Gummel-cycle Algebraic Multigrid Preconditioning for Large-scale Device Simulations

Hiroo Koshimoto*, Hisashi Ishimabushi†, Jaehyun Yoo†,
Yasuyuki Kayama*, Satoru Yamada*, Uihui Kwon† and Dae Sin Kim†
*Samsung R&D Institute Japan (SRJ)
Yokohama 230-0027, Japan
†Data and Information Technology Center, Samsung Electronics Co., Ltd.
Gyeonggi-do 18448, Korea

***Abstract*—It has been proven that the multigrid method is promissing on large-scale scientific simulations. However there still remains some difficulties on applying the multigrid method, which is the system of systems such as FEM on stress analysis or coupled PDEs. Above all, the drift-diffusion model widely used in the device modeling is a typical case belonging to the problems. Because the model has a tight coupling between the electrostatic field and the carrier movements and this property prevents the multigrid method from working effectively. In this paper, we propose a technique to apply the multigrid method to the drift-diffusion model. The technique consists of reflection process between systems coupled in the equation. Consequently the technique helps to solve large-scale device simulations. We show the case of power devices.**

***Index Terms*—power device, device simulation, drift–diffusion, multigrid, large-scale**

I. Introduction

The drift–diffusion (DD) model has been widely used in the device modeling and is still the fundamental tool on devices not focusing on downsizing such as power devices. And many methods have been developed to solve the DD model. However, turning to power devices, the difficulty of large-scale device structures is remaining. In particular, it is difficult for 2D simulations to predicate carrier behaviors caused by dynamic avalanche capability or electrostatic discharge. In such a situation, 3D simulations with large-scale device structures are required recently.

On power devices high biased electric fields shift carries a lot, so a fine discrete mesh is required to find a good approximation of carrier profiles, and besides the coefficient matrices in simulations tend to be ill conditioned and need a lot of fill-ins for robust and fast convergence. Consequently, it is crucial how to handle large memory requirements for power devices. There are two known approacehes, one is the domain decomposition method [5] [6], and another is the multigrid method [2].

We focus on the multigrid method and propose a technique to apply the method to the Newton method on device simulations. The Newton method on device simulations treats the coupled DD model and the strong coupling factors make the equations ill conditioned. The conventional multigrid methods can not handle this difficulties arising from a system of systems like FEM on stress analysis. The Gummel iteration [1] plays an important role in our technique.

II. Drift–Diffusion Model

The DD model in the steady analysis is:

$$\begin{cases} E(\psi,n,p) := -\nabla\cdot(\varepsilon\nabla\psi) - q\,(p-n+C) = 0 \\ F(\psi,n,p) := -\nabla\cdot j_n + qR = 0 \\ G(\psi,n,p) := \nabla\cdot j_p + qR = 0 \end{cases}$$

where ψ is the electrostatic potential, n and p are the electron density and the hole density, ε is the dielectrical permittivity depending on materials, C is the doping concentration, R is the total recombination rate and q is the elementary charge. $j_{n,p}$ are the carrier current densities as following:

$$\begin{cases} j_n = q\,(D_n\nabla n - \mu_n n\nabla\psi) \\ j_p = -q\,(D_p\nabla p + \mu_p p\nabla\psi) \end{cases}$$

Here $\mu_{n,p}$ are the carrier mobilities and $D_{n,p}$ are the carrier diffusivities. R and $\mu_{n,p}$ are nonnlinear models. Besides we apply the Scharfetter–Gummel scheme to discretize F, G. Thus they consist of nonlinear PDEs. Therefore we employ the Newton method to find ψ, n, p defined on a given discrete mesh Ω.

As disscussed in O. Schenk *et al*. [3], solving the DD model by the Newton method introduces highly ill-conditioned and significantly more demanding unsymmetric sparse matrices. We describe its coefficient matrix J and the vectrized function values f formaly here:

$$J = \begin{pmatrix} E_\psi & E_n & E_p \\ F_\psi & F_n & F_p \\ G_\psi & G_n & G_p \end{pmatrix}, f(\psi,n,p) = \begin{pmatrix} E(\psi,n,p) \\ F(\psi,n,p) \\ G(\psi,n,p) \end{pmatrix}$$

$E_\square, F_\square$ and $G_\square$ mean the derivatives of each equation of the variable $\square$, for instance $E_\psi = \frac{\partial E}{\partial \psi}$. The linear equation $J\delta = -f(\psi,n,p)$ arises to update nonlinear iterative solutions in the Newton method. The direct methods could solve these ill-conditioned matrices robustly. However its huge memory consumption is an obstacle for large-scale device simulations. So far, many practical device simulations use the sparse iterative solvers with preconditioners based on the incomplete LU-factorization. Unfortunately these preconditioners need pretty much fill-ins to solve ill-conditioned matrices. The preconditioned iterative solvers also have the large-scale difficulty.

In scientific simulations dominated by diffusion or the Poisson equation, the multigrid method [4] could overcome the difficulty of solving large-scale matrices without computing fill-ins. Instead of computing fill-ins, the multigrid method computes errors defined on a mesh coarsen from the original (fine) mesh to retrieve the factors converging slowly, called smooth errors. However, the matrix J is composed of 3 different systems and it is hard to construct a coarse mesh corresponding to smooth errors. We propose a technique similar to the Gummel iteration to overcome this difficulty.

III. Gummel-cycle Algebraic Multigrid Method

We employ the GMRES method to solve the linear equation $J\delta = -f(\psi, n, p)$ and use the algebraic multigrid method as its preconditioning. For simplicity, we denote the equation as $Ax = b$. The preconditioning K is applied with to compute $w_{k+1} = AK^{-1}v_k$ where w_{k+1} is a new vector of the Hessenberg matrix in the k-iteration and v_k is the computed vectors of the Hessenber matrix. We describe the procedure of applying the algebraic multigrid method in a way limited to preconditioning. So we rewrite $Kw_{k+1} = v_k$ as $Ax = b$ after this.

A. Algebraic Multigrid Method (AMG)

A typical algebraic multigrid iteration solving $Ax = b$ is: (1) Relax a iterative solution x_k^h on the fine mesh Ω^h, (2) Restrict a residual vector $r_k^h = b - Ax_k^h$ to $r_k^H = Rr_k^h$ defined on the coarse mesh Ω^H, (3) Relax an error e_k^H on Ω^H by solving $A^H e_k^H = r_k^H$, (4) Prolongate the error to $e_k^h = Pe_k^H$ defined on the fine mesh Ω^h, (5) Update the iterative solution by $x_{k+1}^h = x_k^h + e_k^h$. Here R is a restriction map: $\Omega^h \to \Omega^H$ and P is a prolongation map: $\Omega^H \to \Omega^h$. For better convergence we expect e_k^H would be an approximation of the smooth error defined on the coarse mesh Ω^H. However it is difficult for systems equation like the DD model because we do not know how to coarse coupled multiple variables.

B. Co-cycle

The complemental procedure: co-cycle is introduced. We suppose that we solve a system composed of systems $Mx = b$:

$$M = \begin{pmatrix} A & A_B \\ B_A & B \end{pmatrix}, b = \begin{pmatrix} b_A \\ b_B \end{pmatrix}$$

Here, the sub systems A and B are defined on the same mesh Ω^h, however, on applying the algebraic multigrid methods, they have two different coarse meshes: Ω_A^H and Ω_B^H. The co-cycle $A^H \Rightarrow B^H$ resolves this discrepancy by following steps(Fig. 1): (1) Prolongate the error e_A^H accompanied with A^H to $e_A^h = P_A e_A^H$ on the fine mesh Ω^h, (2) Map the error e_A^h to the right hand side of the B system, $r_B^h = b_B^h - B_A e_A^h$, (3) Restrict the residual r_B^h to $r_B^H = R_B r_B^h$ on the coarse mesh Ω_B^H, (4) Relax the error e_B^H on Ω_B^H by solving $B^H e_B^H = r_B^H$. On deeper coarse levels than 2, prolongation and restriction steps are applied recursively until an error defined on each coarse mesh could be mapped by the off diagonal sub matrix A_B or B_A. This co-cycle is associative, and we could get the cyclic errors by repeated applications. For instance, the recursive error e_A^H could be given by $A^H \Rightarrow B^H \Rightarrow A^H$. The co-cycle could make convergence of the algebraic multigrid method stabilized however it is a time consuming process because it requires several matrix–vector multiplications depend on the depth. We show the trade off later.

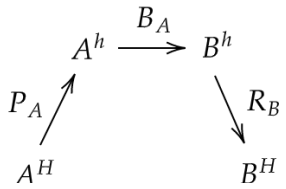

Fig. 1. Multigrid Co-cycle

C. Gummel-cycle AMG

Our approach is, defining an independent coarse mesh $\Omega_\square^H$ for each variable ψ, n, p respectively, and employing the co-cycle to get self-consistent smooth errors. The former trick enables the multigrid method to work well for each system: $E_\psi x_\psi = b_\psi, F_n x_n = b_n$ and $G_p x_p = b_p$. The latter trick recovers the components of smooth errors missed by each system. The overview of the algorithm is Algorithm1.

Algorithm 1 Gummel cycle Algebraic Multigrid

Input: A, b
Output: x
Initialisation :
Build coarsen coefficient submatrices: E_ψ^H, F_n^H, G_p^H
Prepare retrictions $R_\square$, prolongations $P_\square$ for ψ, n, p
LOOP until : $\frac{\|Ax-b\|}{\|b\|} \leq \varepsilon$
  Pre-smoothing: Relax a fine error e_ψ^h
  Restrict a residual $r_\psi^H = R_\psi \left(b_\psi^h - E_\psi^h e_\psi^h\right)$
  Pre-smoothing: Relax a coarse error e_ψ^H
  for $i = 1$ to N **do**
    Resolve the co-cycle: $E_\psi^H \Rightarrow F_n^H \Rightarrow E_\psi^H$
    Resolve the co-cycle: $E_\psi^H \Rightarrow G_p^H \Rightarrow E_\psi^H$
  end for
  Post-smoothing: Relax a coarse error e_ψ^H
  Prolongate and update an error $e_\psi^h \leftarrow e_\psi^h + P_\psi e_\psi^H$
  Post-smoothing: Relax a fine error e_ψ^h
  for $i = 1$ to N **do**
    Resolve the co-cycle: $E_\psi^h \Rightarrow F_n^h \Rightarrow E_\psi^h$
    Resolve the co-cycle: $E_\psi^h \Rightarrow G_p^h \Rightarrow E_\psi^h$
  end for

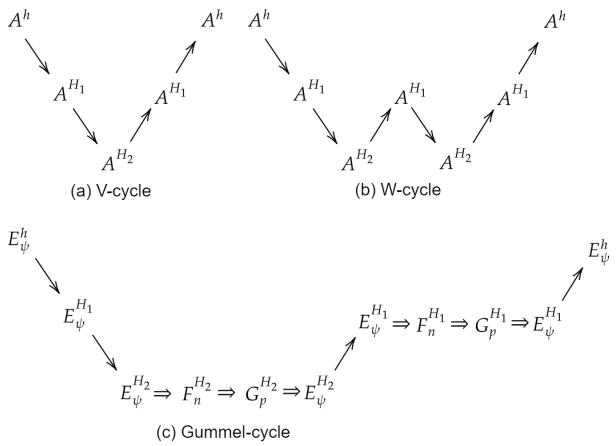

Fig. 2. Multigrid cycles: V-cycle, W-cycle and Gummel-cycle

We call the intermediate cycles: $E_\psi \Rightarrow F_n \Rightarrow E_\psi$ and $E_\psi \Rightarrow G_p \Rightarrow E_\psi$ the Gummel-cycle because this process traces the traditional Gummel iteration [1]. The conventional multigrid methods employ V-cycle or W-cycle to reduce errors but our Gummel-cycle is used to resolve the discrepancy between each system: E, F and G (Fig. 2).

IV. Evaluation

A. Trade-off

As mentioned before, the co-cycle could make convergence of the multigrid method stabilized, but on the other hand, it is a time consuming process consisting of several matrix–vector multiplications. There is a trade off between the number of co-cycles per. a multigrid cycle and the number of iterations required to converge (FIG. 3). To reduce the time of co-cycle, pre-calculated matrices could be provided. However, generating such a matrix requires sparse matrix–matrix multiplications, which are also time consuming. Besides the memory consumption for the pre-calculated matrices is not small. So this issue is still opened.

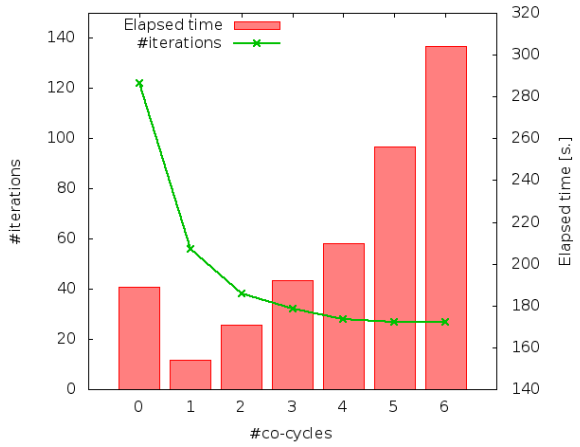

Fig. 3. Trade off btw. #of co-cycles and #of iterations

B. Scaling

To evaluate capability of the algebraic multigrid method with the Gummel-cycle, the strong and weak scaling benchmarks are performed on the matrices generated by the terminal structure of the power device (FIG. 4) under the near breakdown voltage (FIG. 5.) In the both benchmarks, the GMRES method is used with preconditioning. For comparison, the ILUT preconditioning is chosen as the conventional one. As shown in the strong scaling benchmark (Fig. 6), the Gummel-cycle algebraic multigrid method shows almost linear scaling property. Even though the cost of the co-cycle is not cheap, having enough parallel units could defeat the conventional methods. This trend is remarkable on the weak scaling benchmark which loads a constant task per a same compute unit. As using a finer mesh (ie. increasing a number of nodes), the total numbers of fill-ins for solving by the conventional methods are getting larger. Some 15,360 variables are filled for a cocmpute unit and the flat MPI is used for parallelization. As shown in FIG. 7, our method exceeds the other conventional methods at the larger cases.

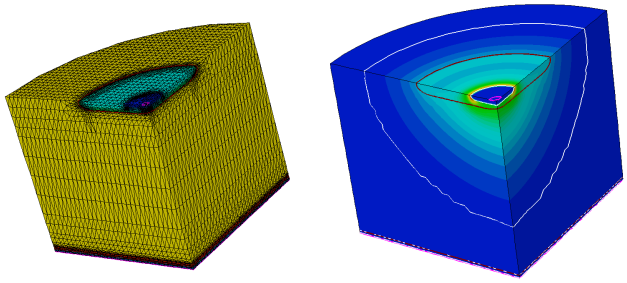

Fig. 4. Device structure for benchmark

Fig. 5. Electric Field under the near breakdown

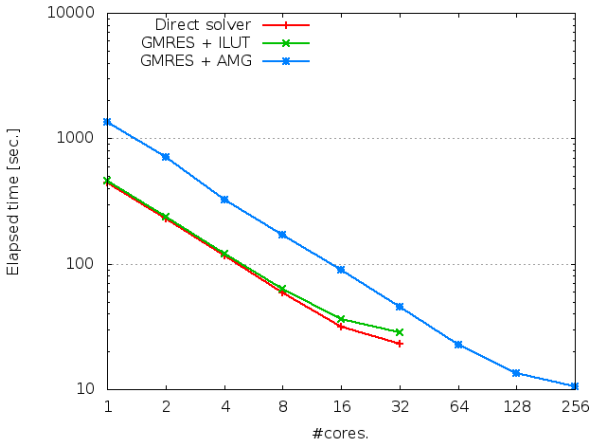

Fig. 6. Comparison on strong scaling with each method

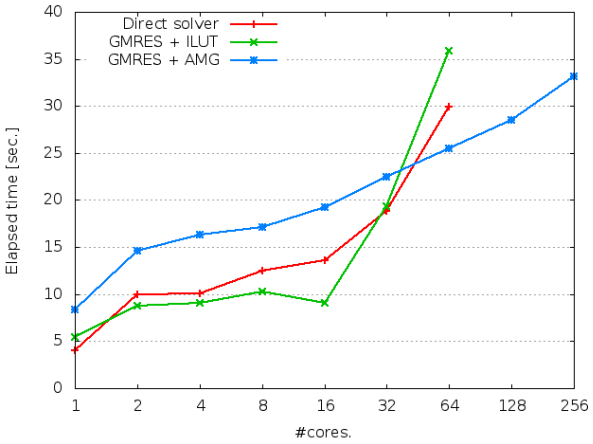

Fig. 7. Comparison on weak scaling with each method

V. Conclusion

In this work, we have proposed a new preconditioning technique based on the algebraic multigrid with Gummel-cycle, which can scale up without having fill-ins. The proposed technique will also meet a situation such as electric fields spread into the bulk owing to the algebraic multigrid manner. And so this approach helps simulations of deep depletion layers or wide bandgap devices like GaN, SiC and Ga2O3. However, there requires a cafeful treatment on applying our method because it has the trade off between its stability and time consumption. The optimal balance of the number of co-cycles or an appropriate relaxation approach should be introduced in future.

References

[1] Gummel, Hermann K., "A self-consistent iterative scheme for one-dimensional state transistor calculations," IEEE Transactions on electron devices, vol. 11, no. 10, pp. 455–465, 1964.

[2] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, "Adaptive smoothed aggregation (α sa) multigrid," SIAM review, vol. 47, no. 2, pp. 317–346, 2005.

[3] O. Schenk, S.Rollin, and A. Gupta, "The effects of unsymmetric matrix permutations and scalings in semiconductor device and circuit simulation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, no. 3, pp. 400–411, 2004.

[4] P. T. Lin, J. N. Shadid, R. S. Tuminaro and M. Sala, "Performance of a Petrov–Galerkin algebraic multilevel preconditioner for finite element modeling of the semiconductor device drift–diffusion equations," International journal for numerical methods in engineering, vol. 74, no. 4, pp. 448–469, 2010.

[5] A. Toselli and O. Widlund "Domain decomposition methods - algorithms and thery," Springer Science & Business Media, vol. 34, 2006.

[6] S. Sho and S. Odanaka, "A hybrid MPI/OpenMP parallelization method for a quantum drift-diffusion model," International Conference on Simulation of Semiconductor Process and Devices (SISPAD), pp. 33-36, 2017.