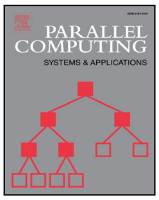
Parallel Computing 000 (2018) 1–16

Contents lists available at ScienceDirect

Parallel Computing

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



Two-sided orthogonal reductions to condensed forms on asymmetric multicore processors

Pedro Alonso[a], Sandra Catalán[b], José R. Herrero[c], Enrique S. Quintana-Ortí[b], Rafael Rodríguez-Sánchez[b,*]

[a] *Departamento de Sistemas Informáticos y Computación, Universidad Politècnica de València, Spain*
[b] *Departamento de Ingeniería y Ciencia de Computadores, Universidad Jaume I, Castellón, Spain*
[c] *Department d'Arquitectura de Computadors, Universitat Politècnica de Catalunya, Spain*

A R T I C L E   I N F O

Article history:
Received 31 March 2017
Revised 7 February 2018
Accepted 25 March 2018
Available online xxx

Keywords:
Dense linear algebra
Condensed forms
Eigenvalue problems
Singular-value problems
Asymmetric multicore processors
Heterogeneous computing
Multi-threading
Workload balancing

A B S T R A C T

We investigate how to leverage the heterogeneous resources of an Asymmetric Multicore Processor (AMP) in order to deliver high performance in the reduction to condensed forms for the solution of dense eigenvalue and singular-value problems. The routines that realize this type of two-sided orthogonal reductions (TSOR) in LAPACK are especially challenging, since a significant fraction of their floating-point operations are cast in terms of memory-bound kernels while the remaining part corresponds to efficient compute-bound kernels. To deal with this scenario: (1) we leverage implementations of memory-bound and compute-bound kernels specifically tuned for AMPs; (2) we select the algorithmic block size for the TSOR routines via a practical model; and (3) we adjust the type and number of cores to use at each step of the reduction. Our experiments validate the model and assess the performance of our asymmetry-aware TSOR routines, using an ARMv7 big.LITTLE AMP, for three key operations: the reduction to tridiagonal form for symmetric eigenvalue problems, the reduction to Hessenberg form for non-symmetric eigenvalue problems, and the reduction to bidiagonal form for singular-value problems.

© 2018 Elsevier B.V. All rights reserved.

1. Introduction

We target the two-sided orthogonal reduction (TSOR) of a dense matrix to a condensed form (namely tridiagonal, Hessenberg or bidiagonal) as an initial step for the solution of dense eigenvalue or singular-value problems [1] on multi-threaded Asymmetric Multicore Processors (AMPs). Our interest in these architectures is motivated by the growing role of energy-efficient multicore systems-on-chip (SoC) on the road to exascale systems, and the challenge of efficiently exploiting the heterogeneous types of cores in these architectures for dense linear algebra operations. An asymmetric big.LITTLE architecture presents the appealing property of being composed of two distinct types of cores, optimized for either raw performance or low power consumption. Energy simply reflects the consumption of power across a period of time. Therefore, a low-power symmetric multicore chip cannot always provide the most energy-efficient solution. In particular, for some applications, a power-hungry but faster processor can provide a more energy-efficient solution, while for others, it is more

* Corresponding author.
E-mail addresses: palonso@upv.es (P. Alonso), catalans@uji.es (S. Catalán), josepr@ac.upc.edu (J.R. Herrero), quintana@uji.es (E.S. Quintana-Ortí), rarodrig@uji.es (R. Rodríguez-Sánchez).

https://doi.org/10.1016/j.parco.2018.03.005
0167-8191/© 2018 Elsevier B.V. All rights reserved.

efficient from the point of view of energy to run the application at a lower pace (i.e., on a low-power processor). A symmetric multicore chip cannot provide an optimal configuration for both types of scenarios. Our motivation to target this type of heterogeneous architecture is that, in the dense linear algebra domain, the primary objective is performance, and therefore it becomes crucial to exploit both types of cores present in the ARM big.LITTLE SoC.

Eigenvalue problems appear, among others, in computational quantum chemistry, finite element modeling, multivariate statistics, and density functional theory [2]. The computation of the singular values of a matrix is relevant, for example, in signal processing, big data, genomics, statistics, natural language text processing, etc. [1,2].

Efficient and numerically reliable algorithms for the computation of the eigenvalues/singular values of a dense matrix consist of two stages [1]. The $m \times n$ input matrix A (with $m = n$ for eigenvalue problems) is first reduced to an $m \times n$ condensed matrix C via a sequence of orthogonal transformations applied from the left and right to A (two-sided transformations). This initial stage is then followed by the application of a specific solver to accurately compute the eigenvalues/singular values of C.

In this paper we describe several optimizations to the routines that perform the TSOR of the matrix A to condensed form specifically designed for an ARM big.LITTLE AMP. The reason for addressing the first stage only is that such transformations cost $O(n^3)$ floating-point arithmetic operations (flops) while, when the eigenvectors/singular vectors are not requested, the second stage has a minor contribution to the total cost and performance of the solver. When (eigen-/singular) vectors are requested, the second stage is more involved, and a third stage is performed to recover the vectors of the original matrix from those of the condensed matrix.

LAPACK (*Linear Algebra PACKage*) [3] provides three main routines for TSOR to distinct condensed forms:

- SYTRD reduces a symmetric matrix to tridiagonal form;
- GEHRD reduces a square matrix to Hessenberg form; and
- GEBRD transforms a general matrix to bidiagonal form.

The first two routines are each the first stage for computing eigenvalues, while the last routine is the first stage for computing singular values. In all three cases, the TSOR preserves the eigenvalues/ singular values. All three routines cast a significant part of their flops in terms of the Level-2 BLAS (*Basic Linear Algebra Subprograms*) [4] for the matrix-vector product, while the remaining flops are performed within the Level-3 BLAS [5]. (Hereafter we will neglect the low order terms in the flop counts. Furthermore, we will only consider the routines and operations that are responsible for the major fraction of the costs. In our case, these correspond to the Level-2 and Level-3 BLAS.)

In a recent work [6], we exposed the poor performance of SYTRD on an ARM big.LITTLE AMP, even if the Level-3 BLAS kernels are optimized to exploit the asymmetry of the architecture. The reason for this behavior is that the memory-bound nature of the Level-2 BLAS limits their scalability and the overall efficiency of SYTRD. In [7] we reported a remarkable acceleration for SYTRD achieved by using architecture-aware micro-kernels for the Level-2 BLAS and an asymmetry-aware dynamic schedule of these kernels. In the present paper, we extend that work making the following contributions:

- We discuss the generalization of our techniques developed in [7] for SYTRD to the two remaining TSOR routines, demonstrating their applicability in the reduction to bidiagonal form implemented in LAPACK routine GEBRD as well as the reduction to Hessenberg form in LAPACK routine GEHRD.
- We propose a performance model that guides the selection of the optimal algorithmic block size and core configuration for the TSOR stage.
- We perform a detailed experimental analysis to illustrate the performance benefits of our architecture- and asymmetry-aware variants of SYTRD, GEBRD and GEHRD on an ARMv7 big.LITTLE SoC.

Overall, we believe that the approach applied to optimize the routines for the TSOR to condensed forms on the target ARMv7 SoC presented in this work carries over to other asymmetric and heterogeneous architectures, including hybrid CPU-GPU systems, as well as multisocket/multicore servers where distinct CPUs/cores operate at different frequencies.

The rest of the paper is organized as follows. In Section 2 we define the reduction to condensed forms for the solution of dense eigenvalue and singular-value problems. Section 3 presents the target AMP and the implementation of Level-2 and Level-3 BLAS on the AMP. Section 4 describes the keys towards performance optimization of TSOR routines. Section 5 presents a performance model that is leveraged to determine the optimal algorithmic block size. Finally, we present the experimental results in Section 6, and we draw some conclusions in Section 7.

2. Reduction to condensed forms

Given a diagonalizable square matrix A, of order n, an eigendecomposition satisfies

$$AX = X\Lambda, \tag{1}$$

where the $n \times n$ diagonal matrix $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ contains the eigenvalues of A, and the columns of the $n \times n$ nonsingular matrix X contain the corresponding eigenvectors [1]. A singular value decomposition (SVD) of an $m \times n$ matrix A is defined as

$$A = U\Sigma V^T, \tag{2}$$

Download English Version:

https://daneshyari.com/en/article/11002810

Download Persian Version:

https://daneshyari.com/article/11002810

Daneshyari.com