UPC CTTC

Parallel multigrid algorithms for computational fluid dynamics and heat transfer

Centre Tecnològic de Transferència de Calor
Laboratori de Termotècnia i Energètica
Departament de Màquines i Motors Tèrmics
Universitat Politècnica de Catalunya

Manel Soria Guerrero
Doctoral Thesis

Parallel multigrid algorithms for computational fluid dynamics and heat transfer

Manel Soria Guerrero

TESI DOCTORAL

presentada al

Departament de Màquines i Motors Tèrmics
E.T.S.E.I.T
Universitat Politècnica de Catalunya

per a l'obtenció del grau de

Doctor Enginyer Industrial

Terrassa, Estiu 2000

Parallel multigrid algorithms for computational fluid dynamics and heat transfer

Manel Soria Guerrero

Directors de la Tesi

Prof. Dr. Assensi Oliva Llena
Prof. Dr. Carlos-David Pérez-Segarra

Tribunal qualificador:

Prof. Dr. Eduard Egusquiza Estévez
(Universitat Politècnica de Catalunya)

Prof. Dr. Miquel Costa Pérez
(Universitat Politècnica de Catalunya)

Prof. Dr. Antonio Lecuona Neumann
(Universidad Carlos III de Madrid)

Prof. Dr. Francisco Tirado Fernández
(Universidad Complutense de Madrid)

Prof. Dr. Jesús Labarta Mancho
(Universitat Politècnica de Catalunya)

Acknowledgements

First I must say that I consider myself very lucky to have had enough time and freedom to write a dissertation about subjects that are of my personal interest. I sincerely have to thank many people for their help:

- I would like to thank all the CTTC people for their support during all these years. It would be virtually impossible to recall all the occasions where the help of one or another has been important for me or my work. Among them:
  - As a code without users would be useless, maybe I should first remember Jaume Salom and Octavio García, the first users of the ACM solvers. And then Carles Oliet, Marcos Quispe and Jesus Castro who started BCM with me. Then it grew and became DPC, mainly due to the work of Jordi Cadafalch, Ricard Consul and Kilian Claramunt.
  - Conxita Lifante who has helped me many times to clarify math concepts, and also with LaTeX "details".
  - Joshua Mora for his collaboration with the MSIP and parallel Krylov algorithms.
  - Debora Faggembauu who continued my work with AGLA and the ventilated facades, allowing me to go on with the parallel solvers.
  - Ramiro Alba who among other things, helped me with UP-UX, then Linux and the JFF cluster and, more important, taught me the importance of being patient (or tried to do so).
  - Carmen López for her friendship and help with the English (although she is not to blame for the mistakes that you might find in the text).
  - Joaquim Rigola who has always been a helpful person.
  - I should also thank Miquel Costa for so many years of collaboration and friendship.
  - In the acknowledgements of the CTTC dissertations, it is traditional to thank Manolo Ordoño for his help with the experimental setups. This is a problem, as I have not been involved with any experiment. However, please let me use this occasion to (abusing the concept of experimental setup) thank him for his help with the water heater in my flat.
  - Carlos David Perez-Segarra, who was my heat transfer professor, then tutor of the Ph.D work and has helped me so much with the correction of the dissertation.
  - Finally, I must thank Assensi Oliva for giving me the oportunity to work in numerical heat transfer and directing my Ph.D work.
- From my stay during spring 1997, at Daresbury Laboratory, I would like to thank David Emerson for his support and Kevin Maguire who helped me with MPI, Emacs and many other things, such as the simulations of section 4.7. Additionally, he went to my wedding dressed in a traditional Scottish kilt that is still remembered in every family meeting.

- From my stay in the summer of 1999 at EPCC in Edinburgh, I would like to thank Erwin Simons from whom I learnt the Schur complement method, and that arriving 5 minutes before departure time is enough to catch a plane. The first technique has been very useful but I think I'll never dare to use the second.

  I would also like to remember Casiano Rodriguez who had the patience to listen to my expositions of the DDV algorithm, and dared to eat my remarkable "vampire chicken" (usually without complaining too much).

  And, of course, all the EPCC staff, specially Mario Antonioletti, my contact Connor Mulholland and Catherine Inglis.

- Finally, I would like to thank all my family for their support. Specially my parents and grandparents, my sister who has always helped me in the difficult moments and my wife (and also my friend) Núria, because without her help I would never have finished my career.

Again, to all of them and others that I might have forgotten, thanks a lot.

Contents

1 Introduction **23**
1.1 Overview 23
1.2 Problem model 25
1.3 Laminar and turbulent flows. Turbulence modeling. 29
1.3.1 Turbulence 29
1.3.2 Turbulence modeling 30
1.3.3 Laminar and turbulent flows from a computational point of view 31
1.4 Additional heat transfer phenomena missing in the problem model 32
1.5 Nomenclature 34

2 Numerical solution of fluid flow equations **35**
2.1 Overview 35
2.2 Convection-Diffusion equation 35
2.3 Spatial discretization. Finite control volume method. 37
2.4 Temporal discretization: time marching 43
2.4.1 Difference between spatial and temporal coordinates 43
2.4.2 Explicit versus implicit discretization. Transition matrices. 44
2.4.3 Pseudo-Transient method 47
2.5 Linearization 48
2.6 Coupling 50
2.6.1 Coupled methods 50
2.6.2 Segregated methods 51
2.7 Solution of linear algebraic equations 51
2.7.1 Linear system used to benchmark the solution algorithms 52
2.7.2 LU decomposition 53
2.7.3 Classic iterative methods 54
2.8 Nomenclature 58

3 Sequential Additive Correction Multigrid **59**
3.1 Multigrid Methods for CFD applications 59
3.1.1 Introduction 59
3.1.2 Overview 59
3.1.3 A classification of MG methods for CFD applications 62
3.1.4 Examples of MG applied to CFD 66
3.2 Additive Correction Multigrid 66
3.2.1 Additive Correction Equations 67
3.2.2 ACM Algorithm 71
3.3 Benchmarking ACM 74
3.3.1 Effect of the parameters on the convergence process 74
3.3.2 CPU time to solve the problem model 79
3.3.3 ACM as a solver for the pressure-correction equation of a time-accurate CFD problem 82
3.4 Coupled Multigrid Solvers 84

3.4.1 CACM benchmark 85
3.5 Nomenclature 91

4 Parallel Computational Fluid Dynamics 93
4.1 Why Parallel Computational Fluid Dynamics ? 93
4.2 Overview of parallel computing technology 95
4.2.1 Hardware 95
4.2.2 Software. Parallel Programming Models. 98
4.3 Parallelism and scalability 99
4.3.1 Dependencies 100
4.3.2 Iterative algorithms dependent on the number of processors 100
4.3.3 Performance measures of a parallel iterative computation 100
4.3.4 Scalability 102
4.4 Control-volume based PCFD 103
4.4.1 Overview 103
4.4.2 Spatial domain decomposition for the solution of elliptic PDEs 104
4.5 Parallel algorithms for the solution of linear equations 105
4.5.1 Block-Jacobi algorithm 105
4.5.2 Krylov subspace algorithms 107
4.6 Software engineering aspects of PCFD 107
4.7 Implementation of a Schwartz Alternating Method to solve reactive flows 109
4.7.1 Physical model 109
4.7.2 Numerical aspects 110
4.7.3 Illustrative results 110
4.7.4 Parallel performance 111
4.8 Nomenclature (4.1-4.6) 115
4.9 Nomenclature (4.7) 115

5 Parallel geometric multigrid 117
5.1 Introduction 117
5.1.1 General aspects of parallel MG algorithms and DDV 117
5.2 Notation 118
5.2.1 Genuine values and halo updates 119
5.3 Parallel smoothing 122
5.4 Parallel V-Cycle. Standard approach 124
5.4.1 Algorithm overview. FAS equations 124
5.4.2 Inter-grid transfer operators 125
5.4.3 Non-recursive formulation of the algorithm 125
5.4.4 Direct solver for level 1 126
5.5 Parallel V-Cycle: DDV approach 127
5.5.1 Data dependencies 127
5.5.2 Avoiding data transfer in the second leg 128
5.5.3 Avoiding data transfer in the first leg 129
5.5.4 Benchmark 132
5.5.5 Numerical efficiency 133
5.5.6 Computing times 134
5.5.7 Breakdown of computing times 135
5.6 Final remarks 136
5.7 Nomenclature 139

6 Schur complement method **141**
6.1 Introduction 141
6.2 Algorithm 141
6.2.1 Reordering 141
6.2.2 Solution of the reordered system 144
6.2.3 Evaluation of $\tilde{A}_{s,s}$ and $\tilde{b}_s$ 144
6.2.4 Implementation. First approach 145
6.2.5 Implementation. Distributed evaluation and storage of $\tilde{A}_{s,s}^{-1}$ 145
6.2.6 Sparse matrix storage 147
6.3 Benchmark 148
6.4 Final remarks 150
6.5 Nomenclature 151

7 Domain Decomposed Additive Correction Multigrid **153**
7.1 Introduction 153
7.2 Domain partition and halos 154
7.2.1 Halo update 155
7.2.2 Benchmarking halo update in JFF cluster 157
7.3 Block Incomplete Lower-Upper Smoothing 160
7.4 Block partitions and MG equations 162
7.5 Algorithm 163
7.5.1 Pre-processing 163
7.5.2 Solution 164
7.6 Benchmarking DDACM 166
7.7 Final remarks 168
7.8 Nomenclature 171

8 Conclusions **173**

A Lewis Fry Richardson **175**

B JFF Cluster at CTTC **177**

C Acronyms **179**

D Main publications done in the context of this work **181**

List of Figures

2.1 Fragment of a Cartesian structured two-dimensional mesh. 38
2.2 Central row coefficients of transition matrix P for different Fo numbers. 47
2.3 CPU time to solve a LU system. 54
2.4 CPU time to evaluate the LU decomposition. 54
2.5 Convergence of Line-by-line Gauss Seidel and MSIP algorithms. 57
2.6 Effect of the frequency distribution on the efficiency of MSIP. 57

3.1 Residual evolution with and without coarse grid correction. 61
3.2 Disposition of blocks of control volumes in ACM. 68
3.3 Non-structured agglomeration of control volumes. 72
3.4 Evolution of the residual norm for a two levels ACM algorithm with different β values. 75
3.5 Evolution of the residual norm for a two levels ACM algorithm for different problem sizes and a fixed β value. 76
3.6 Evolution of the residual norm for a two level ACM algorithm with different α values. 76
3.7 Convergence history of different V ($n_1 = 1$) and W ($n_1 > 1$) ACM cycles without post-smoothing ($n_{2b} = 0$). 77
3.8 Convergence history of different V ACM cycles ($n_1 = 1$). 77
3.9 Convergence history of a static ACM cycle with different frequency distributions of the error. 78
3.10 Iterations per level needed by a dynamic ACM algorithm, with $\alpha = 0.1$ and $\beta = 1.3$ for problems with different p_1 and p_2 parameters. 78
3.11 Time to solve the problem model with ACM versus N. 79
3.12 Number of iterations and cost of the smoothing in each level. 80
3.13 Time to solve the two-dimensional and three-dimensional systems with ACM versus N. 80
3.14 Time to solve the problem model versus N, with ACM and LU. 81
3.15 Norm of the residual versus computing time for ACM and LU. 81
3.16 Instantaneous velocity and temperature fields of a turbulent natural convection simulation. 87
3.17 Instantaneous right wall Nusselt numbers for integrations carried out with different ϵ values. 88
3.18 Number of iterations to achieve convergence for a steady-state flow with ACM and CACM. 89
3.19 CPU time to achieve convergence for a steady-state flow with ACM and CACM. . . 90

4.1 Illustrative results obtained with the parallel code for reactive flows. Top: flat flame. Bottom: Micro slit burner. 112
4.2 Iterations needed by the parallel code for reactive flows as a function of the number of subdomains. 113
4.3 Time to solve each of the subdomains for a flat flame problem. 113
4.4 Speed-ups of the code for reactive flows (flat flame) in the different systems. 114

5.1 Domain decomposition used by the geometric multigrid algorithms. 119
5.2 Propagation of erroneous values after each Red-Black Gauss-Seidel iteration. 121
5.3 Time to do a fixed number of Red-Black Gauss-Seidel iterations on the Cray T3E. . 123

5.4 Parallel efficiency of Red-Black Gauss-Seidel iterations on the Cray T3E. 124
5.5 Data dependencies of operator *down* $\left(u^k, f^k\right) \rightarrow f^{k-1}$. 129
5.6 Data dependencies of operator *up* $\left(u^k, u^{k-1}\right) \rightarrow u^k$. 130
5.7 Evolution of the correct areas during the second DDV leg. 131
5.8 Evaluation of u and f before the first leg of a DDV algorithm in one-dimensional and two-dimensional domain decompositions. 133
5.9 Convergence of the different parallel MG algorithms for a problem with $M = 8$, $N = 2.4 \times 10^6$ equations. 134
5.10 Computing time of the parallel MG variants with 9 processors versus number of unknowns. 135
5.11 Computing time of the parallel MG variants with 16 processors versus number of unknowns. 136
5.12 Speedup of the parallel MG variants for a problem with $N = 768^2 \approx 5.9 \times 10^5$ unknowns. 136
5.13 Speedup of the parallel MG variants for a problem with $N = 1536^2 \approx 2.4 \times 10^6$ unknowns. 137
5.14 Fraction of total execution time spent in halo update operations by the parallel MG variants versus the problem size, with $P = 16$. 137
5.15 Fraction of total execution time spent by the parallel MG variants in the sequential solution of the coarsest level versus the problem size, with $P = 16$. 138

6.1 Schur decomposition. Domain before reordering. 142
6.2 Schur decomposition. Domain after reordering. 142
6.3 Speed-ups of the Schur algorithm in the JFF cluster. 149
6.4 Breakdown of the processing time of the Shur solver, with $P = 16$ for different problem sizes. 150

7.1 Schema of the domain decomposition used by DDACM algorithm. 154
7.2 Minimum, maximum and average time to update a 4 nodes halo with $P = 16$ on the JFF cluster using mode 0. 158
7.3 Minimum, maximum and average time to update a 4 nodes halo with $P = 16$ on the JFF cluster using mode 1. 158
7.4 Minimum, maximum and average time to update a 4 nodes halo with $P = 16$ on the JFF cluster using mode 2. 159
7.5 Comparison of average times to update a 4 nodes halo with $P = 16$ on the JFF cluster using the different modes. 159
7.6 Average communication time to update a 4 nodes halo on the JFF cluster for different numbers of processors, using mode 1. 160
7.7 Maximum communication time to update a 4 nodes halo on the JFF cluster for different numbers of processors, using mode 1. 160
7.8 Average time to update a 4 nodes halo of a group of levels, simultaneously or level by level. 161
7.9 Speedup obtained in a fixed number of BILU iterations on the JFF cluster with different mesh sizes, with $J = 4$ and $\nu_{loc} = 1$. 162
7.10 Block partitions in DDACM. 163
7.11 DDACM correction equations for different numbers of processors. 164
7.12 Second leg in DDACM algorithm. 166
7.13 Number of iterations needed by DDACM to converge the problem model versus the number of processors, for a mesh with $N = 82944$. 168
7.14 Number of iterations needed by DDACM to converge the problem model versus the number of processors, for a mesh with $N = 1327104$. 169
7.15 DDACM Speedup for different meshes. 169
7.16 Breakout of computing time for different number of levels. 170

List of Tables

2.1 Terms of the generic convection-diffusion equation. 36

3.1 Examples of geometric multigrid algorithms for CFD applications. 66
3.2 Examples of algebraic multigrid algorithms for CFD applications. 67
3.3 Averaged number of iterations needed per time step for different precisions in the solution of the pressure correction equation (relative to the initial residual of the equation). 84
3.4 Averaged CPU time (s) to solve each time step, using different algorithms. 84

4.1 Compatibility between parallel computing hardware and programming models. . . . 98
4.2 Number of block Jacobi iterations using a direct solver for each subdomain. 106
4.3 Numerical efficiency of block Jacobi iterations using Gauss-Seidel for each subdomain. 106
4.4 Number of block Jacobi iterations using an iterative solver for each subdomain with an overlapping area. 107
4.5 Main characteristics of the illustrative examples of reactive flows. 111

5.1 Parallel MG variants considered. 132
5.2 Execution times of parallel MG variant 2. 134
5.3 Execution times of parallel MG variant 3. 135
5.4 Execution times of parallel MG variant 4. 135

6.1 Execution times of Dual Schur solver. 149

7.1 Execution times and number of iterations of DDACM for different mesh sizes and number of levels, with one processor. 167
7.2 Execution times and number of iterations of DDACM for different mesh sizes and number of levels, with four processors. 167
7.3 Execution times and number of iterations of DDACM for different mesh sizes and number of levels, with nine processors. 167
7.4 Execution times of DDACM for different mesh sizes and number of levels, with sixteen processors. 168

List of Algorithms

3.1 ACM algorithm. Iterate. 72
3.2 ACM algorithm. Recursive solution of one level. 73
3.3 ACM algorithm. Main function. 74
5.1 Parallel geometric multigrid. Halo update. 120
5.2 Red-Black Gauss-Seidel smoother. 123
5.3 Red-Black Gauss-Seidel smoother for a parallel geometric multigrid. 123
5.4 Parallel V cycle. 126
6.1 Schur complement algorithm. Preprocessing (first approach). 146
6.2 Schur complement algorithm. Solution (first approach). 146
6.3 Schur complement algorithm. Preprocessing (second approach). 147
6.4 Schur complement algorithm. Solution (second approach). 148
7.1 Mode 0 halo update. 156
7.2 Mode 1 halo update. 156
7.3 Mode 2 halo update. 157
7.4 Mode 2 data exchange. 157
7.5 BILU smoother used in DDACM. 162
7.6 DDACM algorithm. Preprocessing. 164
7.7 DDACM algorithm. Solution. 165

Abstract

The main purpose of the dissertation is to contribute to the development of numerical techniques for computational heat transfer and fluid flow, suitable for low cost (loosely coupled) parallel computers. It is focused on implicit integration schemes, using finite control volumes with multigrid (MG) algorithms.

Natural convection in closed cavities is used as a problem model to introduce different aspects related with the integration of the incompressible Navier-Stokes equations, such as the solution of the pressure correction (or similar) equations that is the bottleneck of the algorithms for parallel computers. The main goal of the dissertation has been to develop new algorithms to advance in the solution of this problem rather than to implement a complete parallel Computational Fluid Dynamics (CFD) code.

An overview of different sequential multigrid algorithms is presented, pointing out the difference between geometric and algebraic multigrid. A detailed description of segregated Additive Correction Multigrid (ACM) is given. The direct simulation of a turbulent natural convection flow is presented as an application example. A short description of the coupled ACM variant is presented.

Background information of parallel computing technology is provided and the the key aspects for its efficient use in CFD are discussed. The limitations of low cost, loosely coupled cost parallel computers (high latency and low bandwidth) are introduced. An overview of different control-volume based parallel CFD algorithms and linear equation solvers is done. As an example, a code to solve reactive flows using Schwartz Alternating Method that runs efficiently on Beowulf clusters is given.

Different alternatives for latency-tolerant parallel multigrid are examined, mainly the Domain Decomposed V cycle (DDV) proposed by Brandt and Diskin in a theoretical paper. One of its main features is that, supressing pre-smoothing, it allows to reduce the each-to-neighbours communications to one per MG iteration. In the dissertation, the cycle is extended to two-dimensional domain decompositions. The effect of each of its features is separately analysed, concluding that the use of a direct solver for the coarsest level and the overlapping areas are important aspects. The conclusion is not so clear respect to the suppression of the pre-smoothing iterations.

A very efficient direct method to solve the coarsest MG level is needed for parallel MG. In this work, a variant of the Schur complement algorithm, specific for relatively small, constant matrices has been developed. It is based on the implicit solution of the interfaces of the processors subdomains. In the implementation proposed in this work, a parallel evaluation and storage of the inverse of the interface matrix is used. The inner nodes of each domain are also solved with a direct algorithm. The resulting algorithm, after a pre-processing stage, allows a fast solution of pressure correction equations of incompressible flows in loosely coupled parallel computers.

Finally, all the elements presented in the work are combined in the Domain Decomposed ACM (DDACM) algorithm, an algebraic MG equivalent to the DDV cycle, that is as a combination of a parallel ACM algorithm with Block Incomplete Lower-Upper (BILU) smoothing and a specific version of the Schur complement direct solver. It can be treated as a black-box linear solver and tailored to different parallel architectures.

The parallel algorithms analysed (different variants of V cycle and DDV) and developed in the work (a specific version of the Schur complement algorithm and the DDACM multigrid algorithm) are benchmarked using a cluster of 16 PCs with a switched 100 Mbits/s network.

The general conclusion is that the algorithms developed are suitable options to solve the pressure correction equation, that is the main bottleneck for the solution of incompressible flows on loosely coupled parallel computers.