A DECOMPOSITION-BASED CONTROL STRATEGY FOR LARGE, SPARSE DYNAMIC SYSTEMS

ALEKSANDAR I. ZEČEVIĆ AND DRAGOSLAV D. ŠILJAK

Received 20 July 2004

We propose a new output control design for large, sparse dynamic systems. A graph-theoretic decomposition is used to cluster the states, inputs, and outputs, and to identify an appropriate bordered block diagonal structure for the gain matrix. The resulting control law can be easily implemented in a multiprocessor environment with a minimum of communication. A large-scale problem is considered to demonstrate the validity of the proposed approach.

1. Introduction

The control of large-scale dynamic systems has attracted a great deal of attention over the past few decades (e.g., [15, 16]). Systems of this type are characterized by high dimensionality, information structure constraints, and uncertainty, all of which pose significant theoretical and practical challenges. The central problem in this context is to develop control strategies that are computationally efficient and can ensure robustness with respect to uncertainties and nonlinearities in the system. The obtained feedback laws must also be easy to implement, preferably in a multiprocessor environment and with a minimal exchange of data.

A critical step in any large-scale design is the identification of a decomposition that can fully exploit the sparsity of the system matrix. The advantages of working with sparse matrices have been recognized as early as the 1960s, when it was established that an appropriate permutation can dramatically reduce the computational effort needed for solving large systems of linear equations [11, 19]. Since that time, a substantial body of literature has become available on this subject, including a number of excellent surveys [4, 6, 7].

With the advent of parallel computing, several new objectives emerged for sparse matrix ordering, perhaps the most important one being the identification of structures that are suitable for a multiprocessor environment. The simplest example of such a structure would be a matrix that can be expressed as

$$A = A_0 + \varepsilon A_C, \tag{1.1}$$

DOI: 10.1155/MPE.2005.33

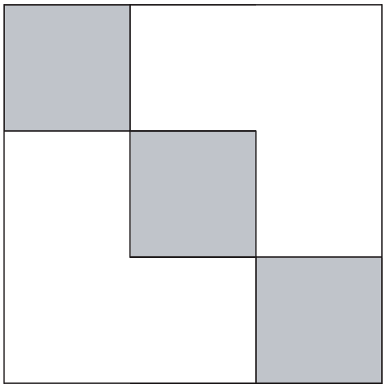

Figure 1.1. Matrix A_0 with a block diagonal structure.

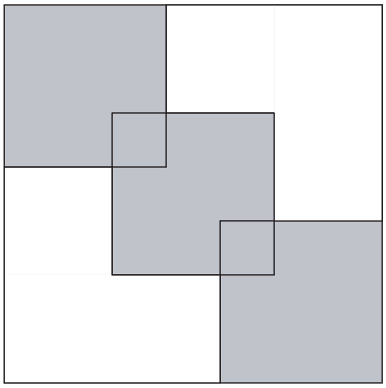

Figure 1.2. Matrix A_0 with overlapping blocks.

where A_0 has the standard block diagonal form shown in Figure 1.1, ε is a small positive number, and all the elements of A_C are smaller than one in magnitude. A matrix with this property is said to have an *epsilon decomposition*, and the necessary permutation can be easily determined even for very large systems [13, 14]. One of the main advantages of this approach stems from the fact that discarding the term εA_C can significantly sparsify the problem. This aspect of the decomposition has been studied extensively, both in the context of parallel computing [2, 20] and in decentralized control (see [16] and the references therein).

An important extension of epsilon decomposition arises when A_0 has the form shown in Figure 1.2. In that case, matrix A is said to have an *overlapping epsilon decomposition* [14, 16, 22]. Problems of this type can be parallelized effectively in the expanded space, in which the overlapping blocks appear as disjoint. Solutions in the original and expanded spaces can be related using the inclusion principle and appropriate rectangular transformation matrices [10, 16].

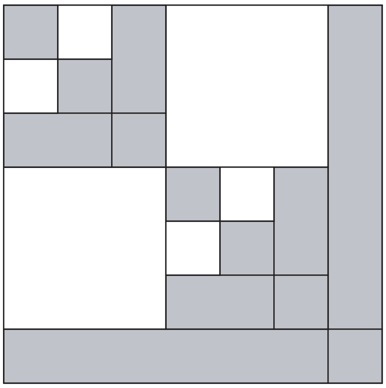

Figure 1.3. Matrix A_0 with a nested BBD structure.

The most general structure for matrix A_0 is the nested bordered block diagonal (BBD) form shown in Figure 1.3, whose potential for parallel processing has been acknowledged by numerous authors (e.g., [5, 9, 12]). Although any sparse matrix can be reordered in this way (at least, in principle), identifying an appropriate permutation matrix turns out to be a difficult graph-theoretic problem. Among the many heuristic schemes that have been developed for this purpose we single out the algorithm proposed in [21], since it represents an essential ingredient of the decomposition that will be developed in Section 2. This method was found to be effective over a wide range of nonzero patterns, and for matrices as large as $500,000 \times 500,000$.

While a considerable amount of research has been devoted to exploiting BBD forms in parallel computing, there have been virtually no applications to control. With that in mind, in this paper we propose a new output control strategy in which the gain matrix has a BBD structure.

We begin by developing an efficient graph-theoretic decomposition that is capable of identifying BBD structures with a suitable input and output assignment. Given such a decomposition, in Section 3, we propose an LMI-based optimization strategy that produces output feedback laws which can be implemented in a multiprocessor environment. A numerical example is provided in Section 4 to illustrate the effectiveness of this approach.

2. Input/output-constrained BBD decomposition

We consider the standard linear system

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx, \end{aligned} \tag{2.1}$$

where $x \in \mathbb{R}^n$ is the state of the system, $u \in \mathbb{R}^m$ is the input vector, and $y \in \mathbb{R}^q$ is the output vector, and $A = (a_{ij})$, $B = (b_{ij})$ and $C = (c_{ij})$ are large, sparse matrices of dimension

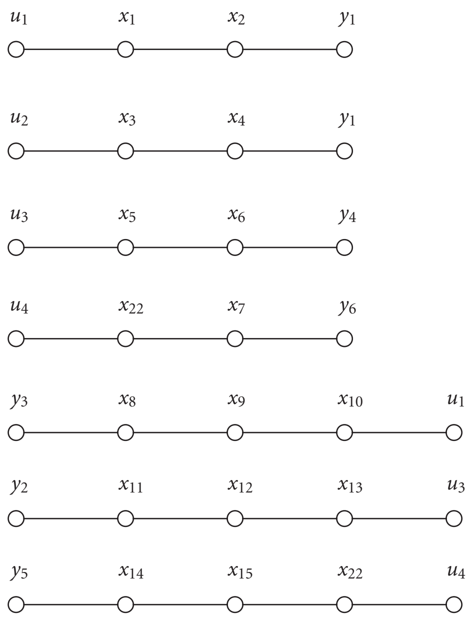

Figure 2.1. Shortest input/output paths for a sample graph.

$n \times n$, $n \times m$, and $q \times n$, respectively. In this section, our objective will be to develop a decomposition that *simultaneously* permutes matrix A into the BBD form *and* secures a compatible block-diagonal structure for matrices B and C. The proposed algorithm represents a modification of the balanced BBD decomposition method [21], with the added requirement that each diagonal block must have at least one input and one output associated with it. In the following, we will refer to such a decomposition as an *input/output-constrained* BBD decomposition.

The graph-theoretic nature of the decomposition problem requires that we associate an undirected graph $\mathbf{G}(\mathbf{V},\mathbf{E})$ with system (2.1), where $\mathbf{V}$ is a set of vertices v_i and $\mathbf{E}$ represents a set of edges e_{ij}. We will assume that $\mathbf{V} = \mathbf{X} \bigcup \mathbf{U} \bigcup \mathbf{Y}$ consists of three disjoint sets of vertices, the state set $\mathbf{X} = \{x_1,\dots,x_n\}$, the input set $\mathbf{U} = \{u_1,\dots,u_m\}$, and the output set $\mathbf{Y} = \{y_1,\dots,y_q\}$. Then, $(x_i,x_j) \in \mathbf{E}$ if and only if $a_{ij} \neq 0$, $(x_i,u_j) \in \mathbf{E}$ if and only if $b_{ij} \neq 0$, and $(x_i,y_j) \in \mathbf{E}$ if and only if $c_{ji} \neq 0$. We can further assume without loss of generality that matrix A is structurally symmetric; if that was not the case, we could simply use $A + A^T$ to obtain the decomposition.

The proposed algorithm can now be described by the following sequence of steps.

Step 2.1. The decomposition is initialized by identifying the shortest path from every input vertex to an output vertex. If any outputs remain unvisited at this point, the procedure is reversed with these outputs as the starting vertices. The initial graph is then formed as a union of the obtained paths. A typical situation after this stage of the decomposition is shown in Figure 2.1, for a system with four inputs and six outputs.

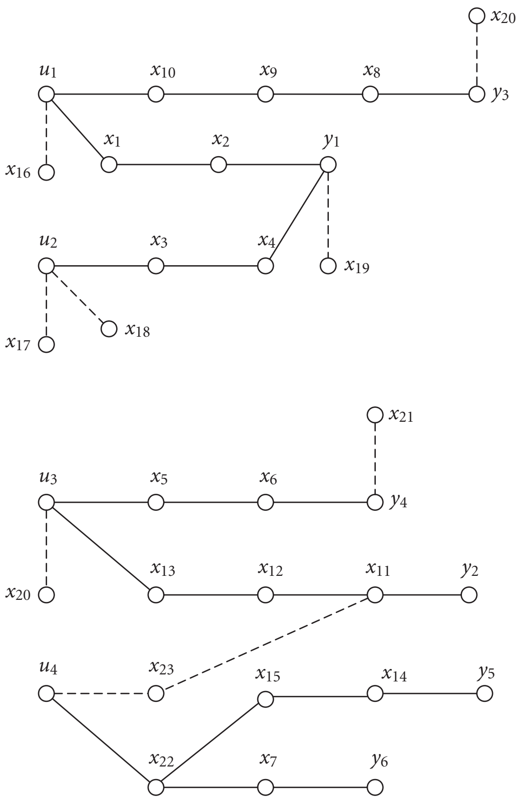

Figure 2.2. Disjoint components after Step 2.2.

Step 2.2. The second step begins by forming incidence sets

$$\mathbf{X}_u^{(i)} \equiv \{x_k \in \mathbf{X} : (x_k, u_i) \in \mathbf{E}\},$$
$$\mathbf{X}_y^{(i)} \equiv \{x_k \in \mathbf{X} : (x_k, y_i) \in \mathbf{E}\}, \tag{2.2}$$

for each $u_i \in \mathbf{U}$ and $y_i \in \mathbf{Y}$. The union of these sets

$$\bar{\mathbf{X}} = \left\{\bigcup_{i=1}^{m} \mathbf{X}_u^{(i)}\right\} \bigcup \left\{\bigcup_{i=1}^{q} \mathbf{X}_y^{(i)}\right\} \tag{2.3}$$

is then added to the graph obtained in Step 2.1. If there are any elements $x_i, x_j \in \bar{\mathbf{X}}$ such that $(x_i, x_j) \in \mathbf{E}$, these edges are added to the graph as well.

Once the new graph $\mathbf{\Gamma}$ is formed, its disjoint components $\mathbf{\Gamma}_i = \mathbf{X}_i \bigcup \mathbf{U}_i \bigcup \mathbf{Y}_i$ $(i = 1, 2, \ldots, k)$ are identified (note that by construction each set $\mathbf{\Gamma}_i$ contains at least one input and one output vertex). A possible scenario corresponding to Figure 2.1 is shown in Figure 2.2, in which all edges added in Step 2.2 are indicated by dashed lines.

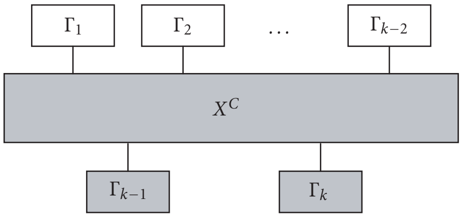

Figure 2.3. A schematic representation of the initial border.

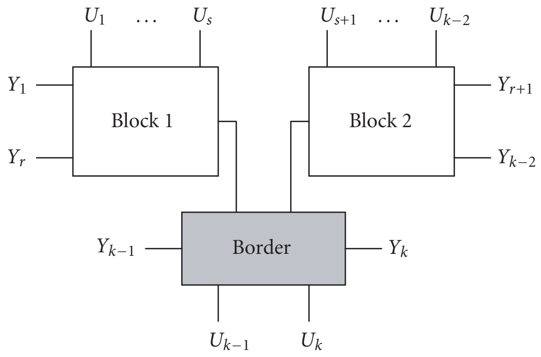

Figure 2.4. The graph after the reconnection is complete.

It is important to recognize at this point that all vertices $x_i \in \mathbf{X}$ that are incident to some input or output have been included in sets $\mathbf{\Gamma}_i$ $(i = 1,2,\dots,k)$. Vertices $x_i \notin \mathbf{\Gamma}$ constitute set $\mathbf{X}^C$, which represents a *separator* for the graph (the term implies that the removal of $\mathbf{X}^C$ from the overall graph results in disjoint sets $\mathbf{\Gamma}_i$). A schematic representation of this property is shown in Figure 2.3.

Step 2.3. In most problems, the separator $\mathbf{X}^C$ is large and needs to be reduced in order to obtain a meaningful BBD decomposition. To perform the necessary reduction, we first need to identify a subset of inputs and outputs that we wish to associate with the border block. For simplicity, let those inputs be contained in sets $\mathbf{\Gamma}_{k-1}$ and $\mathbf{\Gamma}_k$. We then form the *initial border* as the union of sets $\mathbf{X}^C$, $\mathbf{\Gamma}_{k-1}$, and $\mathbf{\Gamma}_k$ (this initial border has been shaded in Figure 2.3). The remaining sets $\mathbf{\Gamma}_i$ $(i = 1,2,\dots,k-2)$ obviously represent the corresponding *initial diagonal blocks*, and the vertices of $\mathbf{X}^C$ are now reconnected to these blocks one by one. The order in which the reconnection proceeds reflects a "greedy" strategy, whose objective is to achieve a minimal increase in the sizes of the diagonal blocks in each step. The reconnection continues for as long as at least two diagonal blocks remain. The situation at the end of this stage is shown in Figure 2.4, in which the final border is shaded.

Several comments are in order at this point.

Remark 2.4. Reconnecting any further vertices from the border would require merging blocks 1 and 2. In that sense, the obtained border is minimal.

Remark 2.5. The decomposition secures that each of the diagonal blocks has *at least* one input and one output associated with it, since only vertices belonging to $\mathbf{X}^C$ are reconnected. In addition, the preselected inputs and outputs remain associated with the border block.

Remark 2.6. The reconnection scheme uses the same data structures and tie-breaking criteria as the one described in [21]. For further details, the reader is referred to this paper.

Remark 2.7. The fact that each diagonal block has at least one input associated with it does not automatically guarantee controllability. As shown in [16], it is possible to construct a graph that is input reachable, but generically uncontrollable due to dilation. However, this type of situation occurs very rarely.

Remark 2.8. The decomposition proceeds in a nested manner by separately applying Steps 2.1, 2.2, and 2.3 to each diagonal block. The algorithm terminates when one of the following two conditions is encountered:

(1) a block has no more than two inputs and/or two outputs associated with it, making a further decomposition impossible,
(2) the resulting border set is unacceptably large. In this case, the block cannot be decomposed in a meaningful way (or, to put it differently, the block is not sufficiently sparse).

The proposed decomposition produces a permutation matrix P that corresponds to the obtained graph-theoretic partitioning. The generic structure of matrix $A_{\text{BBD}} = P^T AP$ is the one shown in Figure 1.3 (possibly with additional levels of nesting), while $B_D = P^T B$ and $C_D = CP$ consist of diagonal blocks compatible with those of A_{BBD}.

It should also be observed that the proposed decomposition can easily be extended to the form

$$\hat{A} = A_{\text{BBD}} + \varepsilon_1 A_C; \qquad \hat{B} = B_D + \varepsilon_2 B_C; \qquad \hat{C} = C_D + \varepsilon_3 C_C, \tag{2.4}$$

where ε_1, ε_2, and ε_3 are small positive numbers, and all elements of A_C, B_C, and C_C are smaller than one in magnitude. This represents a combination of epsilon decomposition and input-constrained BBD decomposition. The structure in (2.4) can be obtained by discarding all elements of A, B, and C such that $|a_{ij}| \le \varepsilon_1$, $|b_{ij}| \le \varepsilon_2$ and $|c_{ij}| \le \varepsilon_3$ *prior* to executing the decomposition. In many cases, this can significantly increase the sparsity of A and produce a more flexible BBD structure. It also guards against poorly controllable or observable blocks, by discarding small entries in matrices B and C.

3. Output feedback design for a multiprocessor environment

We consider a nonlinear system described by the differential equations

$$\begin{aligned} \dot{x} &= Ax + h(x) + B_D u, \\ y &= C_D x, \end{aligned} \tag{3.1}$$

where $x \in \mathbb{R}^n$ is the state of the system, $u \in \mathbb{R}^m$ is the input vector, and $y \in \mathbb{R}^q$ is the output vector. We will assume that A is an $n \times n$ sparse matrix that has been permuted into a BBD form, with diagonal blocks of dimension $n_i \times n_i$ $(i = 1, 2, \ldots, N)$. As described in Section 2, the same permutation ensures that matrices $B_D = \operatorname{diag}\{B_1, \ldots, B_N\}$ and $C_D = \operatorname{diag}\{C_1, \ldots, C_N\}$ consist of $n_i \times m_i$ and $q_i \times n_i$ diagonal blocks, respectively. The nonlinear function $h : \mathbb{R}^n \to \mathbb{R}^n$ is piecewise-continuous in x, satisfying $h(0) = 0$. This term is assumed to be uncertain, but bounded by a quadratic inequality

$$h^T(x)h(x) \le \alpha^2 x^T H^T H x, \tag{3.2}$$

where H is a constant matrix, and $\alpha > 0$ is a scalar parameter.

As shown in [17, 18], a centralized state feedback law $u = Kx$ can be obtained for the system in (3.1) by solving the following LMI problem in γ, κ_Y, κ_L, Y, and L.

Problem 3.1. Minimize $a_1\gamma + a_2\kappa_Y + a_3\kappa_L$ subject to

$$\begin{gathered} Y > 0, \\ \begin{bmatrix} AY + YA^T + B_D L + L^T B_D^T & I & YH^T \\ I & -I & 0 \\ HY & 0 & -\gamma I \end{bmatrix} < 0, \\ \gamma - \frac{1}{\bar{\alpha}^2} < 0, \\ \begin{bmatrix} -\kappa_L I & L^T \\ L & -I \end{bmatrix} < 0, \qquad \begin{bmatrix} Y & I \\ I & \kappa_Y I \end{bmatrix} > 0. \end{gathered} \tag{3.3}$$

If the optimization is feasible, the gain matrix computed as $K = LY^{-1}$ stabilizes the closed-loop system for *any* nonlinearity that satisfies (3.2). From that standpoint, the obtained value for parameter α can be viewed as a measure of robustness.

Since our objective is to design *output* feedback that can be implemented in a multiprocessor environment, it will be necessary to introduce several modifications to optimization Problem 3.1. We will consider three possible scenarios.

A. Decentralized output feedback. In this case, the control law

$$u_i = K_i y_i \quad (i = 1, 2, \ldots, N) \tag{3.4}$$

utilizes only locally available output information. To secure the desired decentralized output feedback structure, the following additional requirements need to be incorporated into Problem 3.1.

Requirement 3.2. Matrix Y must have the form

$$Y = Y_0 + U_D Y_C U_D^T, \tag{3.5}$$

where Y_0 and Y_C are unknown $n \times n$ and $q \times q$ block diagonal matrices, respectively. The diagonal blocks of Y_0 have dimension $n_i \times n_i$, and those of Y_C have dimension $q_i \times q_i$.

Requirement 3.3. Matrix U_D is a user-defined block diagonal matrix of dimension $n \times q$, consisting of $n_i \times q_i$ blocks.

Requirement 3.4. Matrix Y_0 must satisfy the equality constraint

$$Y_0 C_D^T = U_D. \tag{3.6}$$

Note that if U_D is chosen as $U_D = C_D^T$, this condition is automatically satisfied by any matrix Y_0 of the form

$$Y_0 = Q_D Y_Q Q_D^T + C_D^T (C_D C_D^T)^{-1} C_D, \tag{3.7}$$

where Q_D is an $n \times (n-q)$ block diagonal matrix such that

$$Q_D^T C_D^T = 0. \tag{3.8}$$

In that case, we need to compute an $(n-q) \times (n-q)$ matrix Y_Q, which is symmetric and block diagonal, with $(n_i - q_i) \times (n_i - q_i)$ blocks.

Requirement 3.5. Matrix L must have the form

$$L = L_C U_D^T, \tag{3.9}$$

where L_C is a block diagonal $m \times q$ matrix with blocks of dimension $m_i \times q_i$.

To understand the rationale for Requirements 3.2, 3.3, 3.4, and 3.5, we should recall that Y^{-1} can be expressed using the Sherman-Morrison formula as (e.g., [8])

$$Y^{-1} = Y_0^{-1} - SRU_D^T Y_0^{-1}, \tag{3.10}$$

where

$$\begin{aligned} S &= Y_0^{-1} U_D Y_C, \\ R &= [I + U_D^T S]^{-1}. \end{aligned} \tag{3.11}$$

It is now easily verified that Requirement 3.4 implies $LY^{-1} = K_D C_D$, with

$$K_D = L_C (I - U_D^T SR). \tag{3.12}$$

Requirements 3.2, 3.3, and 3.5 also ensure that K_D is a block diagonal matrix, with blocks K_i of dimension $m_i \times q_i$. This gives rise to a decentralized output feedback law of the form

$$u_i = K_i C_i x_i \quad (i = 1, 2, \ldots, N). \tag{3.13}$$

B. BBD output control. The feedback in this case is assumed to have the form

$$\begin{aligned} u_i &= K_{ii} y_i + K_{iN} y_N \quad (i = 1,2,\dots,N-1), \\ u_N &= \sum_{i=1}^{N} K_{Ni} y_i, \end{aligned} \tag{3.14}$$

which corresponds to an $m \times q$ BBD gain matrix

$$K_{\text{BBD}} = \begin{bmatrix} K_{11} & 0 & \cdots & K_{1N} \\ 0 & K_{22} & \cdots & K_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ K_{N1} & K_{N2} & \cdots & K_{NN} \end{bmatrix}. \tag{3.15}$$

A control law of this type can be obtained in the same way as decentralized output feedback, the only difference being that matrix L_C in Requirement 3.5 must now be an $m \times q$ BBD matrix with a block structure that is identical to that of the gain matrix K_{BBD} in (3.15). This property follows directly from the fact that $I - U_D^T SR$ is a block diagonal matrix with $q_i \times q_i$ diagonal blocks, by virtue of Requirements 3.2, 3.3, and 3.4.

Remark 3.6. It is important to note that a BBD output control law is easily implemented in a multiprocessor environment. Indeed, any such structure can be efficiently mapped onto a tree-type parallel architecture, which guarantees a low communication overhead (e.g., [3]).

C. Proportional plus integral (PI) BBD output control. Proportional plus integral BBD output control represents a natural extension of the decentralized feedback strategy proposed in [1]. In this case, the control law is assumed to have the form

$$\begin{aligned} u_i &= K_{ii} y_i + K_{iN} y_N + F_{ii}\beta_i + F_{iN}\beta_N \quad (i = 1,2,\dots,N-1), \\ u_N &= \sum_{i=1}^{N} K_{Ni} y_i + \sum_{i=1}^{N} F_{Ni}\beta_i, \end{aligned} \tag{3.16}$$

where

$$\beta_i(t) = \int_0^t y_i(\tau)\,d\tau \quad (i = 1,2,\dots,N). \tag{3.17}$$

This type of control obviously corresponds to a pair of BBD gain matrices K_{BBD} and F_{BBD}, and can be represented in compact form as

$$u(t) = K_{\text{BBD}} y(t) + F_{\text{BBD}} \beta(t). \tag{3.18}$$

In order to determine such a control, we first define auxiliary matrices $\tilde{A}$, $\tilde{B}$, and $\tilde{H}$ as

$$\tilde{A} = \begin{bmatrix} A & 0 \\ C_D & 0 \end{bmatrix}, \qquad \tilde{B} = \begin{bmatrix} B_D \\ 0 \end{bmatrix}, \qquad \tilde{H} = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}, \tag{3.19}$$

where I_n represents an $n \times n$ identity matrix. We now propose to solve LMI Problem 3.1 using matrices $\tilde{A}$, $\tilde{B}$, and $\tilde{H}$ instead of A, B and H, with the following additional requirements.

Requirement 3.7. Matrix Y must have the form

$$Y = Y_0 + U_D Y_C U_D^T, \tag{3.20}$$

where

$$Y_0 = \begin{bmatrix} Y_{11} & 0 \\ 0 & Y_{22} \end{bmatrix}. \tag{3.21}$$

In (3.21), Y_{11} is an unknown $n \times n$ block diagonal matrix with $n_i \times n_i$ blocks, while Y_{22} and Y_C are unknown $q \times q$ block diagonal matrices with $q_i \times q_i$ blocks.

Requirement 3.8. Matrix U_D is a user-defined block diagonal matrix of dimension $(n + q) \times q$, which can be partitioned as

$$U_D = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}. \tag{3.22}$$

In (3.22), U_1 is an $n \times q$ block diagonal matrix with $n_i \times q_i$ blocks, and U_2 is an $q \times q$ block diagonal matrix with $q_i \times q_i$ blocks.

Requirement 3.9. Matrix Y_{11} must satisfy the equality constraint

$$Y_{11} C_D^T = U_1. \tag{3.23}$$

As discussed earlier, this condition can be automatically satisfied if U_1 is chosen as $U_1 = C_D^T$.

Requirement 3.10. Matrix L must have the form

$$L = L_C U_D^T, \tag{3.24}$$

where L_C is an $m \times q$ BBD matrix with a block structure identical to that of the matrices K_{BBD} and F_{BBD} in (3.18).

If optimization Problem 3.1 is feasible under these conditions, the closed loop system

$$\dot{\tilde{x}} = (\tilde{A} + \tilde{B} L Y^{-1})\tilde{x} + \tilde{h}(\tilde{x}) \tag{3.25}$$

is guaranteed to be stable for any $\tilde{h}(\tilde{x})$ satisfying

$$\tilde{h}^T(\tilde{x})\tilde{h}(\tilde{x}) \leq \alpha^2 \tilde{x}^T \tilde{H}^T \tilde{H} \tilde{x}. \tag{3.26}$$

Furthermore, Requirements 3.7, 3.8, 3.9, and 3.10 ensure that the product LY^{-1} can be factorized as

$$LY^{-1} = WV, \tag{3.27}$$

where

$$W = L_C(I - U_D^T SR),$$
$$V = U_D^T Y_0^{-1} = \begin{bmatrix} U_1^T Y_{11}^{-1} & U_2^T Y_{22}^{-1} \end{bmatrix}, \tag{3.28}$$

and S, R are the matrices introduced in (3.11). Since $U_1^T Y_{11}^{-1} = C_D$ by virtue of Requirement 3.9, we can partition LY^{-1} as

$$LY^{-1} = \begin{bmatrix} WC_D & WU_2^T Y_{22}^{-1} \end{bmatrix}. \tag{3.29}$$

Defining $K = W$ and $F = WU_2^T Y_{22}^{-1}$, it is not difficult to verify that both of these matrix have a BBD structure identical to (3.15). If we now use these matrices to form the control law proposed in (3.18), the closed-loop system (3.1) is guaranteed to be stable for all $h(x)$ such that

$$\left\| \frac{\partial h}{\partial x} \right\| \leq \alpha, \tag{3.30}$$

where $\|\cdot\|$ denotes the Euclidean norm. To see why this is so, we differentiate the state and output equations in (3.1). Setting $v = \dot{x}$, we obtain

$$\dot{v} = Av + B_D \dot{u} + \tilde{h}(x, v),$$
$$\dot{y} = C_D v \tag{3.31}$$

with

$$\tilde{h}(x, v) = \frac{\partial h}{\partial x} v. \tag{3.32}$$

Defining $\tilde{x} = [v^T \quad y^T]^T$ and observing that

$$\dot{u} = KC_D v + Fy \equiv LY^{-1}\tilde{x}, \tag{3.33}$$

the closed-loop system now takes the form (3.25). Since

$$\tilde{h}^T(\tilde{x})\tilde{h}(\tilde{x}) = v^T \left(\frac{\partial h}{\partial x} \right)^T \left(\frac{\partial h}{\partial x} \right) v \leq \alpha^2 v^T v = \alpha^2 \tilde{x}^T \tilde{H}^T \tilde{H} \tilde{x} \tag{3.34}$$

by virtue of (3.19) and (3.30), it follows that $\tilde{x} = 0$ is a stable equilibrium of (3.25). This further implies that the corresponding closed-loop equilibrium of (3.1) must be stable as well. We should note that this equilibrium need not be at the origin, since $\lim_{t\to\infty} u(t) \neq 0$ for the control proposed in (3.18).

4. An illustrative example

In this section, we will compare the three proposed output feedback strategies from the standpoint of robustness and computational complexity. As a typical test case, we consider a sparse system with 55 states, 8 inputs, and 8 outputs (the nonzero elements of

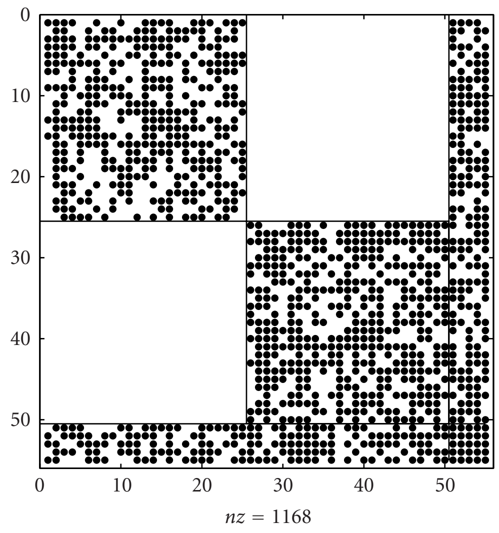

Figure 4.1. Matrix A_0 after a BBD decomposition with $\varepsilon = 0.1$.

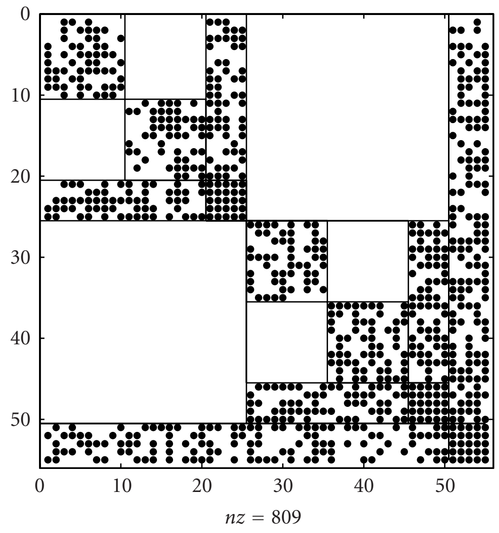

Figure 4.2. Matrix A_0 after a BBD decomposition with $\varepsilon = 0.35$.

matrices A, B, and C were chosen randomly, with a Gaussian distribution). The corresponding input/output constrained BBD decompositions of A_0 for $\varepsilon = 0.1$ and $\varepsilon = 0.35$ are shown in Figures 4.1 and 4.2, respectively.

It should be noted that the matrix in Figure 4.1 has two diagonal blocks of dimension 25×25, and a 5×5 border block. The blocks shown in Figure 4.2 are substantially

Table 4.1. Computational complexity and robustness for $\varepsilon = 0.1$.

	Number of LMI variables	Robustness parameter α
Decentralized output control	825	0.21
BBD output control	955	0.52
PI BBD output control	1245	1.10

Table 4.2. Computational complexity and robustness for $\varepsilon = 0.35$.

	Number of LMI variables	Robustness parameter α
Decentralized output control	325	0.05
BBD output control	515	0.31
PI BBD output control	765	0.98

smaller, due to the fact that more elements are discarded when ε is set to equal 0.35. This additional sparsification reduces the dimension of all border blocks to 5×5, while the remaining diagonal blocks are 10×10. We should also note in this context that the proposed combination of BBD and epsilon decompositions allows for a straightforward adaptation of the problem to a prescribed number of processors. This property, known as *scalability*, is of considerable practical importance in parallel computing.

In order to compare the three designs and the two decompositions, we utilized the proposed LMI optimization to obtain the appropriate output control laws, and established the maximal robustness bound α. As a measure of computational complexity, we also recorded the number of LMI variables associated with matrices L and Y (and also F in the case of PI BBD control). A summary of the obtained results is shown in Tables 4.1 and 4.2.

A comparison of Tables 4.1 and 4.2 shows that the PI BBD control for $\varepsilon = 0.35$ requires fewer LMI variables than any of the three designs corresponding to $\varepsilon = 0.1$. It is also readily observed that the degradation in α as ε increases is smallest in the PI BBD case. This suggests that a PI BBD output control provides the best balance between robustness and computational complexity. Such a conclusion has been supported by a number of additional numerical experiments.

5. Conclusions

In this paper, we developed a new output control strategy that is suitable for large, sparse dynamic systems. The method is based on an efficient input-constrained decomposition that simultaneously clusters the states, inputs, and outputs. The resulting bordered block diagonal form of matrix A induces the structure of the gain matrix, which is computed using LMI optimization. The obtained feedback law can be easily mapped onto a tree-type multiprocessor architecture, which guarantees low communication overhead. A large-scale example was provided to demonstrate the validity of the proposed scheme.

Acknowledgment

The work reported in this paper was supported by the NSF Grant ECS-0099469.

References

[1] A. A. Abdullah and P. A. Ioannou, *Real-time control of a segmented telescope test-bed*, Proc. 42nd IEEE Conference on Decision and Control, Maui, Vol. 1, 2003, pp. 762–767.

[2] M. Amano, A. I. Zečević, and D. D. Šiljak, *An improved block-parallel Newton method via epsilon decompositions for load-flow calculations*, IEEE Trans. Power Syst. **11** (1996), no. 3, 1519–1527.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation*, Prentice Hall, New Jersey, 1989.

[4] I. S. Duff, *A survey of sparse matrix research*, Proc. IEEE **65** (1977), 500–535.

[5] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, The Clarendon Press, Oxford, 1986.

[6] A. George, J. R. Gilbert, and J. W. H. Liu (eds.), *Graph Theory and Sparse Matrix Computation*, Springer-Verlag, New York, 1993.

[7] A. George and J. W. H. Liu, *The evolution of the minimum degree ordering algorithm*, SIAM Rev. **31** (1989), 1–19.

[8] G. Golub and C. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Maryland, 1996.

[9] M. T. Heath, E. Ng, and B. W. Peyton, *Parallel algorithms for sparse linear systems*, Parallel Algorithms for Matrix Computations, SIAM, Pennsylvania, 1990, pp. 83–124.

[10] M. Ikeda, D. D. Šiljak, and D. E. White, *An inclusion principle for dynamic systems*, IEEE Trans. Automat. Control **29** (1984), no. 3, 244–249.

[11] H. M. Markowitz, *The elimination form of the inverse and its application to linear programming*, Management Science **3** (1957), 255–269.

[12] A. Pothen, H. D. Simon, and K.-P. Liou, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM J. Matrix Anal. Appl. **11** (1990), no. 3, 430–452.

[13] M. E. Sezer and D. D. Šiljak, *Nested ϵ-decompositions and clustering of complex systems*, Automatica **22** (1986), no. 3, 321–331.

[14] ______, *Nested epsilon decompositions of linear systems: weakly coupled and overlapping blocks*, SIAM J. Matrix Anal. Appl. **12** (1991), no. 3, 521–533.

[15] D. D. Šiljak, *Large-Scale Dynamic Systems: Stability and Structure*, North-Holland, New York, 1978.

[16] ______, *Decentralized Control of Complex Systems*, Academic Press, New York, 1991.

[17] D. D. Šiljak and D. M. Stipanović, *Robust stabilization of nonlinear systems: the LMI approach*, Math. Probl. Eng. **6** (2000), no. 5, 461–493.

[18] D. D. Šiljak, D. M. Stipanović, and A. I. Zečević, *Robust decentralized turbine/governor control using linear matrix inequalities*, IEEE Trans. Power Syst. **17** (2002), no. 3, 715–722.

[19] W. F. Tinney and J. W. Walker, *Direct solutions of sparse network equations by optimally ordered triangular factorization*, Proc. IEEE **55** (1967), no. 11, 1801–1809.

[20] A. I. Zečević and N. Gačić, *A partitioning algorithm for the parallel solution of differential-algebraic equations by waveform relaxation*, IEEE Trans. Circuits Syst. I **46** (1999), no. 4, 421–434.

[21] A. I. Zečević and D. D. Šiljak, *Balanced decompositions of sparse systems for multilevel parallel processing*, IEEE Trans. Circuits Syst. I **41** (1994), no. 3, 220–233.
[22] ———, *A block-parallel Newton method via overlapping epsilon decompositions*, SIAM J. Matrix Anal. Appl. **15** (1994), no. 3, 824–844.

Aleksandar I. Zečević: Department of Electrical Engineering, Santa Clara University, Santa Clara, CA 95053, USA
E-mail address: azecevic@scu.edu

Dragoslav D. Šiljak: Department of Electrical Engineering, Santa Clara University, Santa Clara, CA 95053, USA
E-mail address: dsiljak@scu.edu