**IEEE International Conference on Cluster Computing,
CLUSTER 2000, November 28.–December 2., 2000,
Chemnitz University of Technology, Saxony, Germany.**

# Multi-Grid Acceleration of a SIMPLE-Based CFD-Code and Aspects of Parallelization

Klaus Bernert   Thomas Frank   Hellfried Schneider   Klaus Pachler

Chemnitz University of Technology
Faculty of Mechanical Engineering and Process Technology
Research Group of Multiphase Flow
Reichenhainer Straße 70, 09107 Chemnitz, Germany
Email: kber@imech.tu-chemnitz.de

## Abstract

*The present study treats the calculation of the continuous phase of a multiphase flow, i.e. the numerical solution of the equations for a turbulent incompressible flow. An existing SIMPLE based Single–Grid algorithm is improved applying the Multi–Grid Method. Numerical experiments for two test cases with up to 3 174 400 finite volumes show the features of the new algorithm, the effect of its parallelization and capability to solve real life problems from engineering on PC clusters.*

## 1 Motivation

Disperse multiphase flows are very common for processes in mechanical and thermal process technology (e.g. gas–particle or gas–droplet flows, coal combustion, pneumatic conveying, erosion phenomena). Furthermore processes for the separation of solid particles from gases or fluids and for the classification and particle size analysis are an important field of interest in process technology.
The numerical simulation of multiphase flows includes both the calculation of the continous phase and the calculation of a high number of particle traces as a basis for deriving statistical quantities as particle density, mean particle velocity and further quantities.
As a first step the continous phase can be calculated independently of the disperse phase, later the interaction with the disperse phase is to be included in the right hand side of the equations of motion in an iterative way. Already a single flow calculation and more than ever the coupled iterative calculation is very time–consuming. Efficient numerical algorithms as well as the power of parallel computing systems are needed to solve real problems from engineering.

The topic of the present study is the calculation of the continous phase flow i.e. the solution of the equations for a turbulent flow. Multi–Grid acceleration of an existing SIMPLE algorithm and some aspects of the parallelization of the numerical algorithm are the primary subjects.
Examples of previously treated applications of our numerical approach MISTRAL/PartFlow–3D with special emphasis on the disperse phase flow can be found in [5, 6, 7, 8, 9]. For current results concerning the parallelization of the disperse phase flow calculation s. [10].

## 2 Mathematical Model

An incompressible, isothermal, turbulent and statistically steady flow can be described by a system of six partial differential equations of the following general form

$$\frac{\partial}{\partial x_j}\Big(\rho_F\, u_j\, \Phi\Big) - \frac{\partial}{\partial x_j}\left(\Gamma\, \frac{\partial \Phi}{\partial x_j}\right) = S, \qquad (1)$$

where $\Phi$ stands for the Cartesian velocity components $u_1 = u, u_2 = v, u_3 = w$, the turbulent kinetic energy $k$ and the energy dissipation $\varepsilon$. The turbulence model used is the standard $k - \epsilon$ model. Table 1 shows the variables and source terms for the different equations. $\Gamma$ is a general transport coefficient and $S$ the source term. In the table $\mu$ denotes the laminar and $\mu_t$ the turbulent viscosity, $\rho_F$ is the fluid density and $f_1$, $f_2$ and $f_3$ are the Cartesian components of outer forces per mass unit.

## 3 Discretization Concept

The discretization is done on a block–structured non–overlapping grid. All blocks are demanded to be topologically equivalent to a cuboid. The grid has a regular hexahedral structure on each block, logically equivalent to a

| Equation for | $\Phi$ | $\Gamma$ | $S$ |
|---|---|---|---|
| Continuity | 1 | 0 | 0 |
| Momentum | $u_i$ | $\mu_{eff}$ | $\frac{\partial}{\partial x_j}\left(\Gamma \frac{\partial u_j}{\partial x_i}\right) - \frac{\partial p}{\partial x_i} + \rho_F f_i$ |
| Turbulent kinetic energy | $k$ | $\frac{\mu_t}{\sigma_k}$ | $P_k - \rho_F \varepsilon$ |
| Dissipation of k | $\varepsilon$ | $\frac{\mu_t}{\sigma_\varepsilon}$ | $\frac{\varepsilon}{k}(c_{\varepsilon_1} P_k - c_{\varepsilon_2} \rho_F \varepsilon)$ |
| $P_k = \mu_t \frac{\partial u_i}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right);\ \mu_{eff} = \mu + \mu_t,\ \mu_t = \rho_F c_\mu \frac{k^2}{\varepsilon}$ | | | |
| $c_\mu = 0.09,\ c_{\varepsilon_1} = 1.44,\ c_{\varepsilon_2} = 1.92,\ \sigma_k = 1.0,\ \sigma_\varepsilon = 1.3$ | | | |

**Table 1. Flow variables, transport coefficients and source terms for the basic equations**

Cartesian grid. At inner block interfaces an accurate matching of the common faces is required.
Structured grids of this type have disadvantages in respect of their geometrical flexibility, the solution of the equations, however, can take advantage of its regular organization.
The discretization of the conservation equations (1) is performed by the finite volume method using a non–staggered grid management. To avoid oscillations, the Rhie and Chow mass flux interpolation [12] is used. The diffusive terms are discretized with central differences, for the convective part a blending between first order upwind and central differences was implemented.

## 4 Solution Methods

### 4.1 The SIMPLE Method

The starting point of this study is the SIMPLE method due to Caretto et al. [3]. After turning the continuity equation into an equation for a pressure correction, equations (1) are corrected in a segregated way in the following sequence: u-momentum, v-momentum, w-momentum, pressure-correction, turbulence energy and dissipation of turbulence energy. This sequence is repeated in an outer iteration cycle up to a prescribed error level. To correct the single equations the SIP method [13] is used. In order to preserve the coupling of the whole system a small number of inner SIP–iterations (e.g. two) proves to be sufficient. There is an exception, however: The pressure correction equation converges very slowly and an increased number of iterations for this equation leads to a better overall convergence.

Due to the block structure the SIP procedure is applied in a Domain Decomposition mode described in [4]. That means SIP works on each block separately and the coupling of the blocks is realized by a data exchange at all inner block boundaries between the iterations. The described block iteration method can be used for serial and parallel calculation without any changes.

From a present-day view the convergence speed of the original SIMPLE method is far from optimal. This is the starting point for implementing the Multi–Grid technique.

### 4.2 The Multi–Grid Method

This section gives a short description of the Multi–Grid (MG) algorithm. Detailed information can be found in [2], [14], [11] and [1].

A TWO–GRID ALGORITHM consists of the following steps:

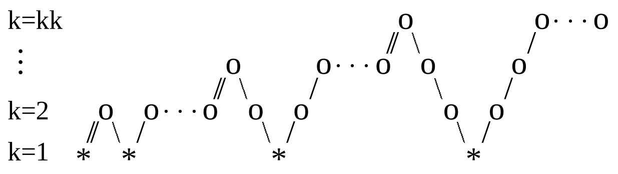
1. $\nu_1$ smoothing sweeps on the fine grid (PRE–SMOOTHING)
2. RESTRICTION to the coarser grid
3. solution of the problem on the coarser grid
4. PROLONGATION of the results to the fine grid and correction of the last fine–grid approximation
5. $\nu_2$ smoothing sweeps on the fine grid (POST–SMOOTHING)

The aim of using two grids is a reduction of computational work: High frequency components of the error are efficiently damped on the fine grid, i.e. $\nu_1$ and $\nu_2$ can be set to one or two in most cases. Low frequency parts of the error can be removed on a coarser grid without loss of accuracy but with less numerical operations than on the fine grid. Steps 1-5 are to be performed several times.

The MULTI–GRID ALGORITHM is a generalization of the two–grid method: The solution of the problem on the coarse grid is now replaced in a recursive way by one or two steps of the two–grid algorithm. This leads to the so–called V– and W–cycles working on a sequence of grids denoted by $\omega_k,\ k = 1, \ldots, kk$ .

If there is no good initial guess for the solution of a problem it is best to use the FULL–MULTI–GRID ALGORITHM (FMG) algorithm. In this case the solution process starts on the coarsest grid $\omega_1$ with a direct solver or a sufficiently high number of iterations. Then a higher order accurate FMG–prolongation gives the initial solution for a two–grid method on $\omega_2$. After $\gamma$ iterations the solution is prolongated to the next finer grid and so on up to the finest grid $\omega_{kk}$. In the ideal case one MG–cycle on every grid–level can be sufficient to get a solution with an error close to the discretization error. Fig. 1 shows how the method works.

The intention of the present study was to use as much as possible from the existing Single–Grid algorithm for the



Notations: $*$ direct solution, o pre- or post-smoothing,
$\backslash$ restriction, / prolongation, // FMG-prolongation

**Figure 1. Full Multi–Grid method (V-cycle, 4 grids)**

Multi–Grid method. The calculation of gradients can be used for restriction and prolongation and the SIMPLE algorithm was hoped to be a usable smoothing procedure and coarse grid solver.

The Multi–Grid method described below is applied to the system of equations (1) as a whole. A Multi–Grid acceleration of the segregated inner iterations of the SIMPLE algorithm would be possible too, but this is not the right way to treat the nonlinear coupling between the equations. The iteration of the pressure correction equation, however, can be accelerated with inner Multi–Grid cycles. This leads to a deciding improvement of the efficiency of the SIMPLE Method, as will be shown in the next section.

For the present no linearization precedes the use of the Multi–Grid method for the system to be solved, i.e. the nonlinearity of the equation is treated by the MG-algorithm.

## 4.3 The improved SIMPLE Method

To improve the efficiency of the original SIMPLE algorithm the influence of the number of inner pressure iterations was investigated. Table 2 shows the results of some test runs for a straight channel described in 6.1 with 32*32*128=131 072 finite volumes. With an increasing number of inner pressure iterations the number of outer iterations decreases considerably while the total execution time

| inner iterations for pressure | outer iterations | T_total in s | T_p in % |
|---|---|---|---|
| 15 | 3024 | 17494 | 28 |
| 50 | 1503 | 12905 | 51 |
| 100 | 902 | 11174 | 66 |
| 200 | 517 | 10356 | 79 |
| 300 | 362 | 10083 | 85 |
| 400 | 273 | 9936 | 88 |
| 500 | 225 | 10679 | 90 |

**Table 2. Convergence of the original SIMPLE algorithm depending on the number of inner pressure iterations**

T_total has an optimum at about 400 inner iterations. The amount of time T_p for the pressure equation shows a further remarkable potential for saving computer time. So an inner Multi–Grid method was implemented for the pressure correction equation. The SIP solver works as smoothing method and can be used as coarse grid solver too. A more efficient solver for the problem on the coarsest grid is the conjugate gradient method with Incomplete Choleskey preconditioning. For a serial computer the usage of the CG method on the coarsest grid is not important, since the dimension of the equations on the coarsest gird is not large. On a parallel machine, however, it is important to minimize the number of iterations because the time for data exchange between the processors has to be regarded too.
Fig. 2 presents the convergence history for the straight channel with $64{*}64{*}256 = 1\,048\,576$ finite volumes for a parallel run with 8 blocks on 8 processors. The oscillating curve belongs to the original SIP method with the optimal number of inner pressure iterations (which is not known a priori!). The two other curves represent the improved SIMPLE method with the inner Multi–Grid method for pressure without (MGP) and with the CG method (MGP/CG) on the coarsest grid. Using the CG method decreases the total number of data exchange operations on the coarsest grid to 7.5% (in absolute numbers from 473 240 to 35 539) The amount of pressure calculation time compared with the total execution time decreases from 86% (SIP) over 37% (MGP) to 26% (MGP/CG) while the time T_total itself reduces from 100% over 7.2% to 6.3%.

REMARK: All convergence history figures in this article present normalized residuals. In order to show the convergence behaviour of the algorithms the calculations are stopped at a very low error level far below the discretization error.

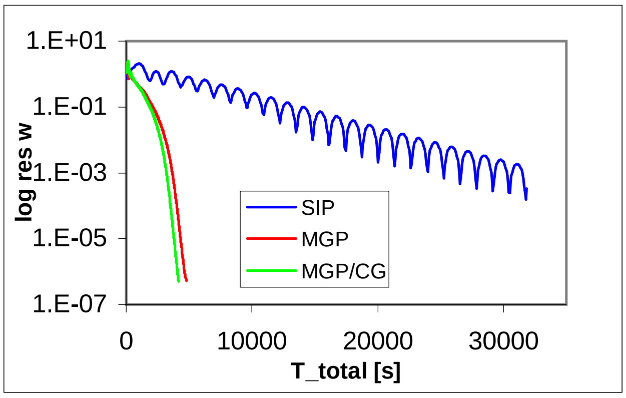


**Figure 2. Convergence history for the Single–Grid SIMPLE algorithm with different solvers for the pressure equation**

## 5 Parallelization

The parallel approach was developed for a machine of MIMD type. Most of the calculations presented in this study were performed on a cluster of 12 PCs with 600MHz AMD/Athlon-CPUs and FastEthernet communication network. The parallel efficiency of the code was investigated on a CRAY-T3E with 300MHz DEC Alpha-CPUs and GigaRing network and on the **C**hemnitz **Li**nux **C**luster **CLiC** consisting of 528 PCs with 800MHz Intel P III-CPUs and FastEthernet network.

So far the parallel code realizes the same algorithm as the serial code. The method starts with an setup step. The host (one of the calculating nodes) reads the input file with grid coordinates, grid block coupling information and boundary conditions for all blocks generated in a preprocessing process and passes them to the nodes according to a block–processor allocation table. This table is determined in a heuristic way with the aim to give all nodes nearly the same amount of work. Normally the number of blocks exceeds the number of processors. So each processor handles a few blocks. The assignment starts with ordering the blocks by their size starting with the biggest one. Then one block after the other is allocated to one of the processors for which the sum of control volumes of all local blocks is minimal.

The code is written in C. All information belonging to a block is saved in a structure. More precisely, the structure contains pointers and each node works on an array of such structures. The nodes dynamically allocate the memory for all information of their blocks. The access on the individual blocks is organized by an additional structure of the same type. This working structure is cyclically mapped on the structures for the blocks by a setup procedure within a loop over the locally stored grid blocks. The typical loop structure is

```
for (n=1; n<=number_of_local_blocks; n++)
{   set_pointers_for_block(n);
    work_on_block(n);
}
```

The serial case can be considered as a special case of the parallel mode – all blocks are treated by the same processor.

In the parallel case there are two types of data exchange. The first type is a global exchange, e.g. for calculating scalar products in the CG algorithm or calculating the maximal residual. This can easily be realized with existing MPI–calls.
The second type of data exchange realizes the coupling at common faces of neighbouring grid blocks. To permit a relatively independent calculation on the blocks they are saved with a small overlapping region which contains the values of the nearest points of the neighbour blocks. In order to ensure the coupling between the blocks this information is to be refreshed regularly. A part of this communication can be done without parallel data exchange if the nodes handle more than one block.
The other part requires data exchange between the nodes which is done as follows: At first all nodes send the information for other nodes by non–blocking MPI calls. Then the nodes receive all information needed for their blocks what can include a small waiting time, if some information has not been sended yet. To save computer time the data exchange is organized in such a way that up to nine functions (this is the case for the gradient of a vector function) can be exchanged simultaneously.

## 6 Numerical Experiments

### 6.1 Test Cases

Two test cases are presented. The first test is a turbulent flow through a STRAIGHT square CHANNEL $(0.1 * 0.1 * 2.0\ [m])$ with a Reynolds number $Re = \frac{ul}{\nu} = 67\,000$. At the inlet a uniform velocity profile is given, at the outlet a zero gradient condition is implemented. Equally spaced grids with $4*4*16$ (coarsest grid) up to $64*64*256 = 1\,048\,576$ (finest grid) finite volumes are used for the calculations.

The second and more real test case is a three times BENDED DUCT with blades in the channel bends (see Fig. 7) which could be used for pneumatic transport. The blades are modeled as infinitely thin solid walls within the flow region (non slip condition). Inlet and outlet conditions are modeled as for the straight channel. The duct has been subdivided into 64 blocks, the number of finite volumes for the finest grid is $80*80*496 = 3\,174\,400$; because of the blades no more than three coarser grids can be used. This means that the coarsest grid with only two cells between the blades has $10*10*62 = 6\,200$ finite volumes. The Reynolds number for the bended duct is $Re = 156\,000$.

### 6.2 Results

A first test run compares the convergence history for the improved SIMPLE method (denoted as Single–Grid method (SG) with the Multi–Grid method, both using the inner MG method for pressure calculation. The test is performed for the straight channel with $64*64*256$ finite volumes splitted into 8 blocks on 8 processors. While the number of SG iterations was 330 the number of MG cycles is 21, if not this number but the time for convergence is minimized. Fig. 3 shows the higher efficiency of the MG method, it needs 27% of the total time compared to the improved SIMPLE method and less than 1.7% compared to the original SIMPLE method.

| Number of blocks | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
|---|---|---|---|---|---|---|---|---|---|---|
| SG-SIP | 216 | 217 | 304 | 317 | 376 | 532 | 554 | 693 | 947 | 1002 |
| SG-MGP/CG | 133 | 133 | 135 | 136 | 137 | 140 | 143 | 145 | 148 | 154 |
| MG | 21 | 23 | 25 | 25 | 26 | 29 | 30 | 32 | 34 | 33 |
| T_total for MG in [s] | | | 205 | 163 | 171 | 212 | 253 | 375 | 779 | 1060 |

**Table 3. Influence of the number of blocks on the number of SIMPLE iterations or MG cycles**

Fig. 4 demonstrates the better starting behavior of the FMG method compared with the MG method for the same test case.

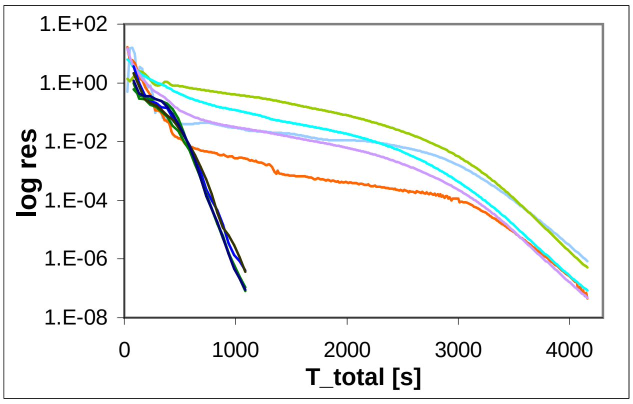


**Figure 3. Convergence history for Single–Grid (gray curves) and Multi–Grid algorithm (black curves), both with inner Multi–Grid solver for pressure, residuals for u, w, mass, k, $\epsilon$**

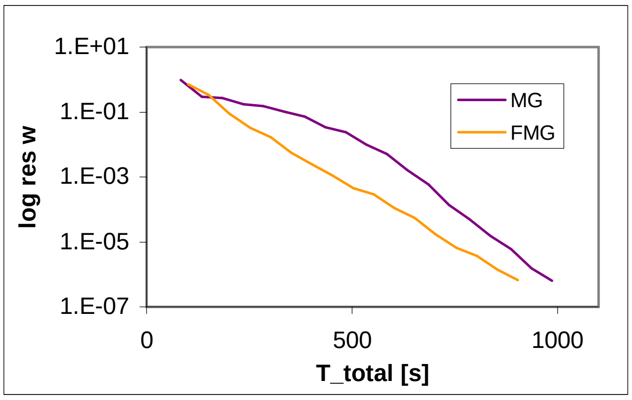


**Figure 4. Convergence history for Multi–Grid and Full–Multi–Grid algorithm**

In a next test the number of blocks for the straight channel with $32 * 32 * 128$ finite volumes is changed from 2 over

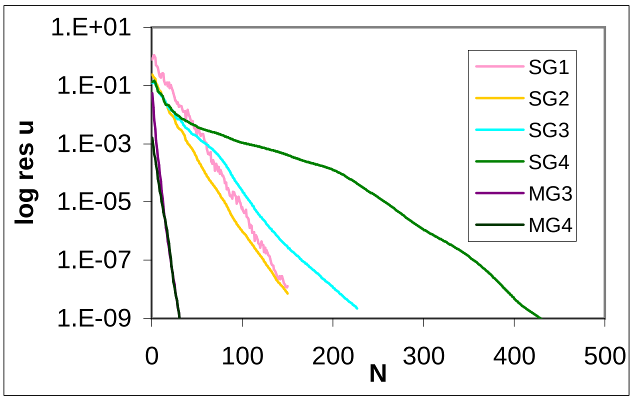


**Figure 5. Convergence of Single– and Multi–Grid method on a sequence of refined grids**

4, 8, 16 up to 512 and 1024. In the case of small block numbers the influence of the number of blocks on the number of iterations is moderate for the original SIMPLE method (SG–SIP) and negligible for the SIMPLE and Multi–Grid method with the inner MG method for the pressure calculation (SG–MGP/CG and MG), see Table 3.
For large block numbers SG–SIP needs noticeable more iterations while the other two methods need only slightly more iterations. The total calculation times are nearly constant for a small or medium number of blocks. For parallel calculations an increase of the number of blocks can be necessary to achieve a better load balance between the processors of the parallel machine. On this behalf a conversion program offers an option to subdivide a prescribed number of the largest blocks of the original grid topology produced by the preprocessor. Of course the total calculation times grow for unreasonably large numbers of blocks e.g. for the MG calculation on 8 processors as shown in the last line of Table 3.

The following test compares the convergence of the Single-Grid method with the Multi–Grid method (both with the inner MG algorithm for pressure calculation) on a sequence of refined grids. The calculations are done for the bended channel with 64 blocks on 4 grids with $10 * 10 * 62$

(coarsest grid) up to $80*80*496$ finite volumes. Fig. 5 includes Single–Grid runs on the four grids (SG1 – SG4) and Multi–Grid runs starting on the two finest grids (MG3, MG4). N denotes the number of SG iterations or MG cycles. The curves show that the number of iterations for the SG method increases while the number of MG cycles remains constant if the grid is refined. The total computation time for the MG run on the finest grid was 10360s, note that a real calculation can be stopped at a higher error level.

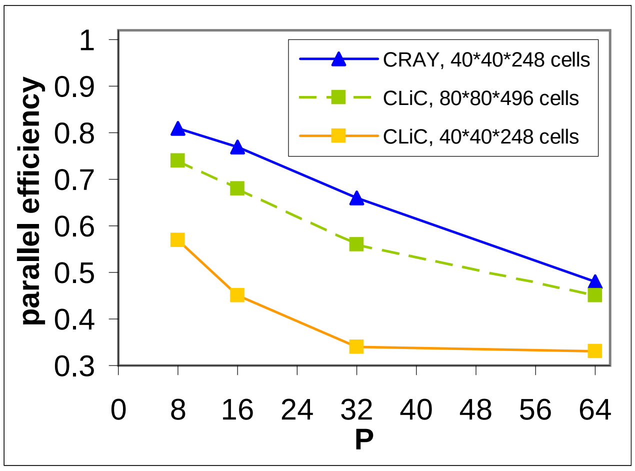


**Figure 6. Parallel Efficiency of the MG algorithm on CLiC and on a CRAY–T3E**

Fig. 6 summarizes the parallel efficiency of the MG method for a series of runs on an increasing number of processors. All calculations are performed for the bended duct with 64 blocks and $396\,800$ or $3\,174\,400$ finite volumes on the finest grid. Generally the parallel efficiency is better for the calculations on the CRAY-T3E. This is due to the faster communication network. Table 4 shows calculation times T_cal and the times needed for data exchange T_e for the runs with $396\,800$ finite volumes. While with a growing number of nodes the ratio T_e/T_cal remains constant on the CRAY it grows from 0.42 to 0.8 on the CLiC. On the CLiC the parallel efficiency is fairly good as long as the number of finite volumes per node is not too small.

The last Fig. 7 gives an impression of the flow through the bended duct. The picture in the lower part shows the pressure distribution in the whole channel (the flow direction is from left to right). In the upper part of the figure there are shown two slices taken from the first bended section. The left slice again shows the pressure distribution with maximal values at the inner sides of the blades. In the corresponding distribution of absolute velocity values (right slice) the minimal values can be found at the inner sides of the blades.

| Number of nodes | | 8 | 16 | 32 | 64 |
|---|---|---|---|---|---|
| CRAY: | T_cal | 1661 | 863 | 499 | 349 |
| | T_e | 396 | 199 | 103 | 80 |
| | **T_e/T_cal** | **0.24** | **0.23** | **0.21** | **0.23** |
| CLiC: | T_cal | 1022 | 669 | 449 | 249 |
| | T_e | 434 | 400 | 330 | 200 |
| | **T_e/T_cal** | **0.42** | **0.60** | **0.73** | **0.80** |

**Table 4. Calculation times and data exchange times for an increasing number of nodes**

## 7 Concluding Remarks

The presented study shows that the Multi–Grid method is very efficient compared with the Single–Grid SIMPLE method. This holds both for its serial and for its parallel implementation. The calculated flow fields can be used to calculate the disperse phase of the multiphase flows under consideration. Further investigations will point to the coupling of the continous with the disperse phase, to nonsteady calculations and to the implementation of alternative turbulence models.

## Acknowledgment

The authors are indebted to Prof. M. Perić for allowing the use of his laminar CFD code FAN–3D, which was the starting point for the presented code Mistral/PartFlow–3D. Further this work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft – DFG) in the framework of the Collaborative Research Centre SFB–393 under Contract No. SFB 393/D2.

## References

[1] Bernert, K.: $\tau$-Extrapolation – Theoretical Foundation, Numerical Experiment and Application to Navier-Stokes Equations, *SIAM J. Sci. Comput.* Vol. 18, No. 2 (1997)

[2] Brandt, A.; Dinar, N.: Multigrid solutions to elliptic flow problems, in: *Numerical methods for partial differential equations* (ed. by S. v. Parter), Academic Press, New York London Toronto Sydney San Francisco, 1979, S. 53-147

[3] Caretto, L.S., Gosman, A.,D., Patankar, S.V., Spalding, D.B.: Two calculation procedures for steady, three-dimensionbal flows with recirculation, *Proc. Third Int. Conf. Numer. Methodes Fluid Dyn.*, Paris (1972)

[4] Ferzinger, J.H., Perić, M.: *Computational Methods for Fluid Dynamics*, Springer, Berlin Heidelberg (1996)

[5] Frank, Th., Wassen, E., Yu, Q.: A 3–dimensional Lagrangian Solver for Disperse Multiphase Flows on Arbitrary, Geometricaly Complex Flow Domains using Block–structured Numerical Grids, *7th Int. Symposium on Gas-Particle Flows, ASME Fluids Engineering Division Summer Meeting*, Vancouver, BC, Canada, June 22–26, 1997, CD–ROM Proceedings, FEDSM97–3590

This picture takes too much memory.

**Figure 7. Bended duct with slices of the first bended section**

[6] Frank, Th., Wassen, E.: Parallel Efficiency of PVM– and MPI–Implementations of two Algorithms for the Lagrangian Prediction of Disperse Multiphase Flows, *JSME Centennial Grand Congress 1997, ISAC '97 Conference on Advanced Computing on Multiphase Flow*, Tokyo, Japan, July 18–19, 1997

[7] Frank, Th., Wassen, E., Yu, Q.: Lagrangian Prediction of Disperse Gas–Particle Flow in Cyclon Separators, *ICMF '98 – 3rd International Conference on Multiphase Flow 1998*, Lyon, France, June 8.–12., 1998, *CD–ROM Proceedings*, Paper No. 217, pp. 1–8

[8] Frank, Th., Schneider, J., Yu, Q. Wassen, E.: Experimental and Numerical Investigation of Particle Separation in a Symmetrical Double Cyclone Separator, *8th Int. Symposium on Gas–Particle Flows, ASME Fluids Engineering Division Summer Meeting*, San Francisco, CA, U.S.A., July 18–22, 1999, *CD–ROM Proceedings*, Paper No. FEDSM99–7865, pp. 1–10

[9] Frank, Th., Bernert, K., Schneider, J.: Numerische Untersuchungen der Gas–Partikel–Strömung in symmetrischen Doppelzyklon–Abscheidern, *VDI Berichte*, Nr. 1511, 1999

[10] Frank, Th., Bernert, K., Scheider, H., Pachler, K.: Efficient Parallelization of Eulerian-Lagrangian Approach for Disperse Multiphase Flow Calculation on MIMD Computer Architectures, *IEEE Int. Conference on Cluster Computing CLUSTER 2000*, Nov. 28. – Dec. 2. 2000, Chemnitz University of Technology, Saxony, Germany

[11] Hackbusch, W.: *Multigrid methods and applications*, Springer–Verlag, Berlin 1985

[12] Rhie, C. M., Chow, W. L.: Numerical study of the turbulent flow past an airfoil with trailing edge separation, *AIAA J.* Vol. 21, 1525-1532, (1983)

[13] Stone, H.L: Iterative solution of implicit approximations of multidimensional partial differential equations, *SIAM J. Numer. Anal.*, 5, 530-558, (1968)

[14] Stüben, K.; Trottenberg, U.: Multigrid methods: fundamental algorithms, model problem analysis and applications, *Lecture Notes in Mathematics 960*, Springer–Verlag, Berlin 1982