PARALLEL TURBULENCE SIMULATION ON THE IBM SP2 USING A SPARSE GRID METHOD

M. GRIEBEL AND W. HUBER*

Institut für Informatik, Technische Universität München
Arcisstraße 21, D-80290 München, Germany
e-mail: griebel/huberw@informatik.tu-muenchen.de

1. The problem. In the area of computational fluid dynamics (CFD), the study of turbulent, 3-dimensional flow of incompressible fluids is one of the most challenging tasks. To understand the arising phenomena and their interaction with walls and boundaries better, driven turbulence[1] in a (periodically closed) pipe or canal is studied frequently as a test problem. Its direct numerical simulation involves the solution of the 3-dimensional Navier-Stokes equations (NS) on relatively fine grids for a long period of observation time to gain a sufficient amount of flow field samples for a statistical evaluation. In the beginning, to generate a turbulent flow field at all, the numerical simulation must run over an extremely large number of time steps. Thus, any direct numerical simulation is very time-consuming and needs, at least for higher Reynolds numbers, an extreme amount of main memory and disk storage which is beyond most existing computers. Therefore, the simulation was limited to relatively small Reynolds numbers. Recently, with the appearance of new numerical methods and powerful parallel computers, the direct numerical simulation for higher Reynolds numbers become possible to some extent. In the following, we consider the fast parallel simulation of turbulent flow in a pipe for a Reynolds number of 6950.

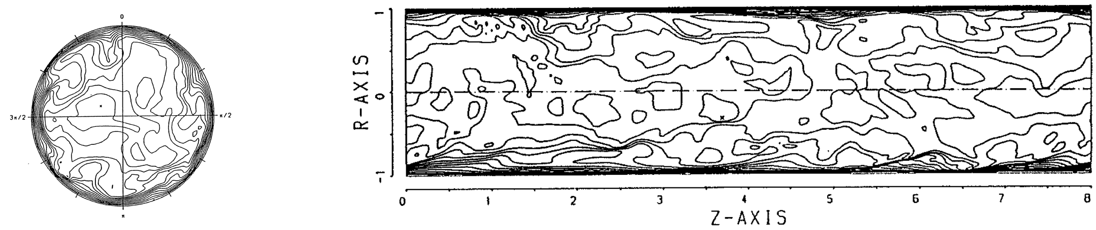

FIG. 1. *Cross sections of a turbulent flow field, contour plots.*

2. The numerical method and its parallelization. Usually, for the direct numerical simulation of driven turbulence, often a (more or less) uniform grid is used to discretize the Navier-Stokes equations with the finite difference or finite volume approach. The time discretization is explicit (Euler-Leapfrog) and

* supported by Deutsche Forschungsgemeinschaft, AZ Ze 281/3-3 and SFB 342 TP B3.
[1] in contrary to decaying turbulence, where no walls are present.

Chorin's projection method results in a Poisson equation for pressure correction at each time step. To cope with the pipe geometry, we use cylindrical coordinates.

Instead of the treatment of the Navier-Stokes equations on a conventional full grid, we apply a new technique, the *combination method*, which works on a so-called sparse grid only. It involves different regular grids $\Omega_{i,j,k}$ with different mesh width h_i, h_j, h_k in r-, φ- und z-direction and the linear combination of the corresponding solution iterates according to the formulae

$$u^c_{n,n,n} = \sum_{i+j+k=n+2} u_{i,j,k} - 2 \sum_{i+j+k=n+1} u_{i,j,k} + \sum_{i+j+k=n} u_{i,j,k}. \tag{1}$$

gives us a solution iterate on the sparse grid Ω^s_n, see Figure 2.

On each of the grids involved, the Navier-Stokes equations are treated explicitly for T successive time steps independently of each other. Then, the sparse grid iterate is assembled by the linear combination process (1) and, from that, the starting iterates for the next cycle of T time steps on the different grids are gained by projection, and so on. For the treatment of the iterates on these different grids, any existing explicit NS-code can be applied directly. Compare also Figure 3.

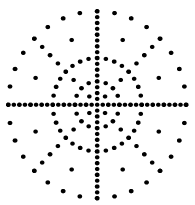

FIG. 2. *The sparse grid Ω^s_4, cross-section.*

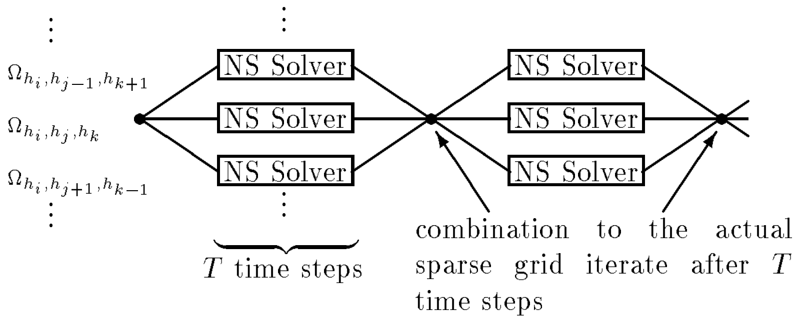

FIG. 3. *Integration of the combination method and the NS-solver.*

Thus, our method involves substantially less grid points (i.e. $O(N \cdot (\log N)^2$ with $n = \log N$) instead of $O(N^3)$ for the full grid discretization) but the accuracy of the obtained solutions is nearly the same (see [2, 4, 5, 6]) as for the conventional approach and is, for turbulent flow applications, within the range of the error of physical measurements anyway. To cope with the possibly extreme distorsion on some of the grids involved, we modified (1) in a way that certain grids are omitted and the combination of the solution iterates of only m grids takes place, see also [2]. Some results of the statistical evaluation (based on 20 samples) for the combination method and the DNS are given in Figure 4.

Furthermore, this method is perfectly suited for parallelization on a relatively coarse grain level involving only minimal communication (see [1, 3]) and allowing for a simple load balancing strategy. Thus, in each time step, the overall complexity (involving computing time, communication time and setup time) of the parallelized combination method is reduced to $O(N)$ in comparison to $O(N^3)$ for

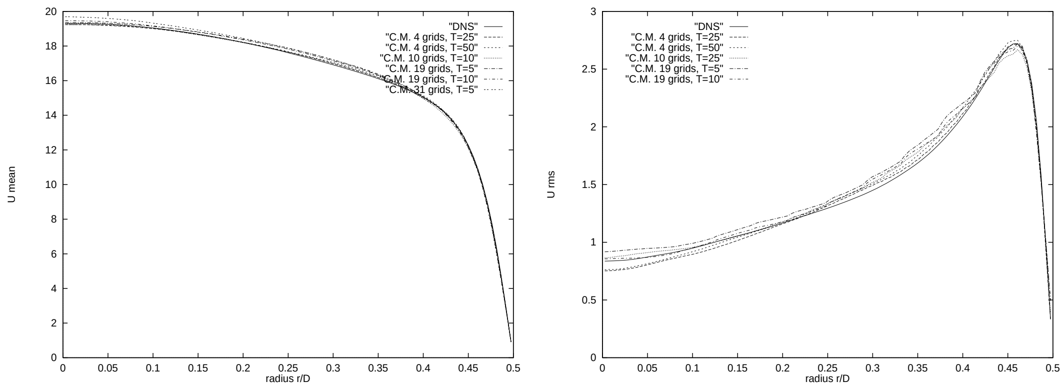

FIG. 4. *Statistical results, mean flow and rms-values of the u-velocity.*

the conventional approach in the sequential setting. To achieve this, $O((\log N)^2)$ processors are necessary.[2] [3]

3. Run time measurements, results and comparison. First, we give the results of our turbulence simulation obtained on the IBM SP2 of the Leibniz Computing Center in Munich. Table 1 gives the numbers of operations needed by different variants of our algorithm for 500 time steps (that means for one sample) with $T = 50$.

TABLE 1
Number of operations in megaflop for a varying number m of grids combined.

$m=$	1	4	10	19	31
	669 150	545 420	272 710	144 620	56 290

The parallel version of our code was implemented in FORTRAN using PVM and PVMe. For the setup time[4], the communication time and the computing time, we obtained the results given in Table 2.

From these results we see that a rate of about 250 megaflop per second was obtained on 19 processors. Note that for reasons of portability, our code was not specially optimized to the SP2. Thus, it runs on one processor with 43 megaflops. Furthermore, we observe that the *numerical* speedup obtained with the method *and* its parallelization is 28.18 (case $m = 31$). Therefore, the corresponding *numerical* efficiency is above 90 percent (for $m = 19$ it is even better)!

[2] Note that, in the parallelized version of the combination method, the sparse grid which would involve $O(N(\log N)^2)$ grid points is never assembled explicitly, but its relevant data are stored in a *distributed* way on the different grids which are associated to different processors. Thus, only necessary data is exchanged in parallel between the different processors resulting in a $O(N)$ communication time.

[3] Of course, it is possible to further parallelize the work for each subproblem arising in the combination method e.g. by conventional domain decomposition techniques.

[4] including the spawn time to unfold the PVMe processes and the time to read the starting data.

TABLE 2

Setup time S, communication time C and computing time W for 500 time steps in sec. using PVMe 3.2, varying m, number of processors $P = m$.

		$T = 10$	$T = 5$
m=P =	1	19	31
W	15 500	242	80
C	-	300	380
S	-	40	90
$W + C + T$	15 500	582	550

Table 3 gives the run times for fixed m but varying numbers of processors using PVM. We see that a speedup of four is obtained on 10 processors resulting in an efficiency of about 40 percent. A short comparison shows that PVMe performs better than PVM especially with respect to communication.

TABLE 3

Overall run time $W + C + S$ for 500 time steps in sec. for P processors using PVM 3.3.7.

P=	1	2	4	8	10	19	31
$m = 19$	4050	2520	1500	1200	1020	956	-
$m = 31$	2400	2040	1620	1260	1200	1185	1065

We also run our code on a vector computer (Cray Y-MP, 1 CPU) and a network of workstations (16 HP720, Ethernet, PVM 3.2.6). It turned out that our code (using 31 processors) performed on the SP2 in parallel by about a factor of 7.5 faster than the Cray Y-MP (1 CPU) and by about a factor of 20 faster than on the workstation network (16 processors).

REFERENCES

[1] M. GRIEBEL, *The combination technique for the sparse grid solution of PDEs on multiprocessor machines*, Parallel Processing Letters, 2 (1992), pp. 61–70.

[2] M. GRIEBEL AND W. HUBER, *Turbulence simulation on sparse grids using the combination method*, SFB Bericht 342/19/94 A, Institut für Informatik, TU München, October 1994.

[3] M. GRIEBEL, W. HUBER, T. STÖRTKUHL, AND C. ZENGER, *On the parallel solution of 3D PDEs on a network of workstations and on vector computers*, in Lecture Notes in Computer Science, Computer Architecture: Theory, Hardware, Software, Applications, A. Bode and M. DalCin, ed., Springer Verlag, 1993, pp. 276–291.

[4] M. GRIEBEL, M. SCHNEIDER, AND C. ZENGER, *A combination technique for the solution of sparse grid problems*, in Iterative Methods in Linear Algebra, P. de Groen and R. Beauwens, ed., IMACS, Elsevier, North Holland, 1992, pp. 263–281.

[5] M. GRIEBEL AND V. THURNER, *The efficient solution of fluid dynamics problems by the combination technique*, Int. J. Num. Meth. Heat and Fluid Flow, 5 (1995), pp. 251–269.

[6] C. ZENGER, *Sparse grids*, in Parallel Algorithms for Partial Differential Equations, Proceedings of the Sixth GAMM-Seminar, Kiel, January 19-21, 1990, W. Hackbusch, ed., Braunschweig, 1991, Vieweg Verlag.