Journal of Computational Science 5 (2014) 1–11

Contents lists available at ScienceDirect

Journal of Computational Science

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

Sub-exponential graph coloring algorithm for stencil-based Jacobian computations

Michael Lülfesmann[a,*], Ken-ichi Kawarabayashi[b,1]

[a] Institute for Scientific Computing, Center for Computational Engineering Science, RWTH Aachen University, D-52056 Aachen, Germany
[b] National Institute of Informatics and JST ERATO Kawarabayashi Large Graph Project, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan

ARTICLE INFO

Article history:
Received 3 October 2012
Received in revised form 16 June 2013
Accepted 22 June 2013
Available online 5 July 2013

MSC:
05C15
05C50
05C85
90C27
65D25
65F50

Keywords:
Grid coloring algorithm
Divide-and-conquer approach
Lipton–Tarjan separator
Stencil discretization
Derivative computation
Sparsity exploitation

ABSTRACT

Partial differential equations can be discretized using a regular Cartesian grid and a stencil-based method to approximate the partial derivatives. The computational effort for determining the associated Jacobian matrix can be reduced. This reduction can be modeled as a (grid) coloring problem. Currently, this problem is solved by using a heuristic approach for general graphs or by developing a formula for every single stencil. We introduce a sub-exponential algorithm using the Lipton–Tarjan separator in a divide-and-conquer approach to compute an optimal coloring. The practical relevance of the algorithm is evaluated when compared with an exponential algorithm and a greedy heuristic.

1. Introduction

In the field of computational science and engineering, partial differential equations (PDEs) are often used to formulate real-world phenomenons. The formulation of the PDEs is given as a mathematical function f, e.g., $f(\phi) = \Delta\phi = 0$ for smooth $\phi : [0, 1]^2 \to \mathbb{R}$, which is implemented as a computer program in a programming language like Fortran, C++, Matlab, etc. PDEs can be solved numerically by first discretizing the computational domain, e.g., a regular Cartesian grid together with a stencil depending on f, and then using a Newton–Raphson-type algorithm. This will require evaluating the sparse Jacobian of f on the discretized domain, which is a very compute-intensive operation for complicated PDEs. Using a Newton–Raphson-type algorithm to solve a non-linear system, Coleman and Xu measured a runtime of around 15,000 s [1]. Furthermore, they stated that with increasing problem size the evaluation of f dominates the solving approach. Compared to the matrices considered in this article their considered system is quite small. It should not be ignored that in time-dependent simulations the Jacobian matrix must be repeatedly evaluated in every timestep.

The Jacobian matrix of the computer program f is often either approximated by divided differencing or computed exactly using automatic differentiation. For both approaches, the computing time to evaluate the Jacobian is a multiple of the time to evaluate the computer program. To reduce the computational effort to determine the matrix in large-scale problems, it is crucial to exploit any available structure. The nonzero structure of the sparse Jacobian matrix is precisely defined by the stencil and the size of the regular grid. The exploitation of the sparsity structure of the matrix can be modeled as a combinatorial optimization problem in terms of the grid.

Explicit solutions of the combinatorial optimization problem for regular grids are known for various special stencils [2–4]. Further solutions are given in [5] where the authors looked at a more

* Corresponding author.
E-mail address: michael.luelfesmann@rwth-aachen.de (M. Lülfesmann).

[1] Research partly supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research, by C&C Foundation, by Kayamori Foundation and by Inoue Research Award for Young Scientists.

1877-7503/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.jocs.2013.06.002

general problem. Recently, stencil-based computations on grids with periodic boundary conditions have been analyzed [6]. A more comprehensive survey and comparison of the optimization problem in terms of the regular grid, the Jacobian matrix and a bipartite graph are given in [7,8]. Nevertheless, investigating a formula for every single stencil is time-consuming and, above all, complex in more than two dimensions. Especially while using semi-automated stencil generators [9], an explicit solution for the generated stencil cannot be determined automatically with this approach.

A graph coloring is an assignment of nonzero values $1, \ldots, p$, the so-called colors, to vertices of a graph by following some rules. The optimization problem can be formulated as a coloring of the grid points. The optimal solution is a coloring of the grid points with the smallest number of colors possible. Such a coloring is denoted as minimal coloring or optimal coloring. For general graphs this optimization problem is NP-hard [10]. Therefore, greedy heuristics are widely used to determine an usually not optimal solution [11,12] in reasonable time. Using an exhaustive search strategy to compute an optimal solution would be very time-consuming and only applicable to very small Jacobian matrices.

We introduce a sub-exponential divide-and-conquer algorithm using separators to exactly solve the optimization problem for regular grids and arbitrary stencils in two or more dimensions. Divide-and-conquer is one of the oldest and most widely used techniques for designing efficient algorithms. Divide-and-conquer algorithms partition their inputs into two or more independent subproblems, solve those subproblems recursively, and then combine the solutions to those subproblems to obtain their final output. This strategy can be successfully applied to several graph problems, provided we can quickly separate the graph into roughly equal subgraphs. A vertex separator is a set of vertices which splits a graph in two non-connected components. This concept was first adapted by Lipton and Tarjan [13]. The underlying planar separator theorem was introduced in [14]. Following this theorem and its extensions [15,16], our algorithm establishes a hierarchy of separators by recursively dividing the original grid in smaller subgrids as long as the number of grid points does not fall below a threshold. After computing all optimal colorings for the smallest subgrids by an exhaustive search, the colorings of the subgrids are combined level by level to get a minimal coloring for the original grid. Our algorithm is implemented in C++ and is parallelized by OpenMP, suitable for recent shared-memory computers. Parts of this article have already been published in a preliminary version in [17,18].

This article is structured as follows. In Section 2, we explain stencil-based Jacobian computations and introduce corresponding matrix- and grid-based representations. Additionally, we briefly explain the technique to exploit the sparsity of Jacobian matrices. In Section 3, the grid coloring problem and the vertex separators are introduced. We describe the sub-exponential coloring algorithm using separators in Section 4 and the implementation details and techniques to reduce the runtime in Section 5. In Section 6, we compare the runtime of our algorithm to the runtime of the exhaustive search coloring algorithm. Furthermore, the difference in the number of colors and the runtime of a greedy coloring heuristic is evaluated. In the last section we give a concluding summary.

2. Stencil-based Jacobian computations on grids

We are looking for the Jacobian matrix of a function f defined by some computer program. We focus on the special case where this function

$$f : \mathbb{R}^{MN} \longrightarrow \mathbb{R}^{MN} \tag{1}$$

is computed by a stencil operation on a regular Cartesian $M \times N$ grid. That is, the value of a quantity on a grid point is updated by the weighted values of the quantity on neighboring grid points. Here, we consider only neighbors in space rather than in time. The derivative of the vector-valued function f with respect to some vector $x \in \mathbb{R}^{MN}$ into the direction of a vector $s \in \mathbb{R}^{MN}$ is defined by

$$\frac{\partial f}{\partial x} s = \lim_{h \to 0} \frac{f(x + hs) - f(x)}{h}.$$

The Jacobian matrix of f is an $MN \times MN$ matrix, which would require $MN+1$ evaluations of f and is therefore very time-consuming. However, for any $MN \times p$ matrix S, the product of the Jacobian and S can be evaluated by $p+1$ evaluations of the function f using divided differencing. Similarly, the forward mode of automatic differentiation [19,20] is capable of computing that matrix–matrix product without truncation error. A good effort estimation is still $p+1$ times the time needed to evaluate f.

Two columns of the Jacobian are *structurally orthogonal* if they do not have any nonzero element in the same row. In the example given in Fig. 1(b), the columns 1 and 6 are structurally orthogonal since there is no row in which both columns have a nonzero element. The idea to reduce the computational effort to compute all nonzero entries of a sparse Jacobian consists of partitioning the columns of the Jacobian into groups of structurally orthogonal columns [11]. Therefore, the vector s can be chosen as binary vector containing ones precisely at the indices corresponding to the columns of such a group. The multiplication of the Jacobian matrix with this vector will then give us the nonzero elements of all these pairwise structurally orthogonal columns simultaneously. This permits us to construct a binary $MN \times p$ matrix S with $p \leq MN$ such that the product $J \cdot S$ will give us all nonzero elements of the Jacobian, by finding p groups of structurally orthogonal columns. Therefore, we can calculate the entire Jacobian using only $p+1$ evaluations of function f.

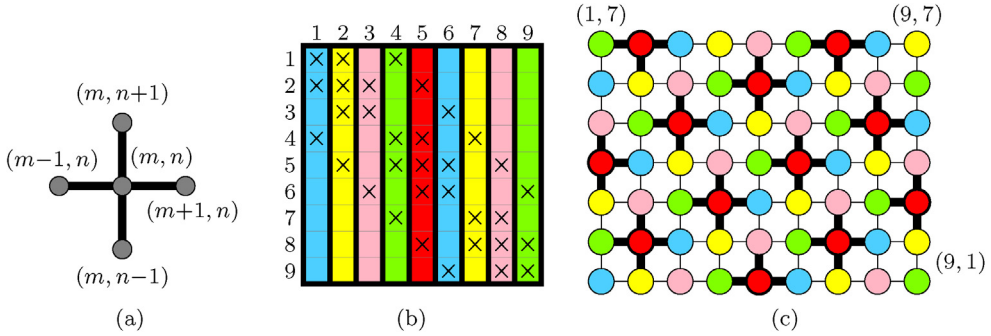

Fig. 1. (a) Five-point stencil $\mathcal{N}_{5\text{pt}}(m, n)$. (b) Nonzero pattern of the Jacobian matrix resulting from $\mathcal{N}_{5\text{pt}}$ using a natural ordering of grid points on a 3×3 grid. Background padding indicates $p = 5$ groups of structurally orthogonal columns. (c) Sequence of covers obtained from using $p = 5$ covers each corresponding to a group of structurally orthogonal center points for $\mathcal{N}_{5\text{pt}}$ on a 9×7 grid.