UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Parallel Complexity of Domain Decomposition Methods and Optimal Coarse Grid Size

Tony F. Chan
Jian Ping Shao

June 1994

CAM Report 94-15

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

Reprinted from

PARALLEL COMPUTING

Parallel Computing 21 (1995) 1033–1049

Parallel complexity of domain decomposition methods and optimal coarse grid size

Tony F. Chan [a,*], Jian Ping Shao [b,1]

[a] *Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90024, USA*
[b] *Center for Computational Studies, 325 McVey Hall, University of Kentucky, Lexington, KY 40506, USA*

Received 19 September 1994; revised 17 November 1994

ELSEVIER

Parallel Computing

Editorial policy. *Parallel Computing* is an international journal presenting the theory and use of parallel computer systems, including vector, pipeline, array, fifth and future generation computers and neural computers. Within this context the journal covers all aspects of high-speed computing.
Its primary objectives are:
- publication of new research results pertaining to the development and application of parallel computer systems;
- dissemination of information on new developments and events in this field.

The editorial policy and the technical content of the journal are the responsibility of the Editors and the Editorial board. The journal is self-supporting from subscription income and contains a minimum amount of advertisements. Advertisements are subject to the prior approval of the Editors.

Scope. *Parallel Computing* features original research work, tutorial and review articles as well as accounts on practical experience with and techniques for the use of parallel computers. Contributions can cover:
- algorithm design for all types of parallel computers;
- all aspects of the application of parallel computers, including those to AI, CAD/CAM, databases, information retrieval, etc. in industrial, business and office environments;
- the impact of high-speed computation on current research, as well as the advancement of knowledge;
- software engineering aspects relating to parallel computing;
- software for parallel computer systems including programming languages, operating systems, utilities, libraries, etc.;
- networking technology for support of high-speed computing via centralised file servers, output servers, interactive access, etc.;
- taxonomy, models and architectural trends of parallel processing;
- general hardware (architecture) concepts, new technologies enabling the realisation of such new concepts as well as details of commercially available systems;
- performance measurement results on state-of-the-art systems;
- peripheral devices for parallel (super-)computers.

(∞) The paper used in this publication meets the requirements of ANSI/NISO 239.48-1992 (Permanence of Paper).

0167-8191/95/$09.50 Published monthly Printed in The Netherlands

ELSEVIER

Parallel Computing 21 (1995) 1033–1049

PARALLEL COMPUTING

Parallel complexity of domain decomposition methods and optimal coarse grid size

Tony F. Chan [a,*], Jian Ping Shao [b,1]

[a] *Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90024, USA*
[b] *Center for Computational Studies, 325 McVey Hall, University of Kentucky, Lexington, KY 40506, USA*

Received 19 September 1994; revised 17 November 1994

Abstract

In this paper, we analyze the parallel complexity of domain decomposition method through estimating the arithmetic time as well as the communication time. In most domain decomposition (DD) methods, a coarse grid solve is employed to provide global coupling and obtain an optimal convergence rate. Generally, a small coarse grid size H improves the convergence rate at the cost of a larger coarse grid problem and a large number P of processors results in more costly global communication, whereas a large H and a small P have the opposite effect. Therefore, there often exists an optimal H such that the execution time on the parallel machine reaches a minimum. Our main goal here is to find this optimal coarse grid size H in the parallel environment. We assume the same solver is used for the subdomains as well as the coarse problem. By expressing the parallel complexity as a function of H and h, we can derive the analytic optimal coarse grid size H_{opt}. When the number of processors is equal to the number of subdomains, we can prove that the choice of $H_{opt} = \sqrt{h}$ asymptotically minimises the execution time as h tends to zero. This optimal coarse grid size is independent of the choice of subproblem solvers and the spatial dimension. However, when the number of processors is much less than the number of subdomains, the optimal coarse grid size depends on the number of processors, the subdomain solver and dimension.

* Corresponding author. Email: chan©math.ucla.edu. This work was supported in part by the Army Research Office under contract DAAL03-91-G-0150 and subcontract under DAAL03-91-C-0047, by the National Science Foundation Grant ASC 92-01266, and by the Office for Naval Research under contract ONR N00014-92-J-1890.

[1] This work was supported in part by the National Science Foundation Grant ASC-9310315.

SSDI 0167-8191(95)00009-7

Keywords: Computation time; Arithmetic time; Communication time; Parallel complexity; Execution time; Domain decomposition; Optimal coarse grid size choice H

1. Introduction

Domain decomposition (DD) methodology has been quickly developed and widely applied in designing efficient parallel algorithms for solving many practical problems, such as partial differential equations [15,6,5,16]. The advantage of this methodology is its suitability of implementation on parallel and high performance computer architectures and its adaptation to special features of a problem, such as irregular domain, discontinuous coefficients, singular problem and boundary layer, etc. There are two main classes of domain decomposition methods: overlapping Schwarz methods [12,13,9,4,31] and iterative substructuring (non-overlapping) methods [3,1,25,23,27]. In all these domain decomposition methods, a coarse grid solve is used to provide global coupling so that the convergence rate can be bounded independent of the grid size h. In general, a small coarse grid size H improves the convergence rate (because the coarse grid problem is a better approximation to the original fine grid problem) at the cost of a more costly coarse grid solve, whereas a large H has the opposite effect. Therefore, an optimal value H_{opt} often exists and indeed has been observed empirically [19,30]. Even though there exist several papers on the parallel complexity of the DD methods, for example see [18,21], there has been almost no systematic study in the literature on the minimization of execution time through choosing the coarse grid size H. The similar timing model used in Hoffmann and Zou's paper [21] will be taken here to derive the optimal coarse grid size H_{opt}. In [8], we already discuss how to minimize the serial complexity of the DD method through choosing the optimal coarse grid size, and give a brief analysis on the minimization of parallel complexity of the DD method as well. Here, a more detailed analysis is given on the estimation of parallel execution time and the choice of the optimal coarse grid size H, when the number of processors is equal to or much less than the number of subdomains. For simplicity, we only consider the additive Schwarz methods with small overlapping subregions. Our study on the parallel complexity can be easily extended to the other domain decomposition methods.

The focus of this paper is on the choice of an optimal coarse grid size H, which asymptotically minimize the *parallel* complexity on a multiprocessor system. On a parallel computer, a partitioning of the underlying computation and their allocation to individual processors are required so that the synchronization and communication cost is kept at a minimum. Hence, by assuming that there are enough available processors, we distribute the subdomain problems to different processors so that neighboring subproblems are allocated to neighbor processors. Then all the subdomain problem can be solved completely in parallel. A crucial issue is how to solve the coarse grid problem in the parallel environment. According to Gropp [18], one of the best ways is to gather the necessary data on one processor, solve it there and then broadcast the result. Another good approach is to gather data in all

processors. For simplicity, we only consider gathering data on one processor when solving the coarse problem. We discuss two cases: the coarse grid subproblem is solved on one processor either (a) sequentially after the subdomain solves, or (b) in parallel to the subdomain solves. It is observed that too small H (that means too many subproblems) can result in very bad workload balance among processors and very expensive global communication, even though a small H improves the convergence rate. The reason is the work load of the processor for solving the coarse grid problem is much heavier than that of the other processors. This implies that increasing the number of processors in the parallel system does not necessarily shorten the total execution time. Therefore, an optimal value H_{opt} often exists. To find the optimal coarse grid size H_{opt} analytically, we need to analyze the parallel complexity, which is proportional to the total execution time on the parallel system. Our approach is to take a simple parallel computer system with a mesh connection structure, a model three dimensional elliptic problem and a particular DD method; allowing a simple but complete and easily understood analysis which we think do give insights for general situations and other DD methods.

This paper is organized as follows. In Section 2, we present the model three dimensional elliptic problem. In Section 3, we introduce some concepts of parallel complexity and give the relation between the complexity and the communication time and arithmetic time. In Section 4, we estimate the parallel execution time of the preconditioned conjugate gradient (CG) method on the parallel architecture as the sum of the times for the conjugate gradient method and the DD preconditioner. We analyze the execution time as a function of the coarse grid size H. Finally, in Section 5, we simplify and extend the estimation of complexity by ignoring the lower order terms so that we can analytically derive the optimal coarse grid size H_{opt}, when the number of processors is equal to or much less than the number of subdomains. Section 6 gives the conclusions and remarks.

2. The elliptic problem and the additive Schwarz method

To make our analysis simple here, we consider the model problem on $\Omega = (0, 1) \times (0, 1) = (0, 1) \subset R^3$:

$$\begin{cases} -\nabla \cdot (a(x)\nabla u) = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{1}$$

with $a(x, y) \in R^{3\times 3}$ uniformly positive definite, bounded on Ω. The extension to more complicated domains Ω can be obtained in the same way. We partition the domain Ω into $N \times N \times N$ non-overlapping sub-domains $\Omega_1, \ldots, \Omega_{N^3}$, which form the elements of a *quasi-uniform* coarse grid triangulation τ^H. The diameter of each subdomain is H and $N = 1/H$. By refining every sub-region Ω_i, we produce a fine grid *quasi-uniform* triangulation τ^h having elements of diameter h. Therefore, there are $n \times n \times n$ grid points on the whole domain Ω_h with $n = 1/h$. Assume that each sub-region Ω_i has $s \times s \times s$ nodes where $s = H/h$. We obtain two finite element spaces V^h and V^H. Corresponding to the coarse grid and fine grid

triangulations, we discretize (1) either by using finite elements or by using finite difference methods. Then, we obtain the following symmetric positive definite linear systems

$$A_h u_h = f_h \tag{2}$$

on the fine grid and

$$A_H u_H = f_H \tag{3}$$

on the coarse grid.

Eq. (2) will be solved by the preconditioned conjugate gradient (PCG) with the additive Schwarz method (ASM) as the preconditioner. The detail description of ASM can be found in [10,11], and we give only a brief description here. We first extend each substructure Ω_i to a larger region $\tilde{\Omega}_i$ such that the distance between the boundaries $\partial\Omega_i$ and $\partial\tilde{\Omega}_i$ is bounded uniformly below by δ. Assume there are m^3 interior points in $\tilde{\Omega}_i$. We take overlap = 1 and then the number of points in one direction of subdomain becomes $m = (H/h) + 2$.

Remark. Theoretically, the condition number of the additive Schwarz method is $(1 + H/\delta)$ where δ is the overlapping size [14] and $\delta = h$ for our case. However, from numerical experiments [7,28], we notice that the convergence rate depends only slightly on the overlapping size of the sub-domains. Hence, only a small overlap is necessary in practice.

We introduce R_i as the pointwise restriction of nodal values to $\tilde{\Omega}_i$ and define the transpose R_H^T as a linear interpolation from V^h to V^H. On each sub-region $\tilde{\Omega}_i$, we obtain a discrete problem from Eq. (1):

$$\tilde{A}_i u_i = f_i,$$

where $\tilde{A}_i \equiv R_i A_h R_i^T$ is a $m^3 \times m^3$ square matrix, u_i is the vector of unknowns corresponding to the interior points of $\tilde{\Omega}_i$ and f_i is the sub-vector of the vector f_h corresponding to the points of $\tilde{\Omega}_i$. For problem (3), we define $A_H \equiv R_H A_h R_H^T$. Then the additive Schwarz preconditioner is defined by its action on a vector r:

$$M^{-1} r = R_H^T A_H^{-1} R_H r + \sum_i R_i^T \tilde{A}_i^{-1} R_i r. \tag{4}$$

3. Basic communicating times

We assume that computation and communication are not overlapped and therefore, the execution time of parallel implementation is the sum of the computation time and the communication time. For simplicity, we consider a parallel system consisting of $P = N \times N$ processors with a mesh connection pattern as illustrated in Fig. 1. To achieve a good balance on the workload of all processors, we map all the subproblems on the N sub-domains $\tilde{\Omega}_i$ in the same column onto a

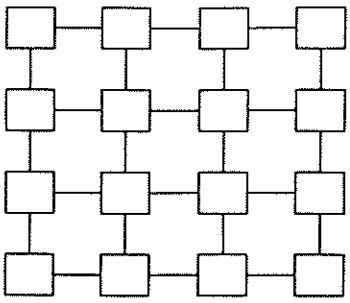

Fig. 1. A mesh-connected array.

processor so that the problems on the neighbor subdomains are solved on the neighbor processors. In this way, the processor synchronization and communication cost is kept low.

For our timing model, we shall use standard models for the arithmetic and communication times [26,21]. First, with assuming no arithmetic vectorization, doing M arithmetic operations on one processor takes time

$$T_a = M * t_a, \tag{5}$$

where t_a is the average unit time to perform one arithmetic operation on one processor.

Next, the *Local communication time* of moving M data from one processor to its neighbor processors is approximated by

$$T_{local} = t_s + M * t_c \tag{6}$$

where t_s is the start-up time and t_c is the incremental time necessary for each of the M words to be sent.

Finally, for the model mesh-connected system, we use the following two kinds of global communication models from [26]:

(1) *Broadcasting* a data packet of size M from one processors to all others: The data packet is broken into smaller segments and the sending of these segments is pipelined with the optimal segment size. It takes time:

$$T_{broadcast} = \left(P^{1/4}\sqrt{t_s} + \sqrt{Mt_c}\right)^2, \tag{7}$$

when there are P processors in the computer system.

(2) *Scattering (gathering)* P data packets, each packet with M/P words, from (in) one processor to (from) all others: The gathering operation is the inverse process of scattering operation, so they take the same amount of time. This yields:

$$T_{gather} = \sqrt{2P}\,t_s + \frac{M}{2}t_c. \tag{8}$$

Again, pipelining of smaller data segments is used.

4. The execution time estimation of the PCG method

In this section, we analyze the total execution time of using a parallel system with the mesh connected processors in Fig. 1 to solve problem (2) by a preconditioned additive Schwarz domain decomposition method. We shall estimate the execution time T_{CG} of performing CG iteration and the execution time T_{prec} of implementing the preconditioner separately.

4.1. The execution time T_{CG} of the CG method

For completeness, the standard estimation of parallel complexity of the CG method [24,21] is given here. We will analyze the calculation time and communication time separately. We first analyze the computational complexity of implementing the CG method in parallel. For elliptic problem (2), we list the computation needed in one iteration of the CG method:

- Matrix-vector multiplication Ap^k:
  - Each element of Ap^k requires 7 multiplications and 6 additions. So $13s^3N$ operations are required on each processor, each with N subdomains.
- Two inner products (p^k, Ap^k) and $(\tilde{r}^{k+1}, r^{k+1})$: These take approximately $2s^3N$ operations on each processor.
- Three linked triads $x^{k+1} = x^k - \alpha_k p^k$, $r^{k+1} = r^k + \alpha_k Ap^k$ and $p^{k+1} = r^{k+1} + \beta_k p^k$: These need about $6s^3N$ operations on every processor.

Summing these up, the total *arithmetic time* T_{ACG} of doing one iteration in CG takes approximately time:

$$T_{ACG} \approx 21s^3Nt_a, \tag{9}$$

according to (5) when $M = s^3N$.

We now estimate the communication time required in each iteration step of the CG method [21]:

(1) Gathering the computational results of two inner products from all processors to the one handling the accumulation takes time:

$$T_{gather} = 2\left(\sqrt{2}Nt_s + \frac{N^2}{2}t_c\right), \tag{10}$$

according to (8) when $P = N^2$ and $M = N^2$.

(2) Broadcasting α_k and β_k from the processor to all others requires the time:

$$T_{broadcast} = 2\left(\sqrt{Nt_s} + \sqrt{t_c}\right)^2, \tag{11}$$

according to (7) when $P = N^2$ and $M = 1$.

(3) Exchanging internal boundary values between neighbor processors when calculating Ap^k. To avoid interruption of sending and receiving data, we color the processors into red and black so that the red processors have only

black processors as their neighbors, and vice versa. Then, the total time of this local communication is

$$T_{local} = 2\left(t_s + Ns^2 t_c\right), \tag{12}$$

according to (6) when $M = Ns^2$.

Summarizing the above communication time, we obtain the approximate total communication time T_{CCG} required in each iteration step of the CG method:

$$\begin{aligned} T_{CCG} &= T_{local} + T_{broadcast} + T_{gather} \\ &= 2\left(t_s + Ns^2 t_c\right) + 2\left(\sqrt{Nt_s} + \sqrt{t_c}\right)^2 + 2\left(\sqrt{2}\,Nt_s + \frac{N^2}{2} t_c\right). \end{aligned} \tag{13}$$

Obviously, the communication time depends not only on the size of the messages but also on the number of processors.

The total execution time T_{CG} of one CG iteration is approximately the sum of the computation time T_{ACG} and the communication time T_{CCG}:

$$\begin{aligned} T_{CG} &= T_{ACG} + T_{CCG} \\ &\approx 21s^3 Nt_a + 2\left(t_s + Ns^2 t_c\right) + 2\left(\sqrt{Nt_s} + \sqrt{t_c}\right)^2 + 2\left(\sqrt{2}\,Nt_s + \frac{N^2}{2} t_c\right). \end{aligned} \tag{14}$$

4.2. Timings for domain decomposition preconditioner

For simplicity, we assign only one processor to solve the coarse problems. The implementation of the additive Schwarz preconditioner (4) can be divided into three steps:

(1) Map the calculation of $R_i^T \tilde{A}_i^{-1} R_i r$ onto the P processors and then calculate these subproblems on all processors simultaneously.
(2) Gather the result $R_H r$ from all processors to one processor, solve the coarse grid problem on the processor, and scatter the result $R_H^T A_H^{-1} R_H r$ from it to all other processors.
(3) Add all these results together to get $M^{-1} r$.

Note that step 1 and step 2 can be implemented *sequentially* (Case (a)) or *simultaneously* (Case (b)).

We first estimate the communication time used in computing $M^{-1} r$. In step 1 and step 3, we need to exchange data on the overlapping region with neighbor sub-domains, which takes time:

$$T_{Cprec}^{(1,3)} = 4\left(t_s + Nm^2 t_c\right), \tag{15}$$

according to (6) when $M = Nm^2$. In step 2, we need to gather the computation result $R_H r$ to one processor. After calculating the coarse grid solution, we have to scatter the results $A_H^{-1} R_H r$ from this processor to all other processors. Together, it takes time:

$$T_{Cprec}^{(2)} = 2\left(\sqrt{2}\,Nt_s + \tfrac{1}{2} N^3 t_c\right) \tag{16}$$

Table 1
The complexity of various methods on m^3 grids

Method	Factorization $f(m)$	Solver $g(m)$
Band-Cholesky [17]	$2m^7$	$4m^6$
Multigrid [20]	0	$40m^3$
FFT Fast Elliptic [22]	0	$7m^3 \ln_2 m$

to gather and scatter the results of $A_H^{-1}R_H r$, according to (8) when $P=N^2$ and $M=N^3$. So the total communication time in one preconditioning step is the sum of (15) and (16):

$$T_{Cprec} = 4(t_s + Nm^2 t_c) + 2\left(\sqrt{2}Nt_s + N^3\frac{t_c}{2}\right). \tag{17}$$

We next estimate the computational time in one preconditioning step. This depends on the solver employed. We list the leading order terms of the complexity

Table 2
The total execution time T of the PCG method

	(a) Sequential Coarse Solve	(b) Simultaneous Coarse Solve
T_{fact}	$\left\{\frac{1}{H}f\left(\frac{H}{h}+2\right)+f\left(\frac{1}{H}\right)\right\}t_a$	$\max\left\{\frac{1}{H}f\left(\frac{H}{h}+2\right), f\left(\frac{1}{H}\right)\right\}t_a$
T_{CG}	$K\left\{\frac{21}{H}\left(\frac{H}{h}\right)^3 t_a + 2\left(t_s + \frac{1}{H}\left(\frac{H}{h}\right)^2 t_c\right) + 2\left(\sqrt{\frac{t_s}{H}}+\sqrt{t_c}\right)^2 + 2\left(\sqrt{2}\frac{t_c}{H}+\frac{t_c}{2H^2}\right)\right\}$	$K\left\{\frac{21}{H}\left(\frac{H}{h}\right)^3 t_a + 2\left(t_s + \frac{1}{H}\left(\frac{H}{h}\right)^2 t_a\right) + 2\left(\sqrt{\frac{t_c}{H}}+\sqrt{t_a}\right)^2 + \left(\sqrt{2}\frac{t_c}{H}+\frac{t_c}{2H^2}\right)\right\}$
T_{prec}	$K\left[\left\{\frac{1}{H}g\left(\frac{H}{h}+2\right)+g\left(\frac{1}{H}\right)\right\}t_a + 4\left(t_a + \frac{1}{H}\left(\frac{H}{h}+2\right)^2 t_c\right) + 2\left(\frac{\sqrt{2}}{H}t_a + \frac{t_a}{2H^2}\right)\right]$	$K\left[max\left\{\frac{1}{H}g\left(\frac{H}{h}+2\right), g\left(\frac{1}{H}\right)\right\}t_a + 4\left(t_s + \frac{1}{H}\left(\frac{H}{h}+2\right)^2 t_c\right) + 2\left(\frac{\sqrt{2}}{H}t_s + \frac{t_c}{2H^2}\right)\right]$
T	$\left\{\frac{1}{H}f\left(\frac{H}{h}+2\right)+f\left(\frac{1}{H}\right)\right\}t_a + K\left\{\frac{21}{H}\left(\frac{H}{h}\right)^3 t_a + 2\left(t_s + \frac{1}{H}\left(\frac{H}{h}\right)^2 t_a\right) + 2\left(\sqrt{\frac{t_c}{H}}+\sqrt{t_c}\right)^2 + 2\left(\sqrt{2}\frac{t_c}{H}+\frac{t_c}{2H^2}\right)\right\} + \left\{\frac{1}{H}g\left(\frac{H}{h}+2\right)+g\left(\frac{1}{H}\right)\right\}t_a + 4\left(t_a + \frac{1}{H}\left(\frac{H}{h}+2\right)^2 t_c\right) + 2\left(\frac{\sqrt{2}}{H}t_s + \frac{t_c}{2H^2}\right)\right\}$	$\max\left\{\frac{1}{H}f\left(\frac{H}{h}+2\right), f\left(\frac{1}{H}\right)\right\}t_a + K\left\{\frac{21}{H}\left(\frac{H}{h}\right)^3 t_a + 2\left(t_s + \frac{1}{H}\left(\frac{H}{h}\right)^2 t_c\right) + 2\left(\sqrt{\frac{t_c}{H}}+\sqrt{t_a}\right)^2 + 2(\sqrt{2}\frac{t_c}{H}+\frac{t_c}{2H^2}) + \max\{\frac{1}{H}g(\frac{H}{h}+2), g(\frac{1}{H})\}t_a + 4\left(t_s + \frac{1}{H}\left(\frac{H}{h}+2\right)^2 t_c\right) + 2\left(\frac{\sqrt{2}}{H}t_s + \frac{t_c}{2H^2}\right)\right\}$

of various sub-domain solvers for the model problem on an $m \times m \times m$ grid in Table 1.

Let T_{fact} be the execution time of performing factorization on the parallel computer. Assume that each processor has N subproblems on N subdomains. Then, on the parallel computer, the computation times of performing factorization T_{fact} and one preconditioning step T_{Aprec}, respectively, are

(1) For sequential case (a),

$$T_{fact} \approx Nf(m) + f(N), \quad T_{Aprec} \approx Ng(m) + g(N); \tag{18}$$

(2) For simultaneous case (b),

$$T_{fact} \approx \max\{Nf(m), f(N)\}, \quad T_{Aprec} \approx \max\{Ng(m), g(N)\}. \tag{19}$$

Using the formula

$$T_{prec} = T_{Cprec} + T_{Aprec},$$

and (17), we obtain the total times of implementing one preconditioning step:

(a) Sequential Coarse Solve:

$$T_{prec} = 4\left(t_s + Nm^2 t_c\right) + 2\left(\sqrt{2}\,Nt_s + \frac{t_c}{2}N^3\right) + \{Ng(m) + g(N)\}, \tag{20}$$

(b) Simultaneous Coarse Solve:

$$T_{prec} = 4\left(t_s + Nm^2 t_c\right) + 2\left(\sqrt{2}\,Nt_s + \frac{t_c}{2}N^3\right) + \max\{Ng(m), g(N)\}. \tag{21}$$

4.3. The total execution time of the PCG method

Let K be the number of iteration steps of the PCG method. Then, the total execution time is the sum of factorization time and the time of doing K PCG iterations:

$$T = T_{fact} + K\left(T_{CG} + T_{prec}\right).$$

By using (18), (19), (14), (20) and (21), we summarize the total execution times T in Table 2 of performing K iterations of the PCG method with the additive Schwarz method (ASM) preconditioner for solving three dimensional model problem on a parallel system of the mesh connected processors.

As a particular example, we consider using FFT fast Poisson solver as subproblem solver for constant coefficient problems. Then, the functions f and g in Table 2 are defined by

$$f(m) = 0 \qquad g(m) = 7m^3 \ln_2 m.$$

When *Band-Choleskey* is used as the sub-problem solver, the functions f and g in Table 2 become

$$f(m) = 2m^7 \qquad g(m) = 4m^5.$$

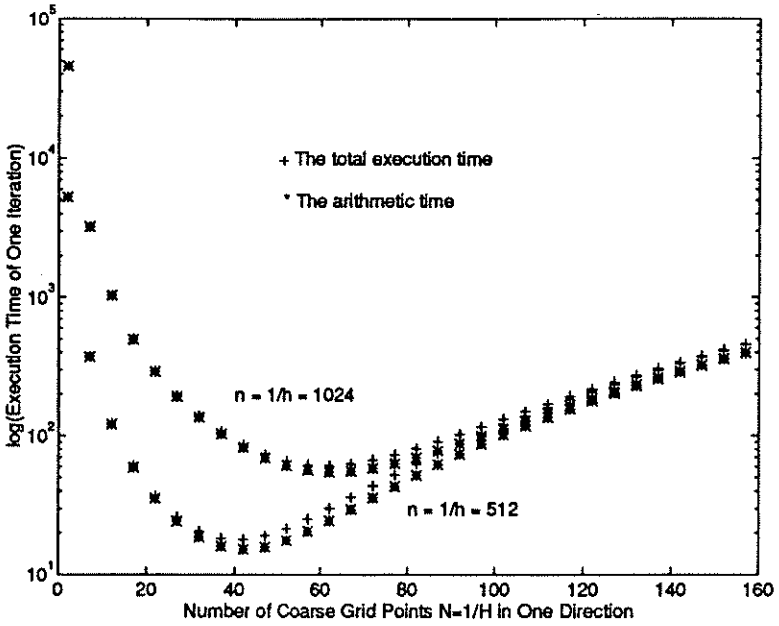

Fig. 2. Additive Schwarz method with FFT solver.

In Figs. 2 and 3, we graph the total execution time and the arithmetic time of performing $K = 20$ iterations of the PCG as a function of $1/H$ with fixed fine grid size h by using time formula defined in Table 2. Based on the numerical results in [7,28], the value $K = 20$ is sufficient to achieve a reduction in the norm of the residual by a factor of 10^{-5}. We determine the communication start-up time t_s and the elemental communication time t_c by matching the communication time model [24]

$$T = t_s + Mt_c$$

with the numerical results on the Intel iPSC/860 provided by Smith [29]. By using a least square method, we obtain $t_s = 196.94 * 10^{-6}$ and $t_c = 0.3574 * 10^{-6}$. In the following figures, the flop rate per unit time is defined as 10 MFLOPS, i.e. $t_a = 10^{-7}$.

Observation. Note that the plots show that, for a fixed h, there is indeed an optimal value H_{opt} for which the execution time is minimal and this optimal value decreases with decreasing h. We also note that the optimal coarse grid size H_{opt} for the total execution time is almost the same as that for the arithmetic time. Note that it is safer to over-estimate $1/H_{opt}$ in practice since the total time $T(H)$ increases slower for $H < H_{opt}$ than for $H > H_{opt}$. Observe that the execution time is almost the same for various fine grid size h when the number of coarse grid points is large because the major contribution to the total execution time is from solving the coarse grid problems. Hence, choosing too fine a coarse grid will result in wasteful computation.

5. Optimal choice of coarse grid size

In this section, we derive analytical formulas for the optimal coarse grid size H_{opt} for a d-dimensional model problem on a parallel system with P processors. From the previous section, we already notice that both the arithmetic time and the total execution time reach their minima at nearly the same optimal coarse grid size, H_{opt}. Hence, we shall use only the arithmetic timing formula in deriving the approximate H_{opt}. Moreover, we shall ignore lower order terms. This makes the derivation of the analytical formulas feasible without losing too much accuracy in the approximate values of H_{opt}.

Assume that there are only P processors in the parallel system. For the problem on a d-dimensional domain, we have $1/H^d$ subproblems on the $1/H^d$ subdomains. We equally distribute the calculation of solving these subproblems on P processors so that the work load is well balanced and the communication time is minimized. Then, by ignoring the lower order terms, the simplified total execution time is:

(a) Sequential Coarse Solve:

$$T \approx \left\{ \frac{1}{H^d P} f\left(\frac{H}{h}\right) + f\left(\frac{1}{H}\right) \right\} t_a + K \left\{ \frac{1}{H^d P} g\left(\frac{H}{h}\right) + g\left(\frac{1}{H}\right) \right\} t_a; \tag{22}$$

(b) Simultaneous Coarse Solve:

$$T \approx \max\left\{ \frac{1}{H^d P} f\left(\frac{H}{h}\right), f\left(\frac{1}{H}\right) \right\} t_a + K \max\left\{ \frac{1}{H^d P} g\left(\frac{H}{h}\right), g\left(\frac{1}{H}\right) \right\} t_a, \tag{23}$$

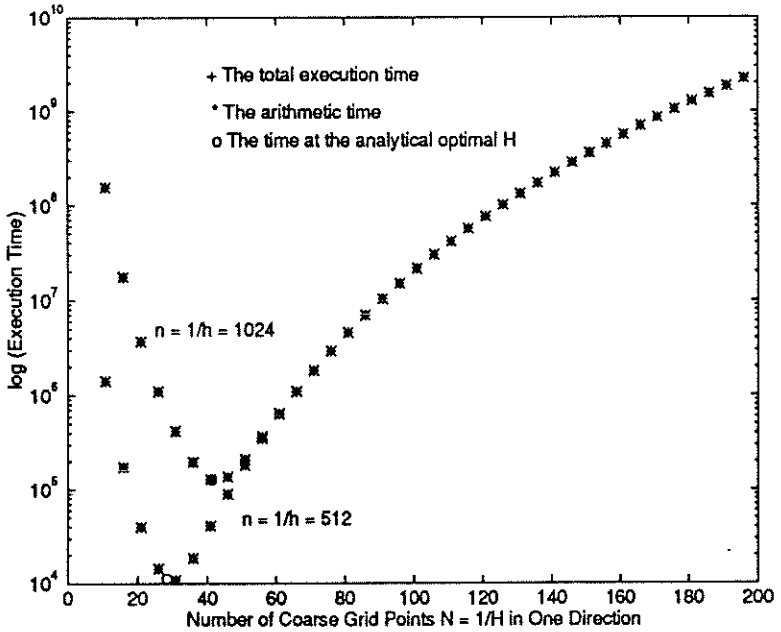

Fig. 3. Additive Schwarz method with band solver.

where $(1/H^dP)f(H/h)$ and $(1/H^dP)g(H/h)$ are the contributions from solving problems on all the subdomains, and the terms $f(1/H)$ and $g(1/H)$ are from solving the coarse problem. We assume that the iteration number K is a finite integer since the convergence rate is independent of the mesh parameters h and H according to the theoretical and numerical results [10,11,28,2,7]. Without loss of generality, we assume that the functions f and g are monotone. Defining

$$w(m) \equiv f(m) + Kg(m),$$

the total execution time can be written as:

(a) Sequential Coarse Solve:

$$T(H) \approx \left\{ \frac{1}{H^dP} w\left(\frac{H}{h}\right) + w\left(\frac{1}{H}\right) \right\} t_a; \tag{24}$$

(b) Simultaneous Coarse Solve:

$$T(H) \approx \max\left\{ \frac{1}{H^dP} w\left(\frac{H}{h}\right), w\left(\frac{1}{H}\right) \right\} t_a. \tag{25}$$

We approximately derive the optimal coarse grid size from these simplified execution time formulae. In principle, the optimal coarse grid size H_{opt} is the solution to the equation:

$$T'(H) = 0 \quad \text{for the sequential case (a)},$$

or

$$\frac{1}{H^dP} w\left(\frac{H}{h}\right) = w\left(\frac{1}{H}\right) \quad \text{for the simultaneous case (b)}.$$

However, in the general case, it is not possible to obtain a closed analytical expression for H_{opt}.

Now, we will consider 3 special cases for which we can indeed derive H_{opt} analytically.

Case 1. $P = N^d$ or $P = N^d + 1$; that is, the number of processors equals the number of subdomains.

(a) Sequential Coarse Solve: By solving

$$T'(H) = \frac{1}{h} w'\left(\frac{H}{h}\right) - \frac{1}{H^2} w'\left(\frac{1}{H}\right) = 0,$$

we obtain the approximate optimal coarse grid size

$$H_{opt} = \sqrt{h}\,. \tag{26}$$

Then, by (22), the total execution time with this H_{opt} is

$$T(H_{opt}) = 2f\left(\frac{1}{\sqrt{h}}\right) t_a + 2Kg\left(\frac{1}{\sqrt{h}}\right) t_a + l.o.t.$$

(b) Simultaneous Coarse Solve: By setting

$$w\left(\frac{H}{h}\right) = w\left(\frac{1}{H}\right),$$

we get the approximate optimal coarse grid size

$$H_{opt} = \sqrt{h}\,, \tag{27}$$

at which, by (23), the total execution time becomes:

$$T(H_{opt}) = f\left(\frac{1}{\sqrt{h}}\right)t_a + Kg\left(\frac{1}{\sqrt{h}}\right)t_a + l.o.t.$$

Note that the optimal choice of H_{opt} is independent of the complexity of the solver and the spatial dimension d of the domain Ω. Note this independence is not true when the algorithm is implemented on a serial machine [8].

For the case $w(m) = cm^{\alpha}$, $\alpha > d$ on a m^d grid, the optimal coarse grid size $H_{opt} = \sqrt{h}$ reduces the total execution time from $O(h^{-\alpha})$ using one processor to the order of $O(h^{-\alpha/2})$ using $P = h^{-d/2}$ processors.

Remark. The optimal choice of $H_{opt} = \sqrt{h}$ has a very intuitive interpretation. It ensures that all the problems solved in parallel (i.e. each subdomain problem and coarse problem) have the same size and can be solved in the same amount of time.

Case 2. $P = N^{\beta}$ where $0 < \beta < d$ is an integer. Unlike the previous case, the optimal value of H in this case depends on the complexity of $w(m)$. We shall consider the specific case of $w(m) = cm^{\alpha}$, with $\alpha \geq d$. Note that the value of α depends on whether the factorization time $f(m)$ or the solver time $Kg(m)$ is dominant.

(a) Sequential Coarse Solve: The approximate total execution time (24) becomes

$$T(H) \approx C\left\{\frac{1}{H^{d-\beta}}\left(\frac{H}{h}\right)^{\alpha} + \left(\frac{1}{H}\right)^{\alpha}\right\}t_a.$$

We minimize this function by setting

$$T'(H) = c\left\{(\alpha+\beta-d)\frac{H^{\alpha+\beta-d-1}}{h^{\alpha}} - \alpha H^{-\alpha-1}\right\} = 0,$$

with the solution

$$H_{opt} = \left(\frac{\alpha}{\alpha+\beta-d}\right)^{1/(2\alpha+\beta-d)} h^{\alpha/(2\alpha+\beta-d)}.$$

For example, in the case of banded factorization on an N by N processor

systems for three dimensional problems, we have $d=3$, $\beta=2$ and $\alpha=7$, and therefore,

$$H_{opt}=\left(\frac{7h^7}{6}\right)^{1/13} \qquad T(H_{opt})\approx C\left\{\left(\tfrac{7}{6}\right)^{6/13}+\left(\tfrac{7}{6}\right)^{-7/13}\right\}h^{-49/13}t_a.$$

We note that the circle plotted in Fig. 3 corresponds to this analytical optimal coarse grid size H_{opt}. It can be seen that this analytical approximation is very close to the true minimum of the total execution time.

(b) Simultaneous Coarse Solve: The approximate total execution time (25) becomes

$$T(H)\approx C\max\left\{\frac{1}{H^{d-\beta}}\left(\frac{H}{h}\right)^{\alpha},\left(\frac{1}{H}\right)^{\alpha}\right\}t_a.$$

By taking

$$\frac{1}{H^{d-\beta}}\left(\frac{H}{h}\right)^{\alpha}=\left(\frac{1}{H}\right)^{\alpha},$$

we obtain the optimal coarse grid size:

$$H_{opt}=h^{\alpha/(2\alpha-d+\beta)}.$$

Note that since $\alpha/(2\alpha-d+\beta)<\frac{1}{2}$, we have in both case (a) and (b) $H_{opt}(\beta<d)<H_{opt}(\beta=d)$ asymptotically.

From the fact that $T(2h)\gg T(H_{opt})$ and $T(\frac{1}{8})\gg T(H_{opt})$ as h tends to zero, we see that a judicial choice of coarse grid size can result in significant reduction on the total complexity.

Case 3. $0<P<N$ is independent of the number of subdomains. We still assume that $w(m)=cm^{\alpha}$ in (24) and (25). The reasons we are interested in this case are: (1) there are many parallel systems consisting of only a small fixed number processors; (2) it is very often that the available processors are limited.

(a) Sequential Coarse Solve: By setting

$$T'(H)=\frac{(\alpha-d)H^{\alpha-d-1}}{h^{\alpha}P}-\frac{\alpha}{H^{\alpha+1}}=0,$$

we obtain the optimal coarse grid size

$$H_{opt}=\left(\frac{\alpha P}{\alpha-d}\right)^{1/(2\alpha-d)}h^{\alpha/(2\alpha-d)}.$$

(b) Simultaneous Coarse Solve: To minimize the execution time, we requires

$$\frac{1}{H^{d}P}\left(\frac{H}{h}\right)^{\alpha}=\left(\frac{1}{H}\right)^{\alpha}.$$

It follows that the optimal coarse grid size is

$$H_{opt} = (P)^{1/(2\alpha-d)} h^{\alpha/(2\alpha-d)}.$$

Note that asymptotically the value of H_{opt} is even smaller than the previous case of $P = N^{\beta}$.

6. Conclusions and remarks

We have analyzed the execution time of the PCG method with additive Schwarz preconditioner for three dimensional elliptic model problem on a parallel system with mesh connected processors. From our timing models, we observe that there exists an optimal coarse grid size H_{opt} and the arithmetic time reaches the minimum at the almost same point as the total execution time. From this fact, we simplify the total execution time by ignoring lower order terms so that we can easily obtain an analytic formula for the optimal H_{opt}. For example, when the number of processors is equal to the number of subdomains, the optimal coarse grid size is $H = \sqrt{h}$. This optimal choice is independent of the spatial dimension and the complexity of the solver for subproblem. However, when the number of processors is less than the number of the subdomains, the optimal choices of the coarse grid depend on the number of the processors, the dimension of the space and the complexity of the solver.

Acknowledgments

The authors would like to thank Barry Smith for providing the numerical data to determine the parameters t_a, t_s and t_c.

References

[1] J.-F. Bourgat, R. Glowinski, P. Le Tallec and M. Vidrascu, Variational formulation and algorithm for trace operator in domain decomposition calculations, in *Domain Decomposition Methods*, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds. (Philadelphia, 1988, SIAM).

[2] J.H. Bramble, J.E. Pasciak and A.H. Schatz, The construction of preconditioners for elliptic problems by substructuring, I, *Math. Comp.* 47 (1986) 103–134.

[3] J.H. Bramble, J.E. Pasciak and A.H. Schatz, The construction of preconditioners for elliptic problems by substructuring, IV, *Math. Comp.* 53 (1989) 1–24.

[4] J.H. Bramble, J.E. Pasciak, J. Wang and J. Xu, Convergence estimates for product iterative methods with applications to domain decomposition, *Math. Comp.* 57 (1991) 1–21.

[5] T. Chan, R. Glowinski, J. Périaux and O. Widlund, eds., Domain Decomposition Methods (SIAM, Philadelphia, 1990), *Proc. Third Int. Symp. on Domain Decomposition Methods*, Hoston, Texas (1989).

[6] T. Chan, R. Glowinski, J. Périaux and O. Widlund, eds., Domain Decomposition Methods (SIAM, Philadelphia, 1989), *Proc. Second Int. Symp. on Domain Decomposition Methods*, Los Angeles, CA (Jan. 14–16, 1988).

[7] T.F. Chan, T.P. Mathew and J.P. Shao, Efficient variants of the vertex space domain decomposition algorithm, SIAM J. Sci. Comp. 15 (6) (1994) 1349–1374.
[8] T.F. Chan and J. Shao, Optimal coarse grid size in domain decomposition, Journal of Computat. Math. 12 (4) (1994) 291–297.
[9] M. Dryja, An additive Schwarz algorithm for two- and three-dimensional finite element elliptic problems, in *Domain Decomposition Methods*, T. Chan, R. Glowinski, J. Périaux and O. Widlund, eds. (Philadelphia, 1989, SIAM).
[10] M. Dryja and O.B. Widlund, An additive variant of the Schwarz alternating method for the case of many subregions, Tech. Rep. 339, also Ultracomputer Note 131, Department of Computer Science, Courant Institute, 1987.
[11] M. Dryja and O.B. Widlund, Some domain decomposition algorithms for elliptic problems, in *Proc. Conf. on Iterative Methods for Large Linear Systems*, Austin, Texas (Oct. 1988) (Academic Press, Orlando, Florida, 1989).
[12] M. Dryja and O.B. Widlund, Towards a unified theory of domain decomposition algorithms for elliptic problems, Tech. Rep. 486, also Ultracomputer Note 167, Department of Computer Science, Courant Institute, 1989.
[13] M. Dryja and O.B. Widlund, Additive Schwarz methods for elliptic finite element problems in three dimensions, in *Fifth Int. Symp. on Domain Decomposition Methods for Partial Differential Equations*, T. Chan, D.E. Keyes, G.A. Meurant, J.S. Scroggs and R.G. Voigt, eds. (Philadelphia, 1992, SIAM).
[14] M. Dryja and O.B. Widlund, Domain decomposition algorithms with small overlap, SIAM J. Sci. Comp. 15 (3) (1994) 604–620.
[15] R. Glowinski, G.H. Golub, G.A. Meurant and J. Périaux, eds., Domain Decomposition Methods for Partial Differential Equations (SIAM, Philadelphia, 1988), *Proc. First Int. Symp. on Domain Decomposition Methods for Partial Differential Equations*, Paris, France (Jan. 1987).
[16] R. Glowinski, Y. Kuznetsov, G. Meurant, J. Périaux and O. Widlund, eds., Domain Decomposition Methods (SIAM, Philadelphia, 1991) *Proc. Fourth Int. Symp. on Domain Decomposition Methods*, Moscow, USSR (1990).
[17] G.H. Golub and C.F.V. Loan, *Matrix Computations* (Johns Hopkins Univ. Press, 1989) second ed.
[18] W.D. Gropp, Parallel computing and domain decomposition, in *Domain Decomposition Methods*, T. Chan, D.E. Keyes, G. Meurant, J.S. Scroggs and R.G. Voigt, eds. (Philadelphia, 1992, SIAM).
[19] W.D. Gropp and D.E. Keyes, Domain decomposition with local mesh refinement, Tech. Rep. RR-726, Yale University, Dept. of Comp. Sci., August 1989, SIAM J. Sci. Stat. Comp. 13 (1992) 967–993.
[20] W. Hackbusk, *Multi-Grid Methods and Applications*, (Springer-Verlag, 1985).
[21] K.-H. Hoffmann and J. Zou, Parallel efficiency of domain decomposition methods, *Parallel Comput.* 19 (1993) 1375–1391.
[22] C.F.V. Loan, *Computational Frameworks for the Fast Fourier Transform*, (SIAM, Philadelphia, 1992).
[23] J. Mandel, Iterative solvers by substructuring for the p-version finite element method, *Comput. Meth. Appl. Mech. Engin.* 80 (1990) 117–128.
[24] J.M. Ortega, *Introduction to Parallel and Vector Solution of Linear Systems* (Plenum Press, New York, 1988).
[25] Y.H.D. Roeck and P.L. Tallec, Analysis and test of a local domain decomposition preconditioner, in *Fourth International Domain Decomposition Methods for Partial Differential Equations*, R. Glowinski, Y.A. Kuznetsov, G. Meurant, J. Périaux and O. Widlund, eds. (Philadelphia, 1991, SIAM).
[26] Y. Saad and M.H. Schultz, Data communication in parallel architectures, *Parallel Comput.* 11 (1989) 131–150.
[27] B.F. Smith, A domain decomposition algorithm for elliptic problems in three dimensions, *Numer. Math.* 60 (1991) 219–234.
[28] B.F. Smith, An optimal domain decomposition preconditioner for the finite element solution of linear elasticity problems, *SIAM J. Sci. Stat. Comput.* 13 (1992) 364–378.

[29] B.F. Smith, A parallel implementation of an iterative substructuring algorithm for problems in three dimensions, *SIAM J. Scientific and Statistical Comput.* 14 (1993) 406–423.
[30] B.F. Smith, Domain decomposition algorithms for partial differential equations of linear elasticity, Tech. Rep. 517, Department of Computer Science, Courant Institute, Sept. 1990. Ph.D. thesis.
[31] J. Xu, Iterative methods by space decomposition and subspace correction, *SIAM Rev.* 34 (1992) 581–613.

Parallel Computing

Publication information
PARALLEL COMPUTING (ISSN 0167-8191). For 1995 volume 21 is scheduled for publication.
Subscription prices are available upon request from the publisher. Subscriptions are accepted on a prepaid basis only and are entered on a calendar year basis. Issues are sent by surface mail except to the following countries where air delivery via SAL is ensured: Argentina, Australia, Brazil, Canada, Hong Kong, India, Israel, Japan, Malaysia, Mexico, New Zealand, Pakistan, PR China, Singapore, South Africa, South Korea, Taiwan, Thailand, USA. For all other countries airmail rates are available upon request.
Claims for missing issues must be made within six months of our publication (mailing) date.
Please address all your requests regarding orders and subscription queries to: Elsevier Science, Journal Department, P.O. Box 211, 1000 AE Amsterdam, The Netherlands. Tel.: 31-20-4853642, fax: 31-20-4853598.
US mailing notice – Parallel Computing (0167-8191) is published monthly by Elsevier Science (Molenwerf 1, Postbus 211, 1000 AE Amsterdam). Annual subscription price in the USA: US$895.00 (US$ price valid in North, Central and South America only), including air speed delivery. Second class postage paid at Jamaica, N.Y. 11431.
USA POSTMASTERS: Send address changes to Parallel Computing, Publication Expediting, Inc., 200 Meacham Avenue, Elmont, NY 11003. Air freight and mailing in the USA by Publication Expediting.

Instructions to authors
Contributions should be written in English and include a 50 to 100 word abstract. They will be published as full papers, preferably of not more than ten pages, or as short communications of not more than two pages, and should be sent in triplicate to the appropriate Regional Editor. The author's mailing address should appear on the manuscript. No page charge is made. Please make sure that the paper is submitted in its final form. Corrections in the proof stage, other than printer's errors, should be avoided; costs arising from such extra corrections may be charged to the authors. Upon acceptance of an article, the author(s) will be asked to transfer copyright of the article to the publisher. This transfer will ensure the widest possible dissemination of information. Manuscripts should be prepared for publication in accordance with instructions given in the "Guide for Authors" (available from the Publisher), details of which are condensed below:

1. The manuscript must be typed on one side of the paper in double spacing (including footnotes, references, and abstracts) with wide margins. A duplicate copy should be retained by the author.
2. All mathematical symbols which are not typewritten should be listed separately.
3. Footnotes should be kept to a minimum and as brief as possible, they must be numbered consecutively and typed on a separate sheet in the same format as the main text.
4. Special care should be given to the preparation of the drawings for the figures and diagrams. Except for a reduction in size, they will appear in the final printing in exactly the same form as submitted by the author; normally they will not be redrawn by the printer. In order to make a photographic reproduction possible, all drawings should be on separate sheets, with wide margins, drawn large size, in India ink, and carefully lettered. Exceptions are diagrams only containing formulae and a small number of straight lines (or arrows); these can be typeset.
5. References should be listed alphabetically, in the same way as the following examples:
   For a book:
   [1] S. Ginsburg, *Algebraic and Automata-Theoretic Properties of Formal Languages* (North-Holland, Amsterdam, 1975).
   For a paper in a journal:
   [2] J. Rhodes and B.R. Tilson, Lower-bounds for complexity of finite semigroups, *J. Pure Appl. Algebra* 1 (1971) 79–95.
   For a paper in a contributed volume:
   [3] A.K. Joshi, L.S. Levy and M. Takahashi, A tree generating system, in: M. Nivat, ed., *Automata, Languages and Programming* (North-Holland, Amsterdam, 1973) 454–455.
   For an unpublished paper:
   [4] J. Goldstine, Abstract families of languages generated by bounded languages, Ph.D. Thesis, Univ. California at Berkeley, 1970.

Authors: Please mention your email address and / or fax number on the first page of your manuscript

---

Authors' benefits
1. 50 reprints of each contribution free of charge.
2. 30% discount on all Elsevier Science books.
   (Order forms will be sent together with the proofs.)

Computer Benchmarks

edited by J.J. Dongarra *and* W. Gentzsch

Advances in Parallel Computing Volume 8

The performance of a computer is a complicated issue and a function of many interrelated quantities. These quantities include: the application, the language, the implementation, the compiler, the architecture, and the hardware characteristics. The usual method to evaluate the performance is to compose a benchmark of programs. This book presents a useful overview on benchmarking. Over twenty experts contributed papers on five important topics concerning benchmarking advanced scientific computer systems: taxonomy and performance metrics, well-known standard and application benchmarks, compiler benchmarks for parallel computers, and benchmarks for database systems.

Contents: Preface to the series. Preface. Introduction. **I. Performance Prediction.** Toward a taxonomy of performance metrics (J. Worlton). Toward a better parallel performance metric (X.-H. Sun, J.L. Gustafson). Performance parameters and benchmarking of supercomputers (R.W. Hockney). A framework for benchmark performance analysis.(R.W. Hockney). Performance estimates for supercomputers. The responsibilities of the manufacturer and of the user (W. Schönauer, H. Häfner). A philosophical perspective on performance measurement (D.F. Snelling). **II. Performance Measurement.** A detailed look at some popular benchmarks (R.P. Weicker). Scientific benchmark characterizations (M. Berry, G. Cybenko, J. Larson). The SPEC benchmarks (K.M. Dixit). The benchmark of the EuroBen group (A.J. van der Steen). Performance of various computers using standard sparse linear equation solving techniques (J.J. Dongarra, H.A. van der Vorst). **III. Compiler Benchmarks.** A comparative study of automatic vectorizing compilers (D. Levine, D. Callahan, J. Dongarra). Parallel loops - A test suite for parallizing compilers: Description and example results (J. Dongarra *et al.*). **IV. Benchmarks for Parallel Computers.** NAS parallel benchmark results (D.H. Bailey *et al.*). Parallel performance of applications on supercomputers (C.M. Grassl). The Genesis distributed memory benchmarks (C.A. Addison *et al.*). Performance of the Intel iPSC/860 and Ncube 6400 hypercubes (T.H. Dunigan). CFD applications on transputer systems (M. Faden *et al.*). Benchmarking parallel programs in a multiprogramming environment: the PAR-Bench system (W.E. Nagel, M.A. Linn). **V. Benchmarks for Database Systems.** A hybrid benchmarking model for database machine performance studies (J.A. McCann, D.A. Bell). Database system benchmarking and performance testing (R. Longbottom).

1993 364 pages
Dfl. 220.00 (US 125.75)
ISBN 0-444-81518-X

Elsevier Science Publishers
P.O. Box 103
1000 AC Amsterdam
The Netherlands

P.O. Box 945
Madison Square Station
New York, NY 10160-0757

The Dutch Guilder (Dfl.) price is definitive. US $ prices are subject to exchange rate fluctuations. Customers in the European Community should add the appropriate VAT rate applicable in their country to the price.