

Fast Physics-Based Electromigration Assessment by Efficient Solution of Linear Time-Invariant (LTI) Systems

Sandeep Chatterjee[†], Valeriy Sukharev[‡] and Farid N. Najm[†]

[†]*ECE Department, University of Toronto, Toronto, ON, Canada, M5S 3G4*

[‡]*Mentor, A Siemens Business, Fremont, CA 94538, USA*

sandeep.chatterjee@mail.utoronto.ca, valeriy_sukharev@mentor.com, f.najm@utoronto.ca

Abstract—Electromigration (EM) is a key reliability concern in chip power/ ground (p/g) grids, which has been exacerbated by the high current levels and narrow metal lines in modern grids. EM checking is expensive due to the large sizes of modern p/g grids and is also inherently difficult due to the complex nature of the EM phenomenon. Traditional EM checking, based on empirical models, cannot capture the complexity of EM and better models are needed for accurate prediction. Thus, recent physics-based EM models have been proposed, which remain computationally expensive because they require solution of a system of partial differential equations (PDEs). In this paper, we propose a fast and scalable methodology for power grid EM verification, building on previous physics-based models. We first convert the PDE system to a succession of homogeneous linear time invariant (LTI) systems. Because these systems are found to be stiff, we numerically integrate them using optimized variable-step backward differentiation formulas (BDFs). Our method, for a number of IBM power grids and internal benchmarks, achieves an average speed-up of over 20x as compared to previously published work and has a runtime of only about 8 minutes for a 4 million node grid.

Index Terms—Power Grids, Electromigration, Reliability, Backward Differentiation Formulas.

I. INTRODUCTION

Electromigration (EM) is a serious issue in modern p/g grids. It is becoming harder to sign off on chip designs using state-of-the-art EM checking tools, due to the very small margins between the allowed failure rates (spec) and the failure rates at which the chips actually operate in the field [1]. This loss of safety margin can be traced back to the simplicity and pessimism inherent in the EM models used in existing tools, such as Black's model for line failure and a series model for power and ground (p/g) grid failure. This simplicity and pessimism is often rationalized on the grounds of necessity (the actual physical system is too complex to be analyzed) and conservatism (the analyzed system is worse than the actual one), and these tools got the job done for the past 30 years. However, over the last decade, technology scaling has exacerbated EM [2]. As such, existing EM tools do not provide any "breathing room" for designers, who are forced to overuse metal resources to design p/g grids, which leads to a sub-optimal design.

Our work is focused on EM in the p/g grid, because the presence of mostly uni-directional currents in the grid make it more susceptible to EM, and because the sheer size of modern grids

makes them very difficult to verify. In existing EM tools, a grid is deemed to have failed as soon as the first failure happens. This has been called a *series model* of grid failure. However, modern power grids are more like a mesh, with a lot of redundancy, and as such can withstand multiple line failures before they actually fail. Thus, a *mesh model* for grid failure has been proposed [3], in which a grid is said to have failed when the voltage drop at any node exceeds some threshold value that would lead to timing errors in the underlying logic. We will build on the mesh model.

Other previous works have focused on improvements to the EM model itself. In Black's model, a multi-branch metal structure is decomposed into separate branches to assess EM degradation. This is highly inaccurate because it ignores the *material flow* between connected branches. Luckily, in modern grids one does not have to treat the whole p/g grid as a fully connected structure when it comes to material (metal) flow. This is because, in modern grids, the metal lines and vias within a metal layer are formed simultaneously using dual damascene process, and are surrounded by a barrier metal liner. This barrier metal prevents the flow of metal atoms from one metal layer to another, so that one can restrict the consideration of metal flow to within individual layers. Furthermore, each metal layer of the grid mostly consists of alternating parallel stripes of power and ground lines. As such, these multi-branch segments on every layer are mostly *trees*, i.e. they contain no loops, and are typically referred to as *interconnect trees*. For accurate EM checking, one must consider the EM degradation of a tree as a whole, rather than focusing on its separate branches.

Physics-based EM models do exactly that, and thus provide more accurate EM assessment. In recent years, many works have proposed p/g EM verification using physics-based EM models. Huang et. al. [4] proposed compact EM models based on Korhonen's model [5] for p/g EM checking, and Li et al. [6] used the simplified EM model proposed by Riege et al. [7] to improve p/g grid resilience to EM. Chatterjee et. al. [8] proposed the *Extended Korhonen's model* (EKM) and used it for their p/g EM verification engine. However, these approaches had some drawbacks. For example, [4] and [6] used simplified EM models that do not provide accurate EM checking, and were slow for some test grids. While [8] used a better EM model, it was not scalable to very large grids because EKM requires solution of a system of partial differential equations (PDEs).

In this paper, building on the work of [8], we propose a very fast and scalable approach for power grid electromigration assessment, using EKM for electromigration checking and the mesh model for determining power grid failure. We begin by converting the PDE system resulting from EKM into a succession of *Linear Time Invariant* (LTI) system(s) of ordinary differential equations

This work was supported in part by the Semiconductor Research Corporation (SRC) and by Mentor, A Siemens Business.

(ODEs) with constant inputs. These LTI systems are found to be *stiff*, so we numerically solve them using variable-coefficient Backward Differentiation Formulas (BDFs). To improve the performance, we implement a number of optimizations in our BDF solver. Specifically, we convert the LTI systems with constant inputs to homogeneous LTI systems to reduce the computational work per step, eliminate the Newton iteration step usually associated with BDFs, and use customized error control for maximum performance. Our overall power grid EM checking approach also includes a couple of significant improvements over the work in [8]: we use 1) preconditioned conjugate gradient method for updating voltage drops, and 2) a multi-process architecture to parallelize the code. Our method, for a number of IBM power grids and internal benchmarks, achieves an average speed-up of $\sim 23\times$ over previously published work. With a runtime of only about 8.2 minutes for a 4.1 million node grid and maximum runtime of 15 minutes among all benchmarks, our approach is extremely fast and should scale well for large integrated circuits.

II. BACKGROUND

A. Electromigration basics

Electromigration is the mass transport of metal atoms due to momentum transfer between electrons (driven by an electric field) and the atoms in a metal line. Time to Failure (TTF) due to EM is a random variable because EM is highly dependent on the specific microstructure of a given line, which varies due to random manufacturing variations. The process of EM degradation can be divided into two phases: *void nucleation* and *void growth*.

Under conditions of high current density, metal atoms are pushed in the direction of the electron flow, which is opposite to the direction of the applied electric field. The number of atoms moving through a cross-section of a metal line per second per unit area is known as the *atomic flux*. If the in-flow of atoms is *not* equal to the out-flow, *atomic flux divergence* or AFD is said to occur. AFD generates points of high tensile and compressive stresses within a metal segment. The amount of compressive stress needed to cause a pile-up of metal atoms (*a hillock*) leading to a short circuit is very high in modern metal systems, hence failure due to a short circuit is not usually observed. However, the build up of tensile stress eventually leads to formation of a *void* when the stress reaches a pre-determined critical threshold. This phase of EM degradation, when stress is increasing over time but no voids have yet nucleated, is called the *void nucleation* phase. In this phase, the resistance of a line remains roughly the same as that of a fresh (undamaged) line.

Once a void nucleates, the *void growth* phase begins. In some cases, depending on the size and the location of the void, nucleation by itself might lead to an *early failure* by disconnecting a via above and causing an open circuit [9]. On the other hand, depending on geometry, a line may continue to conduct current after void nucleation. With time, the void starts to grow in the direction of the electron flow and the line resistance increases towards some finite steady-state value. In testing of single isolated lines, failure is deemed to happen when the increase in resistance reaches 10%–20% of the initial resistance value.

B. Power Grid Model

EM is a long-term failure mechanism. As such, short-term transients in workload typically experienced in chips do not play a significant role in EM degradation. Thus, standard practice in the field is to use an effective-current model [10] to estimate EM degradation, so that the lifetime of a metal line when carrying the constant effective current and the time-varying transient current is the same. As voids nucleate due to EM, branch resistances change fairly quickly. Correspondingly, the currents also change fairly

quickly to their new effective values. Hence, between any two successive void nucleations, the power grid has constant (effective) currents, voltages and conductances, and thus can be modelled as a DC system. Also, given that the power grid is a linear system, effective branch currents can be obtained directly from effective source currents by doing a DC analysis. The power grid model thus can be expressed as

$$\mathbf{G}(t)v(t) = i_s, \quad (1)$$

where $\mathbf{G}(t)$ is the piecewise-constant conductance matrix, $v(t)$ is the corresponding time-varying but piecewise constant node voltage drop vector and i_s is the vector of effective source current values that model the underlying logic blocks.

C. The Korhonen Model

Korhonen et al. [5] proposed a one-dimensional (1D) model to describe the *hydrostatic stress* σ arising under the influence of electromigration. Here, hydrostatic stress is the average of all normal components of the full stress tensor, i.e. $\sigma = (\sigma_{xx} + \sigma_{yy} + \sigma_{zz})/3$. Consider a uniform metal line embedded in a rigid dielectric. Korhonen's model starts with *Hooke's law* to relate the variation in concentration of atoms C , with hydrostatic stress

$$\Delta C(x, t)/C(x, t) = -\Delta\sigma(x, t)/B, \quad (2)$$

where B is the bulk modulus and $\sigma(x, t)$ is the time-varying hydrostatic stress at location x from some reference point, and at time t . Following Korhonen's formulation, σ is positive for tensile stress and negative for compressive stress, and can be obtained by solving the PDE

$$\frac{\partial\sigma}{\partial t} = \frac{B\Omega}{k_b T_m} \frac{\partial}{\partial x} \left\{ D_a \left(\frac{\partial\sigma}{\partial x} - \frac{q^* \rho}{\Omega} j \right) \right\}, \quad (3)$$

where j is the current density in the line, $D_a = D_0 e^{-Q/(k_b T_m)}$ is the lognormally distributed [11] atomic diffusivity with constant coefficient D_0 , Ω is the atomic volume, k_b is the Boltzmann's constant, T_m is the temperature in Kelvin, q^* is the absolute value of the effective charge of the conductor, ρ is the resistivity of the conductor, and Q is the activation energy for vacancy formation and diffusion. A void nucleates once the stress exceeds a predefined threshold value $\sigma_{th} > 0$. The corresponding atomic flux J_a in the line can be written as

$$J_a = \frac{D_a C \Omega}{k_b T_m} \left(\frac{\partial\sigma}{\partial x} - \frac{q^* \rho}{\Omega} j \right). \quad (4)$$

D. Extended Korhonen's model

We will now review the *Extended Korhonen's model* (EKM). This physics-based EM model was proposed in [8] [12], and augments Korhonen's model by introducing boundary laws to track the material flow between connected branches. It also accounts for thermal stresses generated by non-uniform temperature distribution. We will briefly summarize EKM and its use in power grid EM checking for completeness.

EKM evaluates the EM degradation of an *interconnect tree* as a whole. Formally, an interconnect tree is a acyclic graph $\mathcal{T} = (\mathcal{N}, \mathcal{B})$, where \mathcal{N} is a set of tree *junctions* and \mathcal{B} is a set of resistive *branches*. Fig. 1 shows a typical interconnect tree structure. A *branch* is a continuous straight metal line of uniform width. A *junction* is any point on the interconnect tree where a branch ends or where a via is located. Junctions are classified based on their *degree*, which is defined to be the number of branches connected to it, as shown in Fig. 1. Note that vias do not contribute to the degree of a junction.

EKM starts out by using Breadth First Search (BFS) to assign reference directions to all branches in the tree (shown by dashed

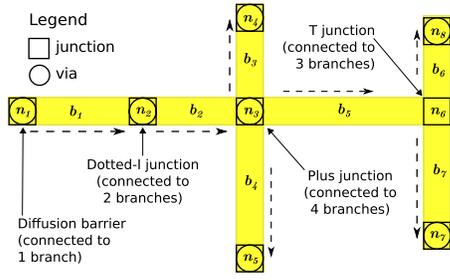


Fig. 1. A typical interconnect tree structure.

arrow lines in Fig. 1). The current density j_k and the atomic flux $J_{a,k}$ for branch b_k is positive if it flows in the reference direction, otherwise it is negative. The initial stress at $t = 0$ in branch b_k is equal to its residual thermal stress $\sigma_{T,k}$, which can be computed as shown in [13]. For any point within branch $b_k \in \mathcal{B}$, i.e. at $x_k \in (0, L_k)$ where L_k is branch length, EKM determines the stress $\sigma_k(x_k, t)$ using the original Korhonen's model (3). The boundary laws govern the behaviour of stress at branch ends ($x_k \in \{0, L_k\}$), i.e. junctions, and is summarized as follows [8]:

Law 1. Before a void nucleates at a junction, stress is continuous across a junction and the number of metal atoms flowing into a junction per unit time is equal to the number of number of metal atoms flowing out from it.

Law 2. After a void nucleates at a junction, there is no flow of atomic flux between the connected branches. The stress gradient at the junction will be

$$\partial\sigma_{jn}/\partial x_k = \pm\sigma_{jn}/\delta, \quad (5)$$

where σ_{jn} is the stress value at the junction and δ is the thickness of the void interface.

EKM assumes that diffusivity (D_a) is the same throughout a branch. As a result, voids nucleate only at junctions of a tree. Once the stress at any junction reaches σ_{th} , a void nucleates at that point and is shared by all the connected branches. Because the initial void growth rate is very high [13], EKM conservatively assumes that once a void nucleates, it reaches the steady state volume in a negligible period of time. Accordingly, the line resistance rises immediately to its steady state value for all connected branches. The steady state void volume for a branch b_k is

$$\mathcal{V}_{k,max} = L_k w_k h_k (\sigma_{T,k}/B + (q^* \rho |j_k| L_k)/(2B\Omega)), \quad (6)$$

where w_k is the width and h_k is the height of branch b_k .

1) *The PDE system for a tree:* EKM scales time, stress and distance (length) for any branch $b_k \in \mathcal{B}$ by introducing the following dimensionless variables

$$\tau \triangleq \frac{B\Omega}{k_b T_m^*} \frac{D_a^* t}{L_c^2}, \quad \eta_k \triangleq \frac{\Omega \sigma_k}{k_b T_m^*}, \quad \xi_k \triangleq \frac{x_k}{L_k}, \quad (7)$$

where D_a^* is the atomic diffusivity at some chosen nominal temperature T_m^* , L_c is some chosen characteristic length and $0 \leq x_k \leq L_k$. The new variables τ , η and ξ are referred to as reduced time, stress and distance, respectively. For any junction $n_p \in \mathcal{N}$, let $\mathcal{B}_{p,in}$ ($\mathcal{B}_{p,out}$) be the set of connected branches for which the reference direction is going into (out of) the junction, and let $\tau_{f,p}$ be its time of void nucleation. Then, the scaled PDE system that describes stress evolution in the whole tree can be stated as (BC and

IC are abbreviations for boundary conditions and initial conditions, respectively)

$$\text{PDE: } \frac{\partial \eta_k}{\partial \tau} = \theta_k \frac{\partial^2 \eta_k}{\partial \xi_k^2} \quad \forall b_k \in \mathcal{B}, \quad (8a)$$

$$\text{BC: } \forall n_p \in \mathcal{N} \text{ s.t. } \tau < \tau_{f,p} \\ \sum_{b_k \in \mathcal{B}_{p,in}} w_k h_k J_{a,k}(1, \tau) = \sum_{b_k \in \mathcal{B}_{p,out}} w_k h_k J_{a,k}(0, \tau), \quad (8b)$$

$$\eta_k(1, \tau) = \eta_i(0, \tau), \quad \forall \{b_k, b_i\} \in \mathcal{B}_{p,in} \times \mathcal{B}_{p,out}, \quad (8c)$$

$$\forall n_p \in \mathcal{N} \text{ s.t. } \tau \geq \tau_{f,p} \\ \frac{\partial \eta_k}{\partial \xi_k} = -\eta_k(1, \tau) \frac{L_k}{\delta} \quad \forall b_k \in \mathcal{B}_{p,in}, \quad (8d)$$

$$\frac{\partial \eta_k}{\partial \xi_k} = \eta_k(0, \tau) \frac{L_k}{\delta} \quad \forall b_k \in \mathcal{B}_{p,out}, \quad (8e)$$

$$\text{IC: } \eta_k(\xi_k, 0) = \eta_{T,k} \quad \forall b_k \in \mathcal{B}, \quad (8f)$$

where $\theta_k = (L_c^2 D_{a,k} T_m^* / L_k^2 D_a^* T_{m,k})$, $\eta_{T,k} = \Omega \sigma_{T,k} / (k_b T_m^*)$, $D_{a,k}$ is the diffusivity, $T_{m,k}$ is the temperature and $J_{a,k}$ is the atomic flux in branch b_k

$$J_{a,k} = \frac{D_{a,k} C T_m^*}{L_k T_{m,k}} \left(\frac{\partial \eta_k}{\partial \xi_k} - \alpha_k \right), \quad (9)$$

with $\alpha_k = (q^* \rho j_k L_k / k_b T_m^*)$. Note that $\eta_k \equiv \eta_k(\xi_k, \tau)$ is the reduced stress in branch b_k at distance $0 \leq \xi_k \leq 1$ and time τ .

E. Power Grid EM analysis using EKM

There are many ways to solve the scaled PDE system (8). In [8], the authors converted (8) to an equivalent Initial Value Problem (IVP) for each tree and solved it using variable step Runge-Kutta method with Butcher tableau as given by Dormand and Prince [14]. We will now present their power grid EM checking approach, which we will refer to as the *standard approach*.

In the standard approach, the power grid lifetime is estimated as per the *mesh model* [3], in which a grid is deemed to have failed when enough voids have nucleated so that the voltage drop at a node exceeds the user provided voltage drop specification. This specification is based on the timing constraints of the underlying logic and is captured in the vector v_{th} . The temperature distribution of the grid is determined at $t = 0$ using compact thermal models [15], which gives the initial stress profile for the trees. A subset of trees, called the *active set*, is chosen such that the first void nucleation time for each tree in the active set is less than some time threshold $t = t_m$. All trees in the active set are then numerically integrated using the IVP formulation to obtain their stress as a function of position and time. Every time a void nucleates at a junction, the steady state void volume is calculated for all connected branches using (6) and the corresponding resistances are updated. If the recently formed void leads to an *early failure*, the corresponding via is removed from the power grid. Next, the node voltage drops and the temperature distribution are updated. Finally, the IVPs for all trees in the active set are reformulated using the updated boundary conditions and the integration is carried on until the next void failure. Due to increasing branch resistances, the conductivity of the grid degrades and the voltage drops eventually exceed v_{th} . The earliest time when a voltage drop violation occurs, is the TTF of the grid as per the mesh model.

In order to account for the randomness in EM degradation, the standard approach performs *Monte Carlo random sampling* to estimate the Mean Time to Failure (MTF) to within a user specified error tolerance. The time threshold t_m for selecting trees in the active set is a part of the Monte-Carlo (MC) process. It is initially set to a sufficiently high value and is updated as more mesh TTF samples are obtained from the subsequent MC iterations.

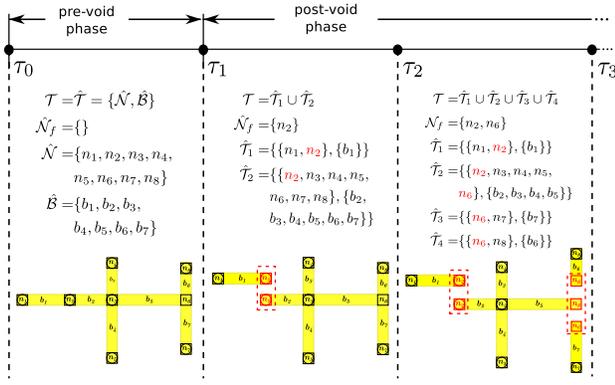


Fig. 2. Notion of subtrees for tree shown in Fig. 1.

III. STATE SPACE MODEL FOR A TREE

We will now present our approach for solving the scaled PDE system by converting it to an equivalent state-space model (SSM), which is a succession of LTI systems. Consider a tree $\mathcal{T} = \{\mathcal{N}, \mathcal{B}\}$ with $|\mathcal{N}|$ junctions and $|\mathcal{B}|$ branches. When a void nucleates at a junction, EKM conceptually treats it as a diffusion barrier for all connected branches, so that there is no material flow between them. Thus, the tree is effectively divided into separate *subtrees*. A *subtree of tree* \mathcal{T} is graph $\hat{\mathcal{T}} = \{\hat{\mathcal{N}}, \hat{\mathcal{B}}\}$ with $\hat{\mathcal{N}} \subseteq \mathcal{N}$ and $\hat{\mathcal{B}} \subseteq \mathcal{B}$. Let τ_p be the time of the p^{th} void nucleation, with $\tau_0 = 0$. For the time-span $[\tau_0, \tau_1]$, a tree has no voids. We will call this time-span as the *pre-void phase*. For all subsequent time-spans, the subtrees will have at least one failed junction that has a void. We will refer to these time-spans as the *post-void phase*.

Fig. 2 illustrates the notion of subtrees using interconnect tree of Fig. 1 as an example. In the pre-void phase, $\hat{\mathcal{T}} = \mathcal{T}$. At $\tau = \tau_1$, the first void nucleates at n_2 , and the tree is divided into two subtrees $\hat{\mathcal{T}}_1$ and $\hat{\mathcal{T}}_2$, as shown in Fig. 2. Note that n_2 appears in both subtrees and as per EKM, is treated as a diffusion barrier with a void. Similarly, at $\tau = \tau_2$, n_6 fails and the whole tree is divided into four subtrees. In general, if \mathcal{N}_f is the set of failed junctions in the tree, then the number of subtrees m can be found using $m = 1 + \sum_{n_p \in \mathcal{N}_f} (d_p - |\mathcal{N}_f|)$, where d_p is the degree of junction n_p before void nucleation.

In the following sections, we will show that for each time-span, the scaled PDE system for a subtree can be represented by an equivalent LTI system with constant inputs. We will describe *state stamps*, that are conceptually similar to *element stamps* used in SPICE for generating circuit matrices. A special case arises for the pre-void phase: the system matrix obtained using the state-stamps is singular. We will then use mass conservation to obtain a non-singular LTI system for the pre-void phase. The complete SSM for a tree in each time-span is composed of the LTI models of each of its subtrees, and is also an LTI system. For example, the LTI model of \mathcal{T} is composed of LTI models of $\hat{\mathcal{T}}_1$ and $\hat{\mathcal{T}}_2$ in $[\tau_1, \tau_2]$.

A. LTI model for a subtree

For each subtree $\hat{\mathcal{T}} = \{\hat{\mathcal{N}}, \hat{\mathcal{B}}\}$, converting the scaled PDE system to an equivalent LTI system requires discretization of the spatial domain. We uniformly discretize each branch b_k into N segments, where N is the same for all branches because all branch lengths are scaled to 1 as in (7). Then, there would be a total of $q + 1$ discretized points, where $q = N|\hat{\mathcal{B}}|$. Each discretized point is given a unique index $i \in i_0 + \{0, 1, 2, \dots, q\}$, where the offset i_0 is to ensure unique indices for all discretized points within tree \mathcal{T} . Let x_i represent the reduced stress at i^{th} discretized point in the tree. Then, using (8a), the time rate of change of x_i in branch b_k is

$$\frac{\partial x_i}{\partial \tau} = \theta_k \frac{\partial^2 x_i}{\partial \xi_k^2}. \quad (10)$$

Approximating the partial derivative with respect to ξ_k in (10) using the central difference formula [16] and solving the boundary conditions at junctions leads to the following *translated* LTI system for a subtree in the time-span $[\tau_p, \tau_{p+1})$

$$\dot{\hat{x}}(\tau - \tau_p) = \hat{\mathbf{A}}\hat{x}(\tau - \tau_p) + \hat{u}, \quad (11a)$$

$$\hat{x}(0) = \hat{x}_0, \quad (11b)$$

where $\hat{x} = [x_i] \in \mathbb{R}^{(q+1)}$ is the *state vector* of the subtree, $\hat{\mathbf{A}} = [a_{i,j}] \in \mathbb{R}^{(q+1) \times (q+1)}$ is the system matrix and $\hat{u} = [u_i] \in \mathbb{R}^{(q+1)}$ is the vector of inputs. The initial condition \hat{x}_0 is obtained from the stress profile of the tree at $\tau = \tau_p$ computed using the LTI model(s) of the previous time-span, or is equal to the thermal stress profile at $\tau = 0$.

Each state x_i contributes some non-zero entries to the i^{th} row of $\hat{\mathbf{A}}$ and \hat{u} , which we will refer to as a *state stamp*. The notion of stamps is useful to quickly assemble the LTI system for a given subtree: we start by initializing $\hat{\mathbf{A}}$ and \hat{u} to all zeros and add the stamps as we traverse through the tree. The state stamps are determined based on the location, the adjacent points and the presence or absence of a void at point i . Two points are said to be *adjacent* to each other if they are physically next to each other in a subtree. We will denote the set of indices for points adjacent to i as $\mathcal{A}(i)$. In order to simplify the presentation going forward, we define the following for any two branches $b_i, b_k \in \mathcal{B}$

$$\begin{aligned} r_{ik} &\triangleq L_i/L_k, & w_{ik} &\triangleq w_i/w_k, & \Delta\xi &\triangleq 1/N, \\ \Upsilon_k &\triangleq \theta_k N^2, & \gamma_{ik} &\triangleq r_{ki} w_{ik} D_{a,i} T_{m,k} / (D_{a,k} T_{m,i}). \end{aligned} \quad (12)$$

1) *State Stamps for branch interior*: Consider state x_i for a discretized point within branch b_k , with $\mathcal{A}(i) = \{i_1, i_2\}$. Then, the state stamps are given as

$$a_{i,i} = -2\Upsilon_k, \quad a_{i,i_1} = a_{i,i_2} = \Upsilon_k, \quad u_i = 0. \quad (13)$$

2) *State Stamps for diffusion barrier*: Consider state x_i for a diffusion barrier n_p at the beginning of branch b_k , with $\mathcal{A}(i) = \{i_1\}$. Let $\tau_{f,p}$ be the time of void nucleation at this barrier. Then, the state stamps for $\tau < \tau_{f,p}$ are

$$a_{i,i} = -2\Upsilon_k, \quad a_{i,i_1} = 2\Upsilon_k, \quad u_i = -2\Upsilon_k \Delta\xi \alpha_k, \quad (14)$$

and for $\tau \geq \tau_{f,p}$

$$a_{i,i} = -2\Upsilon_k (1 + \Delta\xi L_k / \delta), \quad a_{i,i_1} = 2\Upsilon_k, \quad u_i = 0. \quad (15)$$

Similarly, for a diffusion barrier at the end of branch b_k with $\mathcal{A}(i) = \{i_1\}$, the state stamps for $\tau < \tau_{f,p}$ are

$$a_{i,i} = -2\Upsilon_k, \quad a_{i,i_1} = 2\Upsilon_k, \quad u_i = 2\Upsilon_k \Delta\xi \alpha_k. \quad (16)$$

For $\tau \geq \tau_{f,p}$, the state stamps are exactly the same as (15).

3) *State Stamps for higher degree junctions*: Consider a state x_i for a junction n_p with degree $d_p \in \{2, 3, 4\}$ and $\mathcal{A}(i) = \{i_1, \dots, i_{d_p}\}$. Without loss of generality, we will assume that n_p is at the end of branch 1 and at the beginning of branches 2, \dots , d_p . Let $\tau_{f,p}$ be the time of void nucleation at n_p . Then, the corresponding state stamp for n_p for $\tau < \tau_{f,p}$ will be

$$a_{i,i} = -2\varrho_{1d_p} \Upsilon_1 \sum_{k=1}^{d_p} \gamma_{k1}, \quad (17a)$$

$$a_{i,i_k} = 2\varrho_{1d_p} \Upsilon_1 \gamma_{k1}, \quad k = \{1, \dots, d_p\}, \quad (17b)$$

$$u_i = 2\varrho_{1d_p} \Upsilon_1 \Delta\xi \left(\alpha_1 - \sum_{k=2}^{d_p} \gamma_{k1} \alpha_k \right), \quad (17c)$$

where $\varrho_{12} = r_{12}/(r_{12} + w_{21})$, $\varrho_{13} = (r_{12}r_{13})/(r_{12}r_{13} + r_{13}w_{21} + r_{12}w_{31})$ and $\varrho_{14} = (r_{12}r_{13}r_{14})/(r_{12}r_{13}r_{14} + r_{13}r_{14}w_{21} + r_{12}r_{14}w_{31} + r_{12}r_{13}w_{41})$.

As mentioned before, when a void nucleates at junction n_p , it generates new subtrees. Each subtree in the post-void phase will have at least one void located at the newly created *virtual* diffusion barrier. For example, for the time-span $[\tau_1, \tau_2]$ in Fig. 2, n_2 is the *virtual* diffusion barrier (with a void) for both subtrees $\hat{\mathcal{T}}_1$ and $\hat{\mathcal{T}}_2$. For any subtree, let i be the index of the discretized point where the void is present, and let $\mathcal{A}(i) = \{i_1\}$. Then the state stamps for $\hat{\mathbf{A}}$ and u are given by (15). Note that the state stamps for the pre-void phase of $\hat{\mathbf{A}}$ are also stated in [12].

Theorem 1. (properties of $\hat{\mathbf{A}}$) For a subtree $\hat{\mathcal{T}}$, let $\hat{\mathbf{A}}$ be the system matrix obtained using stamps (13)-(17). Then

- (a) All eigenvalues of $\hat{\mathbf{A}}$ have non-positive real parts.
- (b) For the pre-void phase, 0 is a simple eigenvalue of $\hat{\mathbf{A}}$.
- (c) For the post-void phase, $\hat{\mathbf{A}}$ is non-singular, with all eigenvalues having negative real parts.

The proof of this theorem is given in the appendix.

From theorem 1, $\hat{\mathbf{A}}$ is singular in the pre-void phase. This happens because the corresponding boundary conditions model it as a closed system, i.e. there is no exchange of atoms with other trees. This creates a dependency among the states x_i of a tree, which we will use in order to get a corresponding non-singular LTI system, as shown in the next section.

B. LTI system for pre-void phase

In this section, we will consider the whole tree while stating the LTI system because $\hat{\mathcal{T}} = \mathcal{T}$ in the pre-void phase. Define

$$\beta(\tau) \triangleq \sum_{i=0}^q c_i x_i(\tau), \quad (18)$$

where $q = N|\mathcal{B}|$ and the value of c_i coefficients are (\mathcal{B}_p is the set of branches connected to n_p)

$$c_i = \begin{cases} L_k w_k h_k \Delta \xi & x_i \text{ is inside branch } b_k, \\ (\Delta \xi / 2) \sum_{b_k \in \mathcal{B}_p} L_k w_k h_k & x_i \text{ is at junction } n_p. \end{cases} \quad (19)$$

Then, it can be shown using Hooke's Law (2) that in order to satisfy the conservation of mass in a tree, we must have [12]

$$\beta_0 = \sum_{i=0}^q c_i x_i(0) = \beta(\tau) \quad \forall \tau, \quad (20)$$

where the stress values at $\tau = 0$ are known from the initial condition. This gives us a linear dependence between the states so that one state can be eliminated from (11a), which makes the system matrix non-singular as it has only one zero eigenvalue. Note that we can only eliminate state x_i if $u_i = 0$. Without loss of generality, let x_0 be the state (with no input) to be eliminated. If we denote $\tilde{x} = [x_i] \in \mathbb{R}^q$ for $1 \leq i \leq q$ to be the new state vector, we can write

$$x_0(\tau) = -c^T \tilde{x}(\tau) + \beta_0 / c_0, \quad (21)$$

where $c = c_0^{-1} [c_1 \ c_2 \ \dots \ c_q]^T \in \mathbb{R}^q$. Using (21) in the LTI system (11a) for a subtree, we get q ODE equations with q independent states for the pre-void phase $[0, \tau_1]$

$$\dot{\tilde{x}}(\tau) = \tilde{\mathbf{A}} \tilde{x}(\tau) + \tilde{u}, \quad (22a)$$

$$\tilde{x}(0) = [\eta_{T,1} \ \eta_{T,2} \ \dots \ \eta_{T,q}]. \quad (22b)$$

Here, $\eta_{T,i}$ is the initial reduced thermal stress at point i , $\tilde{\mathbf{A}} \in \mathbb{R}^{q \times q}$ is the *non-singular* system matrix and $\tilde{u} \in \mathbb{R}^q$ is the input vector such that

$$\tilde{\mathbf{A}} = -\hat{a}_q c^T + \hat{\mathbf{A}}_q \text{ and } \tilde{u} = \hat{u}_q + (\beta_0 / c_0) \hat{a}_q, \quad (23)$$

with $\hat{a}_q = [a_{i,k}] \in \mathbb{R}^q$ for $1 \leq i \leq q, k = 0$, $\hat{\mathbf{A}}_q = [a_{i,k}] \in \mathbb{R}^{q \times q}$ for $1 \leq i, k \leq q$ and $\hat{u}_q = [u_i] \in \mathbb{R}^q$ for $1 \leq i \leq q$.

IV. IMPROVED SOLUTION METHOD

We will now present our improvements to the standard approach (reviewed in Section II-E) that makes it significantly faster than before. Note that unless stated otherwise, MTF and TTF refer to mesh model based MTF and TTF.

A. Dealing with Stiffness

We found from practical experience that the LTI systems (11a) and (22a) are *stiff* systems. An LTI system with all negative eigenvalues (which is the case for us) is said to be *stiff* if the ratio of its largest to smallest magnitude eigenvalue is very large. We observed this ratio to be of the order of $10^9 - 10^{10}$ for many of the system matrices. Thus, an explicit method like Runge-Kutta is not suitable for solving these systems. Instead, we use Backward Differentiation Formulas (BDFs) to numerically integrate the LTI systems.

BDFs are a type of linear multi-step method that are particularly suited to solve *stiff* ODE systems. Suppose we wish to numerically integrate a stiff ODE system $\dot{z} = f(z, \tau)$, where $z(\tau) \in \mathbb{R}^q$ is a function of independent variable $\tau \in \mathbb{R}$. Let z_n denote the solution computed by the numerical method that approximates the true solution $z(\tau_n)$, i.e. $z_n \approx z(\tau_n)$. A *variable coefficient* p -step BDF method, denoted by BDF $_p$, uses p previous solution(s) (τ_n, z_n) , (τ_{n-1}, z_{n-1}) , \dots , $(\tau_{n-p+1}, z_{n-p+1})$ to compute the solution at the next time point and is defined as

$$h_{n+1} \mathbf{b}_{-1} f(z_{n+1}, \tau_{n+1}) = z_{n+1} + \sum_{i=0}^{p-1} \mathbf{a}_i z_{n-i}, \quad (24a)$$

$$\epsilon_{\text{PLTE}} = \varsigma_p z^{(p+1)}(\tau_n), \quad (24b)$$

where z_{n+1} is the solution at time $\tau_{n+1} = \tau_n + h_{n+1}$ that needs to be computed. ϵ_{PLTE} is the *principal local truncation error* (PLTE) and estimates the error in the computed solution z_{n+1} . $z^{(p+1)}$ denotes the $(p+1)^{\text{th}}$ derivative of z with respect to τ . This method is referred as a *variable coefficient* method because the scalar coefficients $\mathbf{b}_{-1}, \mathbf{a}_0, \dots, \mathbf{a}_{p-1}$ and error constant ς_p are not constant, but a function of the p time steps $h_{n+1}, \dots, h_{n-p+2}$. This makes it easier to implement a change in time-step without using interpolation methods. The formulas for the coefficients and the error constant can be determined using Newton divided differences or linear difference operator [17] [18], and is omitted due to lack of space. BDFs with $p > 6$ are known to be unstable and BDF1 is *Backward Euler*.

A major drawback of an implicit method like BDF is the requirement of Newton iterations to solve (24a) for a non-linear $f(\cdot)$. Fortunately, in our case, the ODE system we are trying to solve is an LTI system with constant inputs. Define $z \triangleq \mathbf{x} + \mathbf{A}^{-1}u$, where we have dropped the accents to avoid clutter. Then, for each time-span, we only need to solve a homogeneous LTI system $\dot{z} = f(z, \tau) = \mathbf{A}z(\tau)$ for each tree. Thus, the solution z_{n+1} at next time-point τ_{n+1} is easily obtained by doing a linear system solve of the following equation

$$(h_{n+1} \mathbf{b}_{-1} \mathbf{A} - \mathbf{I}) z_{n+1} = \sum_{i=0}^{p-1} \mathbf{a}_i z_{n-i}, \quad (25)$$

where \mathbf{I} is the identity matrix. Another factor that affects the performance of BDF methods is the estimation of ϵ_{PLTE} , as it requires the calculation of $(p+1)^{\text{th}}$ derivative of z , which is usually not available or is difficult to compute. However, given that we have a homogeneous LTI system, it is straightforward to compute the $(p+1)^{\text{th}}$ derivative. Thus, the PLTE can be stated as

$$\epsilon_{\text{PLTE}} = \varsigma_p \mathbf{A}^{p+1} z(\tau_n) \approx \varsigma_p \mathbf{A}^{p+1} z_n, \quad (26)$$

We use ϵ_{PLTE} for error control, i.e. keeping the solution within the absolute and relative error bounds provided by the user and in deciding the next time step to be taken. For a given value of z_n , larger ϵ_{PLTE} denotes higher error.

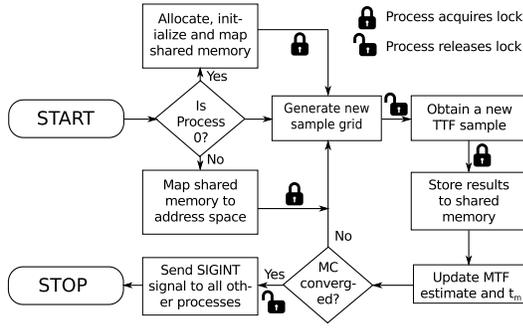


Fig. 3. Workflow for each process in the parallel implementation.

B. Updating voltage drops using PCG

Every time a void nucleates, the node voltage drops need to be updated in order to check for voltage drop violation. As the power grid size increases, updating voltage drops due to changing branch resistances becomes more computationally expensive, which limited the scalability of previous approaches. In our approach, we update voltage drops using Preconditioned Conjugate Gradient (PCG) method. At $t = 0$, we have to factorize the conductance matrix in order to find the initial voltage drops. We use this initial factorization as a pre-conditioner for our Conjugate Gradient (CG) method. This works because the perturbation in \mathbf{G} due to void nucleations is minimal, hence the factorization at $t = 0$ acts as an excellent incomplete factorization for $\mathbf{G}(t)$, which results in very fast convergence within a few iterations.

C. Parallelization

Due to the random nature of EM degradation, estimating power grid MTF requires MC random sampling approach, whereby each MC iteration generates one TTF sample and the average of all TTF samples gives the power grid MTF. The stopping criteria (number of samples required) is determined such that we have $(1 - \zeta) \times 100\%$ confidence that the relative error in MTF estimation is less than a user provided relative error threshold ϵ_{MC} [19]. We generate at least 30 TTF samples before checking the stopping criteria. In each MC iteration, we assign randomly generated diffusivities to all branches in the power grid, which effectively generates a new *sample grid*. Estimating TTFs of sample grids in different MC iterations are independent of each other, and thus it can be parallelized. In our implementation, we use a multi-process architecture, with each process bound to a separate core, to carry out several MC iterations simultaneously. The workflow for each process is shown in Fig. 3. These processes use shared memory for inter-process communication. The first process allocates and initializes the shared memory object, which contains 1) a random number generator to be used for generating sample grids, 2) a table to store the TTF samples (and other data) generated from all processes, 3) the time threshold t_m required for selecting the *active set*, 4) process IDs for all processes using the shared memory and 5) several read-write locks to synchronize the read/write accesses to the shared memory. All subsequent processes map this memory space to their own address space.

Whenever a process obtains a new TTF sample by completing a MC iteration, it writes the results to the shared table, updates the estimated MTF and t_m based on TTF samples obtained so far from all processes and checks if the stopping criteria has been satisfied. If its not satisfied, this process uses the shared random number generator to generate a sample grid and starts a new MC iteration. On the other hand, if the stopping criteria is satisfied, this process sends an interrupt to all other processes to signal the end of the task and then stops. Any time a process receives an interrupt, it assumes

TABLE I
RUNTIME COMPARISON OF BDF SOLVERS

Grid Name	t_{BDF2} (mins)	t_{BDF3} (mins)	t_{BDF4} (mins)	t_{BDF5} (mins)	t_{BDF6} (mins)
ibmpg1	0.588	0.633	0.756	1.460	3.954
ibmpg2	1.235	1.442	1.740	4.803	16.116
ibmpg5	1.944	1.981	2.268	4.092	11.254
ibmpgnew1	4.708	4.814	5.936	15.534	48.023
ibmpgnew2	3.504	3.546	4.111	10.792	35.953

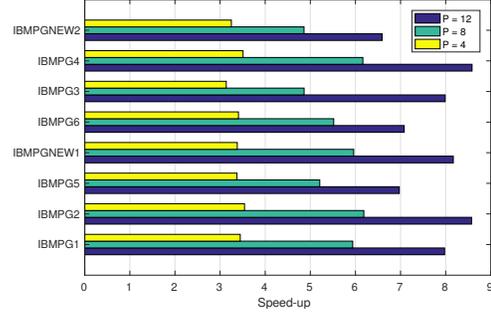


Fig. 4. Bar chart showing speed-ups with using 4, 8 and 12 parallel processes.

that the stopping criteria has been satisfied in some other process, and thus stops execution. The last process to stop deallocates the shared memory object.

V. EXPERIMENTAL RESULTS

A C++ implementation was written to test the proposed methodology. Two types of test grids were used to verify our approach: IBM power grids [20] and internal grids. The internal grids are non-uniform grids, synthesized as per user specifications, including grid dimensions, metal layers, pitch, and width per layer. The current sources are randomly placed on the grid. The technology specifications are consistent with 1V 45 nm CMOS technology. The grids named PG1-PG5 are internal grids. We use a 12 core 3GHz Linux machine with 128 GB of RAM to perform all simulations. The interconnect material is assumed to be Copper, so that the following parameters are used in our EM model: $B = 1.35 \times 10^{11}$ Pa, $\Omega = 1.66 \times 10^{-29}$ m³, $k_b = 1.38 \times 10^{-23}$ J/K, $q^* = 8.0109 \times 10^{-19}$ C, $\sigma_{th} = 600 \times 10^6$ Pa and $\delta = 10^{-9}$ m. An ambient temperature of 300K is used for all simulations. The LTI models are generated with $N = 16$ discretizations per branch. We use a relative tolerance of 10^{-3} and an absolute tolerance of 10^{-6} for the BDF solver. For MTF estimation, we use $\zeta = 0.05$ (95% confidence bounds) and $\epsilon_{MC} = 0.1$ (maximum relative error threshold of 10%).

First, we compare the performance of BDF2-BDF6 solvers, in terms of time taken for mesh MTF estimation. The results are shown in table I. Clearly, BDF2 is fastest of all with BDF3 being a close second. Overall, BDF2 is 1.06x faster than BDF3, 1.25x faster than BDF4, 2.97x faster than BDF5 and 9.2x faster than BDF6. The reason for the slowdown of BDF4-BDF6 solvers can be attributed to the calculation of PLTE in (26) : BDF p requires \mathbf{A}^{p+1} to calculate the PLTE, which results in higher error norms for larger p values (in our case, $\|\mathbf{A}^{p_1} z\| \geq \|\mathbf{A}^{p_2} z\|$ iff $p_1 \geq p_2$). This forces the BDF4-BDF6 solvers to take smaller time-steps in order to maintain the solution accuracy.

Next, we compare the performance of our improved solution approach with the standard approach in [8]. Table II compares the the series and mesh MTF estimated 1) with the standard approach using the RK solver [14] (μ_s^{RK} and μ_m^{RK}) and 2) with the improved approach using the BDF2 solver (μ_s^{BDF2} and μ_m^{BDF2}). Although we

TABLE II
PERFORMANCE COMPARISON OF RUNGE-KUTTA SOLVER AND BDF2 SOLVER

Power Grid				Standard Approach [8]			Proposed Approach with BDF2					
Grid Name	# nodes	#branches	#trees	μ_s^{RK} (yrs)	μ_m^{RK} (yrs)	t_{RK} (hrs) [†]	μ_s^{BDF2} (yrs)	μ_m^{BDF2} (yrs)	t_{BDF2} (hrs) [†]	ϵ_s (%) [§]	ϵ_m (%) [§]	$\frac{t_{RK}}{t_{BDF2}}$
ibmpg1	6K	11K	709	3.32	7.11	0.033	3.28	7.06	0.01	1.17	0.61	3.40x
ibmpg2	62K	61K	462	6.69	11.94	0.27	6.67	11.95	0.02	0.33	0.09	12.95x
ibmpg3	410K	401K	8.1K	4.58	7.02	2.77	4.50	6.83	0.07	1.82	2.75	42.02x
ibmpg4	475K	465K	9.6K	8.97	16.88	2.68	8.97	16.81	0.11	0.00	0.46	24.68x
ibmpg5	249K	496K	2K	4.56	6.20	0.14	4.48	6.39	0.03	1.62	2.99	4.33x
ibmpg6	404K	798K	10.2K	5.67	11.41	1.83	5.60	11.22	0.24	1.23	1.66	7.67x
ibmpgnew1	316K	698K	19.5K	3.81	13.16	0.43	3.84	13.29	0.08	0.90	0.95	5.43x
ibmpgnew2	718K	698K	19.5K	4.72	7.36	0.73	4.67	7.44	0.06	1.00	1.03	12.42x
PG1	560K	558K	2.6K	4.43	17.40	2.77	4.44	17.22	0.04	0.33	1.08	77.11x
PG2	1.2M	1.2M	5.6K	3.55	10.25	2.36	3.64	10.37	0.12	2.52	1.19	20.46x
PG3	1.6M	1.6M	6.9K	3.82	8.55	2.25	3.87	8.52	0.06	1.28	0.26	40.55x
PG4	2.6M	2.6M	12.2K	-	-	-	3.34	14.76	0.15	-	-	-
PG5	4.1M	4.1M	12.7K	-	-	-	4.39	9.21	0.14	-	-	-
Average										1.11	1.19	22.82x

[†] t_{RK} and t_{BDF} denote the wall-time(s) for Runge-Kutta and BDF solvers, respectively.

[§] The percentage error in results of BDF solver with respect to the results from Runge-Kutta Solver.

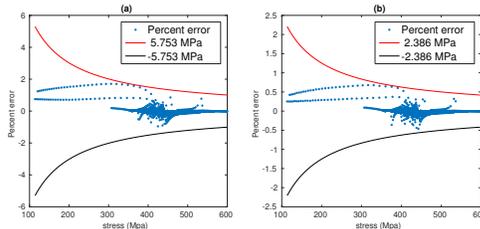


Fig. 5. Percentage error plot between solutions obtained using (a) $N = 10$ and (b) $N = 16$ vs. $N = 64$.

use the mesh model for estimating power grid MTF, we also report the MTF as per the *series* model, which is generated as a byproduct. For a fair comparison, the MC iterations for both approaches are parallelized using 12 parallel processes. As can be seen, the average relative error in series and mesh MTF estimation is 1.1% and 1.2%, respectively. Thus, the MTF estimated by both approaches is almost the same. However, due to the optimizations we implemented, our p/g EM checking methodology is $\sim 23x$ faster than the standard approach and has a runtime of only 8.2 minutes for a 4.1 M node grid. For some of the bigger grids, the results for the standard approach are not shown as it ran out of memory.

We also report the speed-ups obtained with the multi-process architecture in Fig. 4. All speedups are calculated based on sequential runtime in which all MC iterations are performed in a single process. We obtained an average speed-up of 3.38x, 5.6x and 7.75x using 4, 8 and 12 parallel processes, respectively. The reason for this sub-linear speedup is due to ‘slow’ update of t_m in parallel version as compared to the sequential version. Recall that the initial value of t_m is very high, and it is updated as more TTF samples are obtained. A higher value of t_m leads to a longer runtime for a given MC iteration. In the sequential version, only the first few MC iterations run with the initial high value of t_m , after which all subsequent iterations use updated values of t_m . On the other hand, for a parallelized version with P processes, the first P MC iterations will run with the initial high value of t_m , which reduces the speed-ups obtained.

In Fig. 5, we use an interconnect tree form *ibmpg2* to compare the solutions obtained using different discretizations. A finer discretization leads to a larger LTI system that gives accurate

results but takes more time to solve and vice versa. We will use LTI_N to denote an LTI model generated with N discretization per branch. The best solution is considered to be the one obtained with LTI_{64} . As compared to the best solution, the maximum percentage error for solutions obtained using LTI_{16} and LTI_{10} are 0.6% and 1.7% respectively, and the corresponding speed-ups are 7.7x and 11.17x. Given the accuracy vs speed trade-off, we use $N = 16$ discretization per branch for MTF estimation.

VI. CONCLUSION

We proposed a fast and scalable methodology for EM assessment of on-die power grids using previously published physics-based models. We improved the computational speed by converting the PDE system for a tree to an equivalent LTI system and solving it using an optimized BDF solver. Further speedup is obtained by using preconditioned CG for updating the voltage drops and parallelizing the code using a multi-process architecture. Overall, our proposed methodology is $\sim 23x$ faster as compared to previously published work, and is suitable for very large power grids.

REFERENCES

- [1] A. S. Oates, “Interconnect reliability challenges for technology scaling: A circuit focus,” in *2016 IEEE Int. Interconnect Tech. Conf. / Adv. Metallization Conf. (IITC/AMC)*, May 2016, pp. 59–59.
- [2] J. Warnock, “Circuit design challenges at the 14nm technology node,” in *Design Automation Conf. (DAC)*, June 2011, pp. 464–467.
- [3] S. Chatterjee *et al.*, “Redundancy-Aware Electromigration Checking for Mesh Power Grids,” in *IEEE/ACM Int. Conf. on Comput. Aided Design*, San Jose, CA, Nov. 2013, pp. 540–547.
- [4] X. Huang *et al.*, “Physics-based Electromigration Assessment for Power Grid Networks,” in *Design Automation Conf.*, June 2014, pp. 1–6.
- [5] M. A. Korhonen *et al.*, “Stress evolution due to electromigration in confined metal lines,” *J. Appl. Phys.*, vol. 73, no. 8, pp. 3790–3799, 1993.
- [6] D.-A. Li *et al.*, “A method for improving power grid resilience to electromigration-caused via failures,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 1, pp. 118–130, Jan 2015.
- [7] S. P. Hau-Riege *et al.*, “Experimental characterization and modeling of the reliability of interconnect trees,” *J. Appl. Phys.*, vol. 89, no. 1, pp. 601–609, 2001.

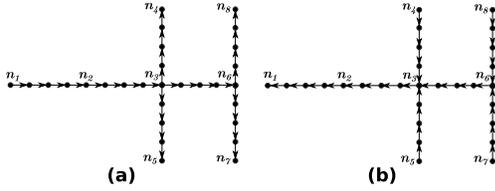


Fig. 6. (a) Graph $\mathcal{G}(\mathcal{T})$ and (b) the converse $\mathcal{G}'(\mathcal{T})$ for interconnect tree shown in Fig. 1, with $N = 4$.

- [8] S. Chatterjee *et al.*, “Fast Physics-Based Electromigration Checking for On-Die Power Grids,” in *IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD)*, Nov 2016, pp. 110:1–110:8.
- [9] B. Li, J. Gill, C. Christiansen, T. Sullivan, and P. S. McLaughlin, “Impact of via-line contact on cu interconnect electromigration performance,” in *IEEE Int. Rel. Phys. Symp.*, April 2005, pp. 24–30.
- [10] L. Ting *et al.*, “AC electromigration characterization and modeling of multilayered interconnects,” in *IEEE Int. Rel. Phys. Symp.*, March 1993, pp. 311–316.
- [11] J. Lloyd *et al.*, “The electromigration failure distribution: The fine-line case,” *J. Appl. Phys.*, vol. 69, no. 4, pp. 2117–2127, Feb 1991.
- [12] S. Chatterjee *et al.*, “Power Grid Electromigration Checking using Physics-Based Models,” *IEEE Trans. on Comput.-Aided Design of Integrated Circuits and Systems*, no. 99, 2017.
- [13] V. Sukharev *et al.*, “Postvoiding stress evolution in confined metal lines,” *IEEE Transactions on Device and Materials Reliability*, vol. 16, no. 1, pp. 50–60, March 2016.
- [14] J. Dormand and P. Prince, “A family of embedded runge-kutta formulae,” *Journal of Computational and Applied Mathematics*, vol. 6, no. 1, pp. 19 – 26, 1980.
- [15] W. Huang *et al.*, “Hotspot: a compact thermal modeling methodology for early-stage VLSI design,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 5, pp. 501–513, May 2006.
- [16] W. Schiesser, *Computational Mathematics in Engineering and Applied Science: ODEs, DAEs, and PDEs*. Taylor & Francis, 1993.
- [17] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, 2nd ed. Springer-Verlag Berlin Heidelberg, 1996.
- [18] F. N. Najm, *Circuit Simulation*. Wiley-IEEE Press, 2010.
- [19] I. Miller *et al.*, *Probability and Statistics for Engineers*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1990.
- [20] S. R. Nassif, “Power grid analysis benchmarks,” in *ASP-DAC*, 2008, pp. 376–381.
- [21] R. A. Horn *et al.*, Eds., *Matrix Analysis*. New York, NY, USA: Cambridge University Press, 1986.

APPENDIX

Definition 1. A square matrix $\mathbf{M} = [m_{i,k}] \in \mathbb{R}^{n \times n}$ induces a directed graph $\Gamma(\mathbf{M})$ whose vertices are $\{0, 1, 2, \dots, n-1\}$, and whose directed edges are $i \rightarrow k$ if $m_{i,k} \neq 0$.

Definition 2. If there is a directed path in the graph from every vertex to every other vertex, then the graph is *strongly connected*.

Theorem 2. A matrix \mathbf{M} is said to be *irreducible* if $\Gamma(\mathbf{M})$ is strongly connected (proved in [21]).

Lemma 3. $\hat{\mathbf{A}}$ is an irreducible matrix.

Proof. For any subtree $\hat{\mathcal{T}}$, consider two weighed directed graphs $\mathcal{G}(\hat{\mathcal{T}})$ and $\mathcal{G}'(\hat{\mathcal{T}})$, where the discretized points are the vertices and any two adjacent points have an edge between them. In $\mathcal{G}(\hat{\mathcal{T}})$, the direction of each edge is the same as reference direction assigned to the branches and in $\mathcal{G}'(\hat{\mathcal{T}})$, the direction of every edge is opposite to the assigned reference direction, so that $\mathcal{G}'(\hat{\mathcal{T}})$ is the *converse* of $\mathcal{G}(\hat{\mathcal{T}})$. For $\mathcal{G}(\hat{\mathcal{T}})$ ($\mathcal{G}'(\hat{\mathcal{T}})$), the weight of a directed edge $i \rightarrow k$ ($k \rightarrow i$) between adjacent vertices is equal to $a_{i,k}$ ($a_{k,i}$), which can be determined using (13)–(17). Fig. 6 shows the graphs $\mathcal{G}(\hat{\mathcal{T}})$ and $\mathcal{G}'(\hat{\mathcal{T}})$ for interconnect tree of Fig. 1. A weighed adjacency matrix

can be used to represent the connectivity of a graph. If a graph has n nodes, then the adjacency matrix will be of size $n \times n$. The entry in the i^{th} row and k^{th} column of a weighed adjacency matrix is equal to weight of edge $i \rightarrow k$ if it exists, otherwise it is 0. Let $\mathbf{W} = [w_{i,k}]$ and $\mathbf{W}' = [w'_{k,i}]$ be the weighed adjacency matrices for graphs $\mathcal{G}(\hat{\mathcal{T}})$ and $\mathcal{G}'(\hat{\mathcal{T}})$, respectively. Then, $w_{i,k} = a_{i,k}$ and $w'_{k,i} = a_{k,i}$. Now, we can write $\hat{\mathbf{A}}$ as:

$$\hat{\mathbf{A}} = \mathbf{W} + \hat{\mathbf{A}}_d + \mathbf{W}' \quad (27)$$

where $\hat{\mathbf{A}}_d$ is simply a diagonal matrix whose i^{th} diagonal entry is equal to the i^{th} diagonal entry of $\hat{\mathbf{A}}$. From (27), it is clear that $\Gamma(\hat{\mathbf{A}}) = \mathcal{G}(\hat{\mathcal{T}}) \cup \mathcal{G}'(\hat{\mathcal{T}}) \cup \Gamma(\hat{\mathbf{A}}_d)$, so that for any two adjacent vertices i and k , $\Gamma(\hat{\mathbf{A}})$ has both edges $i \rightarrow k$ and $k \rightarrow i$. Thus, in $\Gamma(\hat{\mathbf{A}})$ there is always a path from every vertex to every other vertex. Hence $\Gamma(\hat{\mathbf{A}})$ is strongly connected and $\hat{\mathbf{A}}$ is irreducible. \square

Definition 3. A matrix $\mathbf{M} = [m_{i,k}]$ is said to be *non-negative* if $m_{i,k} \geq 0 \forall i, k$.

Definition 4. Let $\lambda_0, \lambda_1, \dots, \lambda_{n-1}$ be the (real or complex) eigenvalues of a matrix $\mathbf{M} = [m_{i,k}] \in \mathbb{R}^{n \times n}$. Then the *spectral radius* is defined as: $\kappa(\mathbf{M}) = \max\{|\lambda_0|, |\lambda_1|, \dots, |\lambda_{n-1}|\}$.

Definition 5. A matrix $\mathbf{M} = [m_{i,k}] \in \mathbb{R}^{n \times n}$ is said to be *irreducibly diagonally dominant* if \mathbf{M} is irreducible, all its rows are diagonally dominant and there is at least one row i that satisfies $|m_{i,i}| > \sum_{k=0, k \neq i}^n |m_{i,k}|$.

Theorem 4. (Perron-Frobenius theorem) Let $\mathbf{M} \in \mathbb{R}^{n \times n}$ and suppose that \mathbf{M} is irreducible and non-negative. Then $\kappa(\mathbf{M}) > 0$ is a simple eigenvalue of \mathbf{M} with an associated positive eigenvector (proved in [21]).

Theorem 5. An irreducibly diagonally dominant matrix is non-singular (proved in [21]).

A. Proof for Theorem 1

(a) From Gershgorin disc theorem [21], all eigenvalues of $\hat{\mathbf{A}}$ are located in the union of $q+1$ discs:

$$\bigcup_{i=0}^q \{z \in \mathbb{C} : |z - a_{i,i}| \leq \sum_{k=0, k \neq i}^q |a_{i,k}|\} \equiv G(\hat{\mathbf{A}}) \quad (28)$$

From construction, we always have $|a_{i,i}| \geq \sum_{k=0, k \neq i}^q |a_{i,k}|$ and $a_{i,i} < 0 \forall i$. Thus, $G(\hat{\mathbf{A}})$ would lie in the left-half of the complex plane touching the imaginary axis at the origin. Hence, all eigenvalues of $\hat{\mathbf{A}}$ have non-positive real parts.

(b) In the pre-void phase, all the row sums in $\hat{\mathbf{A}}$ are zero. Thus, we must have at least one eigenvalue at 0. Define $\hat{\mathbf{A}}_c \triangleq \hat{\mathbf{A}} + c\mathbf{I}$, where $c = \max\{|a_{i,i}|\}$. Then, clearly $\hat{\mathbf{A}}_c$ is non-negative and irreducible because it is obtained by only adding c to the diagonal entries of $\hat{\mathbf{A}}$ (non-diagonal elements are unaffected). Also, if $\lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_q$ are the eigenvalues of $\hat{\mathbf{A}}$ (including multiplicities), then $\lambda_0 + c \geq \lambda_1 + c \geq \dots \geq \lambda_q + c$ are the eigenvalues of $\hat{\mathbf{A}}_c$. From part (a), we know that all eigenvalues are non-positive, thus $\lambda_0 = 0$ is the largest eigenvalue of $\hat{\mathbf{A}}$ and $\lambda_0 + c = c$ is the largest eigenvalue of $\hat{\mathbf{A}}_c$. But, since $\hat{\mathbf{A}}_c$ is non-negative and irreducible, we must have $\kappa(\hat{\mathbf{A}}_c) = c$. By Perron-Frobenius theorem, c is a simple eigenvalue of $\hat{\mathbf{A}}_c$. Hence, 0 is a simple eigenvalue of $\hat{\mathbf{A}}$.

(c) In the post-void phase, the system matrix $\hat{\mathbf{A}}$ will have at least one row i that satisfies $|a_{i,i}| > \sum_{k=0, k \neq i}^q |a_{i,k}|$ [from (15)]. Hence, $\hat{\mathbf{A}}$ is irreducibly diagonally dominant and non-singular. Also, from part (a), all eigenvalues have non-positive real parts. Since 0 cannot be an eigenvalue of $\hat{\mathbf{A}}$, all eigenvalues of $\hat{\mathbf{A}}$ will have negative real parts. \square