Cost-Effective Parallel Processing for H^2/H^∞ Controller Synthesis

Yuzhen Ge[†] Layne T. Watson[‡] Emmanuel G. Collins, Jr.[*]

Abstract

A distributed version of a homotopy algorithm for solving the H^2/H^∞ mixed-norm controller synthesis problem is presented. The main purpose of the study is to explore the possibility of achieving high performance with low cost. Existing UNIX workstations running PVM (Parallel Virtual Machine) are utilized. Only the Jacobian matrix computation is distributed and therefore the modification to the original sequential code is minimal. The same algorithm has also been implemented on an Intel Paragon parallel machine. Our implementation shows that acceptable speedup is achieved and the larger the problem sizes, the higher the speedup. Comparing with the results from the Intel Paragon, the study concludes that utilizing the existing UNIX workstations can be a very cost-effective approach to shorten computation time. Furthermore, this economical way to achieve high performance computation can easily be realized and incorporated in a practical industrial design environment.

1 Introduction

H^2/H^∞ mixed-norm controller synthesis is an important and interesting technique in modern control design which provides the means for simultaneously addressing H^2 and H^∞ performance objectives. In practice such controllers provide both nominal performance (via suboptimal H^2) and robust stability (via H^∞). Hence mixed-norm synthesis provides a technique for trading off performance and robustness, a fundamental objective in control design.

The H^2/H^∞ mixed-norm problem has been addressed in a variety of settings. One treatment utilized an H^2 cost bound as the basis for an auxiliary nonconvex constrained minimization problem, which is very difficult to solve without the global convergence of homotopy methods. A successful homotopy algorithm based on the Ly form parametrization has been developed (Ge *et al.* 1994).

The H^2/H^∞ control design algorithms will be used for controller design of systems such as the four disk system of (Cannon and Rosenthal 1984). This system is especially representative of the type of vibration control problems that arise in industrial problems involving rotating turbomachinery. H^2/H^∞ design will be used to develop controllers that

---

The work of Yuzhen Ge and Emmanuel G. Collins, Jr. was supported in part by Air Force Office of Scientific Research grant F49620-95-1-0244 and the work of Layne T. Watson was supported in part by Air Force Office of Scientific Research grant F49620-92-J-0236 and Department of Energy grant DE-FG05-88ER25068/A004.

† Department of Mathematics and Computer Science, Butler University, Indianapolis, IN 46208. ge@butler.edu.

‡ Departments of Computer Science and Mathematics, Virginia Polytechnic Institute & State University, Blacksburg, VA 24061-0106. ltw@cs.vt.edu.

* Department of Mechanical Engineering, Florida A&M/Florida State University, Tallahassee, FL 32310. ecollins@eng.fsu.edu.

are robust with respect to unmodeled dynamics and also guarantee a certain measure of nominal performance.

It should be mentioned that H^2/H^∞ theory has been used in (Haddad *et al.* 1993)–(Haddad *et al.* 1994b) to develop complex structured singular value synthesis (CSSV) formalisms that *a priori* fix the structure of both the D-scales and the controller. Hence, an extension of the algorithms here will enable fixed-structure CSSV controller synthesis that blends H^2 and H^∞ performance objectives.

Practical applications often lead to large dense systems of nonlinear equations which are time-consuming to solve on a serial computer. For these systems, parallel processing may be the only feasible means to achieving solution algorithms with acceptable speed. One economical way of achieving parallelism is to utilize the aggregate power of a network of heterogeneous serial computers. In industrial environments where interactive design is often the practice, the parallel code can be easily incorporated into interactive software such as MATLAB or Mathematica with proper setup of the network computers. To the engineering users the design environment is identical. However the computations are faster.

The most expensive part of the H^2/H^∞ homotopy algorithm of (Ge *et al.* 1994) is the computation of the Jacobian matrix, which can be parallelized easily to run across an Ethernet network with little modification of the original sequential code, and which has relatively large task granularity. There is a trade-off between the programming effort and the speedup of the parallel program. To obtain a better speedup, other parts of the homotopy algorithm, such as finding the solution to the Riccati equations and the QR factorization to compute the kernel of the Jacobian matrix, need to be parallelized as well.

In this study the homotopy algorithm for H^2/H^∞ controller synthesis is parallelized to run on a network of workstations using PVM (Parallel Virtual Machine) and on an Intel Paragon parallel computer, under the philosophy that as few changes as possible are to be made to the sequential code while achieving an acceptable speedup. The parallelized computation is that of the Jacobian matrix, which is carried out in the master-slave paradigm by functional parallelism, that is, each machine computes a different column of the Jacobian matrix with its own data. Unless the Riccati equation solver is parallelized, there is a large amount of data needed for each slave process at each step of the homotopy algorithm. To avoid sending too many large messages across the network or among different nodes on the Intel Paragon, all slave processes repeat part of the computation done by the master process, which therefore decreases the speedup of the parallel computation.

The speedups of the parallel code are compared as the number of workstations increases or as the number of nodes increases on an Intel Paragon and as the size of the problem varies. A reasonable speedup can be achieved using an existing network of workstations compared to that of using an expensive parallel machine, the Intel Paragon. It is demonstrated that for a large problem, the approach of using a network of workstations to parallelism is feasible and practical, and provides an efficient and economical computational method to parallelize a homotopy based algorithm for H^2/H^∞ controller synthesis in a workstation-based interactive design environment.

2 The H^2/H^∞ controller synthesis problem

The LQG controller synthesis problem with an H^∞ performance bound can be stated as follows: given the n-th order stabilizable and detectable plant

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,u(t) + D_1 w(t), \\ y(t) &= C\,x(t) + D_2 w(t), \end{aligned} \tag{1}$$

where $A \in \mathbf{R}^{n\times n}$, $B \in \mathbf{R}^{n\times m}$, $C \in \mathbf{R}^{l\times n}$, $D_1 \in \mathbf{R}^{n\times p}$, $D_2 \in \mathbf{R}^{l\times p}$, $D_1 D_2^T = 0$, and $w(t)$ is p-dimensional white noise, find a n_c-th order dynamic compensator

$$\begin{aligned} \dot{x}_c(t) &= A_c\,x_c(t) + B_c\,y(t), \\ u(t) &= C_c\,x_c(t), \end{aligned} \tag{2}$$

where $A_c \in \mathbf{R}^{n_c\times n_c}$, $B_c \in \mathbf{R}^{n_c\times l}$, $C_c \in \mathbf{R}^{m\times n_c}$, and $n_c \le n$, which satisfies the following criteria:

(i) the closed-loop system (1)–(2) is asymptotically stable, i.e., $\tilde{A} = \begin{pmatrix} A & B\,C_c \\ B_c C & A_c \end{pmatrix}$ is asymptotically stable;

(ii) the $q_\infty \times p$ closed-loop transfer function

$$H(s) \equiv \tilde{E}_\infty (sI_{\tilde{n}} - \tilde{A})^{-1} \tilde{D},$$

from $w(t)$ to

$$z(t) = E_{1\infty} x(t) + E_{2\infty} u(t), \tag{3}$$

where $\tilde{E}_\infty = (\,E_{1\infty} \quad E_{2\infty} C_c\,)$ ($E_{1\infty} \in \mathbf{R}^{q_\infty\times n}$, $E_{2\infty} \in \mathbf{R}^{q_\infty\times m}$, $E_{1\infty}^T E_{2\infty} = 0$), $\tilde{n} = n + n_c$, and $\tilde{D} = \begin{pmatrix} D_1 \\ B_c D_2 \end{pmatrix}$, satisfies the constraint

$$\|H(s)\|_\infty \le \gamma \tag{4}$$

where $\gamma > 0$ is a given constant; and

(iii) the performance functional

$$J(A_c, B_c, C_c) \equiv \lim_{t\to\infty} \mathcal{E}\left[x^T(t) R_1 x(t) + u^T(t) R_2 u(t)\right] \tag{5}$$

is minimized, where $\mathcal{E}$ is the expected value, $R_1 = E_1^T E_1 \in \mathbf{R}^{n\times n}$ and $R_2 = E_2^T E_2 \in \mathbf{R}^{m\times m}$ ($E_1 \in \mathbf{R}^{q\times n}$, $E_2 \in \mathbf{R}^{q\times m}$, $E_1^T E_2 = 0$) are, respectively, symmetric positive semidefinite and symmetric positive definite weighting matrices.

The closed-loop system (1)–(3) can be written as the augmented system

$$\begin{aligned} \dot{\tilde{x}}(t) &= \tilde{A}\,\tilde{x}(t) + \tilde{D}\,w(t), \\ z(t) &= \tilde{E}_\infty \tilde{x}(t) \end{aligned} \tag{6}$$

where $\tilde{x} = \begin{pmatrix} x \\ x_c \end{pmatrix}$.

Using this notation and under the condition that $\tilde{A}$ is asymptotically stable, for a given compensator the performance (5) is given by

$$J(A_c, B_c, C_c) = \mathrm{tr}\,(\tilde{Q}\,\tilde{R}), \tag{7}$$

where $\tilde{R} = \begin{pmatrix} R_1 & 0 \\ 0 & C_c^T R_2 C_c \end{pmatrix}$ and $\tilde{Q}$ satisfies the Lyapunov equation

$$\tilde{A}\,\tilde{Q} + \tilde{Q}\,\tilde{A}^T + \tilde{V} = 0, \tag{8}$$

with symmetric positive semidefinite $V_1 = D_1 D_1^T$, symmetric positive definite $V_2 = D_2 D_2^T$, and

$$\tilde{V} = \begin{pmatrix} V_1 & 0 \\ 0 & B_c V_2 B_c^T \end{pmatrix}.$$

LEMMA 1 (BERNSTEIN AND HADDAD 1989), (HADDAD AND BERNSTEIN 1990). *Let (A_c, B_c, C_c) be given and assume there exists $\mathcal{Q} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ satisfying*

$$\mathcal{Q} \quad \text{is symmetric and nonnegative definite} \tag{9}$$

and

$$\tilde{A}\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\mathcal{Q} + \tilde{V} = 0, \tag{10}$$

where $\tilde{R}_\infty = \begin{pmatrix} R_{1\infty} & 0 \\ 0 & C_c^T R_{2\infty} C_c \end{pmatrix}$, $R_{1\infty} = E_{1\infty}^T E_{1\infty}$, *and* $R_{2\infty} = E_{2\infty}^T E_{2\infty}$ *are symmetric positive semidefinite matrices. Then*

$$(\tilde{A}, \tilde{D}) \quad \text{is stabilizable} \tag{11}$$

if and only if

$$\tilde{A} \quad \text{is asymptotically stable.}$$

In this case

$$\|H(s)\|_\infty \le \gamma, \tag{12}$$

$$\tilde{Q} \le \mathcal{Q} \qquad (\mathcal{Q} - \tilde{Q} \text{ is nonnegative definite}),$$

and

$$\text{tr } \tilde{Q}\tilde{R} \equiv J(A_c, B_c, C_c) \le \mathcal{J}(A_c, B_c, C_c) \equiv \text{ tr } \mathcal{Q}\tilde{R}.$$

Hence the satisfaction of (9) and (10) along with the generic condition (11) leads to: 1) closed-loop stability; 2) prespecified H^∞ attenuation; and 3) an upper bound for the H^2 performance criterion.

The auxiliary minimization problem is to determine $(A_c,\ B_c,\ C_c)$ that minimizes $\mathcal{J}(A_c, B_c, C_c)$, and thus provides a bound for the actual H^2 criterion $J(A_c, B_c, C_c)$.

For technical reasons $(A_c, B_c, C_c, \mathcal{Q})$ is restricted to the open set

$$\mathcal{S} \equiv \Big\{(A_c, B_c, C_c, \mathcal{Q}) : \tilde{A} \text{ and } \tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R} \text{ are asymptotically stable,}$$
$$\mathcal{Q} \text{ is symmetric positive definite, and } (A_c, B_c, C_c) \text{ is controllable and observable } \Big\}.$$

3 The homotopy algorithm

Treating all the components of A_c, B_c, C_c as unknowns is redundant, since only $n_c(m+l)$ parameters suffice to describe the full or reduced order controller. There are numerous ways to parametrize the model (A_c, B_c, C_c) — the parametrization introduced by (Ly *et al.* 1985) is one way that uses the minimal number of parameters. In this basis, A_c is a 2×2 block-diagonal matrix (2×2 blocks with an additional 1×1 block if n_c is odd) with 2×2 blocks in the form $\begin{pmatrix} 0 & 1 \\ * & * \end{pmatrix}$, B_c is a full matrix, and C_c has its first row fixed as $(1 \quad 0 \quad 1 \quad 0 \quad \cdots)$. Observe that the Ly structure has $n_c(m+l)$ unknowns — n_c from A_c, $n_c l$ from B_c, and $n_c(m-1)$ from C_c. Denote the unknowns by

$$\theta \equiv \begin{pmatrix} (A_c)_\mathcal{I} \\ \text{Vec } (B_c) \\ \text{Vec } (C_c)_\mathcal{T}. \end{pmatrix},$$

where $(A_c)_{\mathcal{I}}$ is a vector consisting of those elements in A_c indexed by the set $\mathcal{I} \equiv \{(2,1),(2,2),\ldots,(n_c,n_c)\}$, Vec of a matrix is the concatenation of its columns, and $(C_c)_{\mathcal{T}.}$ is the matrix obtained from rows $\mathcal{T} \equiv \{2,\ldots,m\}$ of C_c.

Choose a problem for which a solution θ_0 is known, defined by the matrices A_0, B_0, C_0, $R_{1,0}$, $R_{2,0}$, $R_{1\infty,0}$, $R_{2\infty,0}$, $V_{1,0}$, $V_{2,0}$, γ_0; exactly how this initial problem is chosen is described in (Ge *et al.* 1994). Let A_f, B_f, …, γ_f, denote A, B, …, γ, in the above and define $A(\lambda)$, …, $\gamma(\lambda)$, as

$$A(\lambda) = A_0 + \lambda(A_f - A_0), \quad \ldots, \quad \gamma(\lambda) = \gamma_0 + \lambda(\gamma_f - \gamma_0),$$

and for brevity denote them by A, …, γ, respectively in the following. Let

$$\begin{aligned}
H_{A_c}(\theta,\lambda) &= \mathcal{P}_{12}^T\mathcal{Q}_{12} + \mathcal{P}_2\mathcal{Q}_2,\\
H_{B_c}(\theta,\lambda) &= \mathcal{P}_2 B_c V_2 + (\mathcal{P}_{12}^T\mathcal{Q}_1 + \mathcal{P}_2\mathcal{Q}_{12}^T)C^T,\\
H_{C_c}(\theta,\lambda) &= R_2 C_c\mathcal{Q}_2 + B^T(\mathcal{P}_1\mathcal{Q}_{12} + \mathcal{P}_{12}\mathcal{Q}_2)\\
&\quad + \frac{1}{2}\gamma^{-2}R_{2\infty}C_c\big[(\mathcal{Q}_{12}^T\mathcal{P}_1 + \mathcal{Q}_2\mathcal{P}_{12}^T)\mathcal{Q}_{12} + (\mathcal{Q}_{12}^T\mathcal{P}_{12} + \mathcal{Q}_2\mathcal{P}_2)\mathcal{Q}_2\big],
\end{aligned}$$

where $\mathcal{Q}$ and $\mathcal{P}$ satisfy

$$\tilde{A}\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\mathcal{Q} + \tilde{V} = 0, \tag{13}$$

$$\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)^T\mathcal{P} + \mathcal{P}\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big) + \tilde{R} = 0, \tag{14}$$

$$\mathcal{Q} = \begin{pmatrix} \mathcal{Q}_1 & \mathcal{Q}_{12} \\ \mathcal{Q}_{12}^T & \mathcal{Q}_2 \end{pmatrix}, \quad \mathcal{P} = \begin{pmatrix} \mathcal{P}_1 & \mathcal{P}_{12} \\ \mathcal{P}_{12}^T & \mathcal{P}_2 \end{pmatrix},$$

$$\tilde{R}_\infty = \begin{pmatrix} R_{1\infty} & 0 \\ 0 & C_c^T R_{2\infty} C_c \end{pmatrix}, \quad R_{1\infty} = E_{1\infty}^T E_{1\infty}, \quad R_{2\infty} = E_{2\infty}^T E_{2\infty}.$$

The nonlinear equations (corresponding to $\lambda = 1$) that result from the nonconvex constrained minimization procedure (see (Ge *et al.* 1994) for details) are

$$\rho(\theta,\lambda) \equiv \begin{pmatrix} \big[H_{A_c}(\theta,\lambda)\big]_{\mathcal{I}} \\ \mathrm{Vec}\,\big[H_{B_c}(\theta,\lambda)\big] \\ \mathrm{Vec}\,\big[H_{C_c}(\theta,\lambda)\big]_{\mathcal{T}.} \end{pmatrix} = 0.$$

To guarantee a full rank Jacobian matrix along the whole homotopy zero curve except possibly at the solution corresponding to $\lambda = 1$, define the homotopy map to be $\hat{\rho}(\theta,\lambda) = \lambda\rho(\theta,\lambda) + (1-\lambda)(\theta - \theta_0)$. The Jacobian matrix of $\hat{\rho}$ is given by

$$D\hat{\rho}(\theta,\lambda) = \big(\lambda D_\theta\rho(\theta,\lambda) + (1-\lambda)I, \rho(\theta,\lambda) + \lambda D_\lambda\rho(\theta,\lambda) - (\theta - \theta_0)\big).$$

$D_\theta\rho(\theta,\lambda)$ and $D_\lambda\rho(\theta,\lambda)$ are derived from the following. Define

$$\begin{aligned}
\hat{H}_{A_c}\big(\mathcal{P}^{(j)},\mathcal{Q}^{(j)}\big) &= \mathcal{P}_{12}^{T\,(j)}\mathcal{Q}_{12} + \mathcal{P}_{12}^T\mathcal{Q}_{12}^{(j)} + \mathcal{P}_2^{(j)}\mathcal{Q}_2 + \mathcal{P}_2\mathcal{Q}_2^{(j)},\\
\hat{H}_{B_c}\big(\mathcal{P}^{(j)},\mathcal{Q}^{(j)}\big) &= \mathcal{P}_2^{(j)}B_c V_2 + \big(\mathcal{P}_{12}^{T\,(j)}\mathcal{Q}_1 + \mathcal{P}_{12}^T\mathcal{Q}_1^{(j)} + \mathcal{P}_2^{(j)}\mathcal{Q}_{12}^T + \mathcal{P}_2\mathcal{Q}_{12}^{T\,(j)}\big)C^T,\\
\hat{H}_{C_c}\big(\mathcal{P}^{(j)},\mathcal{Q}^{(j)}\big) &= R_2 C_c\mathcal{Q}_2^{(j)} + B^T\big(\mathcal{P}_1^{(j)}\mathcal{Q}_{12} + \mathcal{P}_1\mathcal{Q}_{12}^{(j)} + \mathcal{P}_{12}^{(j)}\mathcal{Q}_2 + \mathcal{P}_{12}\mathcal{Q}_2^{(j)}\big)\\
&+ \frac{\gamma^{-2}}{2}R_{2\infty}C_c\Big[\big(\mathcal{Q}_{12}^{T\,(j)}\mathcal{P}_1 + \mathcal{Q}_{12}^T\mathcal{P}_1^{(j)} + \mathcal{Q}_2^{(j)}\mathcal{P}_{12}^T + \mathcal{Q}_2\mathcal{P}_{12}^{T\,(j)}\big)\mathcal{Q}_{12} + \big(\mathcal{Q}_{12}^{T\,(j)}\mathcal{P}_{12} + \mathcal{Q}_{12}^T\mathcal{P}_{12}^{(j)}\\
&+ \mathcal{Q}_2^{(j)}\mathcal{P}_2 + \mathcal{Q}_2\mathcal{P}_2^{(j)}\big)\mathcal{Q}_2 + \big(\mathcal{Q}_{12}^T\mathcal{P}_1 + \mathcal{Q}_2\mathcal{P}_{12}^T\big)\mathcal{Q}_{12}^{(j)} + \big(\mathcal{Q}_{12}^T\mathcal{P}_{12} + \mathcal{Q}_2\mathcal{P}_2\big)\mathcal{Q}_2^{(j)}\Big],
\end{aligned}$$

where the superscript (j) means $\partial/\partial\theta_j$. Using the above definitions, we have for $\theta_j = \big(A_c\big)_{kl}$, where $(k,l) \in \mathcal{I}$,

$$
\begin{aligned}
\frac{\partial H_{A_c}}{\partial (A_c)_{kl}} &= \hat{H}_{A_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),\\
\frac{\partial H_{B_c}}{\partial (A_c)_{kl}} &= \hat{H}_{B_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),\\
\frac{\partial H_{C_c}}{\partial (A_c)_{kl}} &= \hat{H}_{C_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),
\end{aligned}
$$

for $\theta_j = \big(B_c\big)_{kl}$,

$$
\begin{aligned}
\frac{\partial H_{A_c}}{\partial (B_c)_{kl}} &= \hat{H}_{A_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),\\
\frac{\partial H_{B_c}}{\partial (B_c)_{kl}} &= \hat{H}_{B_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big) + \mathcal{P}_2 E^{(k,l)} V_2,\\
\frac{\partial H_{C_c}}{\partial (B_c)_{kl}} &= \hat{H}_{C_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),
\end{aligned}
$$

and for $\theta_j = (C_c)_{kl}$, where $k > 1$,

$$
\begin{aligned}
\frac{\partial H_{A_c}}{\partial (C_c)_{kl}} &= \hat{H}_{A_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),\\
\frac{\partial H_{B_c}}{\partial (C_c)_{kl}} &= \hat{H}_{B_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big),\\
\frac{\partial H_{C_c}}{\partial (C_c)_{kl}} &= \hat{H}_{C_c}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big) + R_2 E^{(k,l)} \mathcal{Q}_2 + \frac{\gamma^{-2}}{2} R_{2\infty} E^{(k,l)} Y,
\end{aligned}
$$

where

$$
Y = \big(\mathcal{Q}_{12}^T \mathcal{P}_1 + \mathcal{Q}_2 \mathcal{P}_{12}^T\big)\mathcal{Q}_{12} + \big(\mathcal{Q}_{12}^T \mathcal{P}_{12} + \mathcal{Q}_2 \mathcal{P}_2\big)\mathcal{Q}_2
$$

and $E^{(k,l)}$ is a matrix of the appropriate dimension whose only nonzero element is $e_{kl} = 1$. $\mathcal{P}^{(j)}$ and $\mathcal{Q}^{(j)}$ can be obtained by solving the Lyapunov equation

$$
\begin{aligned}
0 &= \big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)\mathcal{Q}^{(j)} + \mathcal{Q}^{(j)}\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)^T + \tilde{V}^{(j)}\\
&\quad + \tilde{A}^{(j)}\mathcal{Q} + \mathcal{Q}\tilde{A}^{T\,(j)} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty^{(j)}\mathcal{Q},\\
0 &= \big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)^T\mathcal{P}^{(j)} + \mathcal{P}^{(j)}\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big) + \tilde{R}^{(j)}\\
&\quad + \big(\tilde{A}^{(j)} + \gamma^{-2}\mathcal{Q}^{(j)}\tilde{R}_\infty + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty^{(j)}\big)^T\mathcal{P} + \mathcal{P}\big(\tilde{A}^{(j)} + \gamma^{-2}\mathcal{Q}^{(j)}\tilde{R}_\infty + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty^{(j)}\big).
\end{aligned}
$$

Similarly for λ, using a dot to denote $\partial/\partial\lambda$,

$$
\begin{aligned}
\frac{\partial H_{A_c}}{\partial \lambda} &= \hat{H}_{A_c}\big(\dot{\mathcal{P}}, \dot{\mathcal{Q}}\big),\\
\frac{\partial H_{B_c}}{\partial \lambda} &= \hat{H}_{B_c}\big(\dot{\mathcal{P}}, \dot{\mathcal{Q}}\big) + \mathcal{P}_2 B_c \dot{V}_2 + (\mathcal{P}_{12}^T \mathcal{Q}_1 + \mathcal{P}_2 \mathcal{Q}_{12}^T)\dot{C}^T,\\
\frac{\partial H_{C_c}}{\partial \lambda} &= \hat{H}_{C_c}\big(\dot{\mathcal{P}}, \dot{\mathcal{Q}}\big) + \dot{R}_2 C_c \mathcal{Q}_2 + \dot{B}^T\big(\mathcal{P}_1\mathcal{Q}_{12} + \mathcal{P}_{12}\mathcal{Q}_2\big) + \frac{1}{2}\gamma^{-2}\dot{R}_{2\infty} C_c Y - \gamma^{-3}\dot{\gamma} R_{2\infty} C_c Y,
\end{aligned}
$$

where $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$ are obtained by solving

$$
\begin{aligned}
0 = {} & \big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)\dot{\mathcal{Q}} + \dot{\mathcal{Q}}\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)^T + \dot{\tilde{V}} + \dot{\tilde{A}}\,\mathcal{Q} + \mathcal{Q}\dot{\tilde{A}}^T + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}}_\infty\mathcal{Q} \\
& - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R}_\infty\mathcal{Q}, \\
0 = {} & \big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big)^T\dot{\mathcal{P}} + \dot{\mathcal{P}}\big(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}_\infty\big) + \dot{\tilde{R}} + \big(\dot{\tilde{A}} + \gamma^{-2}\dot{\mathcal{Q}}\tilde{R}_\infty + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}}_\infty \\
& - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R}_\infty\big)^T\mathcal{P} + \mathcal{P}\big(\dot{\tilde{A}} + \gamma^{-2}\dot{\mathcal{Q}}\tilde{R}_\infty + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}}_\infty - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R}_\infty\big).
\end{aligned}
$$

The homotopy zero curve tracking algorithm (which is a standard globally convergent probability-one homotopy algorithm (Watson *et al.* 1987)) is

1) Set $\lambda := 0$, $\theta := \theta_0$.
2) Calculate $\tilde{R}$, $\tilde{R}_\infty$, $\tilde{V}$, and compute $\mathcal{Q}$ and $\mathcal{P}$ according to (13) and (14).
3) Evaluate the homotopy map $\hat{\rho}(\theta, \lambda)$ and the Jacobian of the homotopy map $D\hat{\rho}(\theta, \lambda)$.
4) Predict the next point $Z^{(0)} = \big(\theta^{(0)}, \lambda^{(0)}\big)$ on the homotopy zero curve using, e.g., a Hermite cubic interpolant.
5) For $k := 0, 1, 2, \cdots$ until convergence do
$$Z^{(k+1)} = Z^{(k)} - \big[D\hat{\rho}(Z^{(k)})\big]^\dagger \hat{\rho}\big(Z^{(k)}\big),$$
where $\big[D\hat{\rho}(Z)\big]^\dagger$ is the Moore-Penrose inverse of $D\hat{\rho}(Z)$. Let $\big(\theta_1, \lambda_1\big) = \lim_{k\to\infty} Z^{(k)}$.
6) If $\lambda_1 < 1$, then set $\theta := \theta_1$, $\lambda := \lambda_1$, and go to step 2).
7) If $\lambda_1 \geq 1$, compute the solution $\bar{\theta}$ at $\lambda = 1$.

4 The parallel algorithm

In the above algorithm, Step 2) involves solving one Riccati equation and one Lyapunov equation. The Riccati equation is solved using Laub's Schur method (Laub 1979). The algorithm of Bartels and Stewart (Bartels and Stewart 1972) is applied to solve the Lyapunov equation. Although both algorithms are $O((n + n_c)^3)$, the Riccati equation, being more complicated, takes much more CPU time to solve. Once $\mathcal{Q}$ and $\mathcal{P}$ are obtained, the homotopy map $\hat{\rho}$ is formed by matrix multiplication operations.

The major part of the computation in Step 3) is that of the Jacobian matrix. The number of variables including λ in this formulation is $n_c(m + l) + 1$. Each column of the Jacobian matrix corresponds to the derivative of the homotopy map with respect to one variable and requires the solution of two Lyapunov equations (Ge *et al.* 1994). Therefore the time complexity of the Jacobian matrix computation is $O\big(n_c(m + l)(n + n_c)^3\big)$. The Bartels and Stewart algorithm finds the real Schur form of $\tilde{A}$ or $\tilde{A}^T$ depending on the Lyapunov equation. At each step along the homotopy path unnecessary factorization can be avoided if the previous factorization results from the computation of $\hat{\rho}$ and $D_\lambda\hat{\rho}$ are used.

The primary goal of this study is to make use of the existing code and to achieve reasonable parallel efficiency economically. The only part of the algorithm that is parallelized is the Jacobian matrix computation in Step 3). To utilize existing computer resources such as a network of workstations, the software package PVM (Parallel Virtual Machine) is used to provide the distributed computing capabilities.

The parallel algorithm follows the master-slave paradigm. The master sends the index of the column of the Jacobian matrix to be computed to a slave. The slave computes the corresponding column of the Jacobian, sends the column back to the master, and waits for

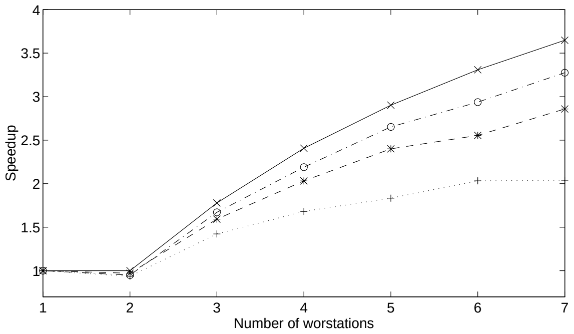

Fig. 1. Speedup with master and slaves on different machines.

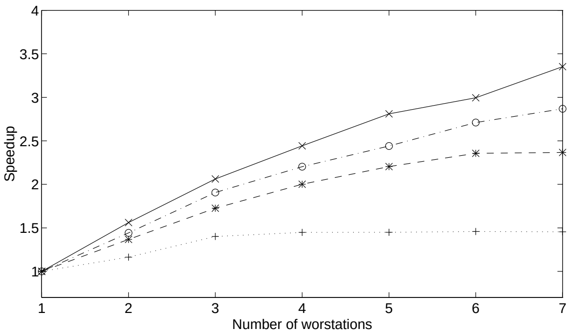

Fig. 2. Speedup when one slave is on the master machine.

the next index from the master to arrive. After receiving a column of the Jacobian, the master sends another index to the idle slave. In the implementation for the Intel Paragon, asynchronous send which sends a message without waiting for completion is used whenever possible to speed up the communication.

When the algorithm is implemented on a network of workstations, the modification to the original sequential source code consists of three parts: the first one is to spawn slave processes and set up the communication links between the master and the slaves; the second is to extract a slave program from the original code and at the same time simplify the master program; the last is to add a mechanism to guarantee correct communication between master and slaves. The first part consists of standard PVM operations, while the second is more problem oriented. To decrease communication, each slave process repeats part of the computation of $\hat{\rho}$ and $D_\lambda\hat{\rho}$ so that $\mathcal{Q}$ and $\mathcal{P}$ are not sent through the network. There is no loss of efficiency since the master is also computing the same quantities. The slave program consists of mainly the original subroutines with additional code for communication.

For the implementation on the Intel Paragon, the modification of the original code is even simpler. There is no need for a separate slave program if control statements use node identification properly. The parent process run on an Intel Paragon always gets node number 0, while other nodes are numbered 1 and higher. The statement *if node_number* == 0 precedes the code that is to be executed by the master, and an *else* following the previous master code will precede the code to be executed by the slave. The remaining modification to the original code is similar to the implementation using PVM. Asynchronous send is used whenever possible. A *wait* is used later when the data is needed, to ensure correct communication between the master and the slaves.

5 Results

The distributed code using PVM was run on a network of seven SGI Indigo^2 workstations. The data came from a control problem for suppressing vibrations in a string under transverse loading from a time varying disturbance force (Erwin 1993). For dimensions $n = 12, 20, 28, 36$ reduced order controllers of dimensions $n_c = 10, 18, 26, 34$ are sought respectively.

The speedups versus the number of workstations are shown in Figs. 1 and 2 ($n = 36$, 28, 20, 12, top to bottom). Fig. 1 shows the speedup versus the number of workstations when the master process and the slave processes are run on different machines, while Fig. 2 corresponds to the situation where the master process and a slave process with lower priority are run on one machine and the rest of the slaves are on other machines. For fair comparison

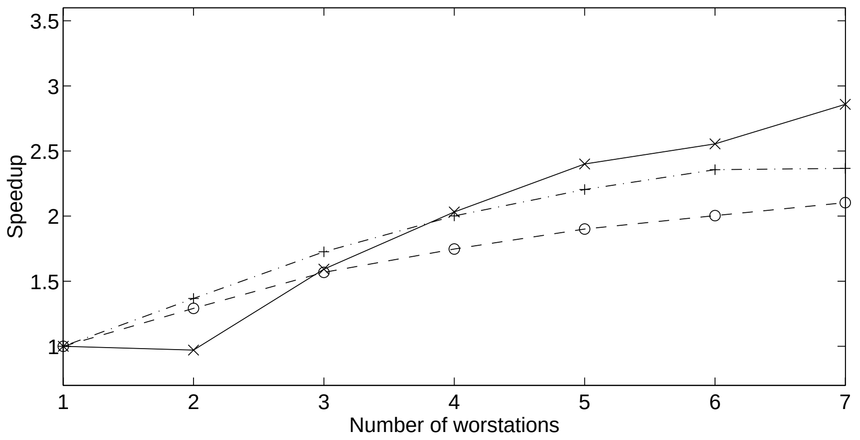

Fig. 3. Comparison of speedups.

all the speedups are computed relative to the results of the best optimized sequential code. In Fig. 1 two workstations correspond to the master process on one machine and the only slave on the other.

As shown in the figures, the speedup increases as the dimension of the problem increases, or as the number of workstations increases for a sufficiently large problem. The speedups from three scenarios (solid line—master and slaves on different machines; dash-dot line—master and a low priority slave on one machine and the rest of the slaves on others; dashed line—master and a slave with the same priority as the master on one machine) for $n = 20$ are plotted against the number of workstations in Fig. 3. If the number of workstations is < 4 it is better to use the second scenario. When the number of workstations is > 4, the speedup is higher if all the processes including the master and the slaves are run on different machines. Similar results obtain for large n.

The same algorithm is implemented using the system function calls of an Intel Paragon and run on one with 28 processors at Virginia Polytechnic Institute and State University. Fig. 4 shows the results obtained from the run for $n = 12$, 20, 28. The number of nodes varies up to 25. The highest curve corresponds to the speedup when $n = 28$ and the lowest corresponds to that when $n = 12$. As n increases, the advantage of parallel processing also increases. The highest speedup achieved for $n = 28$ using seven SGI Indigo^2 workstations is about 3.3 while the highest speedup using 25 nodes on an Intel Paragon is about 5.1. Comparing speedups is meaningful since the performance of a single SGI Indigo^2 processor is roughly comparable to that of a single i860XP Paragon node (actually, depending on the task, the 100MHz R4000 Indigo^2 is faster by a factor of 2). However, the cost of the Intel paragon is a factor of three times the cost of the SGI UNIX workstation network. Much higher speedups are potentially possible with the Paragon, but not without considerable programming effort for this controller design problem.

The above methodology can be easily generalized to industrial design environments where software packages like MATLAB or Mathematica are often used. The sequential program for mixed-norm H^2/H^∞ LQG controller synthesis has been developed into a MATLAB package. It is easy to include this distributed implementation into the MATLAB package. Installation requires two steps: the first one is to install PVM on the network of workstations, and the second is to create a file in which all the worker machines on

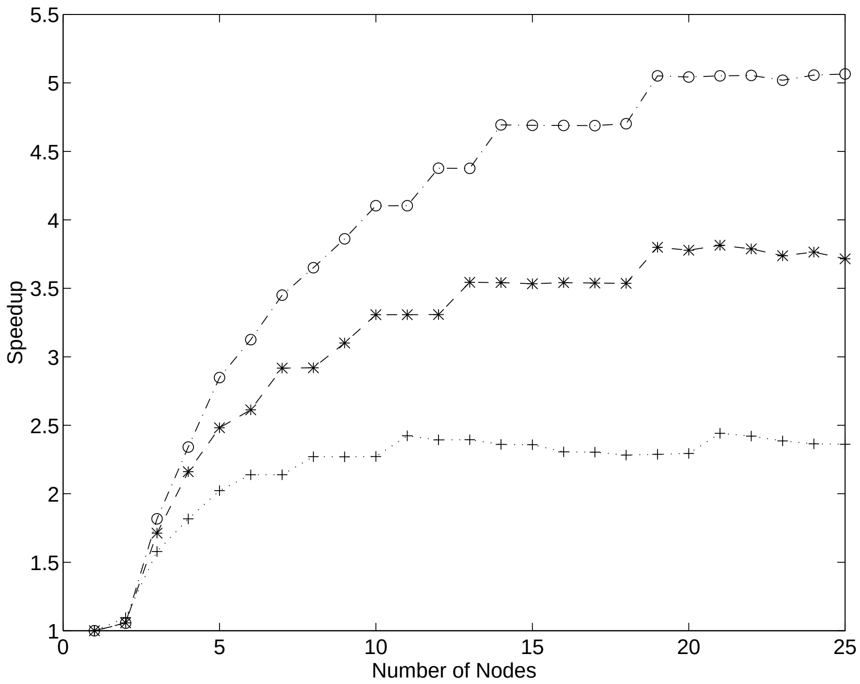

Fig. 4. Results for Intel Paragon XPE-28.

the network are listed (Geist *et al.* 1993). The execution of the distributed program from within an interactive design environment, e.g., MATLAB, can be done by using a MATLAB function defined in a MATLAB .m file, in which UNIX shell commands will start the PVM daemons if they have not already been started, and will then execute the distributed code.

In summary, parallelizing the Jacobian matrix computation in a homotopy algorithm reduces the execution time and is economical, especially for large problems. Acceptable speedups are obtained for a PVM implementation on a network of workstations. The approach can be applied to real industrial design environments to reduce controller design time and effectively utilize existing workstation networks. Compared to the cost of using real parallel computers and the cost of developing highly efficient parallel code, the approach of utilizing a network of workstations with only moderately efficient code is much more cost-effective for H^2/H^∞ controller synthesis.

References

BARTELS, R. H., AND STEWART, G. W. 1972. Solution of matrix equation $AX + XB = C$, *Comm. of the ACM*, **15**, 820–826.

BERNSTEIN, D. S., AND HADDAD, W. M. 1989. LQG control with an H_∞ performance bound: A Riccati equation approach, *IEEE Trans. Autom. Contr.*, **AC-34**, 293–305.

CANNON, R. H., JR., AND ROSENTHAL, D. E. 1984. Experiments in control of flexible structures with noncollocated sensors and actuators, *AIAA J. Guid. Contr. Dynam.*, **7**, 546–553.

ERWIN, R. S.. 1993. Private communication.

GE, Y., WATSON, L. T., COLLINS, E. G., AND BERNSTEIN, D. S.. 1994. Probability-one homotopy algorithms for full and reduced order H^2/H^∞ controller synthesis, submitted to *Optimal Control Appl. Methods* (also see *Proc. 33rd IEEE Conf. Decision Control*, Lake Buena Vista, FL, 1994, 2672–2677).

GEIST, A., BEGUELIN, A., DONGARRA, J., JIANG, W., MANCHEK, R., AND SUNDERAM, V.. 1993. PVM 3 User's Guide and Reference Manual, ORNL/TM-12187, Oak Ridge National Laboratory, Oak Ridge, TN, 1993.

HADDAD, W. M. AND BERNSTEIN, D. S. 1990. Generalized Riccati equations for the full- and reduced-order mixed-norm H_2/H_∞ standard problem, *Systems and Control Lett.*, **14**, 185–197.

HADDAD, W. M., COLLINS, E. G., JR., AND MOSER, R.. 1993. Fixed structure computation of the structured singular value, *Proc. Amer. Contr. Conf.*, San Francisco, CA, 1673–1674.

HADDAD, W. M., COLLINS, E. G., JR., AND MOSER, R.. 1994a. Structured singular value controller synthesis using constant D-scales without D-K iteration, *Proc. American Control Conf.*, Baltimore, MD, 2819–2823.

HADDAD, W. M., COLLINS, E. G., JR., AND MOSER, R.. 1994b. Complex structured singular value analysis using fixed-structure dynamic D-scales, *Proc. IEEE Conf. Decision and Control*, Lake Buena Vista, FL, 3003–3008.

LAUB, A. J. 1979. A Schur method for solving algebraic Riccati equations, *IEEE Trans. Aut. Contr.*, **AC–24**, 913–921.

LY, U.-L., BRYSON, A. E., AND CANNON, R. H. 1985. Design of low-order compensators using parameter optimization, *Automatica*, **21**, 315–318.

WATSON, L. T., BILLUPS, S. C., AND MORGAN, A. P. 1987. HOMPACK: a suite of codes for globally convergent homotopy algorithms, *ACM Trans. Math. Software*, **13**, 281–310.