Theoretical Computer Science 412 (2011) 1470–1483

Contents lists available at ScienceDirect

Theoretical Computer Science

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

# Dynamic normal forms and dynamic characteristic polynomial☆

Gudmund Skovbjerg Frandsen[a,*], Piotr Sankowski[b,c,**]

[a] Department of Computer Science, University of Aarhus, Aabogade 34, DK-8200 Aarhus N, Denmark

[b] Warsaw University, Poland

[c] University of Rome "La Sapienza", Italy

ARTICLE INFO

*Keywords:*
Characteristic polynomial
Matrix normal form
Eigenproblem
Dynamic algorithm
Lower bound

ABSTRACT

We present the first fully dynamic algorithm for computing the characteristic polynomial of a matrix. In the generic symmetric case, our algorithm supports rank-one updates in $O(n^2 \log n)$ randomized time and queries in constant time, whereas in the general case the algorithm works in $O(n^2 k \log n)$ randomized time, where $k$ is the number of invariant factors of the matrix. The algorithm is based on the first dynamic algorithm for computing normal forms of a matrix such as the Frobenius normal form or the tridiagonal symmetric form. The algorithm can be extended to solve the matrix eigenproblem with relative error $2^{-b}$ in additional $O(n \log^2 n \log b)$ time. Furthermore, it can be used to dynamically maintain the singular value decomposition (SVD) of a generic matrix. Together with the algorithm, the hardness of the problem is studied. For the symmetric case, we present an $\Omega(n^2)$ lower bound for rank-one updates and an $\Omega(n)$ lower bound for element updates.

© 2010 Elsevier B.V. All rights reserved.

## 1. Introduction

The computation of the *characteristic polynomial* (CP) of a matrix and the eigenproblem are two important problems in linear algebra and they find an enormous number of applications in mathematics, physics and computer science. Until now almost nothing about the dynamic complexity of these problems has been known. The CP problem is strongly related to the computation of the polycyclic Hessenberg form or the Frobenius Normal Form (FNF)—known also as the rational canonical form. All of the efficient algorithms for CP are based on the FNF computation [1–5]. The algorithms that use fast matrix multiplication work in Las-Vegas time [1], $\tilde{O}(n^\omega)$[1] deterministic time [4] or $O(n^\omega)$ Las-Vegas time [5]. Algorithms obtained using so-called black-box approach work in $\tilde{O}(nm)$ time and are Las-Vegas type [2] or Monte-Carlo type [3]—here $m$ is the number of nonzero entries in the matrix. The latter bound holds only in the generic case, whereas the fastest general algorithm works in $\tilde{O}(\mu nm)$ [3], where $\mu$ is the number of distinct invariant factors of the matrix.

All of these results have been obtained very recently. In this paper, we are trying to understand the dynamic complexity of these fundamental problems by devising efficient algorithms and by proving matching lower bounds. Note that in this paper and in all of the papers cited above, we study the arithmetic complexity of the problem, i.e., the notion of time is equivalent to the count of arithmetic operations and discrete control operations. More strictly speaking, we work in the real RAM model, for details see [6].

☆ Extended version of paper presented at ICALP'08 [G.S. Frandsen, P. Sankowski, Dynamic Normal Forms and Dynamic Characteristic Polynomial, in: Automata, Languages and Programming. Part I, in: Lecture Notes in Comput. Sci., vol. 5125, Springer, Berlin, 2008, pp. 434–446].

* Corresponding author.

** Corresponding address: Institute of Informatics, Warsaw University, Ul. Banacha 2, 02-097 Warsaw, Poland.
*E-mail addresses:* gudmund@cs.au.dk (G.S. Frandsen), piotr.sankowski@gmail.com (P. Sankowski).

[1] Throughout the paper, $f(n) = \tilde{O}(g(n))$ is shorthand for $f(n) = O(g(n) \log^k n)$ for some $k$. Essentially, it is Big-$O$, ignoring logarithmic factors.

0304-3975/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2010.11.049

In the first part of the paper, we consider the problem of computing the normal form of a real (complex) $n \times n$ dynamic matrix $A$. We assume that the matrix can be changed with use of rank-one updates, i.e., for two $n$-dimensional column vectors $a$ and $b$ we allow updates of the form $A := A + ab^T$. We want to dynamically compute matrices $Q$ and $F$ such that $A = Q^{-1}FQ$, where $Q$ is the unitary similarity transformation and $F$ is the normal form in question. The algorithm should support queries to $F$ as well as vector queries to $Q$, i.e., given vector $v$ it should be able to return $Qv$ or $Q^{-1}v$. We present the following fully dynamic Monte Carlo algorithms for this problem:

- for generic symmetric matrices—an algorithm for tridiagonal normal form supporting updates in $O(n^2 \log n)$ worst-case time,
- for general matrices—an algorithm for Frobenius normal form (for definition see Section 2.1.2) supporting updates in $O(kn^2 \log n)$ worst-case time, where $k$ is the number of invariant factors of the matrix.

In the static algorithms saying worst-case usually implies that the algorithm is deterministic. However, here in dynamic setting it means that each update is handled in the same time without amortization over the operations. In the above algorithms, the matrix $F$ is computed explicitly, so element queries for $F$ are answered in $O(1)$ time whereas vector queries can be answered in $O(n)$ time. On the other hand, vector queries to $Q$ are answered in $O(n^2 \log n)$ worst-case time. After each update, the algorithms can compute the characteristic polynomial explicitly and hence support queries for CP in constant time. These are the first known fully dynamic algorithms for the CP and normal form problems. Our results are based on a general result which can be applied to any normal form, under condition of availability of a static algorithm for computing the normal form of a sparse matrix. For the completeness of the presentation, we have included the full algorithm for generic symmetric matrices, whereas the included algorithm for Frobenius normal form is the most universal result.

This algorithm can be extended to solve the dynamic eigenproblem, i.e., we are asked to maintain with relative error $2^{-b}$ the eigenvalues $\lambda_1, \ldots, \lambda_n$ and a matrix $Q$ composed of eigenvectors. In generic case, our algorithm supports updates in $O(n \log^2 n \log b + n^2 \log n)$ worst-case time, queries to $\lambda_i$ in constant time and vector queries to $Q$ in $O(n^2 \log n)$ worst-case time. Note that the relative error $2^{-b}$ is immanent even in the exact arithmetic model, i.e., the eigenvalues can only be computed approximately (for more details please see [7]). Additionally, in $O(n^2 \log n + m \log m)$ worst-case time, we can compute for any polynomial $p$ of degree $m$ and any $1 \leq i \leq n$ the vector $p(A)e_i$, i.e., a given column of the matrix polynomial $p(A)$.

Let $A$ be a real $m \times n$ matrix, $m \geq n$. The singular value decomposition (SVD) for $A$ consists in two orthogonal matrices $U$ and $V$ and a diagonal matrix $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ with nonnegative real entries $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$ (the singular values) such that $A = U\Sigma V^T$. Our dynamic SVD problem considers maintaining the SVD under rank-one updates and two query operations: one operation returns elements of $\Sigma$ and another returns the rank $r$ approximation to $A$, i.e., given $r$ and $v$ return $U\Sigma_r V^T v$, where $\Sigma_r = \mathrm{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$. Our algorithm for SVD supports updates in $O(n \log^2 n \log b + n^2 \log n + nm)$ worst-case time in the generic case, queries to $\Sigma$ in constant time and rank $r$ approximation query in $O(n^2 \log n + nm)$ worst-case time. The computations are with a relative error $2^{-b}$.

Accompanying the above upper bounds, we provide some lower bounds for the problem of computing the characteristic polynomial. The lower bounds are formulated in the model of history-dependent algebraic computation trees [8]. One should note that our algorithms for computing the CP fit into this model. We use the technique developed and used by [8] for proving $\Omega(n)$ lower bounds for several dynamic matrix problems. The technique has been used later to prove an $\Omega(n)$ lower bound for the matrix rank problem [9]. Here, we significantly extend the technique to show an $\Omega(n^2)$ lower bound for the problem of computing the characteristic polynomial in the case of column updates. This is the first known result of this type. A column update can be realized with the use of one rank-one update.

The paper is organized as follows. In the next subsection, we motivate our study by reviewing possible applications within the scope of computer science. Nevertheless, note that the eigenproblem is the main method to study physical systems and our result could be applied to speed-up the physical computations in the case when system parameters can be changed. In Section 2, we introduce the algorithms mentioned above. Section 3 includes the description of the obtained lower bound.

### 1.1. Applications and earlier work

Our result delivers a general framework for solving many problems that are based on the computation of matrix normal forms and on the solution of the matrix eigenproblem. Hence, it generalizes a large number of problem-specific solutions and can be directly applied to a broad spectrum of problems. Until now, it has been known how to dynamically compute the lowest coefficient of the CP, i.e., the determinant [10] and the rank of the matrix [9]. Our paper generalizes these two results as the CP can be used for both computing the determinant and in a rather simple way for computing the matrix rank using reductions presented in [11,12].

Our algorithms can be used to maintain dynamic information about the spectrum of a graph. Spectral graph theory has a large number of applications (see e.g. [13]) and delivers a way to compute numerous information about the graph. One of the possible applications is a dynamic testing of graph isomorphism, where one of the basic tools is a characteristic polynomial of the adjacency or Laplacian matrix [14].

Our algorithms for computing eigenvalues and eigenvectors can be used for both dynamically approximating the size of the graph partition and for finding good candidates for partition, e.g., the second smallest eigenvalue is related to the minimum partition size [15,16]. There are several spectral methods for finding partitions and clusters in the graphs

Download English Version:

https://daneshyari.com/en/article/10333939

Download Persian Version:

https://daneshyari.com/article/10333939

Daneshyari.com