Solving the Generalized Sylvester Equation with a Systolic Library

Gloria Martínez¹, Germán Fabregat¹, and Vicente Hernández²

¹ Dpt. de Informática, Univ. Jaume I, Campus Riu Sec
E-12071 Castellón, Spain
{martine, fabregat}@inf.uji.es

² D.S.I.C., Univ. Politécnica de Valencia, Camino de Vera s/n
E-46071 Valencia, Spain
vherand@dsic.upv.es

Abstract. The study of the solution of the Generalized Sylvester Equation and other related equations is a good example of the role played by matrix arithmetic in the field of Modern Control Theory. We describe the work performed to develop systolic algorithms for solving this equation, in a fast and effective way. The presented results show that the design methodology used allowed us to propose the use of Systolic Libraries, that is, reusable systolic arrays that can be implemented taking profit of the use of FPGA technology. In this paper we show how it is feasible to solve the Generalized Sylvester Equation using basic modules of Linear Algebra that can be implemented on versatile systolic arrays.

1 Introduction

The Generalized Sylvester Equation, \( AXB + CXD = E \), with \( A, C \in \mathbb{R}^{m \times m}, B, D \in \mathbb{R}^{n \times n} \) and \( X, E \in \mathbb{R}^{m \times n} \), and some simpler derived equations such as the Sylvester [7, 15], Lyapunov [13, 17] and Stein [7, 15] have multiple and important applications in the field of Control Theory [9, 7, 15].

Obtaining the solution of these equations is a suitable problem for the efficient use of parallel algorithms, due to the regular structure of the matrices. However, when real-time constraints apply to the system, the use of dedicated processors, usually implementing systolic algorithms in VLSI is required. We have recently presented several works [10, 12] showing that a modular approach to systolic algorithms is a suitable way of building fast, reconfigurable solutions to be implemented in FPGA devices to obtain cost-effective custom processors to solve different problems.

The starting point is a new design methodology [10] based on the Kronecker Product and Vec-Function operators. Algorithms obtained this way are easy to parallelize because they consist of combinations of basic, widely studied operations (Solve a triangular equation system, Gaxpy, Saxpy, QR decomposition of a Hessenberg matrix, . . .), and the required data flow is well structured to pass from one functional block to another without intermediate storage.
Extending these results, we have compiled in a Systolic Library for Linear Algebra all the basic modules, following the same principle of modular programming that generated other sequential and parallel environments [1], [18]. For the modules of this library [11] to be useful to solve any problem in their application field, two restrictions hold: (1) all the systolic arrays must share a compatible data flow, to allow results from one of them be forwarded to another, and (2) the arrays must be designed to process problems of any size. These two restrictions have been satisfied using dynamic arrays and applying the DBT transformation [14] on the basic operations of the linear algebra.

The application described in this paper is a good example of the use of the Systolic Library. The first step to solve the Generalized Sylvester Equation, following the method proposed by Golub, Nash and Van Loan [4], is transforming the original problem $A'X'B' + C'X'D' = E'$ into $AXB + CXD = E$ using orthogonal similarity transformations on the pencils $A' - \lambda C'$ and $D' - \lambda B'$ to obtain their Generalized Schur Forms (that is, $P_1^T (A - \lambda C) P_2^T = A' - \lambda C'$ and $Q_1^T (D - \lambda B) Q_2^T = D' - \lambda B'$). The coefficient matrices of the resulting equation are in a condensed form. We have worked on the solution for three cases [10]: first, when all of them are triangular (Triangular Case). Second, when $A$ is Schur or Hessenberg and the others triangular (Hessenberg Case). Third, when both matrices $A$ and $D$ are Schur (General Case). The study of the two first cases has made possible the development of the basic arrays; the study of the general case allowed us to prove how the collection of routines obtained were efficient (and sufficient) to solve more general and complex problems.

Section 2 presents the basis of the methodology for developing the algorithms: the definition of Kronecker Product and Vector Function of a matrix. Section 3 describes the main operations to be solved when studying the solution of the Generalized Sylvester Equation in the General Case. Then section 4 shows how to use the library to implement this operation. Finally section 5 concludes and presents the ongoing work.

2 Applying the Methodology of Design

The methodology used to solve the Generalized Sylvester Equation, described in [10], is based on the definition of the Kronecker Product and Vec-Function of a matrix. The properties of both operators [3] can be applied to simplify the structure of the problem. Concretely, by applying them to the equation $AXB + CXD = E$, the linear equation system $(B^T \otimes A + D^T \otimes C)vec(X) = vec(E)$, shown in figure 1, is obtained [1]. The resulting system, too huge to be of practical implementation, offers a clear representation of the data dependencies and a simple expression of the basic steps required to solve the problem.

The structure, similar to an upper triangular system, suggests the application of the Back Substitution Algorithm to solve the problem. For example, an intuitive and simple method would be to obtain the value of $x_n$ and then update the value of $x_{n-1}$.