September 2017

A Parallel Scheduling Algorithm to Solve Triangular Band Systems on multicore machine

Sirine MARRAKCHI [1], Mohamed JEMNI
Research Laboratory of Technologies of Information and Communication & Electrical Engineering (LaTICE), ENSIT University of Tunis, Tunisia

Abstract. Solving sparse triangular systems (spTRSV) is considered as one of the principal problems in computational linear algebra. This study investigates a new practical approach to enhance the performance of spTRSV through the reduction of the parallel execution time. Given a different number of cores, the proposed solution is validated using a set of experimentations based on triangular band matrices. Comparing with the previous works, the obtained results indicate the efficiency of our developed method.

Keywords. solving sparse triangular systems, parallel execution time, performance, number of cores, band matrices.

1. Introduction

The growth of advanced-architecture platforms has a substantial impact on various issues of scientific computation in numerical linear algebra [1]. In fact, it should be possible to decrease the parallel execution time. Nowadays, multicore machines are becoming more and more used due to their scalability and performance [2].

Linear algebra, in particularly, solving sparse triangular systems (spTRSV) noted $Mx = b$ is considered as one of the main problems of computational mathematics [3]. Here $M = (m_{ij}) \in R^{n\times n}, 1 \leq i, j \leq n$ a non-singular sparse triangular matrix, x and b represent the solution and the given vectors, respectively.

Several research studies [4,5,6,7,8,9] have appeared to improve the parallel performance for spTRSV. The simplest method (block columns solver) crosses all columns in ascending order and executes a single component of x in each step. Then the code updates in parallel all the positions corresponding to the non-zero elements of the current column. However, the rows of M usually contain very few non-zero elements and the cores synchronize many times resulting in very low performance. Mayer [4] has developed two parallel algorithms on shared memory. The idea behind the first one is to divide the matrix M into blocks by both rows and columns. Blocks in the same anti-diagonal can

[1] Corresponding Author: Sirine Marrakchi, LaTICE laboratory, ENSIT, University of Tunis, Tunisia; E-mail: sirine.marrakchi@yahoo.com.