Control and Cybernetics

vol. **25** (1996) No. 5

Performance of air pollution models on massively parallel computers

by

John Brown*, Per Christian Hansen, Jerzy Waśniewski**, Zahari Zlatev*****

* Pentum Group, Inc., 485 Central Ave., Mountain View, CA 94043, USA,
** UNI•C, Building 304, Technical University of Denmark,
DK-2800 Lyngby, Denmark,
*** National Environmental Research Institute,
Frederiksborgvej 399, DK-4000 Roskilde, Denmark

Abstract: To compare the performance and use of three massively parallel SIMD computers, we implemented a large air pollution model on the computers. Using a realistic large-scale model, we gain detailed insight about the performance of the three computers when used to solve large-scale scientific problems that involve several types of numerical computations. The computers considered in our study are the Connection Machines CM-200 and CM-5, and the MasPar MP-2216.

1. Introduction

Evaluation and comparison of supercomputer performance is an important topic, because of the large investment involved in acquiring a supercomputer. The evaluation can have different goals, depending on the circumstances in which the supercomputer is going to be used. For example, if the computer is mainly going to be used for "production use" in, say, fluid dynamics or optimization, then sheer performance is of uttermost importance, while ease of program development and tuning may have a lower priority. On the other hand, in a computing center where the supercomputer is used for more general purpose computing, aspects such as powerful software development tools, robust software libraries, and good average performance over various applications is important.

Recently, a large effort has been put into the development of benchmark packages; see, e.g., Bailey (1993), Hockney (1993), Steen (1991). Some benchmarks consist of computational kernels that cover a variety of applications, either by extracting important parts of existing large-scale application codes or by

writing kernels that "simulate" such applications. Such benchmarks illustrate—among other things—how fast standard software runs on the particular computer and how well the various operations are suited for the computer. A different approach is to concentrate on the most important processes in a particular large-scale application, and implement this (perhaps simplified) application code on various supercomputers.

The latter strategy is the one used in this paper. Our application comes from air pollution modeling and includes both PDEs, ODEs, and large sparse symmetric and non-symmetric systems of linear equations. Similar simplified application for air pollution problems can be found in Crowley (1968), Marchuk (1985), McRae, Goodin and Seinfeld (1984), Molenkampf (1968). This kind of approach is suited for comparing, say, implementation and tuning efforts. In addition, one also gets an impression of how data structures and data layouts interface from one computational kernel to another. The larger the variety of computational problems involved in the application, the more insight one usually gets.

We feel that the two approaches to supercomputer evaluation described above should be considered as supplementing each other. Each approach has its advantages and disadvantages, and together they shed light on many important aspects of the use of supercomputers.

The comparison that we describe in this paper was carried out in the period mid 1992 through early 1994 and involved the following computers, all of them massively parallel:

- Connection Machine CM-200 with 256 nodes
- Connection Machine CM-5 with 32 and 64 nodes
- MasPar MP 2216 with 16K 32-bit processors

The particular application code from air pollution modeling consists of the most essential modules from the Danish Eulerian Model which was developed in the period 1987–1992 at the National Environmental Research Institute, see for details Gery, Whitten, Killus and Dodge (1989), Zlatev, Christensen and Eliassen (1993), Zlatev, Christensen and Hov (1992). Our test program was originally developed for testing the efficiency and accuracy of the air pollution computer model, Zlatev and Waśniewski (1994), and it was found that this program would also suit as a good evaluation program for supercomputers, due to the variety of tasks performed by the code.

Our paper is organized as follows. In Section 2 we briefly describe the mathematical model underlying the air pollution code, and we outline the numerical methods in use. In Section 3 we summarize the computers used in the test. In Section 4 we comment on the implementation efforts, the difficulties, and the advantages that we found for these computers. In Section 5 we give the measured performance and computing times across the computers, with a breakdown—where possible—on computation and communication. Finally, we summarize our results in Section 6.

2. The mathematical model and its numerical treatment

The physical model that underlies our test program is long-range transport of air pollutants, and the corresponding mathematical model consists of a system of partial differential equations in two dimensions,

$$\frac{\partial c_i}{\partial t} = -\frac{\partial(uc_i)}{\partial x} - \frac{\partial(vc_i)}{\partial y} + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_i}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_i}{\partial y}\right)$$
$$+ E_i - (\kappa_{1i} + \kappa_{2i})c_i + Q_i(c_1, c_2, \ldots, c_q), \qquad i = 1, 2, \ldots, q. \tag{1}$$

The quantities that are involved in the mathematical model have the following meaning:

1. the concentrations are denoted by c_i,
2. u and v are wind velocities,
3. K_x and K_y are diffusion coefficients,
4. the sources are described by the functions E_i,
5. κ_{1i} and κ_{2i} are deposition coefficients,
6. the chemical reactions are described by $Q_i(c_1, c_2, \ldots, c_q)$.

The model is fully described and tested in Harrison, Zlatev and Ottley (1994), Zlatev, Christensen and Eliassen (1993), Zlatev, Christensen and Hov (1992). The particular chemical scheme used in the model is proposed in Gery, Whitten, Killus and Dodge (1989).

The number q of equations is equal to the number of species that are involved in the model and varies in different studies. In the simplest case, where sulphur pollution or nitrogen pollution is studied, $q = 2$. In more complicated studies where ozone and hydro-carbons are involved, q is greater. The largest number of equations used until now was $q = 35$ (but the physicists at the Danish Environmental Research Institute would like to increase this number if the model so obtained could be handled numerically).

The PDEs in (1) describe the five different physical processes: advection, diffusion, emission, deposition, and chemical reactions. The most important processes are advection, i.e., transport of pollutants, and the chemical reactions that take place in the atmosphere (these are normally considered together with the emissions). If these two processes can be treated efficiently it is relatively easy to attach the other processes, i.e., the deposition and the diffusion. Thus, the following test-problem is the most important part of the mathematical model

$$\frac{\partial c_i}{\partial t} = -(1-y)\frac{\partial(c_i)}{\partial x} - (x-1)\frac{\partial(c_i)}{\partial y} + E_i + Q_i(c_1, \ldots, c_{12}) \tag{2}$$

with

$$i = 1, \ldots, 12, \quad x \in [0, 2], \quad y \in [0, 2], \quad t \in [0, 2\pi] \tag{3}$$

Note that the wind velocities are given in a special way, which allows us to find the analytic solution in some important cases. The chemical scheme is

also simplified by reducing the species to 12, but all the numerically difficult reactions are retained, see Hov, Zlatev, Berkowicz, Elliassen and Prahm (1989). The chemical reactions are described by the functions Q_i which are of the form

$$Q_i(c_1, \ldots, c_{12}) = -\sum_{i=1}^{12} \alpha_i c_i + \sum_{i=1}^{12}\sum_{j=1}^{12} \beta_{ij} c_i c_j. \tag{4}$$

If the big model (1) is used in different simulations, then huge amount of input data is needed; the storage needed to keep the input files is several Gbytes. The emissions as well as all meteorological data needed in such simulations were prepared within the "European Monitoring and Evaluation Programme" (EMEP), a common European program for studying air pollution phenomena. In this study we will concentrate on the performance. Therefore the simplified model (2) will be used. The simplified model does not need special input data files (the input data are generated during the runs).

In the test program, which is a generalization of the test proposed by Crowley (1968) and Molenkampf (1968), the wind field is defined so that the trajectories are concentric circles with a centre that is identical with the centre of the space domain, i.e., $x = 1$ and $y = 1$. The angular wind velocity is constant such that a full rotation is performed after a time period of 2π. The initial concentrations are given by

$$c_i(x, y, 0) = \begin{cases} 100(1 - \bar{x}/r), & \bar{x} < r \\ 0, & \bar{x} \geq r \end{cases} \tag{5}$$

where

$$r = 0.25, \quad \bar{x} = \sqrt{(x - x_0)^2 + (y - y_0)^2}, \quad x_0 = 0.5, \quad y_0 = 1. \tag{6}$$

Regular grids are used in the space discretization. The grids are obtained by using square grids where the numbers of grid-points are:

$$N_x = N_y = 32, 64, 128, 256, 512. \tag{7}$$

Splitting is used to solve the PDEs; see, for example, Bagrinoskij and Godunov (1957), Marchuk (1985), McRae, Goodin and Seinfeld (1984). This means that each step of the iterative procedure for solving the PDEs takes the form

1. solve the advection part
2. smooth (in order to remove negative concentrations)
3. solve the chemical part

Thus, after splitting, two systems of ordinary differential equations (ODEs) are obtained; see Zlatev (1990), Zlatev, Christensen, Moth and Waśniewski (1991), Zlatev, Waśniewski, Venugopal and Moth (1993), Zlatev and Waśniewski (1994). The numerical solution of each system of ODEs leads to the solution of large systems of linear algebraic equations whose matrices are banded and very often also symmetric and positive definite. The size of each system of ODEs is

$N_x \times N_y \times 12$ equations. If, for example, $N_x = N_y = 32$, then the number of equations in each of these systems is 12,288.

Finite elements are used in the space discretization of the advection sub-model; the method is fully described in Zlatev (1995). The time-integration of the semi-discretized advection is carried out by predictor-corrector methods with several different correctors which are chosen so that the global time-integration scheme has better stability properties (Zlatev, 1985; 1995). It is important to emphasize here that other numerical methods can also be successfully applied in the space discretization of the advection sub-model: pseudospectral algorithms, finite differences, flux-corrected transport techniques, semi-Lagrangian algorithms and wavelets. Other time-integration schemes can be efficient if correctly applied (the leap-frog is, for example, very often used).

The numerical treatment of the chemical sub-model causes a lot of problems. The chemical schemes are normally very stiff. The use of classical ODE methods leads to very small time-steps. Therefore the so-called "Quasi-Steady-State-Algorithm" (abbreviated as QSSA) is very often used in large air pollution models. This algorithm as well as some other algorithms for the treatment of atmospheric chemistry sub-models are discussed in Zlatev (1995). The QSSA is used in the experiments in this paper. It should be mentioned here that the efficient treatment of the chemical sub-models of large air pollution models is still an open problem and much more efforts are needed in this field.

The results produced by a mathematical model must be reliable. Some checks of the reliability of the numerical algorithms can easily be performed by using the test-model (2) (in a real model such checks must be combined with comparisons of measured and calculated concentrations; Zlatev, 1995). The results in Fig. 1 are given in order to illustrate that the numerical methods used in the simplified model (2) are reliable (even when more species are attached to the chemical schemes, but when the time step is sufficiently small and when a fine grid, 96x96 grid, is used in the space discretization).

The plot in the upper left corner represents the initial distribution of the concentrations. The plot in the right upper corner gives the concentrations after one full rotation when only advection is used. The upper two plots must be identical. It is seen that the numerical method (here a pseudospectral method is used, but the results obtained by the finite elements are quite similar) is performing very well on this test.

The results obtained by using chemical transformations only are given in the lower left plot (the time interval is the same as the time-interval needed to perform a full rotation). The results obtained after a full rotation when both advection and chemical reactions are applied are given on the lower right plot. It is clear that the results on the two lower plots must be identical. It is seen that they are very similar, which indicates that the numerical methods perform rather well (again assuming both that the time-stepsize is sufficiently small and that a fine space grid is used).

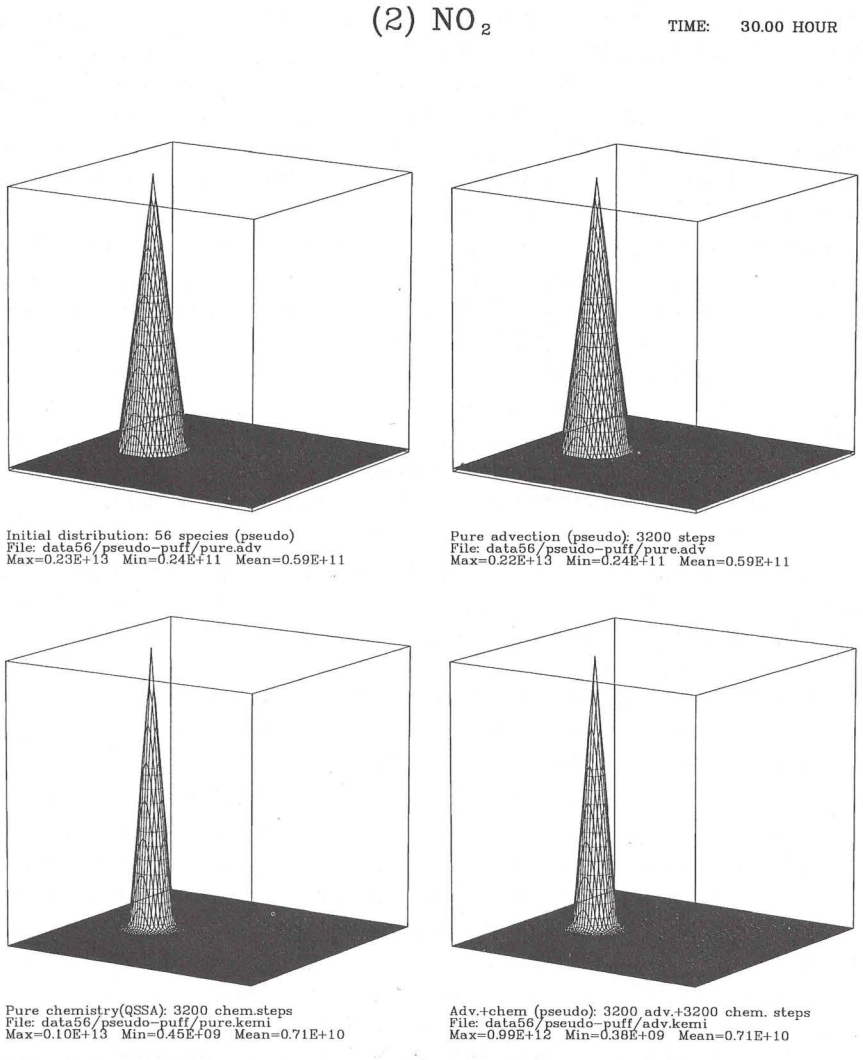

Figure 1. Distribution of the nitrogen di-oxide concentrations: (a) in the beginning (upper, left), (b) at the end of the pure advection test (upper, right), (c) at the end of the pure chemistry test (lower, left) and (d) at the end of the advection-chemistry test (lower, right).

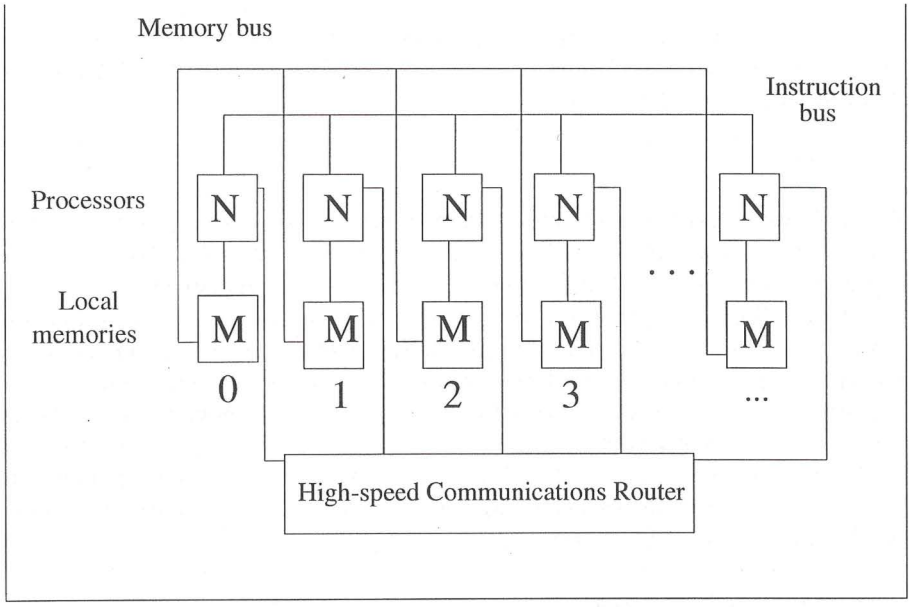

Figure 2. The "generic" hardware of the SIMD computers

3. Main features of the computers

All experiments described in this paper were performed using three SIMD computers: a CM-200 with 256 nodes, a CM-5 with 32 and 64 processor nodes, and a MasPar model 2216 with 16K 32-bit processors.

It should be mentioned here that the architecture of all these massively parallel computers can be illustrated by the "generic" figure shown in Fig. 2. The particular details and differences are described below.

We remark that the actual speed that one obtains is often far below the peak performance. The three main reasons sources to the degradation are:

1. *communication cost* when information must be sent to other processors
2. *reduction cost* when, for example, norms of vectors and/or matrices are calculated
3. *load imbalance* due to the fact that different processors have to carry out different calculations (e.g., when maskings or `WHERE` statements are used).

3.1. CM-200

The CM-200 is a massively parallel computer that contains up to 2,048 floating-point processor nodes that work synchronously. The particular computer used in the experiments contains 256 nodes. Each processor has 1 Gb of local memory and accesses other processors through a general purpose high-speed com-

munication router, which utilizes a hypercube communication network of an appropriate dimension. The theoretical peak performance of the CM-200 with 256 processors is 2.56 Gflops.

3.2. CM-5

A CM-5 system may contain hundreds or thousands of parallel processing nodes which may operate in both the SIMD and MIMD style of programming (in our experiments we use SIMD style). The CM-5 node makes use of industry-standard RISC microprocessor technology and may optionally be augmented with a special high-performance hardware arithmetic accelerator that uses wide data path, deep pipelines, and large register files to improve peak computational performance. Each node has 32 Mbytes of its own local memory and with the vector unit arithmetic accelerator it delivers up to 128 Mflops peak performance.

We ran our model on two CM-5 configurations with 32 processor nodes and 64 processor nodes, respectively, and with 4.096 and 8.192 Gflops theoretical peak performance, respectively.

3.3. MasPar MP 2216

The MasPar MP 2216 consists of a computational array of up to 16,384 processor elements (PEs). The PE array is controlled in a SIMD fashion from instructions issued by the Array Control Unit (ACU). Each PE is a 32-bit processor with an aggregate peak performance of 2 Gflops for 64 bit double precision, and has 64 Kbytes of local memory. Interprocessor communications can occur via two distinct communications network. The Xnet network connects processors in a Cartesian grid and allows communications to occur along horizontal and vertical directions, and this net is capable of over 20 GB/s communication rates. More random communications, as well as parallel I/O, are handled with the router network which connects any pair of processors using a three stage circuit-switched network. The peak bandwidth of the router is about 1 GB/s. The code was compiled with MPFORTRAN which supports the Fortran 90 language standard.

4. Porting the code to the parallel computers

The original test program was written in Fortran 77, and consists of about 2400 Fortran statements.

4.1. Connection Machines

To run the test program on the CM-200, the program had to be re-written in CM Fortran and tuned to the computer. CM Fortran uses the Fortran 90 array constructs to express parallelism. This required extensive modifications to convert the *do* loops to Fortran 90 array notation.

The resulting CM Fortran program from the CM-200 ran straightforwardly without any problems when it was moved to the CM-5. Additionally tuning was required to improve the performance.

On the Connection Machines, the PRISM system was used for performance measurements. This system was found to be very user friendly. It is possible to measure the computing time not only for the total subroutine, but for a single statement. One can measure the global CPU time, or one can measure various CPU times like CM CPU, FE TIME, and different communication CPU times (grid communication and reduction).

4.2. MasPar

MasPar Fortran is similar to Thinking Machines in using Fortran 90 array notation to express parallelism. To port the program to the MasPar, the CM Fortran 90 version was used as a starting point. While, in principle, a straightforward port, differences between the two vendor's implementation of Fortran 90 required fairly extensive modifications of the Connection Machine base.

By far, the biggest difference between the two vendors is in their choice of data layout conventions. The Thinking Machine compilers, by default, layout the last two dimensions of multidimension arrays across processors, while the MasPar compilers, by default, layout the first 2 dimensions across processors. For example:

- Connection Machine default data layout (CM-200 & CM5).

```
        REAL, ARRAY (NEQ, NX, NY) :: X
CMF$    LAYOUT X(:SERIAL, :NEWS, :NEWS)
```

- MasPar default data layout.

```
        REAL, ARRAY (NX, NY, NEQ) :: X
CMPF    MAP X(XBITS, YBITS, MEMORY)
```

  The first two co-ordinates go automatically to DPU (Data Parallel Unit) while the last one will become serial (in memory).

Both machines allow alternate layouts to be specified through the use of compiler directives. An example of these directives is shown in the previous code fragment. However, the use of these directives is fairly awkward and requires extensively using the Fortran 90 INTERFACE construct to pass compiler directives on the data layout conventions for subroutine calls. More importantly, using non-default data layouts also generally impacted performance. Therefore, it was chosen to remap the arrays to match the default layout that the MasPar compilers would use.

Another fairly extensive modification had to do with data communications. The fundamental data communication required is nearest neighbor access to grid point information. On both the Connection Machine and the MasPar, the default Fortran 90 data layouts will result in very efficient communications operations for finite difference operations for Cartesian grids. Fortran 90 offers

several alternative constructs to represent the array communications. Two common alternatives are the CSHIFT construct and the array subsection construct. In principle, either alternative should yield equivalent performance. In practice, the Thinking Machine compilers achieved better performance with the CSHIFT operation while the MasPar compilers achieved better performance with array subsection operations.

We can illustrate this with a matrix-vector multiplication example, where the matrix is tridiagonal. The operations can be written in the form

$$y_{k,i,j} = a^{(1)}_{i-1,j}\, x_{k,i-1,j} + a^{(2)}_{i,j}\, x_{k,i,j} + a^{(3)}_{i+1,j}\, x_{k,i+1,j},$$
$$k = 1, \ldots, N_{eq}, \quad i, j = 1, \ldots, N.$$

Let A1, A2 and A3 denote the arrays that hold $a^{(1)}_{i,j}$, $a^{(2)}_{i,j}$ and $a^{(3)}_{i,j}$, respectively.

- Connection Machine code (CM-200 & CM5).

```
      DO I = 1, NEQ
         B1(I,:,:)  = A1
         B2(I,:,:)  = A2
         B3(I,:,:)  = A3
      END DO
       Y = B1 * CSHIFT(X,DIM=2,SHIFT=-1) +
     1     B2 * X +
     2     B3 * CSHIFT(X,DIM=2,SHIFT=1)
```

- MasPar code.

```
      DO I = 1, NEQ
         B1(:,:,I) = A1
         B2(:,:,I) = A2
         B3(:,:,I) = A3
      END DO
      BC = X(:,:,:NEQ)
      Y = B2*BC
      Y(2:,:,:) = Y(2:,:,:) + B1(2:,:,:)*BC(:NX-1,:,:)
      Y(:NX-1,:,:) = Y(:NX-1,:,:) + B3(:NX-1,:,:)*
     *                BC(2:,:,:)
```

On the MasPar, profiling information is available on the subroutine and statement level for CPU costs for both front end and DPU costs. For final timings the system clock was used.

5. Performance results

The results given in Fig. 3 show that the total computing time is steadily increasing as a function of problem size. It is seen that the rate of increase is less when MP-2216 and CM-5 64pn are used.

The total amount of computations is broken up into three main parts:

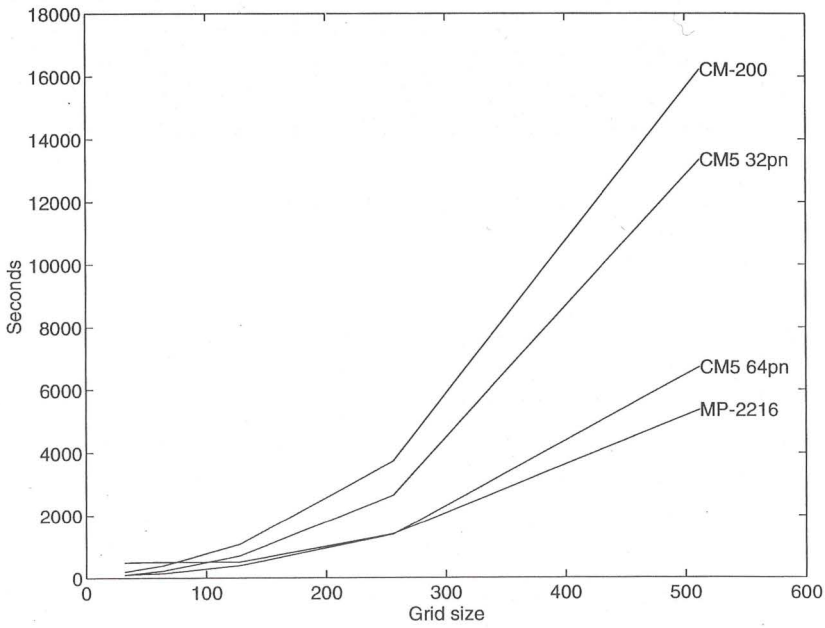

Figure 3. Total computing times of the four SIMD computers

Computer	Part of work	Grid size				
		32 × 32	64 × 64	128 × 128	256 × 256	512 × 512
CM-200 256 pn	Part 1	64	169	320	480	598
	Part 2	332	685	908	961	850
	Part 3	259	449	537	566	572
CM-5 32 pn	Part 1	120	233	297	361	380
	Part 2	1163	1915	2226	2326	1814
	Part 3	566	1004	1217	1265	1261
CM-5 64 pn	Part 1	118	366	537	686	755
	Part 2	1163	3256	4270	4590	3621
	Part 3	559	1600	2270	2497	2516
MP-2216	Part 1	40	171	690	1036	1325
	Part 2	56	226	901	1416	1713
	Part 3	62	244	979	1257	1265

Table 1. Speed of computations measured in MFLOPS. For all grids the number of time-steps is 3456.

Part 1 The space discretisation of the advection term.
Part 2 The time integration of the advection term.
Part 3 The chemical reactions with smoothing.

A breakdown of the total computing time into these three parts is given in Table 1. Below, we give more details about these three parts of the computations.

5.1. Part 1: The space discretization

The space discretization of the advection process i.e. the application of the finite elements method to discretize the partial derivatives in (2) (combined with the splitting procedure used this leads to a large system of ordinary differential equations), requires some communication. Therefore the speed of computations, measured in MFLOPS, is not very high for this part of the computational work—compare the results given in Fig. 4 with those in Fig. 6. The speed achieved on the MP-2216 is greatest.

The efficiency is measured by dividing the actually achieved MFLOPS with the top-performance reported by the architectures under consideration. For all tested machines, the efficiency increases with the problem size. The best efficiency achieved by the products of Thinking Machine Corporation is in general less than 20 % for this part of the computational work, see Fig. 5.

5.2. Part 2: The time integration

The time-integration of the system of ordinary differential equations obtained after the application of the finite elements method to discretize the space derivatives in (2) (see the previous paragraph) is a relatively easy part. There are no communications. Moreover, the loops are rather simple (without `WHERE` and `IF` statements). Therefore the speed of computations achieved in this part is greater than the speed of computations for the other two parts—compare the results given in Fig. 6 with the results given in Fig. 4 and Fig. 8.

If the grid is not very fine all machines seem to show a rather quick increase of the efficiency, see Fig. 7. The products of Thinking Machine Corporation are more efficient by at least 20 % for this problem. However, their efficiency starts to go down when the grid becomes very fine. The efficiency achieved on MP-2216 is monotonically increasing when the grid is refined (although the slope is not as steep as in the beginning).

The speed of computations is degraded for the products of Thinking Machine Corporation. However, the speed of computations for the two CM-5 machines remains faster than that of the other two machines (see Fig. 6). It is not very clear what is the reason for the somewhat strange behaviour for this part of the computational work. There may be some problems with page faults.

5.3. Part 3: The chemical reactions

The chemical part is extremely difficult for the numerical methods. However, if the numerical method is fixed, then it is normally rather easy to achieve high speed of computations on massively parallel machines. The main reason for this is the fact that there are no communications in this part. The CM-5 64pn is the fastest, see Fig. 8. If the efficiency is taken into account, then the MP-2216 is the best one for this part of the computational work, see Fig. 9.

6. Summary of results

We believe that our study of the SIMD computers has given a balanced view of the involved computers. In particular, it is clear that different numerical computations give rise to very different efficiencies on the computers.

A general conclusion is that it is relatively easy to achieve high speeds of computations when partial differential equations are discretized on simple space domain by the use of regular grids. One should therefore expect to obtain similar results for the numerical treatment of other problems arising in science and engineering when they are formulated by similar partial differential equations.

References

BAGRINOSKIJ, K.A. and GODUNOV, S.K. (1957) Difference schemes for multi-dimensional problems. *Dokl. Akad. Nauk, USSR*, **115**, 3, 431-433.

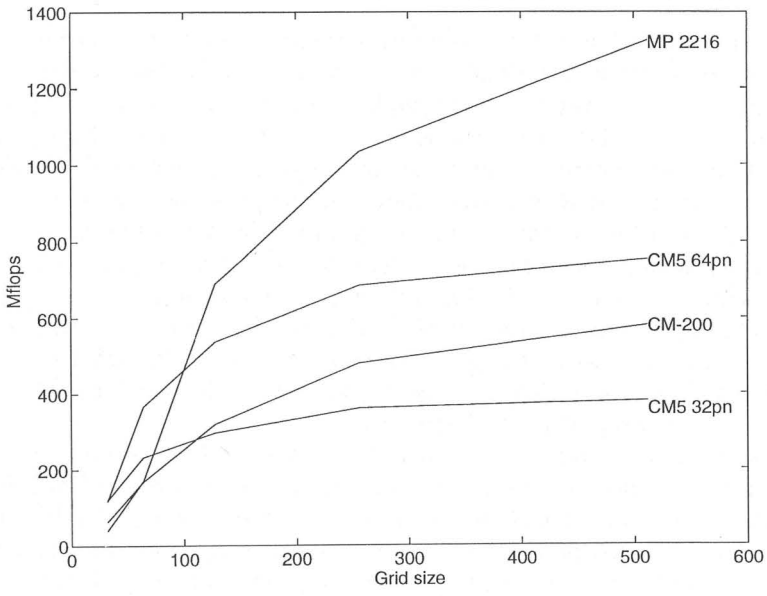

Figure 4. Mflops of Part 1

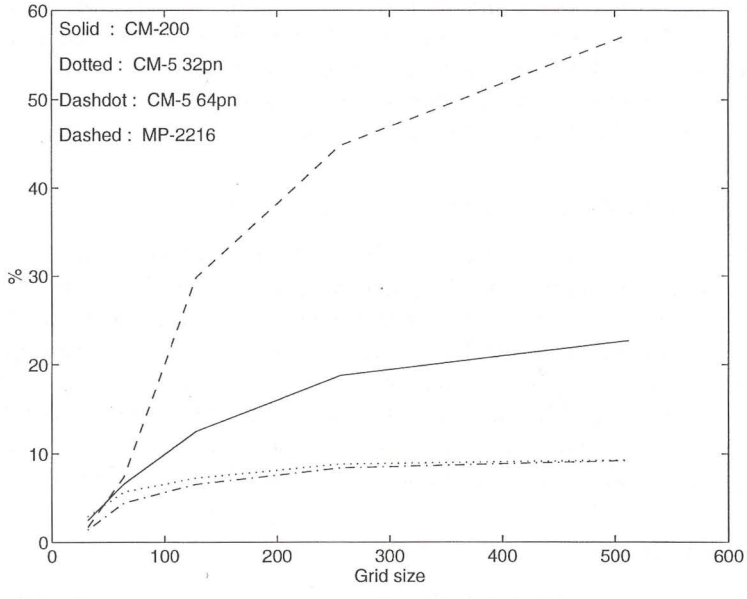

Figure 5. Efficiency achieved during Part 1

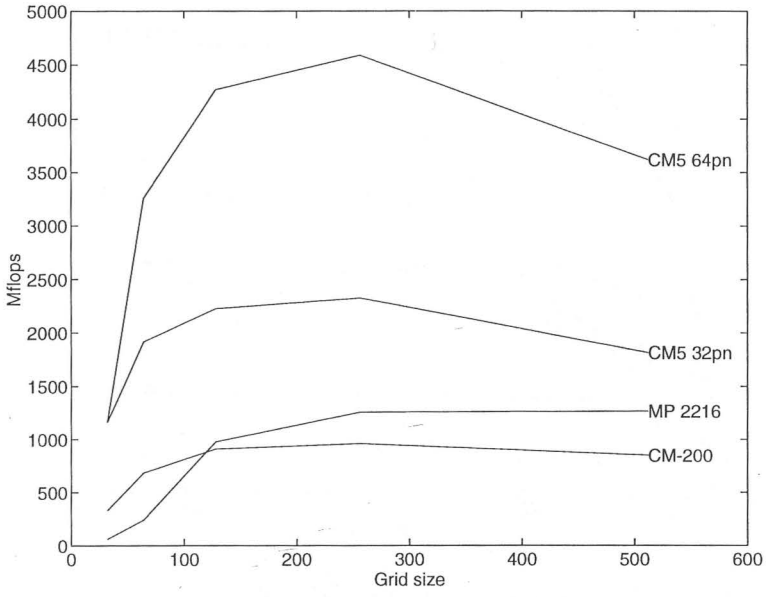

Figure 6. Mflops of Part 2

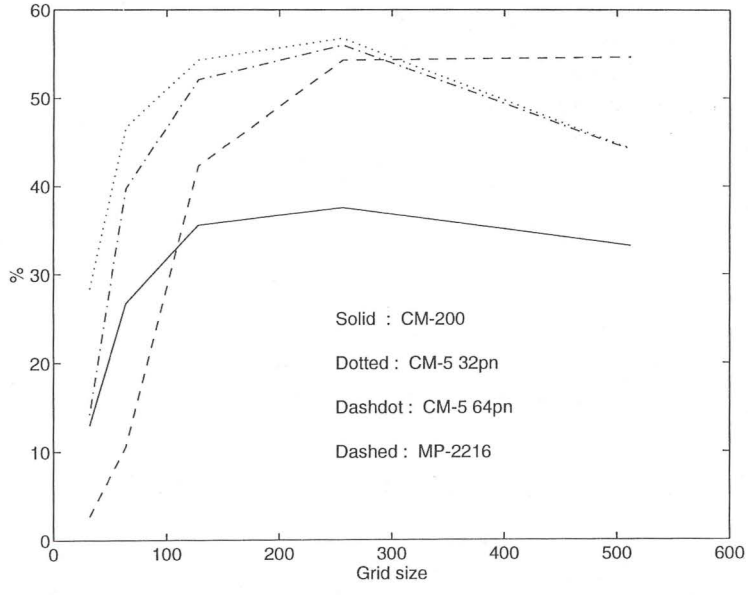

Figure 7. Efficiency achieved during Part 2

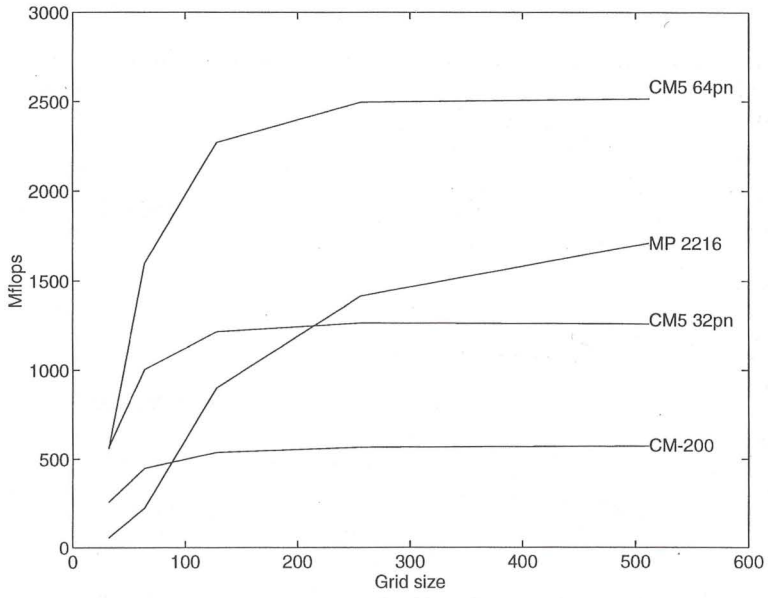

Figure 8. Mflops of Part 3.

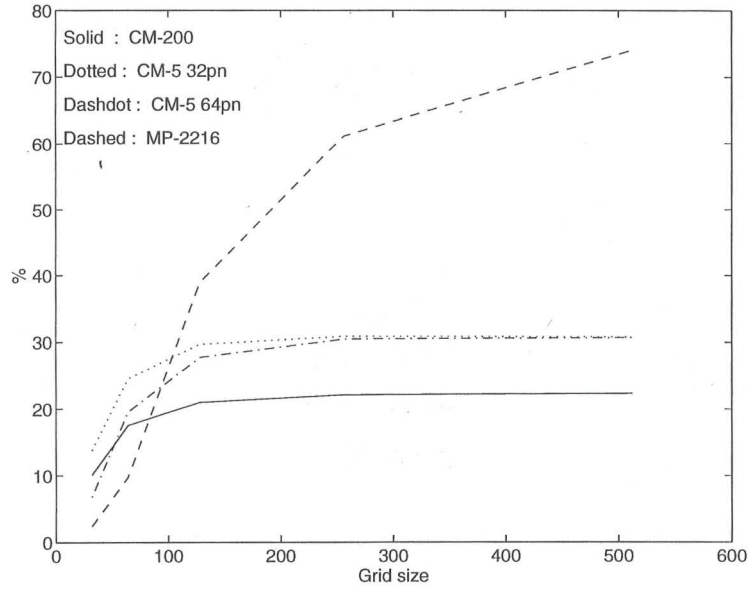

Figure 9. Efficiency achieved during Part 3

BAILEY, D.H. (1993) Experience with parallel computers at NASA Ames. *Int. J. High Speed Comput.*, **5**, 51–62.

CROWLEY, W.R. (1968) Numerical advection experiments. *Month. Weath. Rev.*, **96**, 1-11.

GERY, M.W., WHITTEN, G.Z., KILLUS, J.P. and DODGE, M.C. (1989) A photochemical kinetics mechanism for urban and regional computer modeling. *J. Geophys. Res.*, **94**, 12925-12956.

HARRISON, R., ZLATEV, Z. and OTTLEY, C.J. (1994) A comparison of the predictions of an Eulerian atmospheric transport-chemistry model with experimental measurements over the North Sea. *Atmospheric Environment*, **28**, 497-516.

HOCKNEY, R. & BERRY, M. (1993) *Public international benchmarks for parallel computers.* Report CS-93-213, Computer Science Department, Univ. of Tennessee, November 1993.

HOV, Ø., ZLATEV, Z., BERKOWICZ, R., ELLIASSEN, A. and PRAHM, L.P. (1989) Comparison of numerical techniques for use in air pollution models with non-linear chemical reactions. *Atmos. Environ.*, **23**, 967-983.

MARCHUK, G.I. (1985) Mathematical modeling for the problem of the environment. *Studies in Mathematics and Applications*, **16**, North-Holland, Amsterdam.

MCRAE, G.J., GOODIN, W.R. AND SEINFELD J.H. (1984) Numerical solution of the atmospheric diffusion equations for chemically reacting flows. *J. Comp. Phys.*, **45**, 1-42.

MOLENKAMPF C.R. (1968) Accuracy of finite difference schemes applied to advection equations. *J. Appl. Meteor.*, **7**, 160-167.

VAN DER STEEN A.J. (1991) The benchmark of the EuroBen group. *Parallel Computing*, **17**, 1211-1221.

ZLATEV, Z. (1980) Running large air pollution models on high speed computers. *Mathl. Comput. Modelling*, **14**, 737-740.

ZLATEV, Z. (1985) Application of predictor-corrector schemes with several correctors in solving air pollution problems. *BIT*, **24**, 700-715.

ZLATEV Z. (1995) *Computer treatment of large air pollution models.* Kluwer Academic Publishers, Dordrecht-Toronto-London.

ZLATEV, Z., CHRISTENSEN, J. and ELIASSEN, A. (1993) Studying high ozone concentrations by using the Danish Eulerian Model. *Atmospheric Environment*, **27A**, 845-865.

ZLATEV, Z., CHRISTENSEN, J. and HOV, Ø. (1992) An Eulerian air pollution model for Europe with nonlinear chemistry. *J. Atmos. Chem.*, **15**, 1-37.

ZLATEV, Z., CHRISTENSEN, J., MOTH, J. and WAŚNIEWSKI, J. (1991) Vectorising codes for studying long-range transport of air pollutants. *Mathematical and Computer Modelling*, **15**, 37-48.

ZLATEV, Z., DIMOV, I. and GEORGIEV, K. (1994) Modeling the long-range transport air pollutants. *Computational Science and Engineering*, **1**, 3, 45-52.

ZLATEV, Z. & WAŚNIEWSKI, J. (1994) Running air pollution models on the Connection Machine. *Mathematical and Computer Modelling*, **15**, 1-17.

ZLATEV, Z., WAŚNIEWSKI, J., VENUGOPAL, M. and MOTH, J. (1993) Optimising Air Pollution Models on Two Alliant Computers. In: *Parallel Computation*, A. E. Fincham and B. Ford, eds., 115-133. Clarendon Press, Oxford.