Parallel multiblock CFD computations applied to industrial cases

H. Nilsson, S. Dahlström and L. Davidson

Chalmers University of Technology, Department of Thermo and Fluid Dynamics,
SE-412 96 Göteborg, Sweden

A parallel multiblock finite volume CFD (Computational Fluid Dynamics) code CALC-PMB [3,6,7] (Parallel MultiBlock) for computations of turbulent flow in complex domains has been developed. The main features of the code are the use of conformal block structured boundary fitted coordinates, a pressure correction scheme (SIMPLEC [4] or PISO [5]), cartesian velocity components as principal unknowns, and collocated grid arrangement together with Rhie and Chow interpolation. In the parallel multiblock algorithm, two ghost cell planes are employed at the block interfaces. The message passing at the interfaces is performed using either PVM (Parallel Virtual Machine) or MPI (Message Passing Interface).

This work was performed on a 64-processor shared memory SUN Enterprise 10000 at Chalmers and on a 170-node distributed memory IBM SP at the Center for Parallel Computing at KTH (Royal Institute of Technology).

Parallel aspects of computations from two different industrial research areas, hydraulic machinery and aerospace, and an academic test case are presented. The parallel efficiency is excellent, with super scalar speed-up for load balanced applications using the best configuration of computer architecture and message passing interface [2].

1. Industrial cases

This work presents the parallel aspects of computations from two different industrial research areas, hydraulic machinery - *Numerical investigations of turbulent flow in water turbines* [6] and aerospace - *Large eddy simulation of the flow around a high-lift airfoil* [1]. The background of each case is briefly described below.

1.1. Hydraulic machinery

This work is focused on tip clearance losses in Kaplan water turbines, which reduce the efficiency of the turbines by about 0.5%. The work is part of a Swedish water turbine program financed by a collaboration between the Swedish power industry via ELFORSK (Swedish Electrical Utilities Research and Development Company), the Swedish National Energy Administration and GE (Sweden) AB. The turbine investigated (fig. 1) is a test rig with a runner diameter of $0.5m$. It has four runner blades and 24 guide vanes (fig. 2). The GAMM Francis runner [8] (fig. 3) is used for validation of the computational code, since there are no detailed measurements for the Kaplan runner. The tip clearance between the Kaplan runner blades and the shroud is $0.25mm$. In order to resolve the turbulent

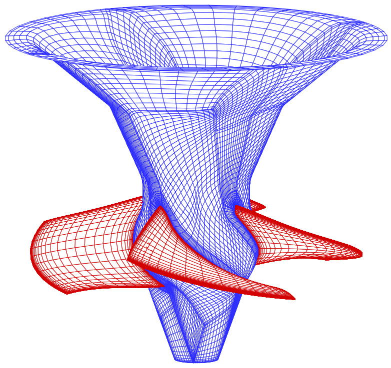

Figure 1. A Kaplan turbine runner multiblock grid.

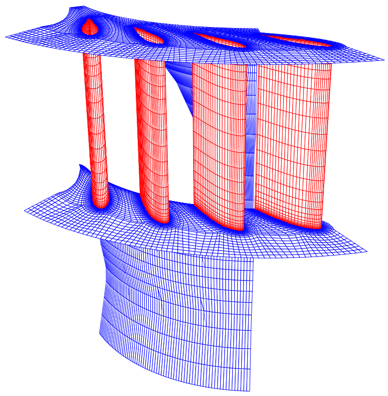

Figure 2. Four (of 24) guide vanes.

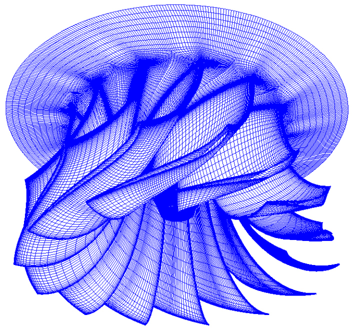

Figure 3. A Francis turbine runner multiblock grid.

flow in the tip clearance and in the boundary layers, a low Reynolds number turbulence model is used. Because of computational restrictions, complete turbine simulations usually use wall functions instead of resolving the boundary layers, which makes tip clearance investigations impossible.

Although the computations assume that the flow is periodic, which allows only one blade passage to be computed, and that the boundary conditions are assumed to be axisymmetric, these kinds of computations tend to become computationally heavy and numerically challenging. This, together with the complicated geometry requiring complex multiblock topologies, makes a parallel multiblock CFD solver a suitable tool.

1.2. Aerospace

This work is part of the ongoing Brite-Euram project called LESFOIL (Large Eddy Simulations of Flows around Airfoils). One of the main objectives of the project is a demonstration of the feasibility of LES for simple 2D airfoils (fig. 4). The test case chosen is the flow around the Aerospatiale A-airfoil at an angle of attack equal to 13.3° and where the chord Reynolds number is $2.1 \cdot 10^6$. This is a challenging case for LES because of the high Reynolds number and because of the different flow situations around the airfoil, including transition from the laminar flow near the leading egde and separation near the trailing edge (fig. 5). Even at the low Reynolds number, from an aeronautical point of view, a wall-resolved LES is too expensive. The use of approximate boundary conditions in the near-wall region is thus necessary. By using a *20-nodes per boundary-layer thickness estimate* [10] in each direction, 50−100 million nodes are needed for this case [9]. However, with a good method of prescribing and controlling the transition we hope that a 2 million node mesh is sufficient for LES with wall-functions. Still, the requirements on the mesh are demanding and result in meshes with a large number of nodes. For this reason, an efficient numerical method with an effective parallelization is needed.

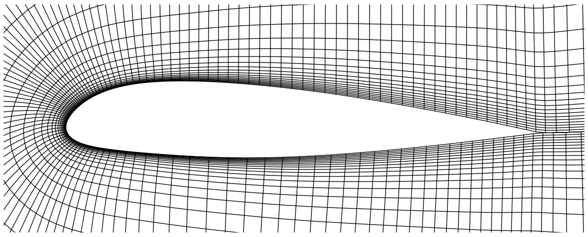

Figure 4. Zoom of mesh around the Aerospatiale A-airfoil (every 4th node in the wrap around direction and every 2nd node in the surface normal direction is plotted).

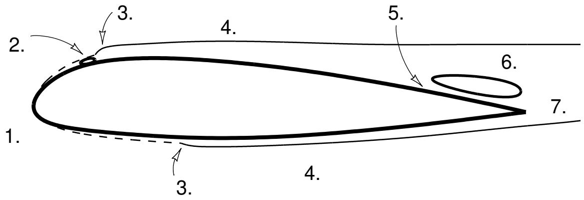

Figure 5. Schematic sketch of the flow regimes around the Aerospatiale A-profile: 1. laminar boundary layer, 2. laminar separation bubble, 3. transition region, 4. turbulent boundary layer, 5. separation point, 6. separation region, 7. wake region.

2. CALC-PMB: The Parallel MultiBlock CFD solver

A single structured block sequential finite volume CFD solver, CALC-BFC (Boundary Fitted Coordinates) [3], has been extended with message passing utilities (PVM or MPI) for parallel computations of turbulent flow in complex multiblock domains [6,7] (fig. 6). The main features of the resulting SPMD (Single-Program-Multiple-Data, all the processes run the same executable on different data) code, CALC-PMB, are the use of conformal block structured boundary fitted coordinates, a pressure correction scheme (SIMPLEC [4] or PISO [5]), cartesian velocity components as principal unknowns, and collocated grid arrangement together with Rhie and Chow interpolation. In the parallel multiblock algorithm, two ghost cell planes are employed at the block interfaces. The message passing at the interfaces is hidden behind a high level parallel multiblock library with data structures that fit CALC-PMB (fig. 7) and the underlying message passing interface (PVM or MPI) is chosen at compile time. The calls for parallel multiblock routines in the code are thus completely independent of the message passing interface that is used. Thus most of the parallelism is hidden from the user, who can easily manipulate the code for his/her purposes using the high level parallel multiblock library if necessary. The advanced user may easily add optional message passing interfaces if needed. The code may be run on everything from inhomogenous NOW (Network Of Workstations) to Beowulf Linux clusters and distributed and shared memory supercomputers. It may read a predefined multiblock topology with connectivity information from the disk or subdivide single block domains into equal sized sub-blocks for load balanced parallel computation. The gains of this operation are several: the computational speed may be increased, larger problems may be solved since the memory requirement is divided between the processors (when using distributed machines), more exact solutions may be obtained because of the extra memory available and parallel supercomputers may be employed.

2.1. Numerical procedure

The parallel SIMPLEC [4] numerical procedure can briefly be summarized as follows (see [6] for a more thorough description, or see [1] for a description of the parallel PISO [5]

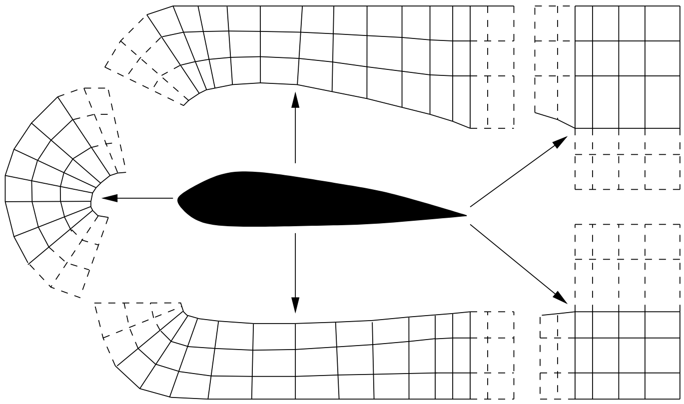

Figure 6. An example of an airfoil multiblock topology that can be computed using CALC-PMB. The blocks overlap two ghost cell planes (dashed lines).

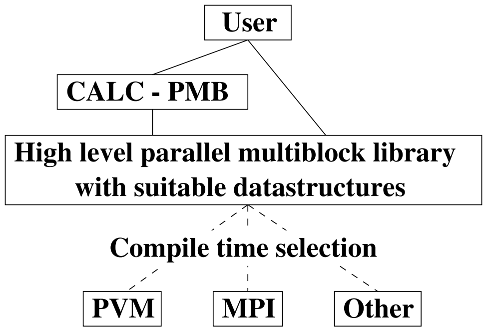

Figure 7. The message passing is hidden behind a high level parallel multiblock library with data structures that fit CALC-PMB.

algorithm).

Iterate through pts. I - X until convergence.

I The discretized momentum equations are solved.

II The inter-block boundary conditions for the diagonal coefficient from the discretized momentum equations are exchanged since they are needed for the Rhie & Chow interpolation.

III The convections are calculated using Rhie & Chow interpolation.

IV The continuity error, needed for the source term in the pressure correction equation, is calculated from these convections.

V The discretized pressure correction equation is solved.

VI The inter-block boundary conditions for the pressure correction are exchanged since they are needed for the correction of the convections.

VII The pressure, convections and velocities are corrected and the pressure field level is adjusted to a reference pressure in one point of the global computational domain. The velocity correction is actually not necessary but has proven to increase the convergence rate.

VIII Inter-block boundary conditions for all variables are exchanged.

IX Other discretized transport equations are solved.

X The residuals are calculated and compared with the convergence criteria. When converged, update the *old* (previous time step) variables and continue with the next time step for transient computations or conclude the computations for stationary computations.

3. Parallel aspects

CALC-PMB computes multiblock cases in parallel by assigning the blocks to separate processes. The computational grid may be given directly as a multiblock grid where the given block size distribution will be kept during the computations. If possible, it may also be given as a single structured block allowing for load balanced (user defined) domain decomposition in CALC-PMB. This work presents parallel aspects from two industrial research areas and one academic test case that has been used for validation and parallel efficiency tests.

Some comments on parallel efficiency should first be given. Commonly, the parallel efficiency is displayed on a 'per iteration' basis. However, for some methods such as the domain decomposition method used in this work, the number of iterations to convergence changes with the number of blocks/processors used. The time to convergence used in this work is then a better measure of the parallel efficiency.

3.1. Academic test cases - parallel aspects

The code has been validated and the parallel efficiency investigated for common academic test cases [7], such as the backward-facing step. Here, parallel aspects from the investigations of both 2D and 3D backward-facing step flow are presented. The 2D case has 22 512 nodes (134x42x4) and a Reynolds number of $Re = 24\,000$, and the 3D case has 874 120 nodes (130x82x82) and a Reynolds number of $Re = 5\,000$. Both cases use the $k - \epsilon$ turbulence model. The computational grid is decomposed in a load balanced way in CALC-PMB. The results of the investigations are presented in table 1. Since the 2D

Table 1
Speed-up and efficiency of 2D and 3D backward-facing step computations. The numbers are based on elapsed wall time to convergence, normalized by the respective single block computation. The domain decompositions are specified by the number of blocks in each direction.

Case	# processors	Domain decomposition	Speed-up	Efficiency
2D	1	1x1x1	1	100%
2D	2	2x1x1	2.0	100%
2D	4	2x2x1	3.1	78%
2D	8	4x2x1	5.7	71%
3D	1	1x1x1	1	100%
3D	4	2x1x2	4.3	108%
3D	8	4x1x2	8.3	104%

case is a rather small problem, with large block surface to volume ratios, the delay times caused by communication are already apparent for four processors. For larger problems, such as the 3D case or common industrial CFD applications, this is not a problem and the parallel efficiency is excellent for load balanced applications. For the 3D case shown in table 1 the speed-up is actually superscalar, probably because of a reduction of cache misses in the smaller subproblems. It is important to notice that, since the computational times are normalized with the computational times of a non-decomposed grid, both domain decomposition and communication effects are included. However, the convergence rates for these cases were affected only slightly by the domain decomposition. The computations were performed on a SUN Enterprise 10000 machine at Chalmers.

3.2. Hydraulic machinery - parallel aspects

The geometries in hydraulic machinery are generally very complicated and several regions of turbulent boundary layers must be resolved. For computational and numerical reasons, the grid size should be kept as small as possible and the control volumes as orthogonal as possible. This requires a complicated multiblock topology. Further challenges arise when there are large differences in geometrical scales, such as in tip clearance computations where the tip clearance block is an order of magnitude smaller than the largest block. If the blocks assigned to separate processes are not of equal size some blocks will wait for others to finish. Thus processors will be temporarily idle and the parallel efficiency will decrease rapidly. The level of parallelization is therefore determined by the block size distribution and the distribution of the processes over the available processors. This is the major problem in these kinds of computations. A load balancing procedure for these cases requires re-distribution of the multiblock topology and re-meshing, which are very time consuming. To a certain extent, this can be avoided by distributing the large blocks on separate processors and the small blocks on shared processors. On a shared memory supercomputer with many processors available, the computational speed will simply adjust to the computational speed of the largest block since the smaller blocks will be run using time sharing.

The CPU usage of each block is thus determined by the block sizes. Summing up the CPU usage from the different processes and normalizing it with the CPU usage of the largest block (which is a measure of the total load of the machine), this 12-block Kaplan runner tip clearance computation (with large differences in block sizes) runs in average on about 7.8 processors. If these computations are to be performed on a distributed system, the blocks must be distributed on eight processors in a way that guarantees load balancing.

An example of the load balancing problem is given in table 2, where the CPU usage of the processors is compared with the block sizes of the GAMM Francis runner computations. The distribution of the CPU usage is quite similar to the distribution of gridpoints, except for some overhead CPU usage that might arise from the message passing procedures.

3.3. Aerospace - parallel aspects

In the airfoil computations presented in this work, the computational grid may be represented as a single structured C-grid wrapped around the airfoil (fig. 4). This configuration allows the user to decide upon the multiblock topology using some parameters in CALC-PMB. The resulting domain decomposition is load balanced, and the multiblock topology and number of blocks may easily be changed without re-meshing.

The computations were done on a 64-processor SUN Enterprise 10000 at Chalmers and

Table 2
Load balance problem (example from a Francis runner computation)

Block	#gridpoints	Normalized #gridpoints (%)	Normalized Instant. CPU %
1	126,360	87.8	94.4
2	144,000	100.0	100.0
3	144,000	100.0	100.0
4	114,660	79.6	86.6
5	72,150	50.1	59.2
Total	601,170	417.5	440.2

on a 170-node IBM SP at the Center for Parallel Computing at KTH. Table 3 compares the elapsed time per time step for different combinations of domain decompositions, message passing interfaces and computers. A comparison of mesh size effects was also made. Two

Table 3
Elapsed time per time step on the coarse and fine meshes (722568 and 1617924 computational nodes, respectively).

Mesh	Computer & message passing system	Number of processors 8	16	32
Coarse	SUN, PVM, socket based	$48s$	$38s$	$36s$
	SUN, PVM, shared memory based	$24s$	$12s$	$6s$
	SUN, MPI	$24s$	-	-
	IBM SP, PVM (based on MPI)	$12s$	-	-
	IBM SP, MPI	-	$5.4s$	$2.8s$
Fine	IBM SP, MPI	-	-	$6.0s$

different versions of PVM are available on the SUN computer: a shared memory-based PVM and a socket-based PVM. When eight processors are used for the present case, the shared memory-based PVM is twice as fast as the socket-based PVM. Some preliminary tests have shown that, when using the shared memory-based PVM, CALC-PMB may scale linearly at least up to 32 processors but that it does not scale at all when the socket based PVM is used. The use of MPI yields the same execution time as the shared memory PVM for the eight processor case, and it is reasonable to believe that it will scale linearly as well. However, since it did not perform better than the shared memory-based PVM for eight processors, the tests were moved to another computer architecture.

Using PVM on the distributed memory IBM SP computer, the eight processor case was twice as fast as when the SUN shared memory PVM was used. Since the IBM SP PVM is based on MPI, using MPI should yield the same computational time as when using PVM. The 16 and 32 processor cases were performed using MPI, and approximately linear speed-up is obtained between the 8 and 32 processor cases.

When the mesh size was scaled up by a factor of 2.3, the computational time was increased by a factor of 2.1 for the 32 processor IBM SP MPI case, which shows that CALC-PMB scales very well for large CFD problems using the most appropriate combination of message passing interface and computer architecture.

4. Conclusions

A parallel multiblock finite volume CFD code, CALC-PMB, for computations of turbulent flow in complex domains has been developed. Most of the parallelism is hidden from the user, who can easily manipulate the code for his/her purposes using a high level parallel multiblock library if necessary. The advanced user may easily add optional message passing interfaces, besides PVM and MPI, if needed.

The parallel efficiency of the code is excellent, with super scalar speed-up at least up to 32 processors for large 3D load balanced applications using the best configuration of computer architecture and message passing interface. However, it has been shown that the parallel efficiency may decrease drastically if the problem size is small, the load balancing poor or the configuration of computer architecture and message passing interface is not good.

REFERENCES

1. S. Dahlström. Large Eddy Simulation of the Flow Around a High Lift Airfoil. Thesis for the degree of Licentiate of Engineering 00/5, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 2000.
2. S. Dahlström, H. Nilsson, and L. Davidson. LESFOIL: 6-months progress report by Chalmers. Technical report, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1998.
3. L. Davidson and B. Farhanieh. CALC-BFC: A Finite-Volume Code Employing Collocated Variable Arrangement and Cartesian Velocity Components for Computation of Fluid Flow and Heat Transfer in Complex Three-Dimensional Geometries. Rept. 92/4, Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1992.
4. J.P. Van Doormaal and G.D. Raithby. Enhancements of the SIMPLE method for predicting incompressible fluid flows. *Num. Heat Transfer*, 7:147–163, 1984.
5. R.I. Issa. Solution of Implicitly Discretised Fluid Flow Equations by Operator-Splitting. *J. Comp. Physics*, 62:40–65, 1986.
6. H. Nilsson. A Numerical Investigation of the Turbulent Flow in a Kaplan Water Turbine Runner. Thesis for the degree of Licentiate of Engineering 99/5, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1999.
7. H. Nilsson and L. Davidson. CALC-PVM: A parallel SIMPLEC multiblock solver for turbulent flow in complex domains. Int.rep. 98/12, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1998.
8. G. Sottas and I. L. Ryhming, editors. *3D-Computations of Incompressible Internal Flows - Proceedings of the GAMM Workshop at EPFL, September 1989, Lausanne - Notes on Numerical Fluid Mechanics.* Vieweg, Braunschweig, 1993.
9. P.R. Spalart. Private communication. Boeing Commercial Airplanes, 2000.
10. P.R. Spalart, W-H. Jou, M. Strelets, and S.R. Allmaras. Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach. *1st AFOSR Int. Conf. on DNS/LES, Aug. 4-8, Ruston, LA. In Advances in DNS/LES, C. Liu & Z. Liu Eds., Greyden Press, Columbus, OH*, 1997.