East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.300510.250311a

Vol. **1**, No. 2, pp. 187-196
May 2011

# Optimization of the Multishift QR Algorithm with Coprocessors for Non-Hermitian Eigenvalue Problems

Takafumi Miyata[*,1], Yusaku Yamamoto[2], Takashi Uneyama[3], Yoshimasa Nakamura[4] and Shao-Liang Zhang[1]

[1] *Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan.*
[2] *Graduate School of System Informatics, Kobe University, 1-1 Rokkodai-cho, Nada-ku, Kobe 657-8501, Japan.*
[3] *Institute for Chemical Research, Kyoto University, Gokasho, Uji 611-0011, Japan.*
[4] *Graduate School of Informatics, Kyoto University, 36-1 Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan.*

*Received 30 May 2010; Accepted (in revised version) 25 March 2011*

*Available online 7 April 2011*

**Abstract.** The multishift QR algorithm is efficient for computing all the eigenvalues of a dense, large-scale, non-Hermitian matrix. The major part of this algorithm can be performed by matrix-matrix multiplications and is therefore suitable for modern processors with hierarchical memory. A variant of this algorithm was recently proposed which can execute more computational parts by matrix-matrix multiplications. The algorithm is especially appropriate for recent coprocessors which contain many processor-elements such as the CSX600. However, the performance of the algorithm highly depends on the setting of parameters such as the numbers of shifts and divisions in the algorithm. Optimal settings are different depending on the matrix size and computational environments. In this paper, we construct a performance model to predict a setting of parameters which minimizes the execution time of the algorithm. Experimental results with the CSX600 coprocessor show that our model can be used to find the optimal setting.

**AMS subject classifications**: 65F15, 65Y10, 65Y20

**Key words**: Eigenvalues, multishift QR algorithm, bulge-chasing, performance modeling.

## 1. Introduction

We consider computing all the eigenvalues of the standard eigenvalue problem

$$A\boldsymbol{x} = \lambda \boldsymbol{x}, \ \ \boldsymbol{x} \neq \boldsymbol{0} \tag{1.1}$$

[*]Corresponding author. *Email addresses:* miyata@na.cse.nagoya-u.ac.jp (T. Miyata), yamamoto@cs.kobe-u.ac.jp (Y. Yamamoto), uneyama@scl.kyoto-u.ac.jp (T. Uneyama), ynaka@amp.i.kyoto-u.ac.jp (Y. Nakamura), zhang@na.cse.nagoya-u.ac.jp (S.-L. Zhang)

 

where $A$ is an $n \times n$ complex non-Hermitian matrix. We suppose that the matrix $A$ is dense and large-scale. Examples of non-Hermitian eigenvalue problems arising in scientific computing and many other applications are given in [1].

In the standard procedure to solve the problem in Eq. (1.1), the non-Hermitian matrix $A$ is first reduced to a Hessenberg form by the Householder algorithm [10]. After that, the Hessenberg matrix is transformed to a Schur form by the QR algorithm [8, 9, 13]. These two steps consist of unitary transformations, i.e., the eigenvalues $\lambda$ of the matrix $A$ are not changed by these steps and are finally given by the diagonal elements of the Schur form. Among these steps, the second step by the QR algorithm requires more computational work. Hence it is necessary to speed up the QR algorithm. The promising approach is a block implementation of the QR algorithm which is referred to as the multishift QR algorithm [2, 3]. The algorithm can perform most of the computational work by matrix-matrix multiplications and is therefore suitable for modern processors with hierarchical memory.

Recently, a recursive implementation of the multishift QR algorithm was proposed [18] to perform more computational parts by matrix-matrix multiplications. The algorithm is especially appropriate for recent coprocessors which have many processor-elements, e.g., GRAPE-DR [11] and CSX600 [4]. The performance of the recursive algorithm highly depends on the setting of parameters in the algorithm. Optimal settings are different depending on the matrix size $n$ and computational environments.

In this paper, we optimize the recursive multishift QR algorithm for coprocessors to attain the minimal execution time of the algorithm for a given matrix size $n$ and a computational environment. We adopt the hierarchical modeling approach [5–7] and construct a performance model of the algorithm. For a given set of parameters, our model provides the prediction of the execution time of the algorithm. By using the model, we can predict the execution time for several sets of parameters and choose the optimal one among them before running the algorithm.

Note that the hierarchical modeling approach has long been exploited to optimize the multishift QR algorithm [14, 17]. Here we consider optimizing the recursive version of the multishift QR algorithm. This approach enables us to fully exploit the potential of recent coprocessors.

This paper is organized as follows. The multishift QR algorithm and its recursive version are described in Section 2. We give performance modeling of the recursive algorithm in Section 3. Experimental results with the CSX600 coprocessor are presented in Section 4. Section 5 gives some concluding remarks and future work. Throughout this paper, the conjugate transpose is denoted by $(\cdot)^*$. The $n \times n$ identity matrix is denoted by $I$.

## 2. The Multishift QR Algorithm and Its Recursive Implementation

In this section, we review some of the standard facts on the multishift QR algorithm [3] and its variant to fully utilize coprocessors [18]. We focus on computing all the eigenvalues of a complex Hessenberg matrix.