CHAPTER 34

Parallel Multigrid Methods on Composite Meshes

Hannes Sbosny*

Abstract. The emphasis of many practical research activities in the area of multigrid methods today lies in constructing highly parallel multigrid algorithms for complex problems with general geometries. Examples are algorithms based on domain decomposition approaches. A straight forward way to parallelize multigrid methods for local memory MIMD machines is based on the method of grid partitioning. This leads to parallel programs with no (or minor) numerical changes in the sequential code. An alternative is using the Schwarz Alternating Procedure as the smoothing process due within the multigrid algorithm. Both approaches will be compared from a practical point of view with particular emphasis on aspects of communication complexity and software development.

1. Introduction. For a wide class of problems in scientific computing the (standard sequential) multigrid principle has been proved to yield highly efficient numerical methods. Practical research activities today lie – among others – in constructing highly parallel multigrid algorithms.

One (natural) parallelization approach for MG methods is governed by the grid partitioning principle, which leads to parallel programs with no (or minor) numerical changes in the sequential code. Whereas all grids in multigrid context have to be treated in a sequential manner, the different multigrid components (relaxation, interpolation, restriction) can be processed in parallel, provided each processor knows certain data from its neighboring processors. In practice, the most efficient implementation is to store appropriate overlapping zones to adjacent processors in its own local memory. Data exchange by message passing is required to update the overlap areas after each computational step, a relaxation sweep for instance.

The development of new parallel multigrid algorithms, which solve the problems with the same accuracy without being numerically equivalent to a standard sequential method, might

*Mathematisches Institut der Heinrich–Heine–Universität Düsseldorf, Universitätsstr. 1, D-4000 Düsseldorf 1, Fed. Rep. of Germany. This work is partially funded by means of German Federal Deptartment of Research and Technology (PARANUSS, grant no. ITR900689).

lead to a more efficient use of distributed memory multiprocessor systems (MIMD). Examples are MG algorithms using domain decomposition ideas. As these algorithms contain less communication without significant degradation of convergence, numerical experiments show higher efficiency.

The earliest domain decomposition method is believed to be the "Schwarz Alternating Procedure (SAP)" [13], which involves decomposition with overlapping regions. For a two–dimensional domain Schwarz used the SAP method to prove the solvability of Dirichlet's problem for Laplace's equation on the union of two overlapping plane regions, provided that it is solvable on each region separately.

An alternating solver is developed by creating composite meshes containing a number of subgrids that cover a subregion and overlap where they meet. Then multigrid methods can be used to solve iteratively on each component. Functions (e.g. local solutions defined on the local grid) are matched by interpolation at the overlapping grid boundaries.

In this way it is easy to parallelize multigrid methods for MIMD machines. First, the problem is decomposed into loosely coupled subproblems mapped onto different processors. Then, a Jacobi–like iteration involves a parallel solution process and the communication needed is limited to the "pseudo"–boundaries of the overlapping zones. Unfortunately, it turns out that the rate of convergence depends on the width of the overlapping zone.

Following earlier ideas of W. Hackbusch [4] and J. Linden [6], the alternative is to use the multigrid iteration, with some steps of Schwarz iteration as the smoothing process. Thus defining the prolongations and restrictions blockwise on each local region we indeed get a parallel multigrid algorithm.

Since the (SAP-) smoothing processes do not only operate on one fine grid but also on coarser grids the overlapping topology is given on all coarser grids. The width of the overlapping zone, however, decreases on coarse grids and leads to unreasonable coarse grid corrections. To overcome this problem we extend the SAP smoothing technique to *moving pseudo–boundaries*: On each MG level the overlap area remains unchanged in terms of the mesh size. This approach turns out to be very efficient.

2. The Schwarz Alternating Procedure. We consider the solution of a second–order uniformly elliptic partial differential equation $Lu = f$ on Ω , a closed, bounded region in $\mathbb{R}^n$ with a Lipschitz continuous boundary $\Gamma := \partial\Omega$. To simplify our presentation we assume that L is the Laplacian and we have homogenious Dirichlet boundary conditions. Thus,

$$-\Delta u = f \quad \text{in} \quad \Omega\,, \qquad u = \varphi \quad \text{on} \quad \Gamma\,. \tag{1}$$

In order to define the SAP algorithm we decompose Ω into some overlapping subdomains. Let $\bar{\Omega} := [0,1] \times [0,1]$. We assume that Ω is partitioned into $\wp \geq 2$ intersecting subdomains, namely

$$\Omega := \bigcup_{k=1}^{\wp} \Omega^k \quad , \quad \Omega^k := \{\, (x,y) \in \Omega \mid x_a^k < x < x_b^k, 0 < y < 1 \,\} \tag{2}$$

with $\Omega^k \cap \Omega^{k+1} \neq \{\}$ for $k = 1,2,\ldots,(\wp-1)$. In particular, we consider $\wp \geq 2$ stripes with $x_a^1 = 0, x_b^\wp = 1$ and $x_a^k < x_a^{k+1} < x_b^k < x_b^{k+1}$, $k = 1,2,\ldots,\wp$.

The width of the overlapping zones is denoted by $d^k > 0$, thus $x_b^k = x_a^{k+1} + d^k$, and the boundaries of Ω^k are defined as $\Gamma^k := \partial\Omega^k \cap \Gamma$, $\gamma_a^k := \{(x_a^k, y) \in \bar{\Omega} \mid 0 < y < 1\}$, $\gamma_b^k := \{(x_b^k, y) \in \bar{\Omega} \mid 0 < y < 1\}$, for $k = 1,2,\ldots,(\wp)$. γ_a^k, γ_b^k are called the pseudo boundaries.

To discretize our model problem we use standard 5-point centered finite difference discretization for Ω and a uniform mesh Ω_h with grid size $h := (h_x, h_y)$ on $\bar{\Omega}$. Similar to (2) we introduce the grid decomposition $\Omega_h = \cup_{k=1}^{\wp} \Omega_h^k$ by

$$\Omega_h^k := \left\{ (ih_x, jh_y) \in \bar{\Omega} \mid h_x = \frac{x_b^k - x_a^k}{N_x}, h_y = \frac{1}{N_y} \right\} , \tag{3}$$

for any $k = 1, 2, \ldots, \wp$. With this notations the SAP iteration consists of solving the following problems:

$$\left.\begin{aligned} -\Delta u_h^{k,(i+1)} &= f \quad in\ \Omega_h^k \\ u_h^{k,(i+1)}|_{\Gamma_h^k} &= 0 \\ u_h^{k,(i+1)}|_{\gamma_{a,h}^k} &= \begin{cases} 0 & \text{if } k = 1 \\ u_h^{k-1,I_l} & \text{otherwise} \end{cases} \\ u_h^{k,(i+1)}|_{\gamma_{b,h}^k} &= \begin{cases} 0 & \text{if } k = \wp \\ u_h^{k+1,I_r} & \text{otherwise} \end{cases} \end{aligned}\right\} \quad k = 1, 2, \ldots, \wp , \tag{4}$$

for $i = 0, 1, 2, \ldots$, with an arbitrary initial approximation u_h^0 and indices $I_l, I_r \in \{i, i+1\}$.

If $I_l = i+1$ and $I_r = i$ we observe a generalization of the classical (multiplicative) Schwarz Alternating Procedure to an arbitrary number of subdomains. Note that there is no parallelism in this algorithm as we must solve for the subdomains sequentially.

A possibility to parallelize the SAP iteration is to use the same strategy as for Gauss–Seidel or SOR methods, namely using red–black ordering of the subdomains: Define $I_l = I_r = i$ if k is odd (red stripes) and $I_l = I_r = i+1$ if k is even (black stripes). This means that a subdomain of one color is related to subdomains of the other color only. Thus, we can compute all subdomains of the same color in parallel (degree of parallelism $\wp/2$).

Choosing $I_l = I_r = i$ we observe a synchronous, fully parallel algorithm, which can easily be adapted to parallel MIMD machines (degree of parallelism $\wp$).

The convergence properties of this method have been studied in many papers, e.g. recently, see P.L. Lions [7] , [8] . Summarizing the results, the Schwarz Alternating Procedure is geometrically convergent as $i \longrightarrow \infty$ and the rate of convergence depends on the width of the overlapping zone. Roughly speaking, the larger the width of the overlapping zone, the faster the convergence. Similar results hold for more general classes of symmetric boundary value problems (higher order problems for instance) or for different boundary conditions on Γ ([7], [8]). On the other hand, the larger the overlap region, the larger is the overhead of supplementary computational work. Furthermore, numerical and theoretical results show that the rate of convergence turns smaller if the number of subdomains is decreased:

Theorem 1 *The convergence factor of the SAP iteration for model problem (1), (2) considering special overlapping geometry, $d^k = d = \frac{D}{2}$, where D denotes the width of $\wp \geq 2$ (equally sized) stripes, then is*

$$\rho(\wp, d) = 2 \frac{\sinh(\pi\, d)}{\sinh(\pi\, D)} \cos(\frac{\pi}{\wp + 1}) \quad .$$

3. Parallel multigrid methods. In this section we shortly summarize the standard multigrid methods for MIMD architectures: On each grid level we perform each of the multigrid component (relaxation, computing of the defect, interpolation, restriction) in parallel. For more details consider [5], [9] or [12] and the literatur therein.

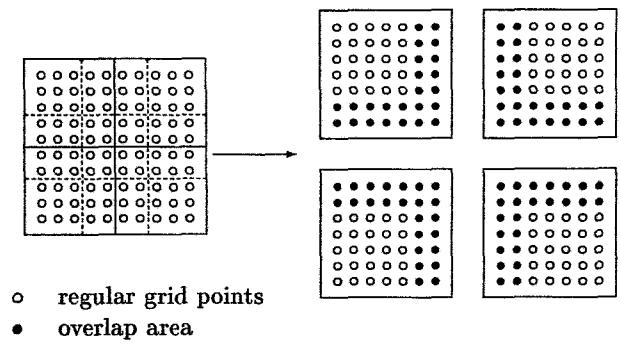

Figure 1: Grid partitioning into overlapping (width=2) boxes

3.1 Notes on grid partitioning. One general parallelization concept for multigrid methods on MIMD architectures is the method of grid partitioning. Since this method is is described in many papers ([14] or [15] for instance) we will only summarize its basic concepts:

- A set of equally sized subgrids is created and then mapped onto the multiprocessor machine in such a way that the distribution of grid points to the processes leads to a balanced load of computation and communication among all nodes.
- All subgrids are processed independently in parallel.
- The numerical algorthim is **NOT** changed, i.e. its sequential and its parallel version are numerically equivalent and give exactly the same results.
- After each computational step the overlapping zones of the subgrids have to be updated.

processors	p=4	p=16
$\|u - u_h\|_\infty$	8.3D-6	8.3D-6
CPU time [sec]	12.6	4.8
mp–speed up $S(N,p) := T(N,1)/T(N,p)$	3.3	8.7
mp–efficiency $E(N,p) := S(N,p)/p$	0.83	0.54

Table 1: Multiprocessor efficiency / speed up of grid partitioning

Let us now consider a multigrid component – a (parallel) relaxation step for instance. On each local subgrid this relaxation step can be carried out in parallel using the grid points of the corresponding process, because of the strictly local dependencies of the operator in each grid point. However, grid points defining an interior (pseudo) boundary need the data from neighboring subgrids. After each relaxation step the values stored in the overlap areas are exchanged and updated via message–passing communication mechanisms. The introduction of overlap areas does not change the numerical algorithm since a strict synchronization following each relaxation step is ensured. Thus, the well known optimal numerical behaviour of standard sequential multigrid methods can be obtained also for parallel algorithms using grid partitioning.

Table 1 presents results (evaluated on Intel's iPSC) based on Dirichlet's problem using a 129×129 global grid, which is decomposed into equally sized squares. Due to an increase in the communication cost / computation cost ratio when going from 4 to 16 processors mp–efficiency decreases. For a more detailed discussion the reader is refered to [14].

3.2 Multigrid using Schwarz iteration as a smoother. In the following subsection, instead of the "naive" combination of Schwarz' method with multigrid techniques, we propose more direct multigrid approaches to the composite mesh systems (4). In this method a multigrid hierarchy of composite meshes is used: The principle of Schwarz' Alternating Procedure is applied exclusively within the relaxation process. Similar smoothing processes are proposed by Linden [6], Hackbusch [4] and Chesshire, Henshaw [2].

We now assume that the composite mesh discretization of our model problem (1), (2) is given. Similarly, we define a sequence of coarser composite meshes and corresponding discretizations on all multigrid levels, e.g. the overlapping topology is given on all coarser grids. We now define the standard multigrid method to be a parallel multigrid algorithm performing each grid level $\ell = 1, \ldots, \ell_{max}$ sequentially, but performing subgrids within each level simultaneously. The usual restriction and interpolation are used. They are however applied blockwise in parallel to all Ω_h^k, $h = h_\ell$ and the following smoothing processes. Clearly, there are different possibilities concerning the order of the subdomains, the order of (local) smoothing and the order of updating the pseudo boundaries.

ALGORITHM ZS : Schwarz–Gauss–Seidel smoother

Let $h = h_\ell$ be the current multigrid level and $u_h^{2k+1,(i)}$ an initial approximation in Ω_h^{2k+1} (including the left and right pseudo boundaries $\gamma_{a,h}^{2k+1}$ and $\gamma_{b,h}^{2k+1}$, respectively).

STEP 1 Apply $\nu \geq 0$ (Gauss–Seidel) relaxation steps to $u_h^{2k+1,(i)}$ in Ω_h^{2k+1}, $1 \leq 2k+1 \leq \wp$, e.g. define

$$S_h^\nu(u_h^{2k+1,(i)}) := u_h^{2k+1,(i+1)} .$$

STEP 2 Update all odd (black) pseudo boundaries, e.g. let

$$u_h^{2k,(i)} := \Pi_{a(b)}^{2k} u_h^{2k+1,(i+1)} \qquad \text{on} \quad \gamma_a^{2k} \; (\gamma_b^{2k}) \; ,$$

where $\Pi_{a(b)}^{2k}$ denotes the interpolation of the pseudo boundaries.

STEP 3 Apply $\nu \geq 0$ (Gauss–Seidel) relaxation steps to $u_h^{2k,(i)}$ in Ω_h^{2k}, $1 \leq 2k \leq \wp$, e.g. define

$$S_h^\nu(u_h^{2k,(i)}) := u_h^{2k,(i+1)} .$$

STEP 4 Update all even (red) pseudo boundaries, e.g. let

$$u_h^{2k+1,(i)} := \Pi_{a(b)}^{2k+1} u_h^{2k,(i+1)} \qquad \text{on} \quad \gamma_a^{2k}(\gamma_b^{2k}) \; .$$

The smoothing rate of SAP is (for local analysis) the reduction of high frequency errors along the pseudo boundaries. If multigrid is applied with the smoothing described above it is essential that not d but $ovl := \frac{d}{h}$ has a not too small minimum on all grids (cf. [11], [12]). Therefore it is better to use subdomains Ω_h^i with fixed overlap ovl for each h than h–discretizations of fixed subdomains Ω^i. This method is more similar to grid partitioning. We call it moving pseudo boundary method.

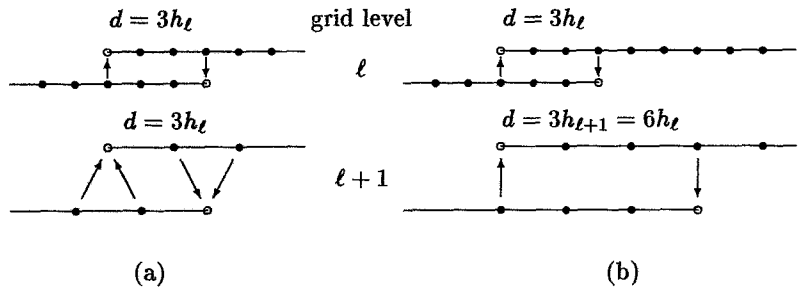

Figure 2: Interpolation of (a) fixed and (b) moving pseudo boundaries

3.2.1 Moving pseudo boundary technique. The smoothing properties for standard applications of the Schwarz–Gauss–Seidel relaxation may become worse on coarser multigrid levels. This is due to the fact that the amount of overlap *ovl* decreases on coarser grids. Additionally, computional work slighly increases for fixed pseudo boundaries: Higher order interpolation is essential for preserving good approximations to the solution restricted to the pseudo boundaries.

To overcome these problems we extend the SAP smoothing technique to so–called moving pseudo boundaries (cf. figure 2): On each multigrid level the overlap area remains unchanged in terms of the mesh size (fixed overlap *ovl* for all $h_\ell, \ell = 1, 2, \ldots$) .

3.3 Numerical results. For illustration we present results for the parallel multigrid method using Schwarz–Gauss–Seidel relaxation (algorithm ZS). First we show the convergence properties of this algorithm. Again the model problem used is equation (1). We apply standard coarsening (for all grids) and coarse–grid operators using the same composite mesh discretization as on the fine grids. The transfers are done by full weigthing (restriction) and bilinear interpolation (prolongation), respectively. The results given below are based on W–cycling and zebra–stripe (ZS) odering of the subdomains:

- ZS1: Update of the pseudo boundaries after every single Gauss–Seidel step. In particular, W(ν_1, ν_2)–cycling leads to ν_1 SAP–Gauss–Seidel (SAP–GS) pre– and ν_2 SAP–GS post–smoothing steps, respectively. Only one Gauss–Seidel smoothing step is performed on each local grid.

- ZS2: Update of the pseudo boundaries after $\nu_1, \nu_2 \geq 0$ Gauss–Seidel steps. In particular, W(ν_1, ν_2)–cycling leads to just one SAP–GS step before (after) coarse grid correction using ν_1 (ν_2) Gauss–Seidel steps on each local subgrid.

The exchange (update) of the pseudo boundaries is done by single injection and (on coarser grids) by cubic interpolation. This turns out to be important: Using linear interpolation for instance, results in much slower multigrid convergence.

In table 2 and 3 we compare the convergence of this algorithms for fixed overlap areas d on all grid levels. The given results refer to different values for d and to a global mesh size of 81×81 grid points.

For very small values of d ($\leq 6h_0$) we illustrate the influence of the decreasing overlap *ovl* on coarser grids as was described above. Comparing version ZS1 with ZS2, version ZS1 differs significantly for small d. The approach of just one Gauss–Seidel smoothing step to solve the local subproblems in the SAP iteration leads to slower error reduction

	W(2,1)–cycling			W(3,0)–cycling		
	$d = 32h_0$	$d = 8h_0$	$d = 4h_0$	$d = 32h_0$	$d = 8h_0$	$d = 4h_0$
ZS1	0.0533	0.0586	0.2753	0.0544	0.0626	0.3955
ZS2	0.0523	0.0529	0.0681	0.0522	0.0530	0.1587

Table 2: Numerical multigrid convergence (2 stripes)

	W(2,1)–cycling	
	$d = 0.1 = 8h_0$	$d = 0.05 = 4h_0$
ZS1	0.0559	0.3081
ZS2	0.0531	0.0524

Table 3: Numerical multigrid convergence (8 stripes)

for high frequency components. Similar results hold for W(ν_1,0)–cycling (which leads to less communications; *cf.* table 4) as shown in table 2 and 3 (the latter for more than two subdomains). In table 4 results are shown for algorithm ZS2 using the moving pseudo boundary method. The results given refer to a global grid containing 65 points in X– and Y–direction, respectively.

ovl	ρ	
	2 stripes	4 squares
4	0.121	0.153
6	0.034	0.038
8	0.032	0.031

Table 4: Numerical multigrid convergence: Moving pseudo boundaries, W(2,1)–cycling

Method	# SEND/RECEIVE	% S/R
MG2	$2 \times (2^{L-1} - 1) \times (\nu_1 + \nu_2 + 1) + 2^{L-1}$	100
ZS1	$2 \times (2^{L-1} - 1) \times (\nu_1 + \nu_2) + 2^{L-1}$	~ 75
ZS2 $\nu_2 > 0$	$2 \times (2^{L-1} - 1) \times 2$	~ 53
ZS2 $\nu_2 = 0$	$2 \times (2^{L-1} - 1) \times 1$	~ 30

Table 5: Number of messages per multigrid W–cycle

Communication on distributed memory machines: Typically, certain highly parallel computers with a large number of processors (100–1000 or more) do not necessarily provide shared memory for all these processors. Thus, for MIMD machines with local memory, a concept has turned out to be suitable that is based on message–passing communication. Here start up time for the initialization of communication is needed as well as transfer time for each item of the message, e.g.

$$T_{message} = T_{startup} + T_{length} \cdot$$

It turns out, that for message lengths obtained in the above problem class the critical term is start up time. Thus, we have to construct parallel algorithms which minimize the messages to be sent and received, respectively. Table 5 shows the number of messages of the

grid partitioning method MG2 (see section 3.1) and the parallel multigrid using Schwarz–Gauss–Seidel relaxation (algorithm ZS). We obtain that algorithm ZS1 needs only about 75 % of the communications used by MG2. In contrast to MG2 and ZS1, the total number of messages exchanged by ZS2 is independent of the number of relaxation steps ν_1 and ν_2, respectively. Hence, this algorithm needs only 53 % of the communications of MG2 for $\nu_1, \nu_2 > 0$ and 30 % for $W(\nu_1, 0)$–cycling.

4. Conclusions. We analyzed and implemented parallel multigrid methods using domain decomposition based on the "Schwarz Alternating Procedure". It turns out that the smoothing rate of SAP depends on the width of the overlapping zone. For fixed pseudo boundaries this overlap may be decreased on coarser grids leading to unreasonable coarse–grid corrections. To overcome this problem we introduced so–called moving pseudo boundaries. This results in algorithms which contain much less communications without significant loss of convergence–speed.

5. Acknowledgements. The author would like to thank Prof. Dr. K. Witsch (University of Düsseldorf, FRG) for many helpful discussions. I also thank K. Ressel for his support.

References

[1] **T.F. Chan, R. Glowinski, J. Périaux, O.B. Widlund (ed.) :** *"Domain Decomposition Methods"*, SIAM, Philadelphia, 1989

[2] **G. Chesshire, W.D. Henshaw :** *"Composite overlapping meshes for the solution of partial differential equations"*, Journal of Computational Physics 90 (1990), p. 1–64

[3] **R. Glowinski, G.H. Golub, G.A. Meurant, J. Périaux (ed.) :** *"Domain Decomposition Methods"*, SIAM, Philadelphia, 1988

[4] **W. Hackbusch :** *"Multi–Grid Methods and Applications"*, Springer, Berlin, 1985

[5] **M. Lemke :** *"Erfahrungen mit Mehrgitterverfahren für Helmholtz–ähnliche Probleme auf Vektorrechnern und Mehrprozessor–Vektorrechnern"*, Arbeitspapiere der GMD 278, St. Augustin, 1987

[6] **J. Linden :** *"Mehrgitterverfahren für die Poisson-Gleichung im Kreis und Ringgebiet unter Verwendung lokaler Koordinaten"*, Diplomarbeit, Universität Bonn, 1981

[7] **P.L Lions :** *"On the Schwarz alternating method. I* in [3], p. 1

[8] **P.L Lions :** *"On the Schwarz alternating method. II* in [1], p. 47

[9] **O.A. McBryan, U. Trottenberg :** *"Multigrid methods on parallel computers"*, Arbeitspapiere der GMD 370, St. Augustin 1989

[10] **K. Ressel :** private communications

[11] **H. Sbosny :** *"Domain decomposition methods and parallel multigrid Algorithms"*, To appear in: W. Hackbusch, U. Trottenberg (ed.): Proceedings of the Third European Conference in Multigrid Methods, Bonn, October 1 to 4, 1990

[12] **H. Sbosny, K. Witsch :** *"Parallel multigrid using domain decomposition"*, Notes on Numerical Fluid Mechanics, Vol. 31 (1991), p. 200–215

[13] **H.A. Schwarz :** *"Gesammelte Mathematische Abhandlungen"*, Springer, Berlin, Vol. 2 (1890), p. 133–134

[14] **K. Solchenbach, C.A. Thole, U. Trottenberg :** *"Parallel multigrid methods: Implementation on SUPRENUM–like architectures and applications"*, SUPRENUM Report 4, 1987

[15] **K. Solchenbach, U. Trottenberg :** *"SUPRENUM–system essentials and grid applications: Implementation and evaluation"*, SUPRENUM Report 13, 1988