Thirteenth International Conference on Domain Decomposition Methods
Editors: N. Debit, M.Garbey, R. Hoppe, J. Périaux, D. Keyes, Y. Kuznetsov

32 On the use of iterative Schwarz algorithms in the solution of an optimal control problem

A. Bounaïm[1]

Introduction

We present two methods for solving an optimal control problem governed by a partial differential equation. Our methods combine optimal control techniques and Schwarz algorithms using an overlapping domain decomposition at each step of the minimization process. We design parallel algorithms based on the iterative Schwarz methods used either as solver or as preconditioner. Numerical results are presented to show the behavior of the optimization solver with respect to some parameters related to domain decomposition.

As a model problem, we consider a boundary control problem of which the state variable is the solution of an elliptic partial differential equation:

$$\left\{\begin{array}{lcll} -\Delta y(v) & = & f & \text{in} \quad \Omega, \\ y(v) & = & 0 & \text{on} \quad \Gamma_N \cup \Gamma_S, \\ \dfrac{\partial y(v)}{\partial n} & = & v & \text{on} \quad \Gamma_E \cup \Gamma_W. \end{array}\right. \tag{1}$$

The control v is taken on the east and west boundaries of a rectangular 2D domain Ω whereas the observations y_d are distributed over the whole domain Ω. The solution of such a problem involves the techniques of a cost function J that minimizes, in a least-square formulation, the quadratic distance between the solution of the state equation (1) and given observations:

$$J(v) = \frac{1}{2}\Big(\int_\Omega |y(v) - y_d|^2 dx + \nu \int_{\Gamma_E \cup \Gamma_W} |v|^2 d\sigma\Big).$$

And we set the optimal control problem as:

$$(\mathcal{P}) \qquad \inf_{v \in \mathcal{U}_{ad}} J(v) = J(u), \qquad u \in \mathcal{U}_{ad},$$

where $\mathcal{U}_{ad}$ is a set of admissible controls. The solution of $(\mathcal{P})$ is commonly based on descent methods [Lio68]: At the nth iteration, from the known u^n, we compute successively the direct state $y(u^n)$ and the adjoint state $p(u^n)$. We then get the value of $J(u^n)$ and the gradient $\nabla J(u^n)$ which is an expression of $p(u^n)$ and u^n (See [Lio68]). A minimization step is shown in Figure 1.

Discretization and numerical framework

The domain Ω is meshed by a uniform grid $\Delta x = \Delta y = h = \frac{1}{N+1}$ (N is the number of points in the y-direction). A finite difference scheme is used to discretize the direct state $y(u_n)$. The discrete adjoint state $p(u_n)$ is then deduced from the transpose system of $y(u_n)$ with the appropriate right hand side.

[1]Department of Informatics, University of Oslo, Norway, aichab@ifi.uio.no

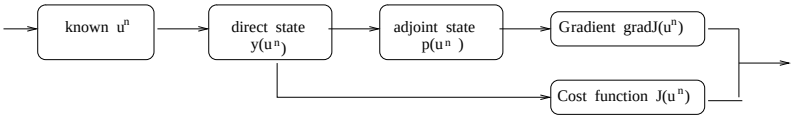

Figure 1: One minimization step: calculation of the cost function and its gradient.

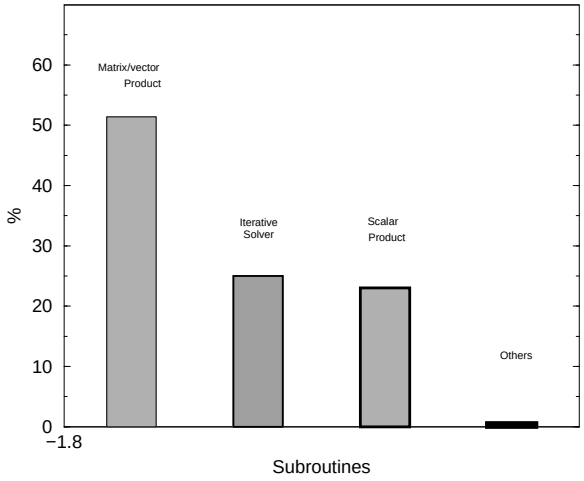

Figure 2: Profiling of the sequential code on the whole domain, $N = 64$.

The resulting linear systems are solved by a Krylov solver: `Bicgstab` (Stabilized bi-conjugate gradient). The minimization phase is carried out by the quasi-Newton method with the BFGS formula [2][GL89].
For the numerical tests, we have the following:
$\Omega =]0, 4[\times]0, 1[$ is the domain of computation, and for (x, y) in Ω:

$$\begin{aligned} f(x,y) &= 2(-x^2 - y^2 + 4x + y), \\ y_d(x,y) &= (x^2 - 4x)(y - y^2) - 8\nu. \end{aligned}$$

Remark. The computation of the discrete gradient of the cost function is the main step of the minimization process, since the precision of the descent method depends on the precision of the discrete gradient calculation.

Motivation

When we solve sequentially the optimal control problem $(\mathcal{P})$ on the whole domain, we find that most of the CPU time required for the minimization process is related to the scalar and

[2]The M1QN3 code is developped in the MODULOPT project of INRIA by J.-C. Gilbert and C. Lemaréchal. We have used its double precision version: N1QN3.

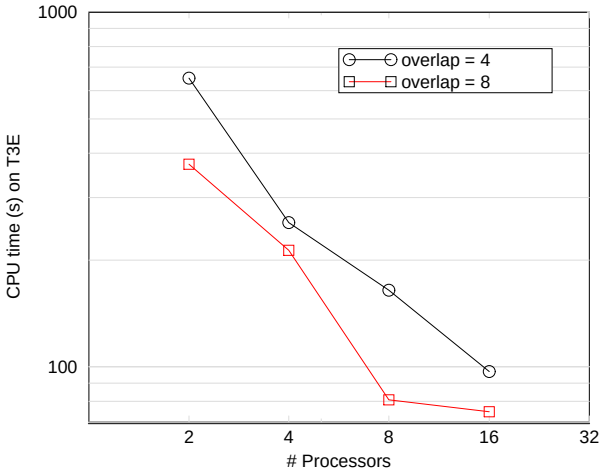

Figure 3: CPU time versus the number of processors. Effect of the overlap size on the behavior of N1QN3 with multiplicative Schwarz method. $N = 64$.

matrix-vector products that are the base in the calculation of the cost function and its gradient. Figure 2 shows the time percentage of each code part. We need 10 iterations to achieve the given precision $\epsilon = \frac{\|\nabla J(u^n)\|}{\|\nabla J(u^0)\|} = 10^{-6}$ in N1QN3.
So, we propose to implement efficient algorithms for parallel architectures using a load allocation of the solvers of both the discrete direct and adjoint states.

Domain decomposition techniques

The main idea of the proposed domain decomposition method consists in using iterative Schwarz methods either as solver or preconditioner for the direct and adjoint linear systems required at each step of the minimization algorithm. In contarst, the minimization instead remains global over the domain of calculation, i.e., the control in N1QN3 is not decomposed. All the results are given for the parallel machine CRAY-T3E using the message passage interface library MPI.

The Schwarz algorithm as a solver

Description

We consider an overlapping decomposition of the domain Ω and using the multiplicative version of the Schwarz algorithm with Dirichlet boundary conditions to solve the direct state so that we get on each subdomain Ω_i:

$$A_i y_i^{n+1} = F_i(f_i, u_i^{n+1}, y_j^n|_{\partial\Omega_i \cap \Omega_j}). \tag{2}$$

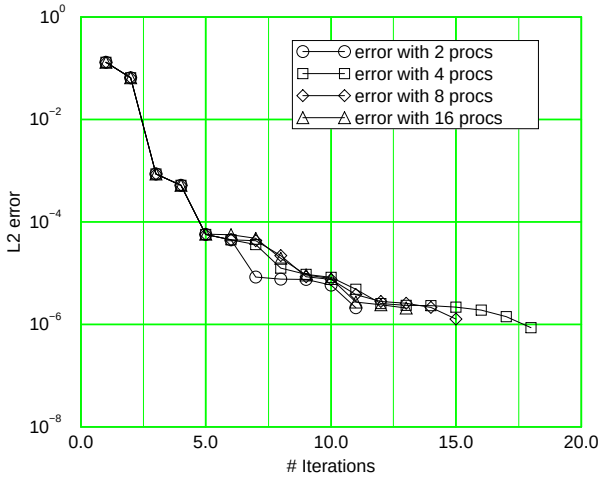

Figure 4: L^2 error, $N = 64$, $\delta_i = 4$. Behavior of N1QN3 with multiplicative Schwarz method as solver.

The discrete adjoint state is then computed by transposing the y_i^{n+1}-local system (2) such that we get formally the p_i^{n+1} system:

$$A_i^t p_i^{n+1} = G_i(y_i^{n+1} - y_{d,i}, p_j^n|_{\partial\Omega_i \cap \Omega_j}) \tag{3}$$

Analysis of numerical results

The tests were carried out on a mesh of $4N \times N$ nodes where $N = 64$ and for a stopping criterion $\epsilon = \frac{\|\nabla J(u^n)\|}{\|\nabla J(u^0)\|} = 10^{-6}$ in the minimization method N1QN3. The local linear systems are solved by the `Bicgstab` method.
We study the behavior of the N1QN3 "minimizer" with respect to different parameters such as the overlap size, the type of the decomposition and the number of processors. Furthermore, to make the implementation possible on the parallel machine, we have used a coloring technique such that neighbouring subdomains have different colors.

¿From Figure 3, it is shown that the CPU time drops when the overlap gets large (in fact, we consider in the figures the relative overlap δ_i which is linked to the real overlap δ between two subdomains by $\delta = 2\delta_i h$). This reflects one of the properties of the multiplicative Schwarz method [SBG96].

In addition, to show the effect of the multiplicative Schwarz method mixed with the N1QN3 otpimizer, we present in Figure 4 the L^2-error between the computed solution (the direct state associated with the computed optimal control) and the analytical solution against the number of iterations in N1QN3. For different numbers of processors and with a relative overlap equal to 4, it is shown that it is only from the 5th iteration that the precision deteriorates.
The number of iterations in the optimiser also varies slightly (in fact, N1QN3 needs only 10 iterations for the whole domain). Thus, the best result is obtained with 2 processors but for

more processors the precision is almost lost. One can conclude that this is due to the "oscillations" of the precision quantity of N1QN3 within the communications between subdomains.

The Schwarz algorithm as a preconditioner

Let $\mathbf{x}_0$, ϵ, ϵ_{stop} be given
refresh_news($\mathbf{x}_0$)
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{v} = \mathbf{p} = 0$, $\rho_0 = \alpha = \omega = 1$
While $\dfrac{\text{Global\_}\|\mathbf{r}_k\|_2}{\text{Global\_}\|\mathbf{r}_0\|_2} > \epsilon$ and Global_ $\|\mathbf{r}_k\|_2 > \epsilon_{stop}$ Do
$\rho = \text{Global\_}({\mathbf{r}_0}^T\mathbf{r})$
$\beta = \alpha\rho/\rho_0\omega$, $\rho_0 = \rho$
$\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega\mathbf{v})$
Solve $\mathbf{C}\widehat{\mathbf{p}} = \mathbf{p}$
refresh_news($\widehat{\mathbf{p}}$)
$\mathbf{v} = \mathbf{A}\widehat{\mathbf{p}}$
$\alpha = \rho_1/\text{Global\_}({\mathbf{r}_0}^T\mathbf{v})$
$\mathbf{s} = \mathbf{r} - \alpha\mathbf{v}$
Solve $\mathbf{C}\mathbf{z} = \mathbf{s}$
refresh_news($\mathbf{z}$)
$\mathbf{t} = \mathbf{A}\mathbf{z}$
$\omega = \dfrac{\text{Global\_}(\mathbf{t}^T\mathbf{s})}{\text{Global\_}(\mathbf{t}^T\mathbf{t})}$
$\mathbf{x} = \mathbf{x} + \alpha\widehat{\mathbf{p}} + \omega\mathbf{z}$
$\mathbf{r} = \mathbf{s} - \omega\mathbf{t}$
Endwhile

Figure 5: Partitioned Bicgstab algorithm

Description

It is well known that when used as a preconditioner of a parallel Krylov solver, the overlapping domain decomposition methods allow us to improve the convergence rate of such iterative linear system solvers and to limit the time of communications needed for their implementation on parallel architectures. In the preconditioning step of the distributed `Bicgstab` (Figure 5), we first extend the local contribution of `s` or `p` to the subdomain enlarged by the overlap in the four cardinal directions. Then, on each subdomain, we solve exactly the local problem with Dirichlet boundary condiditons. And finally, the global solution $\hat{\mathbf{p}}$ or `z` is deduced from the projections of the solution of each local problem. We have used the same notations as in [KA98] (see also [KST95]).

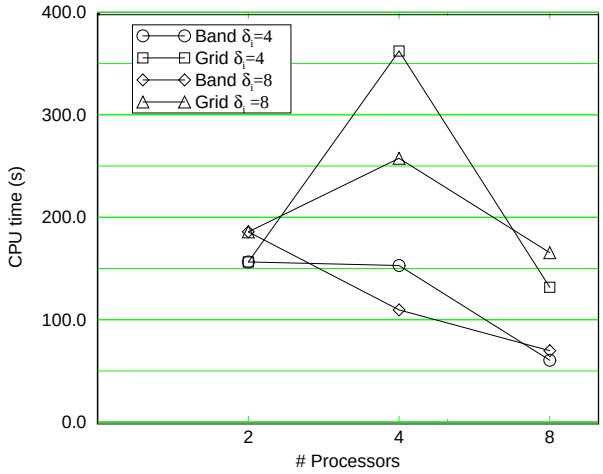

Figure 6: CPU time(s) on Cray-T3E versus the number of processors. $N = 64$. Effect of the decomposition type on the CPU time of N1QN3 algorithm used with preconditioned `Bicgstab`.

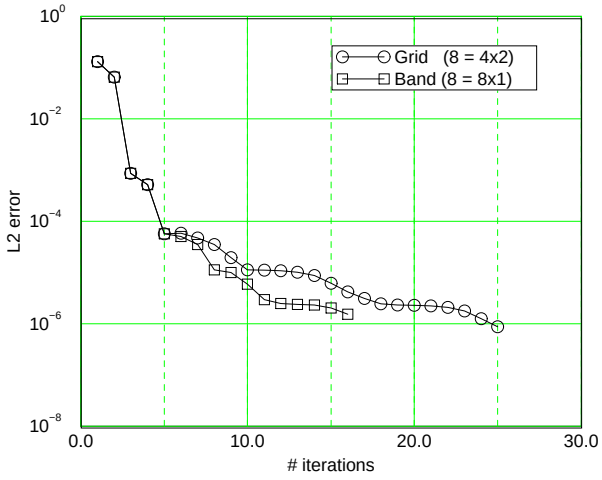

Figure 7: L^2 error versus the iteration number of N1QN3. $N = 64$, $\delta_i = 4$. Effect of the decomposition type on the behavior of N1QN3 with preconditioned `Bicgstab`.

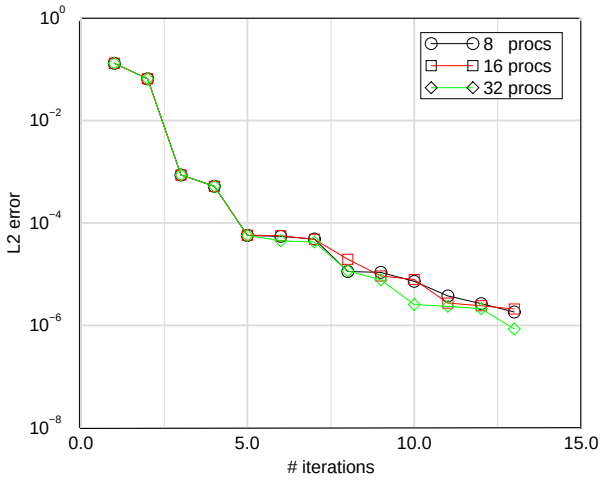

Figure 8: L^2 error versus the iterations number of N1QN3. $N = 64$, $\delta_i = 4$. Behavior of N1QN3 with `Bicgstab` preconditioned versus the number of processors.

Analysis of numerical results

We first remark that the cpu times are better than those obtained with the multiplicative Schwarz method as a solver. On Figure 6, we observe that for $\delta_i = 8$, the cpu time is halved when the number of processors goes from 4 to 8. From Figure 7, we observe that the cpu times are small when $\delta_i = 4$ and the computations are done on 16 processors. We observe in the same figure the important effect of the decomposition type on the behavior of the optimiser N1QN3: with 8 band-disposed processors, we need more iterations than the grid disposition of the processors (in this case, we have 4 processors in x-direction and 2 in y-direction) to reach the given precision in the optimiser. Obviously, in the case of a small size of the problem, a grid decomposition involves more communication than a band one.
From this test series, the relative overlap $\delta_i = 4$ should be an optimal one for the preconditioned distributed `Bicgstab` since the behavior of N1QN3 is not affected by the number of processors (Figure 8).

Conclusion

The methods presented mix minimization algorithms and iterative Schwarz methods (solver or preconditioner). In both cases, the optimal control is computed for a given stopping criterion and the influence of the decomposition parameters on the behavior of the minimization method is shown.
The multiplicative Schwarz method used in the solution of an optimal control problem yields a robust but time consuming method, whereas the additive method used as a preconditioner at each step of the minimization process is less time consuming. The best results are obtainded for a relative overlapping of 25%. Moreover, we have compared this method with the direct

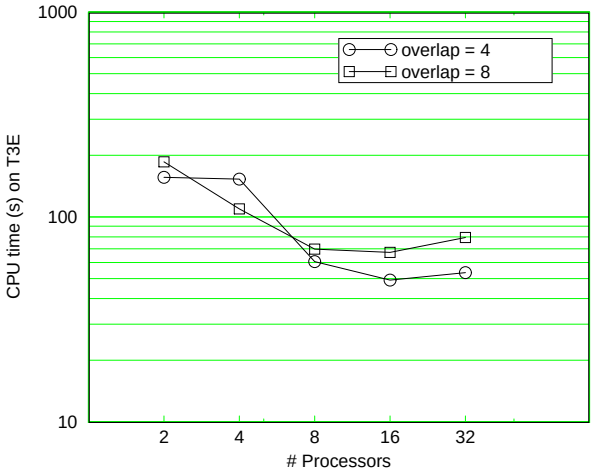

Figure 9: CPU time (s) on Cray-T3E versus the number of processors, $N = 64$. Effect of the overlapping on the behavior of N1QN3 with `Bicgstab` preconditioned by the additive Schwarz method.

parallelization of `Bicgstab` [Bou99] and it is expected (from the curved look of the Figure 9) to be more competitive for a large number of degrees of freedom since the minimizer is only slightly affected by the second method.

References

[Bou99]Aïcha Bounaim. *Domain Decomposition Methods: Application to the Solution of Optimal Control Problems*. PhD thesis, Joseph Fourier University, Grenoble, France, June 1999. (in french).

[GL89]Jean-Charles Gilbert and Claude Lemaréchal. Some numerical experiments with variable storage quasi-Newton algorithms. *Mathematical Programming*, 45:407–435, 1989.

[KA98]Samuel Kortas and Philippe Angot. Parallel preconditioners for a fourth-order discretization of the viscous Bürgers equation. In Petter E. Bjørstad, Magne Espedal, and David Keyes, editors, *Proceedings of the 9th international conference on domain decomposition method*, pages 387–405, 1998.

[KST95]David E. Keyes, Youcef Saad, and Donald G. Truhlar, editors. *Domain-Based Parallelism and Problem Decomposition Methods in Computational Sciences and Engineering*. SIAM, 1995.

[Lio68]J.L. Lions. *Contrôle optimal des systèmes gouvernés par des équations aux dérivées partielles*. Paris. Dunod, Gauthier Villars, 1968.

[SBG96]Barry F. Smith, Petter E. Bjørstad, and William Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.