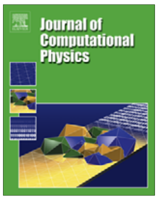
Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp



Computing entries of the inverse of a sparse matrix using the FIND algorithm

S. Li [a], S. Ahmed [b], G. Klimeck [c], E. Darve [a,*]

[a] *Institute for Computational and Mathematical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Room 209, Stanford, CA 94305-4040, USA*
[b] *Electrical and Computer Engineering Department, Southern Illinois University Carbondale, IL 62901, USA*
[c] *Network for Computational Nanotechnology, Birck Nanotechnology Center, Purdue University, West Lafayette, IN 47907, USA*

ARTICLE INFO

Article history:
Received 17 April 2007
Received in revised form 27 May 2008
Accepted 23 June 2008
Available online 17 July 2008

PACS:
02.60.Dc
02.70.−c
73.23.−b
73.63.−b

Keywords:
Nested dissection
Green's function
NEGF
Nanotransistor
Gaussian elimination
Sparse matrix

ABSTRACT

An accurate and efficient algorithm, called fast inverse using nested dissection (FIND), for computing non-equilibrium Green's functions (NEGF) for nanoscale transistors has been developed and applied in the simulation of a novel dual-gate metal-oxide-semiconductor field-effect transistor (MOSFET) device structure. The method is based on the algorithm of nested dissection. A graph of the matrix is constructed and decomposed using a tree structure. An upward and downward traversal of the tree yields significant performance improvements for both the speed and memory requirements, compared to the current state-of-the-art recursive methods for NEGF. This algorithm is quite general and can be applied to any problem where certain entries of the inverse of a sparse matrix (e.g., its diagonal entries, the first row or column, etc.) need to be computed. As such it is applicable to the calculation of the Green's function of partial differential equations. FIND is applicable even when complex boundary conditions are used, for example non reflecting boundary conditions.

1. Introduction

The non-equilibrium Green's function (NEGF) approach is being considered as a state-of-the-art modeling tool in predicting performance and designing emerging nanoscale devices. Development of multi-dimensional simulators based on the NEGF approach is crucial to capture both the quantum mechanical effects and the effect of scattering with phonons and other electrons. Despite the fact that transport issues for nano-transistors, nanowires and molecular electronic devices are very different from one another, they can be treated with the common formalism provided by the NEGF [1]. The approach is based on the coupled solution of the Schrödinger and Poisson equations. So far, the difficulties in understanding the various terms in the resultant equations and the computational burden needed for its actual implementation are perceived as great challenges. A successful utilization of the Green's function approach commercially is the nano-electronics modeling (NEMO) simulator [2], which is effectively 1D and is primarily applicable to resonant tunneling diodes. Accurate and reliable multi-dimensional modeling of realistic future nanoscale devices requires enormous computational efforts, yet the currently

[*] Corresponding author. Tel.: +1 650 725 2560; fax: +1 650 723 3525.
E-mail address: darve@stanford.edu (E. Darve).

doi:10.1016/j.jcp.2008.06.033

available algorithms are prohibitively expensive. This paper focuses on an accurate and efficient implementation of the NEGF approach for 2D MOSFET device structures.

Our algorithm, fast inverse using nested dissection (FIND), reduces the computational cost of the most expensive part of NEGF, which is the solution of the Green's function equation for the electron density, which is then used in the Poisson equation. In a typical simulation, the Poisson equation needs to be solved self-consistently with the Schrödinger equation. Consequently, the electron density gets typically computed multiple times until convergence is achieved. This leads to huge computational costs which FIND can reduce by orders of magnitude.

The most expensive calculation is computing some of (but not all) the entries of the matrix G^{r} [1]:

$$G^{\mathrm{r}}(E) = [EI - H - \Sigma]^{-1} = A^{-1} \quad \text{(retarded Green's function)} \tag{1}$$

and $G^{<}(E) = G^{\mathrm{r}}\Sigma^{<}(G^{\mathrm{r}})^{\dagger}$ (less-than Green's function). In these equations, I is the identity matrix, and E is the energy level. † denotes the transpose conjugate of a matrix. The Hamiltonian matrix H describes the system at hand (e.g., nano-transistor). It is usually a sparse matrix with connectivity only between neighboring mesh nodes, except for nodes at the boundary of the device which may have a non-local coupling (e.g., non-reflecting boundary condition). The matrices Σ and $\Sigma^{<}$ correspond to the self energy and can be assumed to be diagonal matrices. See Svizhenko [3] for this terminology and notations. In this work, all these matrices are considered to be given and we will focus on the problem of efficiently computing some entries in G^{r} and $G^{<}$. As an example of entries which must be computed, the diagonal entries of G^{r} are required to compute the density of states, while the diagonal entries of $G^{<}$ allow computing the electron density. The current can be computed from the upper diagonal entries of $G^{<}$.

Even though the matrix A in Eq. (1) is, by the usual standards, a mid-size sparse matrix of size typically 10,000 × 10,000, computing the entries of $G^{<}$ is a major challenge since this operation is repeated at all energy levels for every iteration of the Poisson–Schrödinger solver. Overall, the diagonal of $G^{<}(E)$ for the different values of the energy level E can be computed as many as thousands of times.

The problem of computing certain entries of the inverse of a sparse matrix is relatively common in computational engineering. Examples include:

- *Least square fitting: in the linear least-square fitting procedure, coefficients* a_k are computed so that the error

  $$\sum_i \left[Y_i - \sum_k a_k \phi_k(x_i) \right]^2$$

  is minimal, where (x_i, Y_i) are the data points. It can be shown, under certain assumptions that, in the presence of measurement errors in the observations Y_i, the error in the coefficients a_k is proportional to C_{kk} where C is the inverse matrix of A:

  $$A_{jk} = \sum_i \phi_j(x_i)\phi_k(x_i)$$

- *Eigenvalues of tri-diagonal matrices: the inverse iteration method attempts to compute the eigenvector* $\boldsymbol{v}$ associated with eigenvalue λ by solving iteratively the equation

  $$(A - \hat{\lambda}I)\boldsymbol{x}_k = s_k \boldsymbol{x}_{k-1}$$

  where $\hat{\lambda}$ is an approximation of λ and s_k is used for normalization. Varah [4] and Wilkinson [5–7] have extensively discussed optimal choices of starting vectors for this method. An important result is that, in general, choosing the vector $\boldsymbol{e}_l$ (lth vector in the standard basis), where l is the index of the column with the largest norm among all columns of $(A - \hat{\lambda}I)^{-1}$, is a nearly optimal choice. A good approximation can be obtained by choosing l such that the lth entry on the diagonal of $(A - \hat{\lambda}I)^{-1}$ is the largest among all diagonal entries.
- *Accuracy estimation: when solving a linear equation* $A\boldsymbol{x} = \boldsymbol{b}$, one is often faced with errors in A and $\boldsymbol{b}$, either because of uncertainties in physical parameters or inaccuracies in their numerical calculation. In general the accuracy in the computed solution $\boldsymbol{x}$ will depend on the condition number of A: $\|A\|\|A^{-1}\|$, which can be estimated from the diagonal entries of A and its inverse in some cases.
- *Sensitivity computation: when solving* $A\boldsymbol{x} = \boldsymbol{b}$, the sensitivity of x_i to A_{jk} is given by $\partial x_i / \partial A_{jk} = x_k (A^{-1})_{ij}$.

Many other examples can be found in the literature.

Currently the state-of-the-art is a method developed by Klimeck and Svizhenko et al. [3], called the recursive Green's function method (RGF). This approach can be shown to be the most efficient for "nearly 1D" devices, i.e. devices which are very elongated in one direction and very thin in the two other directions.

Assume that the matrix A is the result of discretizing a partial differential equation in 2D using a local stencil, e.g., with a 5 point stencil. Assume the mesh is the one given on Fig. 1.

For a 5 point stencil, the matrix A can be written as a tri-diagonal block matrix where blocks on the diagonal are denoted by A_q $(1 \leqslant i \leqslant n)$, on the upper diagonal by B_q $(1 \leqslant i \leqslant n-1)$, and on the lower diagonal by C_q $(2 \leqslant i \leqslant n)$.

RGF computes the diagonal of A^{-1} by computing recursively two sequences. The first sequence, in increasing order, is defined recursively as [3]: