Journal of Computational Physics 242 (2013) 915–945

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

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

A fast algorithm for sparse matrix computations related to inversion

S. Li [a,*], W. Wu [b], E. Darve [a,c]

[a] Institute for Computational and Mathematical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Stanford, CA 94305, USA
[b] Department of Electrical Engineering, Stanford University, 350 Serra Mall, Packard Building, Room 268, Stanford, CA 94305, USA
[c] Department of Mechanical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Room 209, Stanford, CA 94305, USA

ARTICLE INFO

Article history:
Received 4 February 2011
Received in revised form 10 October 2012
Accepted 26 January 2013
Available online 5 March 2013

Keywords:
Nested dissection
Green's function
NEGF
Decomposition
Gaussian elimination
Sparse matrix
Inverse
Elimination tree

ABSTRACT

We have developed a fast algorithm for computing certain entries of the inverse of a sparse matrix. Such computations are critical to many applications, such as the calculation of non-equilibrium Green's functions $\mathbf{G}^r$ and $\mathbf{G}^<$ for nano-devices. The FIND (Fast Inverse using Nested Dissection) algorithm is optimal in the big-O sense. However, in practice, FIND suffers from two problems due to the width-2 separators used by its partitioning scheme. One problem is the presence of a large constant factor in the computational cost of FIND. The other problem is that the partitioning scheme used by FIND is incompatible with most existing partitioning methods and libraries for nested dissection, which all use width-1 separators. Our new algorithm resolves these problems by thoroughly decomposing the computation process such that width-1 separators can be used, resulting in a significant speedup over FIND for realistic devices — up to twelve-fold in simulation. The new algorithm also has the added advantage that desired off-diagonal entries can be computed for free. Consequently, our algorithm is faster than the current state-of-the-art recursive methods for meshes of any size. Furthermore, the framework used in the analysis of our algorithm is the first attempt to explicitly apply the widely-used relationship between mesh nodes and matrix computations to the problem of multiple eliminations with reuse of intermediate results. This framework makes our algorithm easier to generalize, and also easier to compare against other methods related to elimination trees. Finally, our accuracy analysis shows that the algorithms that require back-substitution are subject to significant extra round-off errors, which become extremely large even for some well-conditioned matrices or matrices with only moderately large condition numbers. When compared to these back-substitution algorithms, our algorithm is generally a few orders of magnitude more accurate, and our produced round-off errors stay at a reasonable level.

1. Introduction

The Non-Equilibrium Green's Function (NEGF) approach is currently being considered as a state-of-the-art modeling tool in the design and performance analysis of emerging nanoscale devices. Development of multi-dimensional simulators based on the NEGF approach is crucial for capturing both quantum mechanical effects as well as the effect of scattering with phonons and other electrons.

* Corresponding author. Tel.: +1 650 714 8221.
E-mail address: lisong@stanford.edu (S. Li).

0021-9991/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2013.01.036

List of Main Notations

N_x, N_y	Size of the mesh from the discretization of 2D device. p. 2
$\mathbf{A}, \mathbf{\Sigma}$	Two given matrices of the same sparsity pattern. $\mathbf{A}$ is the sparse matrix from the discretization of the device. $\mathbf{\Sigma}$ is the matrix associated with the self energy. p. 2
$\mathbf{G}^r, \mathbf{G}^<$	$\mathbf{G}^r = \mathbf{A}^{-1}$ and $\mathbf{G}^< = \mathbf{A}^{-1}\mathbf{\Sigma}\mathbf{A}^{-\dagger}$ are two Green's function matrices we want to compute. p. 2
M	The set of all the nodes in the mesh. p. 10
$\mathsf{C}, \mathsf{D}, \mathsf{C}_g, \mathsf{D}_g$	C_g is the set of mesh nodes after partitioning M by divider set D_g. p. 29
P, P_g	Peripheral sets. P_g includes the neighbors of C_g with two corner nodes added. p. 10
S, S_g	Separator sets. S_g separates cluster C_g. They are generalization of the divider sets. p. 10
$\mathsf{L}_k, \mathsf{R}_k$	The parts of P_k from its left child and right child, respectively. $\mathsf{P}_k = \mathsf{L}_k \uplus \mathsf{R}_k$. Here $\uplus$ is to emphasize the disjointness: $\mathsf{L}_k \cap \mathsf{R}_k = \emptyset$. p. 11
negative sets	$\mathsf{C}, \mathsf{S}, \mathsf{P}, \mathsf{L}$, and R with negative indices are also defined. p. 12
T_0, T_r^+	All positive clusters C_g are organized as a basic (binary) cluster tree T_0. For every target cluster C_r, there is a partitioning of M into positive clusters and negative clusters, which are organized as an augmented tree T_r^+. The root of T_0 is $\mathsf{C}_1 = \mathsf{M}$; the root of T_r^+ is C_{-r}. pp. 5, 10, 12
$\prec, \|$	Partial order relations, equivalent to tree structure. $\mathsf{S}_g \prec \mathsf{S}_k$ (or $g \prec k$ for short) if and only if C_g is a descendant of C_k in T_r^+. $\mathsf{S}_g \| \mathsf{S}_k$ or $g \| k$ if they are incomparable. p. 15
$<$	Total order relation, equivalent to an order of elimination. $\mathsf{S}_g < \mathsf{S}_k$ if and only if the columns corresponding to S_g appear before columns corresponding to S_k in an ordering of $\mathbf{A}$, i.e., S_g columns are eliminated before S_k. p. 15
$\mathbf{L}, \mathbf{U}$	The LU factors of the sparse matrix $\mathbf{A} = \mathbf{LU}$. p. 4
$\mathbf{L}_g$	The LU factorization of $\mathbf{A}$ is decomposed into multiple partial factorizations: $\mathbf{L} = \prod_g \mathbf{L}_g; \mathbf{L}_g^{-1}$ eliminates S_g columns. The order of multiplications is associated with the order of elimination. p. 16
$\mathbf{A}^{(r)}$	$\mathbf{A}$ after a reordering such that all columns corresponding to some separator set S_g stay together. The total order among S_g given by $\mathbf{A}^{(r)}$ is a linear extension of the partial order among S_g given by T_r^+. p. 15
$\mathbf{A}_k^{(r)}, \mathbf{A}_{k+}^{(r)}$	Intermediate results in the process of elimination on $\mathbf{A}^{(r)}$. All the columns before S_k have been eliminated in $\mathbf{A}_k^{(r)} = \left(\prod_{\mathsf{S}_g < \mathsf{S}_k} \mathbf{L}_g\right)^{-1} \mathbf{A}_k^{(r)}$. In addition to these columns, the S_k columns are also eliminated in $\mathbf{A}_{k+}^{(r)} = \mathbf{L}_k^{-1}\mathbf{A}_k^{(r)}$. p. 16
$\mathring{\mathbf{A}}_k$	Intermediate results in the elimination process on $\mathbf{A}$ with an ordering given by a subtree of some augmented tree; the root of the subtree is C_k. p. 16
$\mathbf{M}_k, \mathbf{V}_k$	Multiplication part of each step of elimination and their sums. $\mathbf{M}_k = \mathring{\mathbf{A}}_{k+} - \mathring{\mathbf{A}}_k$. $\mathbf{V}_k = \sum_{g \preccurlyeq k} \mathbf{M}_g$. pp. 17, 18
$\mathcal{M}_k, \mathcal{L}_k$	Non-zero submatrices of $\mathbf{M}_k$ and $\mathbf{L}_k$. $\mathcal{M}_k = \mathbf{M}_k(\mathsf{P}_k, \mathsf{P}_k)$; $\mathcal{L}_k = \mathring{\mathbf{A}}_k(\mathsf{P}_k, \mathsf{S}_k)\mathring{\mathbf{A}}_k(\mathsf{S}_k, \mathsf{S}_k)^{-1}$. pp. 17, 20

Despite the fact that the transport issues for nano-transistors, nanowires, and molecular electronic devices are very different from one another, these issues can all be treated with the common formalism provided by the NEGF approach, which treats the devices as 2D [1]. The NEGF approach is based on the consistent solution to the coupled Schrödinger and Poisson equations. The most time-consuming aspect of this approach in a typical simulation is the repeated solving of the following Schrödinger equation for the density of states and electron density:

$$[\hat{H} + U(\mathbf{r}) - E]\Psi(\mathbf{r}) = 0 \tag{1}$$

Here E is a constant, U is a real function of $\mathbf{r}$, and $\hat{H}$ is the Hamiltonian operator. Solving of Eq. (1) continues until consistency between the Schrödinger and Poisson equations is achieved.

Typically, Eq. (1) is solved numerically by reducing it to a matrix computation problem after discretization. In this paper, we will focus on a discretization scheme for $\hat{H}$ on a 2D Cartesian grid with a 5-point stencil finite difference scheme. However, both the existing algorithms and our algorithm can be extended to more general schemes, including finite elements and arbitrary stencils with local connectivity on 3D devices. Fig. 1(a) illustrates a mesh of size $N_x \times N_y$ for a nanodevice. Fig. 1(b) shows the corresponding matrix using a 5-point stencil finite difference scheme.

After discretization, solving Eq. (1) is reduced to computing the diagonal entries of the retarded Green's function $\mathbf{G}^r(E) \triangleq \mathbf{A}^{-1}$ and $\mathbf{G}^<(E) \triangleq \mathbf{A}^{-1}\mathbf{\Sigma}^<\mathbf{A}^{-\dagger}$ in matrix form, where $\mathbf{\Sigma}$ and $\mathbf{\Sigma}^<$ correspond to the self-energy in the physical problem and are considered given, and $\mathbf{A} \triangleq E\mathbf{I} - \mathbf{H} - \mathbf{\Sigma}$.

Several algorithms exist for computing $\mathbf{G}^r$ and $\mathbf{G}^<$ using direct methods [2–5]. Among them, the FIND (Fast Inverse using Nested Dissection) algorithm has the lowest asymptotic runtime for 2D problems. However, this runtime includes a large constant factor due to FIND's usage of width-2 separators, which are natural to achieve independence between subproblems. This large constant factor makes FIND less competitive for 2D problems of medium size.

To address this problem, in this work we present a new algorithm called FIND-SS that makes the usage of width-1 separators [6–9] possible by relaxing the independence required by FIND. This relaxation of independence is not straightforward, and is achieved through further decomposition of the elimination process, and full utilization of independence between subproblems. Using width-1 separators not only reduces computational costs, but also makes FIND-SS compatible