Proceedings of the 32nd Conference
on Decision and Control
San Antonio, Texas • December 1993

Efficient Parallel Algorithm for Optimal Block Decomposition of Transfer Function Matrices for Decentralized Control of Large-Scale Systems

A.Balluchi A.Balestrino A.Bicchi

Dipartimento di Sistemi Elettrici e Automazione
Università degli Studi di Pisa

Abstract

The purpose of this paper is to present a parallel algorithm which can be used to determine how a large-scale system can be best decomposed in simpler subsystems to which decentralized control can be applied. The algorithm is based on an analysis in the frequency domain which employs the notion of block-diagonal dominance. The decentralization design problem is set as a combinatorial optimization problem and a "branch and bound" approach is applied to solve it. The proposed algorithm is conceived for parallel computational architectures. A discussion on the computational efficiency gained by parallelization is presented, and the implementation of the algorithm on a transputer network is briefly illustrated.

1 Introduction

Given a MIMO system which consists of a number of weakly coupled SISO subsystems, the direct or the inverse Nyquist array method, proposed by Rosenbrock [1] can be applied for designing a decentralized control in the hypothesis of diagonal dominance of the transfer function matrix $G(s)$. In the case of weakly interacting MIMO subsystems we cannot expect to achieve the condition of diagonal dominance of the transfer function matrix for every s on the Nyquist contour. In order to take advantage of weak coupling among MIMO subsystems, one needs to introduce the notion of block diagonal dominance reformulating the Rosenbrock's methods for blockwise decompositions. A generalization of the Nyquist array methods is due to Bennett and Baras [2], who used the notion of diagonal dominance introduced by Feingold and Varga [3] providing extensions of Gerschgorin bands and Ostrowsky bounds. Subsequently, Limebeer [4] proposed modifications of the stability criteria, that are based upon the concept of generalized block diagonal dominance introduced originally by Fiedler [5]. Robert [6] and Pearce [7] introduced a new notion of block diagonal dominance. Stability and perfomance robustness have been studied by Nwokah [8] who showed how decentralization and robustness are related.

The objective of this paper is to present a "branch and bound" algorithm which can be used to choose the optimal decentralization according to a chosen "control cost" function. In the general case, the elements of $G(s)$ are not in the right order to verify a possible dominance condition. This means that if we apply a suitable input and output permutation as

$$G'(s) = P \cdot G(s) \cdot P^T$$

a block diagonal dominance condition on the matrix $G'(s)$ for some block partition could be met. We provide an algorithm, based on a "branch and bound" approach, that, for a given frequency sample vector $\omega = (\omega_1, .., \omega_k)^T$, returns a number of solutions for decentralized control design, ordered according to a "control cost" function. Frequency axis sampling is necessary in order to obtain a finitely computable algorithm. Obsviously, dominance on a set of sampled frequencies is only a necessary condition for applying Nyquist array methods. Analysis is conducted for the "worst case" ω_ℓ in ω, that is the one with lower variance of elements of $G(j\omega_\ell)$. When a feasible solution is found, we verify whether it is feasible for all other matrices $G(j\omega_i)$ with $i \neq \ell$. Finally the algorithm returns a set of decompositions for which the block diagonal dominance condition is met for all given frequency samples. A partial ordering is provided on the set, based on the size of diagonal sub-blocks and on the residual level of interaction. In order to verify the block diagonal dominance condition for every s on the Nyquist contour, a more detailed analysis can be done on selected decompositions in the solution set.

2 Problem formulation

Given the transfer function matrix of a MIMO system $G(s)$ with n inputs and n outputs, and a k-vector of real $\omega = (\omega_1, .., \omega_k)^T$ specifying the set of k frequency samples at which dominance must be checked, the problem of decentralization design consists of finding, amoung all decompositions that afford a prescribed level of non-interaction, these allowing the finest decentralization. This can be cast in a combinatorial optimization problem as

$$(P) \quad \begin{array}{l} min \ f(x) \\ ||\mathbf{I}(\omega, x)||_\infty < I_M \\ x \in X \end{array}$$

where X is the set of solutions, $\mathbf{I}(\omega, x) : \mathbb{R}^k \times X \rightarrow \mathbb{R}^k$ is a constraint function, and $f : X \rightarrow \mathbb{N}^+$ with $0 < m \leq f(x) \leq M$ is the cost function. Definitions of the above terms follow.

2.1 The set of solutions X

The set of solutions X consists of structured elements x, henceforth called "decompositions". Each decomposition x is comprised of a permutation matrix P and a subdivision S

$$X = \{(P,S): \quad P \in \mathcal{P}_n, \quad S \in \mathcal{S}_n\}. \tag{1}$$

$\mathcal{P}_n$ is the set of all $n \times n$ permutation matrices. The cardinality of $\mathcal{P}_n$ is $n!$. Let H be the set of the first n integers: $H = \{1,2,\ldots,n\}$. A partition of H is a family of subsets of H $\{J_1, J_2, \ldots, J_r\}$ such that each $j \in H$ appears in one and only one subset J_i. If 2^H stands for the family of all subsets of H, $\mathcal{S}'$ denotes the set of partitions of H

$$\mathcal{S}' = \{I \subset 2^H: \quad \forall J_i, J_j \in I \quad J_i \bigcap J_j = \emptyset, \ \cup_i J_i = H\}$$

$\mathcal{S}_n$ is the set of all partitions of H $\{J_1, J_2, \ldots, J_r\}$ such that all J_i for $i = 1, \ldots, r$ consist of consecutive integers

$$\begin{aligned}\mathcal{S}_n = \{ \quad & \{J_1, J_2, \ldots, J_r\} \in \mathcal{S}' : 1 \in J_1, \ n \in J_r, \\ & \forall l \in J_i \setminus \{1,n\} \ \ (l+1) \in J_i \cup J_{i+1} \wedge \\ & \wedge (l-1) \in J_i \cup J_{i-1}, \quad i = 1,.,r\}\end{aligned}$$

Since each element $S \in \mathcal{S}_n$, as it is shown below, can be represented with a $(n-1)$-vector of binary variables, the cardinality of $\mathcal{S}_n$ is 2^{n-1}. The number of possible solutions for problem P is

$$|X| = |\mathcal{P}_n| \cdot |\mathcal{S}_n| = n! \cdot 2^{n-1}.$$

Given a solution $x = (P,S) \in X$ let us apply the permutation matrix P to $G(s)$ obtain $B(s) \rightarrow \mathbf{C}^{n \times n}$

$$B(s) \ = \ P \cdot G(s) \cdot P^T$$

$S = \{J_1, J_2, \ldots, J_r\}$ defines a partition of $B(s)$ with square diagonal blocks

$$\begin{aligned}\forall \quad & J_s, J_t \in S \ \rightarrow \ Z_{st} = \{b_{ij}\} \\ & i \in J_s, \ j \in J_t, \qquad s,t = 1, \ldots, r\end{aligned}$$

We drop the variable s from matrices Z_{st} without confusion.

$$B(s) \ = \ \begin{pmatrix} Z_{11} & \cdots & Z_{1r} \\ \vdots & \ddots & \vdots \\ Z_{r1} & \cdots & Z_{rr} \end{pmatrix}$$

2.2 The constraint function $\mathbf{I}(\omega, x)$

The function $\mathbf{I}(\omega, x)$ which appears in the constraint condition of problem P is a k-vector whose components are interaction indices

$$\mathbf{I}(\omega, x): \mathbb{R}^k \times X \longrightarrow \mathbb{R}^k$$

$$\mathbf{I}(\omega, x) = \begin{pmatrix} I(\omega_1, x) \\ \vdots \\ I(\omega_k, x) \end{pmatrix}; \quad \omega = \begin{pmatrix} \omega_1 \\ \vdots \\ \omega_k \end{pmatrix}.$$

In order that a solution $x \in X$ be feasible we demand that all components $I(\omega_i, x)$, with $i = 1, \cdots, k$, be lower than a value I_M

$$\|\mathbf{I}(\omega, x)\|_\infty < I_M \tag{2}$$

The interaction index is a function of the frequency ω and of the decomposition $x \in X$, whose values are not negative real

$$I: \mathbb{R} \times X \longrightarrow \mathbb{R} \qquad \text{with } I(\omega, x) \geq 0.$$

Associate to each decomposition $x = (P,S)$ a matrix $C(\omega, x) \rightarrow \mathbb{R}^{r \times r}$ obtained from $B(s)$

$$C(\omega, x) = \{c_{ij}\} = \begin{cases} 0 & \text{if } i = j \\ \| Z_{ij} \cdot Z_{ii}^{-1} |_{s=j\omega} \|_\infty & \text{if } i \neq j \end{cases} \tag{3}$$

and define the interaction index $I(\omega, x)$ as follows

$$I(\omega, x) \ = \ \|C(\omega, x)\|_\infty \tag{4}$$

For a chosen frequency value $\hat{\omega}$ the inequality

$$I(\hat{\omega}, x) < I_M = 1$$

defines the condition of block diagonal dominance of the matrix $B(j\hat{\omega})$, obtained from the solution x which includes a matrix block partition. This definition includes the notions of block diagonal dominance introduced by Robert [6] and Pearce [7]. However, the user is free to choose I_M in order to achieve either higher decentralization (I_M large) or stronger control robustness (I_M small) [8]. The set of feasible solutions for the optimization problem P is therefore

$$Y = \{x \in X: \ \|\mathbf{I}(\omega, x)\|_\infty < I_M\}.$$

2.3 The objective function $f(x)$

The choice of the objective function is a crucial point in the definition of an optimization problem. The best situation is to obtain the maximum decentralization given by n SISO control systems, so in this case $f(x)$ must be equal to the minimum value m. Vice versa, the worst situation is to have no decentralization, with only one MIMO control system correspondingly, $f(x)$ must reach the maximum value M. A wide variety of objective functions is possible, e.g. to maximize the number of sub-blocks, minimize the dimension of the largest sub-block, and so on. In the following, a minimum squared dimension objective function is considered, that is defined as follows

$$f(x) = \sum_{i=1}^{r} |J_i|^2. \tag{5}$$

Example

Consider the transfer function matrix $G(s)$

$$G(s) = \begin{pmatrix} \frac{j40}{s+j10} & \frac{j10}{s} & 0 \\ \frac{j15}{s+j5} & \frac{s+j50}{s+j10} & 0 \\ \frac{j150}{s+j20} & 0 & \frac{j40}{s+j10} \end{pmatrix}.$$

At $\hat{\omega} = 10$ we obtain $A = G(j\hat{\omega})$

$$A = G(j10) = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 0 \\ 5 & 0 & 2 \end{pmatrix}$$

Choose a value $I_M = 0.8$, and consider the solutions

$$x_1 = (\ P_1, S_1) \qquad x_2 = (\ P_2, S_2)$$

with

$$S_1 = \{\{1,2\},\{3\}\} \quad P_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad S_2 = \{\{1\},\{2,3\}\} \quad P_2 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Evaluating $B(\jmath 10) = P \cdot A \cdot P^T$ we have

$$x_1 \leftrightarrow \left(\begin{array}{cc|c} 2 & 1 & 0 \\ 1 & 3 & 0 \\ \hline 5 & 0 & 2 \end{array}\right); \; x_2 \leftrightarrow \left(\begin{array}{c|cc} 3 & 0 & 1 \\ \hline 0 & 2 & 5 \\ 1 & 0 & 2 \end{array}\right).$$

$\mathbf{I}(\omega, x)(\jmath 10, x_1) = 4 > I_M \qquad \mathbf{I}(\omega, x)(\jmath 10, x_2) = 0.5 < I_M$
$f(x_1) = f(x_2) = 1^2 + 2^2 = 5$

Therefore x_1 is not feasible while x_2 is.

2.4 Equivalent optimization problem

Let $I(x)$ denote $\|\mathbf{I}(\omega, x)\|_\infty$ and let us consider a new optimization problem Q which is equivalent to problem P

$$(Q) \quad \min_{x \in X} g(x)$$

where

$$g(x) = \begin{cases} f(x) + \frac{I(x)}{I_M} & \forall x \in X : I(x) < I_M \\ M + 1 & \forall x \in X : I(x) \geq I_M. \end{cases} \tag{6}$$

In the new formulation we removed the constraint condition and the feasible set is X. Also, we have assigned the maximum value of the objective function $g(x)$ to solutions that do not satify the non-interaction constraint in P. Indeed we have

$$\forall x \in Y \;\Leftrightarrow\; g(x) = f(x) + \frac{I(x)}{I_M} < M + 1.$$

Note that the addition of the term $\frac{I(x)}{I_M}$ in $g(x)$, reflects a further partial ordering related to the interaction index value.

3 Branch and Bound Algorithm

We propose a "branch and bound" algorithm to solve the optimization problem Q. To this purpose we need a lower bound function of $g(x)$, which allows to apply implicit visit of subtrees during the exploration of enumeration tree. A lower bound function must be defined on subsets of X and must give a value which is lower or equal to $g(x)$ for all solutions that belong to the subset. Furthermore, the computational complexity of the lower bound function ought to be lower than that of the objective function, involving the $I(x) = \|\mathbf{I}(\omega, x)\|_\infty < I_M$ test.

3.1 The lower bound function $w(x)$

Since $\|\mathbf{I}(\omega, x)\|_\infty \geq I(\omega_\ell, x)\ \forall \ell$ any of them can be chosen for building a lower bound function. The program chooses ℓ such that the variance of the elements of $G(\jmath\omega_\ell)$ is the lowest. However the program let the user free to make different choices. Introducing $v(x)$

$$v(x) = \max_i \left\{ \sum_{j=1, j\neq i}^{r} \frac{\|Z_{ij}|_{s=\jmath\omega_\ell}\|_\infty}{\|Z_{ii}|_{s=\jmath\omega_\ell}\|_\infty} \right\} \tag{7}$$

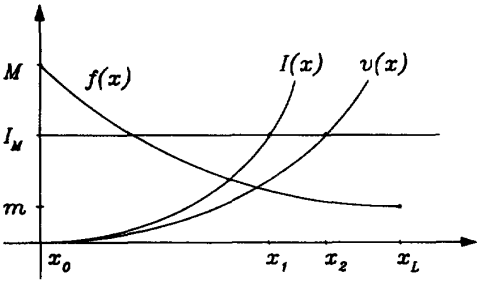

Figure 1: Objective function $f(x)$, constraint function $I(x)$ and lower bound function $v(x)$

and applying the well-known matrix norm proprety

$$\frac{\|G\|}{\|F\|} \leq \|F^{-1} \cdot G\|, \qquad F \in \mathbb{R}^{n\times n}, \; G \in \mathbb{R}^{n\times m}$$

we have $\|C(\omega_\ell, x)\|_\infty \geq v(x)$. Therefore the following inequality holds

$$\forall x \in X : \; \|\mathbf{I}(\omega, x)\|_\infty \geq I(\omega_\ell, x) \geq v(x) \tag{8}$$

Let us introduce the notion of sub-decomposition.

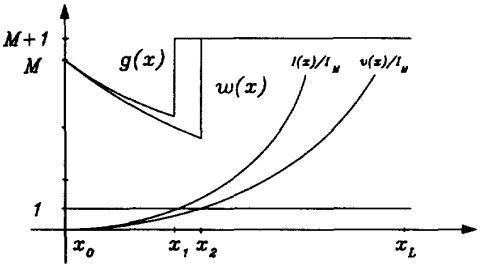

Figure 2: Objective function $g(x)$ and lower bound function $w(x)$

Given two decompositions $x_1 = (P_0, S_1)$ and $x_2 = (P_0, S_2)$, having the same permutation matrix P_0, with $|S_1| = r$ and $|S_2| = r + 1$, x_2 is sub-decomposition of x_1 if there exists t such that

$$S_1 = \{J_1, J_2, \ldots, J_{t-1}, J_t, J_{t+1}, \ldots, J_r\}$$
$$S_2 = \{J_1, J_2, \ldots, J_{t-1}, J'_t, J''_t, J_{t+1}, \ldots, J_r\}$$
$$J_t = J'_t \cup J''_t.$$

For all $x \in X$ define the set of all sub-decomposition $D(x)$ of x

$$D(x) = \{y \in X : y \text{ is sub-decomposition of } x\} \tag{9}$$

Claim 1 *If solution $x_1 = (P_0, S_1)$ and $x_2 = (P_0, S_2)$ are given, with x_2 sub-decomposition of x_1, then $v(x_1) \leq v(x_2)$.*

$$\forall x_1 \in X, \qquad \forall x_2 \in D(x_1) \implies v(x_1) \leq v(x_2).$$

Proof: The proof is algebraic in nature, and is reported in [11].

Corollary 1 *For all $x \in X$, function $v(x)$ is a lower bound of $\|\mathbf{I}(\omega, x)\|_\infty$ on the subset $D(x) \subset X$ of x sub-decompositions*

$$\forall x \in X: \; v(x) \leq \|\mathbf{I}(\omega, y)\|_\infty \; \forall y \in D(x). \qquad (10)$$

Proof:

$$v(x) \leq v(y) \leq \|\mathbf{I}(\omega, y)\|_\infty \qquad \forall y \in D(x). \; \square$$

We are looking for a lower bound function of $g(x)$ defined on subsets of X $w: 2^X \longrightarrow \mathbb{R}$ such that

$$w(T) \leq g(y) \qquad \forall y \in T \subset X$$

We now show that there exists a biunique relationship between $\mathcal{S}_n$ and sub-graphes of the graph G with n nodes and $(n-1)$ $(i, i+1)$ arcs (see figure 3). Associate the binary variable a_i to each arc $(i, i+1)$, $a_i = 0$ indicating that the $(i, i+1)$-arc is missing and $a_i = 1$ indicating that the $(i, i+1)$-arc exists. Put in J_i all nodes belonging to ith connected sub-graph. Consider a subset of X, T obtained choosing n_1 variables $a_i = 0$, n_2 variables $a_i = 1$, and leaving free the others $n_3 = n - n_1 - n_2 > 1$. Let $x_{[1]}$ denote the solution belonging to T with the n_3 variables a_i not assigned yet set to 1 and $x_{[0]}$ the solution with $a_i = 0$. Obviously we have $T = D(x_{[1]})$. Consider a lower bound for g(x) on the subset T

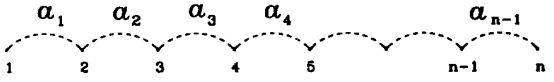

Figure 3: Graph G

$$w' = \min\{g(y): \; y \in D(x_{[1]})\} =$$

$$= \begin{cases} \min\{f(x) + \frac{I(x)}{I_M}: \; \forall y \in D(x_{[1]})\} \\ \qquad\qquad \text{if } \exists y \in D(x_{[1]}): I(y) < I_M \\ M+1 \qquad\qquad \text{if } \forall y \in D(x_{[1]}), I(y) \geq I_M, \end{cases}$$

and

$$w'' = \begin{cases} \min\{f(y) + \frac{v(y)}{I_M}: \forall y \in D(x_{[1]})\} \\ \qquad\qquad \text{if } \exists y \in D(x_{[1]}): v(y) < I_M \\ M+1 \qquad\qquad \text{if } \forall y \in D(x_{[1]}), v(y) \geq I_M. \end{cases}$$

Clearly, we have $w'' \leq w'$ (see figure 2). Also

$$\forall y \in D(x_{[1]}), v(y) \geq I_M \iff v(x_{[1]}) \geq I_M.$$

$$\min\{f(y) + \frac{v(y)}{I_M} \forall y \in D(x_{[1]})\} = f(x_{[0]}) + \frac{v(x_{[1]})}{I_M}.$$

Therefore

$$w(T) = \begin{cases} f(x_{[0]}) + \frac{v(x_{[1]})}{I_M} & \text{if } v(x_{[1]}) < I_M \\ M+1 & \text{if } v(x_{[1]}) \geq I_M \end{cases} \qquad (11)$$

is a lower bound function of $g(x)$ on $T = D(x_{[1]})$.

Remark: The computation of $w(T)$ is approximately the same as that needed for $v(x_{[1]})$, since computation of $f(x_{[0]})$ only involves the sum of r squared terms. In the following time needed to perfom the computation of a function will be used as a measure of computation complexity. Since evaluation of $v(x)$ involves r^2 infinity-norms and divisions the computation complexity of $w(T)$ is $n^2 \cdot \Delta$, where Δ denotes a reference unit of time. For many subsets T has the same $x_{[1]}$, the last values of $v(x_{[1]})$ computed are stored in a table for repeated use.

3.2 The enumeration tree

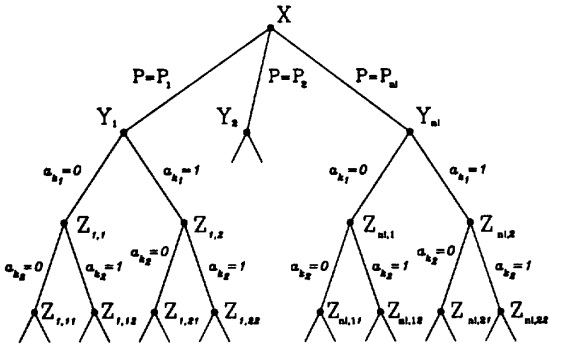

Figure 4: Enumeration tree visited by "branch & bound algorithm".

Let us consider a partition of X into $n!$ subsets Y_i, obtained by choosing the permutation matrix

$$Y_i = \{(P, S) \in X: \; P = P_i \in \mathcal{P}_n\}$$

Each Y_i has 2^{n-1} elements and we have $Y_i = D(x_{i0}) \subset X$, where $x_{i0} = (P_i, \{H\})$. The root of the enumeration tree is the set X. At depth 1 there are the $n!$ subsets Y_i. From this point a bipartite separation rule is applied:

- Y_i is parted into two subsets Z_{i1}, Z_{i2}. Let k_1 be $\lceil \frac{n}{2} \rceil$, we insert into Z_{i1} all solutions with $a_{k_1} = 0$ and into Z_{i2} the ones with $a_{k_1} = 1$.

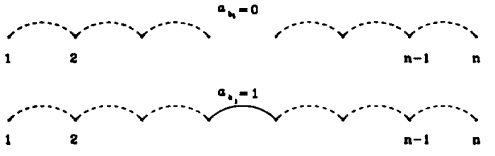

Figure 5: First assignment: $a_{k_1} = 0$ and $a_{k_1} = 1$.

- choose $k_2 = \lceil \frac{k_1}{2} \rceil$ and bipart Z_{i1}, Z_{i2} assigning $a_{k_2} = 0$ and $a_{k_2} = 1$.
- consider a_{k_3} with $k_3 = n - \lceil \frac{n-k_1}{2} \rceil$, and so on.
- bisection of the graph G is applied since we assign all arc variables.

This separation rule allows us to use the lower bound function $w(T)$ for subset evaluations. The visit

of a subtree rooted in Y_i involves computing n times function $w(T)$ then whose complexity is $h(n)\Delta = n^3 \cdot \Delta$.

Enumeration tree visit is implemented handling a priority queue of candidate nodes Q, which contains the next nodes that must be visited. Let us divide the priority queue Q into two subsets $Q = Q_1 \cup Q_2$

$$Q_1 = \{q' \in Q : depth(q') = 1\}$$
$$Q_2 = \{q' \in Q : depth(q') > 1\}.$$

where $depth(q')$ denotes the depth of the node q' in the enumeration tree. The initialization phase sets

$$Q_1 = \{Y_1, Y_2, \ldots, Y_{n!}\} \quad Q_2 = \emptyset$$

There is no insertion into Q_1 but just extractions. Instead of storing all Y_i in a queue, an extraction procedure, which generates the next subset to be extracted using the previous one, is provided. This is possible by defining an ordering on Q_1. Before each extraction from Q_1, Q_2 is empty, following separation of a Y_i new elements are inserted in Q_2. We apply on Q_2 a FIFO selection strategy which generates a depth visit of the enumeration tree. We choose to visit firstly the left son ($a_k = 0$) and then the right son ($a_k = 1$). In such a way at each step we firstly try to divide in two equal parts a sub-block J_t of B which involves the maximum decrement $-\Delta f$ of objective function $-\Delta f = \frac{|J_t|^2}{2}$. Therefore, we achieve a kind of visit that is also a minimum search. This strategy supplies small queue dimension (depth strategy) and quick algorithm steps (minimum search). If the visit of the enumeration tree reaches a leaf x_L then the objective function $g(x_L)$ is evaluated (which involves $I(x_L, \omega_i) < I_M$ test $\forall i$). And if $g(x_L)$ is lower than the current upper bound x_L becomes the current best solution.

4 Algorithm parallelization

"Branch and bound" algorithms are easy to parallelize, in fact they are based on the enumeration tree visit, so that if we have a number of elaboration units, we can divide the tree in several subtrees and assign a subtree visit to each elaboration unit. We can use a master-slave model where the master process handles the high part of the enumeration tree and gives all slave processes the task to explore a particular subtree. In this case we achieve that the priority queue of candidate nodes is not concentrated on a single process enviroment, but it is distributed. In our "branch and bound" algorithm the master process owns the priority queue Q_1, whereas each slave process has its own priority queue for subtree evaluation, so that a parallel evolution of Q_2 is obtained. The only consistency problem regards upper bound local copies, since both the master process and all the slave processes have their own copy. When a slave process reaches a solution better than the current upper bound value, it sends this solution to the master process. A parallel high priority update process, supplying transmission and reception of upper bound updates, is provided.

4.1 Parallel architecture

The parallel architecture is based on a message passing model and the software network is a tree-shaped graph. The host process runs user interface and file management, the master process handles high level algorithm and the four slaves processes implement subtree visit. Note the hierarchical structure of the adopted model. The hardware network is supported by an IBM 486, host computer, and a network of four transputers IMS T800. the hardware network and the master process, with high priority, plus a slave process, with low priority, on the root transputer.

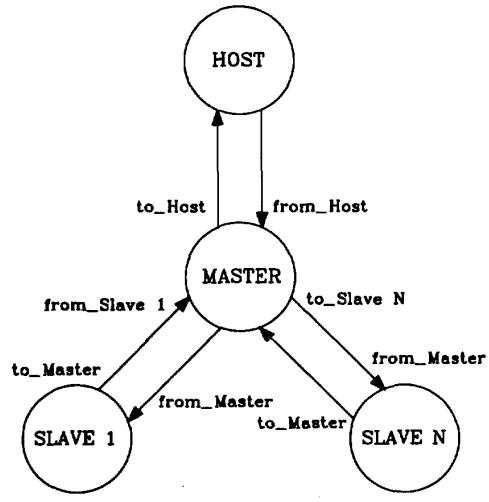

Figure 6: Software network.

4.2 Parallel algorithm efficiency

Let us consider a slave process elaboration. Each slave instance starts with a waiting which ends when the slave process receives a new subtree exploration task. After data receiving elaboration takes place. When the elaboration is finished a waiting time follows until the master process is ready to receive data from slave processes. To ensure a efficient performance of the parallel algorithm we must control the relationship between waiting time, communication time and elaboration time. Elaboration time is a function of the number of subtrees m that must be visited for all slave instances and of the subtree visit complexity $h(n)$, $T_{elab} = m \cdot h(n) \cdot \Delta = m \cdot n^3 \cdot \Delta$.

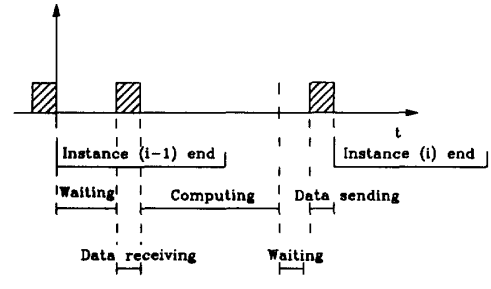

Figure 7: Time diagram of a typical slave process instance.

k	number of nodes m				
Slaves	10	30	50	70	100
1	5682	17087	28450	39768	56895
2	5690	17049	28453	39830	56920
3	5689	17090	28440	39698	56931

Table 1: Elaboration time, expressed in $msec$, as a function of m and k

Communication time is proportional to data length, $T_{comm} = d(n) \cdot \Delta = d_0 \cdot n \cdot \Delta$. To achieve a specified value of efficiency let fix a minimum value to the quotient between them

$$E = \frac{T_{elab}}{T_{comm}} = \frac{m \cdot h(n)}{d(n)} > E_{min} \qquad (12)$$

This relationship allow us to choose m value

$$m = \left\lceil \frac{E_{min} \cdot d_0}{n^2} \right\rceil . \qquad (13)$$

To limit waiting time, we demand that probability of prompt access to communication P be greater than a P_{min} value

$$P = 1 - \frac{k-1}{E+1} > P_{min}. \qquad (14)$$

This allows us to choose the optimal number of slave processes

$$k = \lfloor (1 - P_{min}) \cdot (E_{min} + 1) + 1 \rfloor \qquad (15)$$

4.3 Algorithm performance

Several tests have been carried out in order to assess algorithm performance. Table 1 shows a typical performance obtained for $n = 10$. Note that elaboration time is proportional to m. From this values we get the reference unit of time $\Delta = 569msec$, communication time $T_{comm} = 11.1msec$, and waiting time $T_{wait} = 8.3msec$. Observe that communication time and waiting time raise proportionally with n, but they are always much lower than elaboration time. Indeed we have large E values. We can conclude that the speed-up obained is proportional to the number k of slave processes used, that is the number of transputer on the network. However parallelization of the algorithm is not rewarding if the transfer function matrix dimension n is too small, $n < 10$.

5 Conclusions

If a square transfer function matrix of a MIMO system is given, the possibility of applying a decentralized control should be examined. The notion of block diagonal dominance allows to extend decentralized control to a large class of systems. Decentralized control design, which means choosing a permutation matrix and a block partition, can be set as a combinatorial optimization problem by defining a suitable "control cost" function. Block diagonal dominance condition for every s on the Nyquist contour can be tested on a suitable set of frequency samples. The dominance for all frequency samples gives a constraint condition for the optimization problem. A suitable separation rule of the set of the solutions allowed us to define a lower bound function for application of a branch and bound algorithm. Branch and bound algorithms are easily parallelizable by subdividing enumeration tree visit. A transputer network is used as hardware support for the parallel algorithm. Experimental data support use of the proposed technique for decomposing large scale ($n > 10$) systems.

Acknoledgment

The authors would like to thank Prof. M. Holcombe, head of the Department of Computer Science at the University of Sheffield UK, who allowed the development of the algorithm in his department, and Prof. G. Manson for his unvaluable help during algorithm parallelization.

References

[1] H.H.Rosenbrock, "Computer-Aided Control System Design", London, Academic Press, 1974.

[2] W.H.Bennet and J.S.Baras, "Block diagonal dominance and design of decentralized compensator", in Proc.IFAC Symp.Large Scale Syst. Theory, Appl., Toulouse, France, pp. 93-102, 1962.

[3] D.G.Feingold and R.S.Varga, "Block Diagonally Dominant Matrices and Generalizations of the Gerschgorin Circle Theorem", Pacific Journal of Math., vol 12, pp. 1241-1250, 1962.

[4] D.J.N.Limebeer, "The application of generalized diagonal dominance to linear stability theory", Int. J.Control, vol 36 no 2 pp. 183-212, 1982.

[5] M. Fiedler, "Some estimates of spectra of matrices", in Symp. on Numerical Treatment of Ordinary Differantial, Integral, and Integro-Differential Equations. Basel Switzerland: Birkhauser-Verlag, pp. 33-36, 1961.

[6] F.Robert, "Blocs-H-Matrices et convergence des methodes iteratives classiques par blocks", Linear Algebra Appl., vol 2, pp. 223-265, 1966.

[7] I.F.Pearce, "Matrices with dominating diagonal blocks", J. Econ. Theory, vol 9, pp. 159-170, 1974.

[8] O.D.I. Nwokah, "The robust decentralised stabilisation of complex feedbak systems", IEE Proceedings, vol 134, Pt. D, no 1, pp. 43-47, January 1987.

[9] M.K.Sundareshan and R.M.Elbanna, "A Constructive Procedure for Stabilization of Large-Scale Systems by Informationally Decentralized Controllers", IEEE Trans. Automat. Contr., vol 36, no 7, pp. 848-852, July 1991.

[10] Yuzo Ohta, D.D.Siljak, and T.Matsumoto, "Decentralized Control Using Quasi-Block Diagonal Dominance of Transfer Function Matrices", IEEE Trans. Automat. Contr., vol AC-31, no 5, pp. 420-429, May 1986.

[11] A. Balluchi "Algoritmo Parallelo per la Decomposizione di Sistemi a Larga Scala", degree thesis, Dipartimento di Sistemi Elettrici e Automazione, Università degli Studi di Pisa, 1993.