include a total of 209 total conductors. For each configuration we extract four different capacitances. The largest of the 5 MTMs, each associated with one of the 5 layers, has size 1536×1636, and is 95% sparse (Fig. 8). The time required to compute all 5 MTMs is approximately 15 minutes in a code implemented in Matlab and running on a Intel Duo CPU at 2.4GHz with 2GB of memory. Such time can be significantly reduced by using

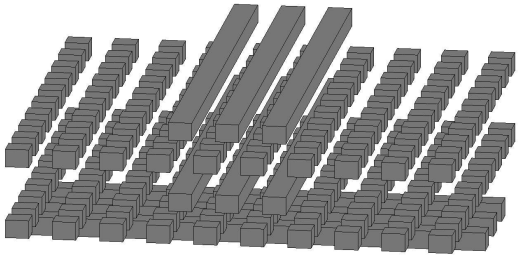

Figure 7: A five layers, 209 conductor structure. Each layer is treated as a motif.

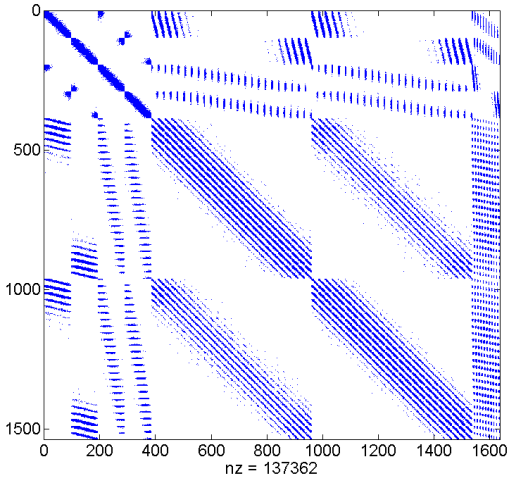

Figure 8: A typical sparsity pattern of the Markov Transition Matrix (MTM) for a motif including 100 cubic conductors.

C/C++ code and by using a parallel implementation of the algorithm. After the MTMs are computed, the subsequent computational time required to solve all possible 120 configurations is 15 minutes in Matlab on the same machine (5min in Matlab on three parallel processes) as opposed to an estimated time of about 1800min required to solve all configurations using the standard FRW. This corresponds to a 60× speedup. Running FastCap (C code) on the same set of 120 configurations with the same 5% accuracy requires a total of 270min. Hence, our Matlab HFRW is 9× faster than C-code FastCap.

7. CONCLUSION

In this paper we have presented a hierarchical floating random walk (HFRW) algorithm for computing the 3D capacitances of a large number of layout configurations that, although topologically different, are all composed of the same layout motifs. Our algorithm is not a standard hierarchical domain decomposition extension of the well established floating random walk technique. It is rather a novel algorithm that employs Markov Transition Matrices (MTM) to integrate the different subdomains (motifs) together. More specifically, our approach is exact and does not rely on approximations and tradeoff between accuracy and computational efficiency, as opposed to standard hierarchical domain decomposition techniques which use a far-field approxima-

tion to gain computational efficiency. Furthermore, the computation in our approach of the MTM associated with every motif is extremely efficient and requires minimal memory, since a motif is typically small and includes dense metal structures. The main advantage of our algorithm is its extreme efficiency in extracting the capacitance matrix of a large number of configurations constructed by different recompositions of the same set of motifs. We have observed that its complexity is almost independent of the number of configurations. In particular, in a large example, the total time required to compute all the capacitance matrices of 120 different 3D configurations is only two times the time required for solving a single configuration using the standard FRW. This is equivalent to a 60× speedup.

Acknowledgment

The authors acknowledge the support of Mentor Graphics, AMD, IBM, the Semiconductor Corporations and the Interconnect Focus Center, one of five research centers funded under the Focus Center Research Program, a DARPA and Semiconductor Research Corporation program.

8. REFERENCES

[1] L. Liebmann, "DfM, the Teenage Years," *Proc. of SPIE*, Vol. 6925, pp. 692502-1 – 692502-13, 2008.

[2] T. Jhaveri, V. Rovner, L. Pileggi, A. Strojwas, D. Motiani, K. Y. Tong, T. Hersan, D. Pandini, "Maximization of layout printability/manufacturability by extreme layout regularity," *J. of Micro/Nanolithography MEMS MOEMS*, Vol. 6, No. 3, pp. 031011-1 – 031011-15, 2007.

[3] S. Banerjee, P. Elakkumanan, L. Liebmann, J. Culp, M. Orshansky, "Electrically Driven Optical Proximity Correction," *Proc. of SPIE*, Vo. 6925, pp. 69251W-1 – 69251W-9, 2008.

[4] T. El-Moselhy, I. M. Elfadel, L. Daniel, "A Capacitance Solver for Incremental Variation-Aware Extraction," *ICCAD'08*, pp. 662–669, Nov. 2008.

[5] L. Jiang, B. Rubin, J. Morsey, H. T. Hu, I. M. Elfadel, "Novel Capacitance Extraction Method Using Direct Boundary Integral Equation and Hierarchical Approach," *Workshop on Electrical Performance of Electronic Packaging*, pp. 331 – 334, Oct. 2006.

[6] J. R. Phillips and J. K. White, "A precorrected-FFT method for electrostatic analysis of complicated 3D structures" *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1059–1072, 1997.

[7] K. Nabors and J. K. White, "FASTCAP A multipole-accelerated 3D capacitance extraction program" *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1447–1459, 1991.

[8] R. B. Iverson and Y. L. Le Coz, "A Stochastic Algorithm for High Speed capacitance Extraction in Integrated Circuits," *Solid-State Electronics*, Vol. 35, No. 7, pp. 1005–1012, 1992.

[9] J. Kemeny and J. Snell, "Finite Markov Chains," Springer-Verlag, 1976. Chapter 3.