ESPRIT Project #5404

GP MIMD

Extension of Workpackage 6

Final Report

30. March 1995

Activity :	6.2.3 — Disperse Multiphase Currents of Air Pollutants
Partner :	University of Chemnitz–Zwickau, Germany
Person in Charge :	Dr. Thomas Frank
Email :	Frank@IMech.TU-Chemnitz.DE
Phone :	+49 (371) 531 46 43
Fax :	+49 (371) 531 46 44
Status :	**O**ngoing

Effort (MM) :	Q1	Q2	Q3	Q4	Q5	
	—	3	3	3	—	scheduled
	—	3	3	3	3	actual

Due Date :	M09
Actual Completion :	M15

Max. # processors :	64
Speedup achieved :	53.8
Efficiency achieved :	84.2 %

Abstract : In this activity of the GP MIMD project different parallelization methods for numerical algorithms for the prediction of disperse two–phase flows (gas–droplet and gas–particle flows) were investigated. The numerical algorithm was based on the Lagrangian approach. For the parallelization of the Navier–Stokes solver the method of grid partitioning developed by M. Perić [9] was used. For the parallelization of the Lagrangian trajectory solver 3 different parallelization methods were investigated. The different parallelization methods were evaluated and compared to each other. The parallelization was based on EXPRESS and PVM message passing libraries.

1 Objective

In many engineering flow situations particulate two–phase flows play an important role. The motion of particles or droplets in a turbulent flow has been studied theoretically, numerically and experimentally for more than 40 years. In the last decade the modelling of two–phase flows has been performed in several different ways. The continuous phase is usually predicted from an Eulerian approach and the behaviour of discrete particles is predicted from either an Eulerian or a Lagrangian approach. The performances of each approach have been studied in detail in the literature by e.g. Durst et al. [3] and Crowe [2].

The objective of this work was to develop, test and validate a numerical algorithm for the prediction of particulate two–phase flows in particular for flow regimes in gas cleaning and spray drying facilities. An Eulerian/Lagrangian stochastic–deterministic (LSD–)model was incorporated in the FAN–2D (**F**low **A**nalysis, **N**umericaly) code developed by M. Perić and Ž. Lilek [10]. The Lagrangian particle trajectory solver was specially adapted to the non–orthogonal, boundary–fitted arbitrary numerical grids used by the Navier–Stokes solver and to the full multigrid solution scheme implemented in FAN–2D. Phase interaction was taken into account using the PSI–cell model developed by C.T. Crowe [1].

The main objective of this activity in the GP MIMD Project was the investigation of various parallelization methods for the Eulerian/Lagrangian approach for the prediction of disperse multiphase flows. For the parallelization of the Navier–Stokes solver the method of grid partitioning developed by M. Perić [9] was used. Grid partitioning methods were investigated in the past by many authors and so this parallelization algorithm was applied without greater changes. For the parallelization of the 2–dimensional Lagrangian trajectory solver 3 different parallelization methods were investigated. The main problem in the parallelization of the Lagrangian approach for disperse multiphase flows is the dependence of flow data of the particulate phase on the flow field data of the continuous phase. This can lead to a great amount of node communication during the calculation of particle trajectories, source terms due to phase interaction and mean characteristics of the disperse phase. The 3 different parallelization algorithms were evaluated and compared to each other. The parallelization was based on EXPRESS and PVM message passing libraries. As hardware platforms we used a FDDI–linked workstation cluster of 3 HP 735 with a minimum node memory of 80 Megabyte and (from M13 to M15) the Parsytec Power–GC with up to 128 processors and an amount of 32 Megabyte node memory.

2 Approach

2.1 Fundamental equations of fluid motion

The turbulent two–phase flow under consideration is described by assuming that the particulate phase is dilute, but the particle loading is appreciable. Inter–particle effects are neglected, but the effects of the particles on the gas flow are taken into account. The two–phase flow is statistically stationary, incompressible and isothermal. The gas phase has constant physical properties and is Newtonian. Under these assumptions the time–averaged form of the governing gas–phase equations can be cast into the following form of the general transport equation :

$$\begin{aligned}\frac{\partial}{\partial x}(\rho_F\, u_F\, \Phi) \;&+\; \frac{1}{r}\frac{\partial}{\partial r}(r\,\rho_F\, v_F \Phi) = \frac{\partial}{\partial x}\left(\Gamma\,\frac{\partial \Phi}{\partial x}\right) \\ &+\; \frac{1}{r}\frac{\partial}{\partial r}\left(r\,\Gamma\,\frac{\partial \Phi}{\partial r}\right) + S_\Phi + S_\Phi^P \end{aligned} \qquad (1)$$

where Φ stands for u_F, v_F, k and ε. The terms S_Φ and Γ represent the ”source” and the effective diffusion coefficient, respectively, and S_Φ^P represents the coupling through the particle–fluid interaction. This last term is calculated by solving the Lagrangian equation of particle motion. The continuity equation is obtained by setting $\Phi = 1$, $\Gamma = 0$.

For modelling of fluid turbulence the standard k–ε turbulence model together with isotropic eddy viscosity and standard model constants are used :

$$\begin{aligned} \mu_t &= C_\mu\,\rho_F\,\frac{k^2}{\varepsilon} \quad , \quad \Gamma_\Phi^t = \frac{\mu_t}{\sigma_\Phi} \\ S_k &= P_k - \rho_F\,\varepsilon \\ S_\varepsilon &= C_{\varepsilon_1}\,\frac{\varepsilon}{k}\,P_k - C_{\varepsilon_2}\,\rho_F\,\frac{\varepsilon^2}{k} \end{aligned} \qquad (2)$$

$$C_\mu = 0.09 \quad , \quad C_{\varepsilon_1} = 1.44 \quad , \quad C_{\varepsilon_2} = 1.92$$
$$\sigma_k = 1.0 \quad , \quad \sigma_\varepsilon = 1.33$$

where P_k is the rate of production of turbulence. The influence of particle motion on fluid turbulence characteristics was neglected ($S_k^P = S_\varepsilon^P = 0$).

2.2 Particle momentum source term

In the PSI-cell model [1, 2], the force excerted on a fluid control volume by a single particle is calculated from the residence time of a particle in the control

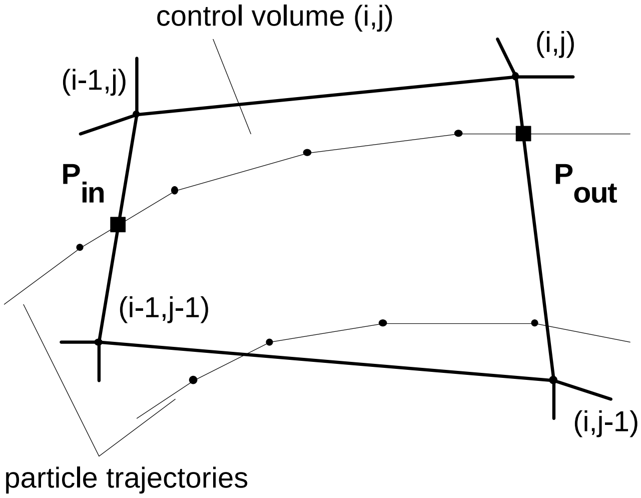

Figure 1: The PSI-cell model. Calculation of particle momentum source terms.

volume and the change in particle momentum in that time. In order to calculate the particle momentum source terms $S^P_{u_F}$ and $S^P_{v_F}$ in the momentum equations the points of intersection of the particle trajectory with the faces of the control volume have to be calculated and the particle and fluid properties have to be interpolated in this points (Fig. 1). The particle momentum source term is then as follows :

$$S^P_{u_{Fi}} = \sum \dot{N}_P \, \rho_F \, \frac{A_P}{2} \int_{t_{in}}^{t_{out}} C_D \, v_{rel} \, (u_{Fi} - u_{Pi}) \, dt \tag{3}$$

where the summation is taken over all the representative particles crossing the control volume. Because the number of test particles used for simulation is limited and different from that of particles which would actually cross the control volume, $\dot{N}_P$ characterises the particle flow rate for a calculated representative particle trajectory.

2.3 Solution procedure for the continuous phase

The above equations of fluid motion were solved with the FAN–2D program package developed by Perić and Lilek [10]. The code is designed for prediction of two-dimensional (plane or axisymmetric), laminar or turbulent, incompressible flows of Newtonian fluid in domains of arbitrary geometry. The numerical solution method employed in the code is based on the finite volume discretization of the governing equations. Characteristics of the method are : use of non–orthogonal, boundary fitted arbitrary numerical grids; colocated arrangement of variables on numerical grids; use of Cartesian vector and tensor components; segregated solution approach of SIMPLE kind [7]; full multigrid solving methodology using local bisectional grid refinement strategy [8].

The original computer code was extended by introduction of the particle momentum source terms in the momentum equations of fluid motion. Efficiency of the solution method was ensured by employing an optimized underrelaxation practice concerning not only the fluid variables but also the additional source terms.

2.4 Equations of motion of the dispersed phase

The disperse phase was treated by the Lagrangian approach where a large number of particles were followed in time along their trajectories through the flow domain. Each particle trajectory is assoziated with a particle flow rate $\dot{N}_P$ and so represents a number of real particles with the same physical properties. This representation is used in order to allow the consideration of the particle size distribution and to simulate the appropriate particle mass flow rate at the injection locations. The particle trajectories were determined by solving the ordinary differential equations for the particle location and velocity components. For the formulation of the particles equation of motion it was assumed that forces due to particle rotation, the pressure gradient in the flow, the added mass force and the Basset history force are negligible since a large density ratio ρ_P/ρ_F is considered. The equations of particle motion than can be written as follows :

$$\begin{aligned}
\frac{d\,x_P}{dt} &= u_P \quad , \quad \frac{d\,y_P}{dt} = v_P \\
\frac{d}{dt}\begin{bmatrix} u_P \\ v_P \end{bmatrix} &= \frac{3}{4}\frac{\nu\,\rho_F}{\rho_P\,d_P^2} Re_P\, C_D(Re_P) \begin{bmatrix} u_F - u_P \\ v_F - v_P \end{bmatrix} \\
&+ \frac{\rho_P - \rho_F}{\rho_P}\begin{bmatrix} -g \\ 0 \end{bmatrix} \qquad (4)
\end{aligned}$$

with :

$$\begin{aligned}
Re_P &= \frac{d_P\,v_{rel}}{\nu} \qquad (5) \\
v_{rel} &= \sqrt{(u_F - u_P)^2 + (v_F - v_P)^2} \qquad (6)
\end{aligned}$$

where d_P – particle diameter; u_P, v_P – particle velocity components in Cartesian coordinate system (x, y); C_D – coefficient of drag; g – gravity acceleration; ν – fluid kinematic viscosity; ρ – density of the fluid (F) and the particles (P) respectivly. The drag coefficient C_D is calculated as a function of Re_P using the correlations obtained by Morsi and Alexander [6].

The boundary conditions for the particle tracking procedure are specified as follows. Trajectories are calculated until the particle leaves the flow domain through a inlet or outlet cross section. Particles leaving the computational domain at the symmetry line ($y = 0$) are replaced by particles entering the domain with opposit radial velocity. For the particle–wall interaction reflection with a restitution coefficient k and a coefficient of kinetic friction f is assumed.

2.5 Solution procedure for the coupled system of equations

The equations of motion of the dispersed phase were solved using a standard Runge–Kutta solution scheme of 4th order accuracy with automatic time step correction. In order to ensure sufficient resolution of the influence of fluid flow turbulence on the particle motion the time step Δt was limited to a maximum of 1/10 the Lagrangian time scale T_L of the generated eddy.

The numerical procedure to obtain a converged solution for both phases is than as follows :

1. A converged solution of the gas flow field was calculated without source terms of the dispersed phase.
2. A large number of particles were traced through the flow field, and the values of the source terms were calculated for all control volumes of the numerical grid.
3. The flow field was recalculated by considering the source terms of the dispersed phase, where appropriate underrelaxation factors were considered.
4. Repetition of steps 2 and 3 until convergence was reached.

2.6 Parallelization of the Lagrangian particle trajectory solver

First efforts on parallization of the Lagrangian solver were based on a FDDI–linked workstation cluster of 3 HP 735 with 80 Megabyte of node memory running under EXPRESS and PVM message passing libraries.

Method 1 :

We introduced a host–node or divide–and–conquer parallelization scheme where the host generates the starting locations of the dispersed particles within the flow domain, checks their location on the numerical grid and distributes them to the nodes for trajectory and source term calculation. The nodes are calculating trajectories, the corresponding contributions to the overall source terms and mean

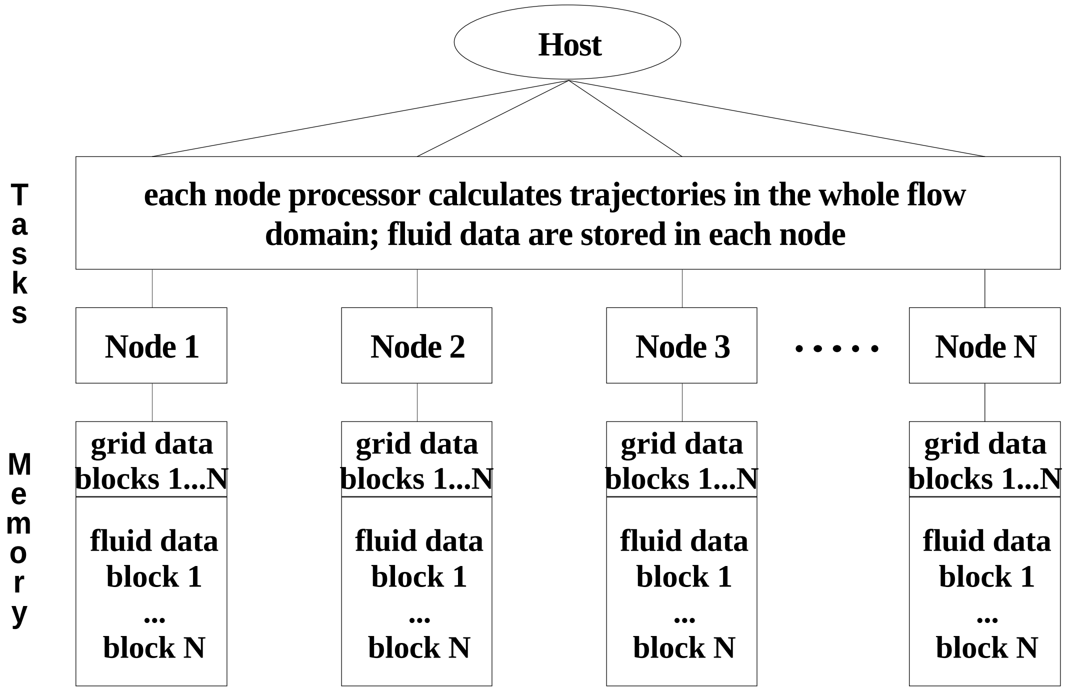
Host
T a s k s
each node processor calculates trajectories in the whole flow domain; fluid data are stored in each node
Node 1
Node 2
Node 3
• • • • •
Node N
M e m o r y
grid data blocks 1...N
fluid data block 1 ... block N

Figure 2: Parallelization method 1 for the Lagrangian solver.

values of particle phase characteristics (e.g. volume concentration, mean particle velocity and mean particle diameter). After particle trajectory calculations the host is summing up the contributions to the source terms over all nodes and over all grid elements and submits this values to the Navier–Stokes solver for recalculation of the modified fluid flow. Because of the large amount of node memory on the workstation cluster it was possible to store the whole fields of fluid characteristics in each node. Therefor we have only a small amount of node communication due to the distribution of initial values and collection of source terms and mean values of the particulate phase. Load balancing is automatically established due to the large number of calculated particle trajectories in comparison to the number of processor nodes.

Method 2 :

For the implementation of the Lagrangian solver on a dedicated MIMD machine like the Parsytec Power–GC it was necessary to let the Lagrangian solver operate on a distributed set of fluid flow data because the node memory on such machines is rather limited and does not allow the storage of the whole fields of fluid characteristics in each processor node. In the second method we use the same assignment of processor nodes to the blocks of the numerical grid as

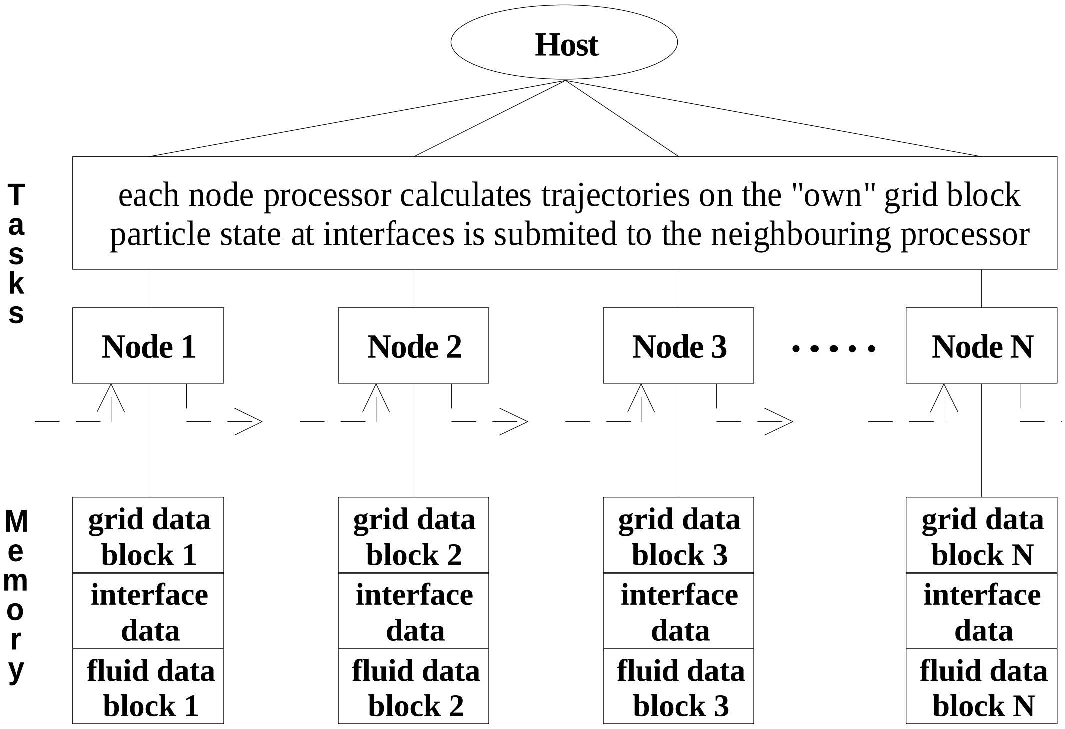

Figure 3: Parallelization method 2 for the Lagrangian solver.

used by the grid partitioning method for the Navier–Stokes solver. In each node are stored the fluid flow characteristics of the corresponding grid block. Now the processors are calculating particle trajectories from their entry point to the current grid block (from an inlet cross section or from a boundary to a neighbouring grid block) to the exit point (block boundary or outlet cross section). The amount of communication between nodes is very small because it is reduced to the delivery of the particle state to the neighbouring processor (grid block) in the case if a particle trajectory leaves the current block through a block boundary. The calculation of global sums over all processor nodes is no longer necessary because the contributions to the source term fields are calculated and stored at the right location during the calculation process. But load balancing can be a serious problem of that method. That can be illustrated by a simple example of a pipe or channel flow where grid blocks are arranged one behind the other along the pipe or channel axis. In this case all trajectories have to be calculated at first by the first procesor and so the calculation process only slowly propagates throughout the parallel machine. The same situation can be observed to the end of the calculation process for the last processor at the end of the pipe or channel. Similiar situations of poor load balancing can occur for flows around nozzles, recirculating and highly separated flows where most of the numerical effort has to be performed by a small subset of all processors used.

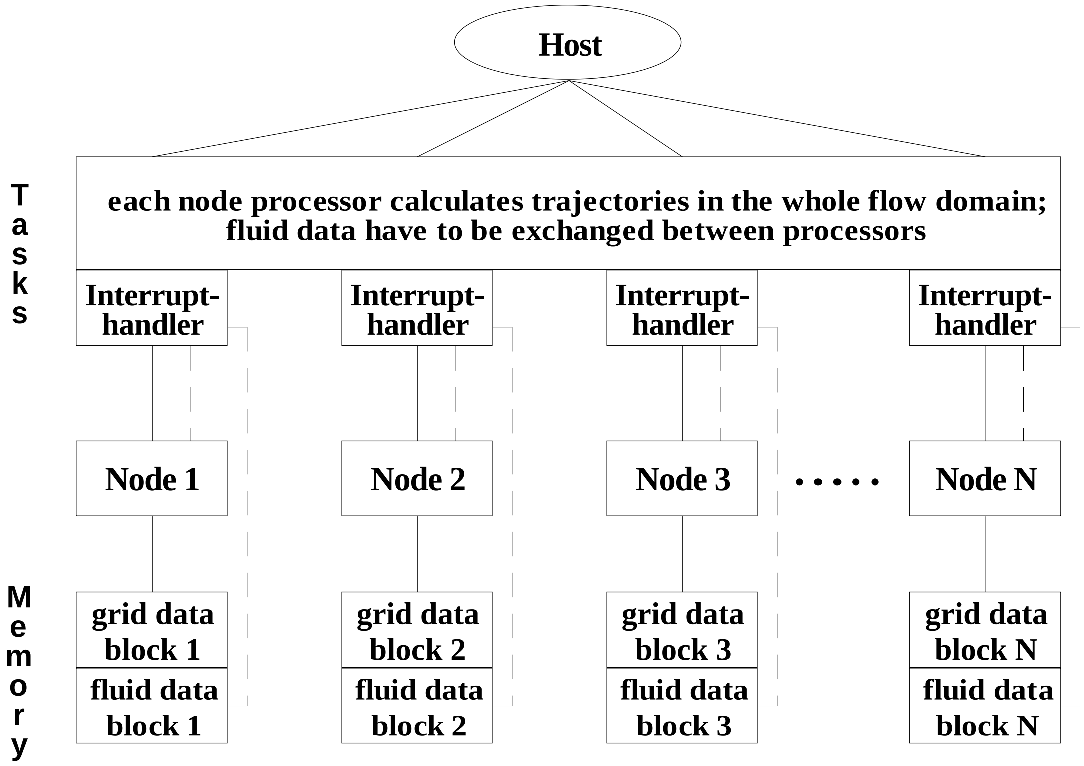

Figure 4: Parallelization method 3 for the Lagrangian solver.

Method 3 :

The third investigated parallelization method again uses the same distribution of fluid flow data over the nodes of the parallel machine as in the grid partitioning method of the Navier–Stokes solver. But in contrast to the second method a processor node calculates a particle trajectory from its entry point to the flow domain to its final exit location at an outlet cross section. While the particle is moving in the processors "own" grid block fluid flow data needed for the particle trajectory calculation can be taken from the processors node memory. If the trajectory leaves this grid block, fluid flow data have to be made available by node communication. In order to disturbe the work of the other processors as less as possible a feature of the EXPRESS library is used called message induced procedure calls or interrupt messages (EXHANDLE). Using this functionality of the EXPRESS library it is possible to implement an efficient handling of fluid flow data which are stored in the node memory of other processor nodes. Sending a message with a special message tag and the coordinates of the needed fluid flow data to the processor with the appropriate grid block number starts at that processor a message induced procedure with the same adress space like the main routine on that processor. This procedure can read the needed fluid flow data from the local node memory and sends them back to the calling processor. This algorithm can be enhanced by various caching and look–forward algorithms for the transfered fluid flow data. A similiar algorithm can be used for the calculation and distributed storage of the global source terms due to phase interaction and for the mean characteristics of the disperse phase.

This method requires a larger amount of node communication than the first two methods, but it was found to work very efficient. It operates with distributed fluid flow data and therefor needs the same amount of node memory as the grid partitioning algorithm of the Navier–Stokes solver. And this method has automaticaly a good load balancing due to the large number of calculated particle trajectories.

3 Results

Most of the implementation effort and most investigations and performance evaluations for the 3 described parallelization methods were performed on the HP 735 workstation cluster. Only after January 1995 the Parsytec Power–GC with Power–PVM for PARIX 1.2–PPC (a subset of PVM 3.2 which excludes support for heterogeneous platforms) became available for program implementation and testing. Therefor not all of the program development could be transfered to the dedicated parallel machine in the project period.

3.1 Results and application

During the project period the three described parallelization methods for the Lagrangian particle trajectory solver were implemented on the basis of the EXPRESS message passing library. In the last months of the project period method 1 was implemented on the Parsytec Power–GC using PVM — the only availabel high–level communication standard on this machine at the present time. This latest version of the program was also the basis for first performance and scalability evaluations. All versions of the program were used for calculations of several test cases of gas–particle and gas–droplet flows in flow configurations which are relevant for technical applications.

The comparison of the 3 methods has shown minor differences between methods 1 and 3. The increase in node communication in method 3 due to distributed storage of fluid flow characteristics has led only to a slight increase in execution time and a slight decrease of the parallel efficiency (for 4 to 16 node processes) in comparison with method 1. As allready mentioned in the previous section method 2 shows a strong dependence of its efficiency on the flow regime of the

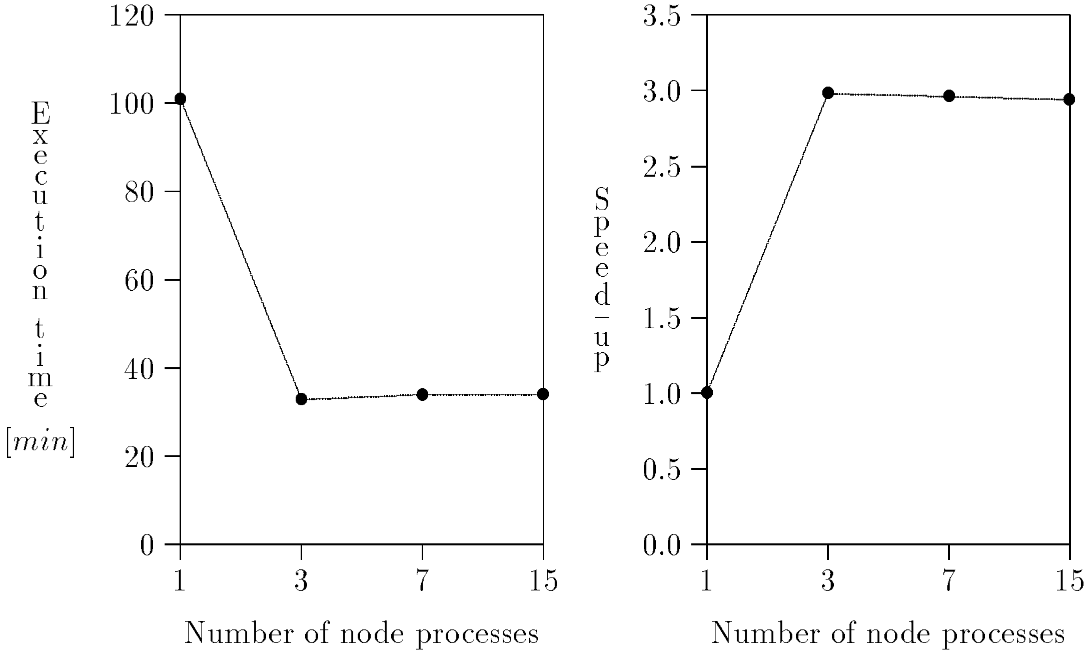

Figure 5: Execution time and speed–up for execution on a HP–workstation cluster.

investigated multiphase flow. Separation and inhomogeneous volume concentration of the dispersed phase leads to a decrease in parallel efficiency due to load balancing problems.

3.2 Evaluation of the performance

For the evaluation of the performance of the algorithm the time of one iteration cycle of the Lagrangian solver was measured using the TIME utillity of the underlaying UNIX operation system (HP–UX, PARIX). Therefor the measured execution time includes the start–up and stopping period of the message passing system and all I/O–processes which are necessary for the submission of the pre–calculated fluid flow data to the processor nodes of the (virtual) parallel machine. Experiments have shown that often in scalability experiments over a wide range of node numbers (from 8 to 128) either the computational task is to big to compute it on the smallest number of processor nodes in a reasonable time or the ratio of start–up and I/O–process time to the real calculation time is unfavourable if the task was executed on the largest number of processor nodes possible.

Three different scalability and efficiency experiments are shown in the following figures (Fig. 5...7).

- In the first case the Lagrangian solver with method 1 and for a constant number of particle trajectories (5000) was executed on the network of 3

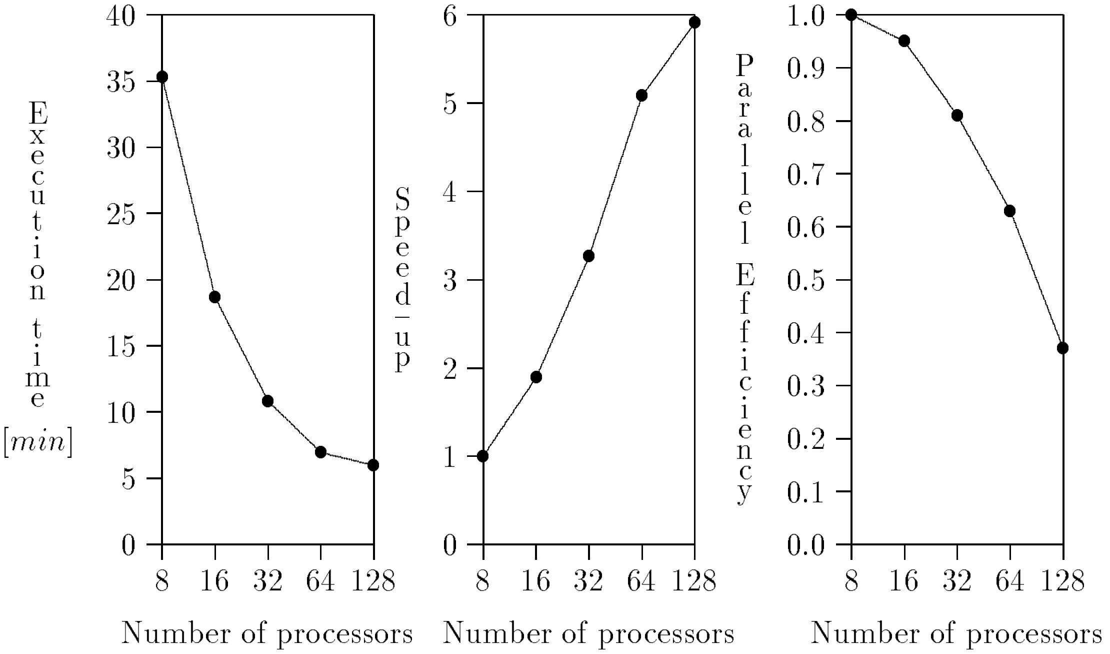

Figure 6: Performance results for execution on the Parsytec Power–GC 128 (5.000 particle trajectories in a test geometry).

HP 735 (HP–UX 9.05, EXPRESS 3.2) with a variable number of node programs. Fig. 5 shows the decrease in execution time and the nearly ideal speed–up of 2.95 for 3 node programs on 3 real processor nodes. For the increased number of node programs up to 15 on the same number of 3 real processors the diagrams show only a very slight increase in execution time what stands for a small increase in node communication with increasing number of processes (node programs).

- In the second case the Lagrangian solver was executed for the calculation of 5000 particle trajectories on the Parsytec Power–GC (PARIX, Power–PVM) using from 8 to 128 processor nodes of the parallel machine. Fig. 6 shows good speedup results for up to 32 processors and acceptable speed–up for 64 processors. Further increase in processor number gives no further speed–up of the program. For 128 processors most of the measured execution time was used for system start–up time and process I/O. Also it seems that the Parsytec Power–GC has a bottleneck for node communication between the upper and the lower 64 processor partition which contributes to the worse results for the execution of the program on 128 processors. This has to be further investigated in the future.

- In order to reduce the influence of start–up time and process I/O on the performance measurements a third experiment with the calculation of 20000

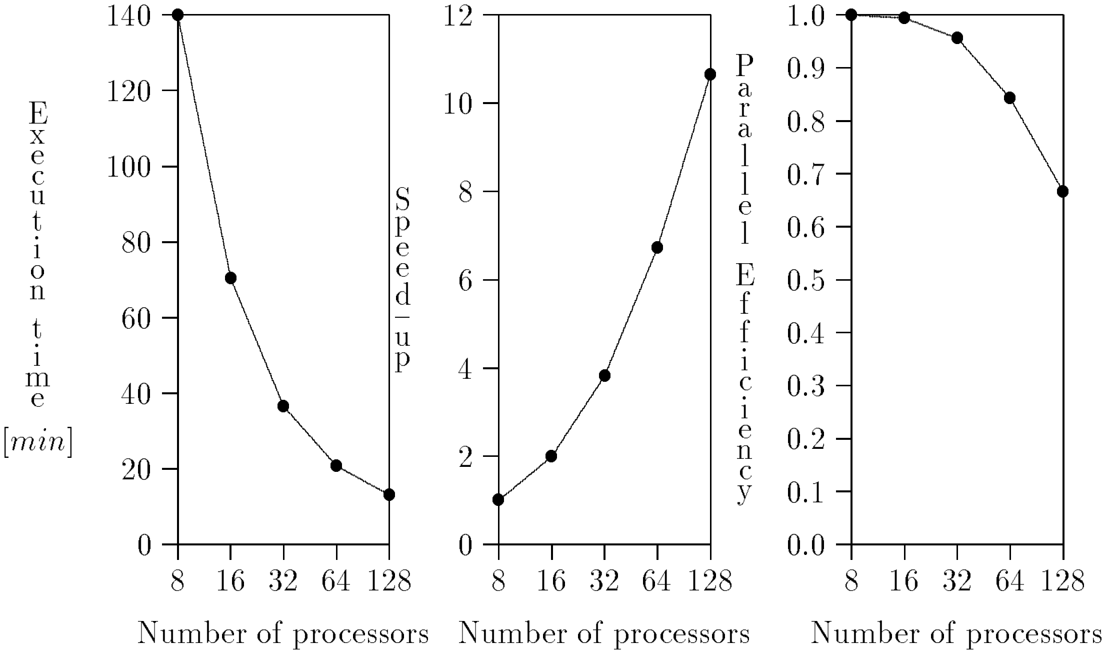

Figure 7: Performance results for execution on the Parsytec Power–GC 128 (20.000 particle trajectories in a test geometry).

particle trajectories on the same test geometry was carried out. This experiment (Fig. 7) shows improved speed–up and parallel efficiency especially for execution on 64 and 128 processor nodes.

4 Further investigations

The main objective of further investigations and development will be the implementation of the third parallelization method for the Lagrangian approach on a dedicated parallel machine. This implementation can be based either on a subset of the EXPRESS communication library running on the PARIX operation system or on a adequate implementation of the algorithm under Power–PVM using multiple threads per processor node. After optimization of the node communication (asynchronous communication, cache sizes,...) we expect comparabel performance and speed–up results.

References

[1] Crowe C.T., Sharma M.P., Stock D.E., 1977, "The Particle–Source–In Cell (PSI–Cell) Model for Gas–Droplet Flows", Trans. of ASME, J. Fluids Eng., Vol. 99, pp. 325–332.

[2] Crowe C.T., 1982, "REVIEW — Numerical Models for dilute Gas–Particle Flows", Trans. of ASME, J. Fluids Eng., Vol. 104, pp. 297–303.

[3] Durst F., Milojevic D., Schönung B., 1983, "Eulerian and Lagrangian Predictions of Particulate Two–Phase Flows — A Numerical Study", SFB 80/T/222, Preprint of the Sonderforschungsbereich 80, Ausbreitungs– und Transportvorgänge in Strömungen, University Karlsruhe, Germany.

[4] Frank Th., Schulze I., 1994, "Ein numerisches Verfahren zur Berechnung disperser Mehrphasenströmungen auf parallelen Hochleistungsrechnern", Arbeitssitzung des GVC–Fachausschusses Mehrphasenströmungen, 17./18. Februar 1994, Würzburg, Germany.

[5] Frank Th., Schulze I., 1994, "Numerical simulation of gas–droplet flow around a nozzle in a cylindrical chamber using Lagrangian model based on a multigrid Navier–Stokes solver", International Symposium on Numerical Methods for Multiphase Flows, 19.–23. Juni 1994, Lake Tahoe (NV), USA.

[6] Morsi S.A., Alexander A.J., 1972, "An Investigation of Particle Trajectories in Two–Phase Flow Systems", J. Fluid Mech., Vol. 55, part 2, pp. 193–208.

[7] Patankar S.V., 1980, "Numerical Heat Transfer and Fluid Flow", McGraw–Hill, New York.

[8] Perić M., 1989, "A Finite Volume Multigrid Method for Calculating Turbulent Flows", Proc. 7th Symposium on Turbulent Shear Flows, Vol. 1, pp. 7.3.1.–7.3.6., Stanford University, USA.

[9] Perić M., 1992, "Ein zum Parallelrechnen geeignetes Finite–Volumen–Mehrgitterverfahren zur Berechnung komplexer Strömungen auf blockstrukturierten Gittern mit lokaler Verfeinerung", Abschlußbericht zum DFG–Vorhaben Pe 350/3–1 im DFG–Habilitandenstipendiumprogramm, Stanford University, USA.

[10] Perić M., Lilek Ž., 1993, "Users Manual for the FAN–2D Software for the Calculation of Incompressible Flows", Institut für Schiffbau der Universität Hamburg, Germany.