Parallel Spatial Direct Numerical Simulation of Boundary-Layer Flow Transition on IBM SP1 *

Ulf R. Hanebutte [1] **Ronald D. Joslin** [2]
Mohammad Zubair [3]

[1] Reactor Analysis Division
Argonne National Laboratory
Argonne, IL 60439

[2] NASA Langley Research Center
Hampton, VA 23681

[3] International Business Machines Corporation
Thomas J. Watson Research Center
Yorktown Heights, NY 10598

Abstract

The spatially evolving disturbances that are associated with laminar-to-turbulent transition in three-dimensional boundary-layer flows are computed with the PSDNS code on an IBM SP1 parallel supercomputer. By remapping the distributed data structure during the course of the calculation, optimized serial library routines can be utilized that substantially increase the computational performance. Although the remapping incurs a high communication penalty, the parallel efficiency of the code remains above 40 percent for all performed calculations. By using appropriate compile options and optimized library routines, the serial code achieves 52–56 Mflops on a single node of the SP1 (45 percent of theoretical peak performance). The actual performance of the PSDNS code on the SP1 is evaluated with a "real world" simulation that consists of 1.7 million grid points. Comparisons to the Cray Y/MP and Cray C-90 are made for this large scale simulation.

*This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480 while the first and third author were in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681.

1 Introduction

In a recent review article, Fischer and Patera [3] summarize current work in the area of parallel simulation of viscous incompressible flows. However, a discussion of parallel three-dimensional (3D) spatial direct numerical simulation (DNS) algorithms for laminar-to-turbulent transition (the subject of this paper) is not included in their review article.

This paper summarizes our research effort on the IBM SP1 as a follow-up to the work by Joslin and Zubair [8], in which the performance of the parallel spatial direct numerical simulation (PSDNS) code on the the relatively small and slow INTEL iPSC/860 computer was analyzed. A detailed report on the performance and scalability of the PSDNS code on the IBM SP1 is given in [5].

2 Parallel Computing Environment

The IBM SP1 [4] scalable parallel computer utilized in the presented performance study consists of 128 processing nodes. Each node is essentially an IBM RS/6000 model 370 workstation with a clock rate of 62.5 MHz. The local memory is 128 Mb, and the

processor data and instruction cache is 32 Kb each. The individual nodes are connected by a multistage network that consists of high-performance switches (50μsec latency, 8.5 Mb bandwidth); each switch can support up to 16 nodes. The peak performance obtained by performing one multiplication and one addition on 64-bit floating point numbers per clock cycle is 125 Mflops for each processing node. However, in practice, a FORTRAN code delivers 15–75 Mflops. Although the next-generation parallel computer from IBM, called the SP2 [11], is identical to the SP1 architecture, its node performance has more than doubled and the communication network bandwidth has increased fourfold. For the SP2, the increase in communication bandwidth relative to the computing performance will provide a better balanced system, which should further improve the performance results of the presented code. The access to Argonne's SP1 is controlled by a scheduler, which ensures that the requested node partition is operated in a dedicated mode.

3 Parallel Application

The physical domain of a boundary layer flow with an inflow disturbance is graphically shown in Figure 1. With the PSDNS code the spatially evolving disturbances that are associated with laminar-to-turbulent transition in three-dimensional boundary-layer flows are computed. The interested reader is referred to references [8] and [9] for algorithmic details of the spatial DNS code.

The PSDNS code developed by Joslin and Zubair [8] has been ported to the SP1 with only minor changes. The original parallel code is based on the message-passing paradigm with explicit data distribution, which enables good portability among a broad class of parallel computers. In the PSDNS code, the data are distributed among the n_p processors in block form with a z-mapping. That is, the 3D data are partitioned into n_p blocks that contain n_z/n_p two-dimensional (2D) planes of $n_x n_y$ data items each. To perform local FFT's in the spanwise direction n_z, the data must be remapped. As indicated in Figure 2, an x-mapping allows the utilization of optimized serial FFT library routines [7] in the z direction. The INTEL implementation of the PSDNS code relies on the *xor* algorithm [2] for the global data exchange; the IBM implementation makes use of a global index routine provided by the AIX-parallel environment [6]. As shown in the study by Joslin and Zubair, a significant performance gain can be achieved by utilizing a machine-specific basic linear algebra subprogram (BLAS) level 3 routine for the matrix by matrix multiplication. Because this routine is also available on the IBM as part of the ESSL library [7], the advantage can also be taken in the present implementation. The performance of the application code is further improved through appropriate selection of the compile options. As a result, the run time of the serial code can be reduced by a factor of 2.3 compared with a compilation without any options. For a small test problem (for which Joslin and Zubair [8] obtained 189 Mflops on the Cray Y/MP and 5 Mflops on a single node of the iPSC/860), a single node of the SP1 delivers 52.5 Mflops for the double-precision (i.e., 64-bit) computation.

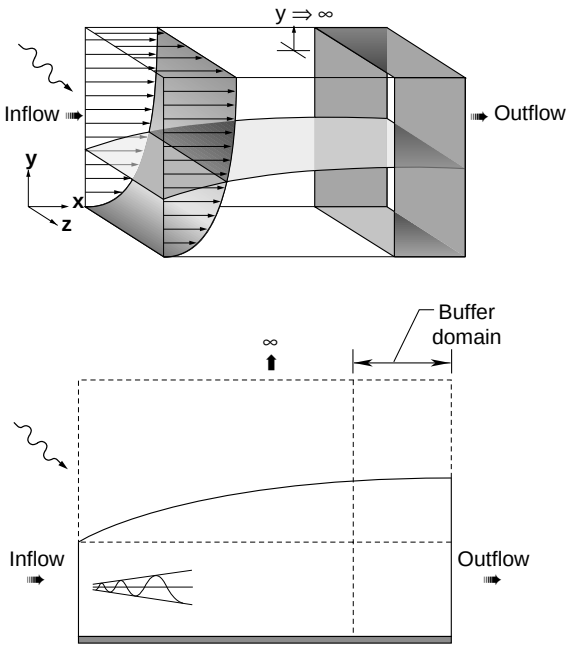

Figure 1: Physical Domain.

4 Performance Study

Although thousands of time steps are required for a single simulation, performance figures for only one time step of the PSDNS code are presented here. Performance figures for one time step are sufficient because the workload for each time step is constant throughout the simulation.

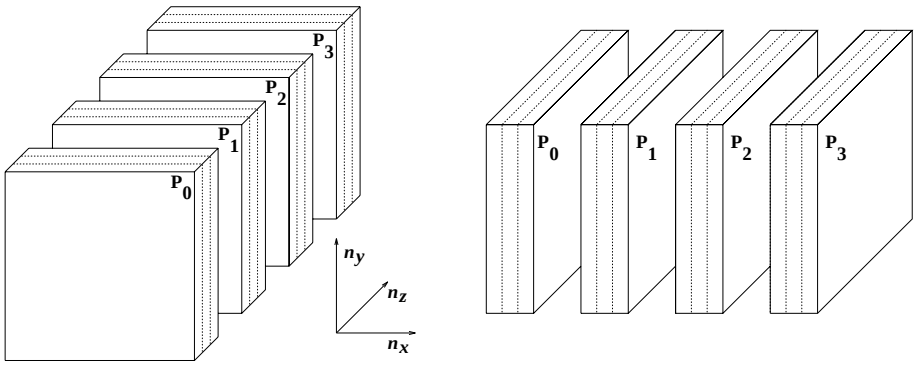

Figure 2: Global remapping results in local FFT's in spanwise direction.

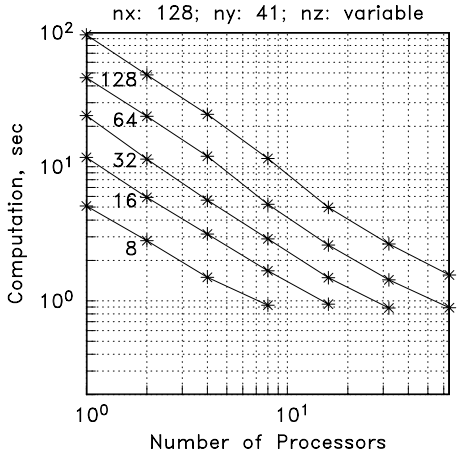

Figure 3: Computational costs for test suite.

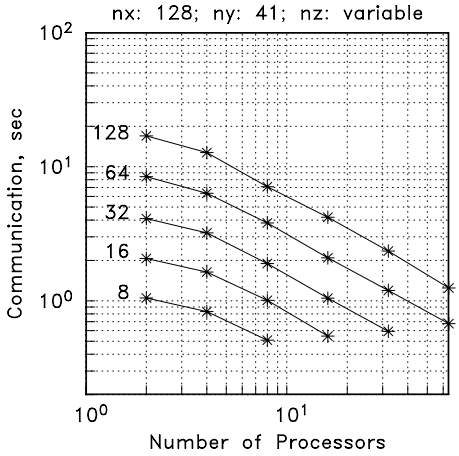

Figure 4: Communication costs for test suite.

Performance data are collected for the serial code on a single node of the SP1 and for the parallel code on up to 64 processing nodes. The chosen problem dimensions are representative of actual simulations that are currently performed on Cray-class supercomputers. The wall-normal dimension is fixed at 41 grid points and the streamwise direction consists of 128 grid points. The spanwise dimension is varied from 8 to 128 Fourier modes in powers of 2. For brevity, the results for varied streamwise dimension can not be given here, these results can be found in [5]. The PSDNS code is instrumented with a set of timers to record separate performance data for different parts of the computation (the total and four dominating algorithmic kernels) and the communication. These measurements are wall-clock time. By including the idle time that results from the necessary synchronization points of the code in the time data, processor-independent performance figures can be obtained. Processor idle time is discussed below in conjunction with the large simulation for which the small serial fraction of the PSDNS code is experimentally determined.

In Figure 3 the computational times for a single time step of the test suite are given in double logarithmic graphs. The associated communication times are given in Figure 4. The excellent scaling of the code on the SP1 can be observed immediately. However, large communication costs relative to the computation costs are incurred because of the unbalanced architecture of the current SP1 (ie., network performance lags behind compute performance of processing nodes) on the one hand and the algorithmic communication penalty on the other hand. The communication penalty must be incurred in order to utilize highly optimized serial FFT routines in the spanwise

direction. The good scaling of the communication cost with respect to the number of processors is noteworthy because the communication that occurs in the PSDNS code involves a complete exchange, which represents a stringent test to the communication network.

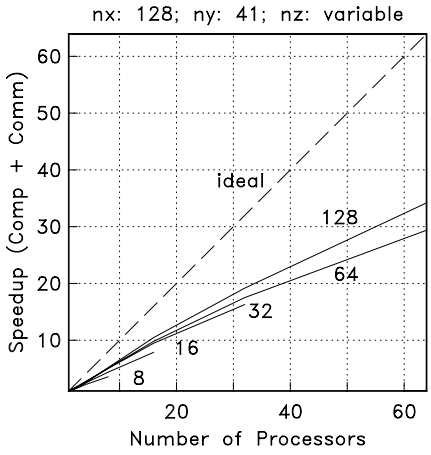

Figure 5: Speedup of test suite based on complete calculation.

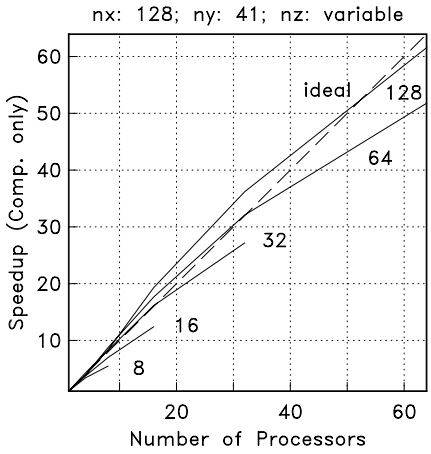

Figure 6: Speedup of test suite based on computation only.

The speedup of a parallel code for fixed-size problems is an important performance metric. In Figure 5, the actual speedup of the complete calculation is given. The performance of the algorithm can be improved by scaling the problem size by either increasing the number of spanwise grid points or increasing the number of streamwise grid points. However, the code is less sensitive to changes in the size of the streamwise dimension than it is to changes in the number of spanwise grid points. For all test cases, the parallel efficiency of the PSDNS code stays above 40 percent, even when 64 processing nodes are utilized.

A theoretical speedup metric can be obtained by ignoring all communication costs. This metric is given in Figures 6. A superlinear speedup is observed. The superlinear theoretical speedup seen for the large problems is not a surprise. The good scalability of the algorithm, combined with the better memory access of the local portion of the distributed data structure is an obvious explanation.

5 Large Scale Simulation

The nonlinear evolution of a crossflow vortex packet on a swept wing has been computed with the spatial DNS code described by Joslin and Street [10]. Because this study required substantial computational resources (i.e., approximately 125 CPU hours on a Cray-2 with a single processor), it is representative of a large-scale simulation. For the SP1 compatible simulation, a grid with 896 streamwise, 61 wall-normal, and 32 spanwise grid points was used. Thus, the computational grid contains over 1.7 million grid points. The Cray Y/MP performs one time step of this simulation in 54 seconds and delivers 240 Mflops. Therefore, the computational expense of one time step is 12 960 Mflop.

The large core memory of the SP1 allows a problem of the same size to be computed on as few as eight processing nodes. The computational costs of the PSDNS algorithm for 8, 16, and 32 nodes of the SP1 are presented in Figure 7. The dashed line gives the total time required by the algorithm to perform one time step. If we compare these SP1 timings with the times required by a single node of the Cray Y/MP and Cray C-90 (marked with solid squares in the same plot) we see that the PSDNS code is highly competitive with these serial supercomputer performances for as few as 8 and 32 processing nodes of the SP1, respectively.

The actual measured execution time, which includes communication and idle time, is given in Table 1 for 8, 16, and 32 nodes of the SP1. An idealized execution time can be obtained by subtracting those times that each processing node spends idle or in communication. Because the serial part of the algorithm is performed only on the first node, two idealized times must be recorded; one value for the first node

Table 1: Performance of large simulation on 8, 16, and 32 Nodes of SP1

Number of processors n_p	Time, sec			Performance, Mflops				Executable code, Mb
	Actual	Idealized node		Actual		Idealized		
		1	2–n_p	Per proc.	Total	Per proc.	Total	
8	53.75	32.4	29.1	30	241	55	440	79
16	29.75	17.7	14.5	27	436	55	880	60
32	18.75	10.2	7.1	22	691	56	1760	50

and another for the remaining processing nodes. The performance of the PSDNS code (in Mflops), based on the actual time and the idealized time, is given in Table 1. The idealized performance of 55 Mflops per processor is noteworthy. Recall that even though the peak performance of a single node is 128 Mflops, 15–75 Mflops are generally observed for actual applications. The last column of Table 1 shows the memory requirements of the executable code; these numbers show that the code is far from reaching the local memory limit of 128 Mb. A performance summary for the communication part of the algorithm is given in Table 2. Here, the reported bandwidth represents a measured value which includes the message startup costs.

By using the idealized execution times for node 1 and for nodes 2–n_p in Table 1 (the idealized execution time excludes all idle time and communication costs), one can determine experimentally the serial and parallel fractions of the PSDNS algorithm. The difference between the two execution times is the time spent in the serial part of the parallel algorithm. If we multiply the execution time of node 2 by the number of processors, we obtain the execution time of the parallel portion. In this context, total execution time is equal to the time spent by all processing nodes combined. If we normalize the time spent in the serial and parallel portions of the algorithm with the total execution time we obtain the serial fraction s and the parallel fraction p, respectively. Surprisingly, the serial fraction is only 1.4 percent, and the parallel fraction is 98.6 percent of the total. Amdahl's law [1] provides a theoretical speedup that is derived from these two quantities:

$$S_p = \frac{1}{s + \frac{p}{n_p}}$$

For 8, 16, and 32 processing nodes, the theoretical speedup S_p of the PSDNS code is 7.29, 13.22, and 22.32, respectively. In the limit of $n_p \to \infty$, the speedup asymptotically reaches the value $1/s$. Even though the parallel granularity of the PSDNS code is restricted for this problem to 32 processing nodes, the theoretical maximum speedup is 71.

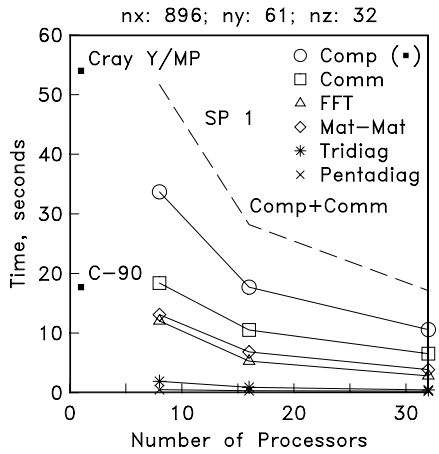

Figure 7: Computational costs for large simulation: IBM SP-1 versus Cray Y/MP and Cray C-90.

6 Conclusions

The expectations raised in reference [8] for the performance of the PSDNS code on a larger and more powerful distributed memory machine may be realized with its implementation on the SP1. In reference [8], due to hardware limitations, only a vague estimate for the performance of a large-scale simulation on a 32-node INTEL iPSC/860 with sufficient core memory was given. Joslin and Zubair concluded that

Table 2: Total Data Exchange for Single Iteration Step of Large Simulation

Number of processors n_p	Comm., sec	Message Volume, Mb	Message Size, Mb	Bandwidth Total, Mb/sec	Bandwidth Per Processor, Mb/sec
8	19.0	595	0.208	63	7.9
16	11.0	638	.052	116	7.3
32	7.5	659	.013	176	5.5

the execution time for the PSDNS code on the 32-node iPSC/860 would be twice the time required by a Cray supercomputer. In this work, we have shown that only eight nodes of the more powerful SP1 are needed to perform such a large-scale simulation in the same amount of time as required by a Cray Y/MP. Furthermore, the utilization of 32 processing nodes on the SP1 reduces the execution time to roughly one-third. Both the parallel efficiency of the PSDNS code (above 40 percent for all performed calculations) on the SP1 and the high serial performance of 52–56 Mflops on a single SP1 node (45 percent of theoretical peak performance) contribute to this success. On 32 processing nodes of the SP1, the PSDNS code is also highly competitive in comparison with the advanced Cray C-90 on large-scale simulations.

Acknowledgments

The authors gratefully acknowledge use of the Argonne National Laboratory High-Performance Computing Research Facility (HPCRF). The HPCRF is funded principally by the U.S. Department of Energy Office of Scientific Computing.

References

[1] Amdahl, G. (1967). *Validity of the single-processor approach to achieving large scale computing capabilities.* Proceedings of the AFIPS Conference, 483–485.

[2] Bokhari, S.H. (1991). *Complete Exchange on the iPSC/860.* ICASE Report No. 91-4.

[3] Fischer, P.F., and Patera, A.T. (1994). Parallel simulation of viscous incompressible flows. *Annu. Rev. Fluid. Mech.*, **26**, 483–527.

[4] Gropp, W., Lusk, E., and Pieper, S.C. (1994). *Users Guide for the Argonne National Laboratory IBM SP1.* Argonne National Laboratory.

[5] Hanebutte, U.R., Joslin, R.D., and Zubair, M. (1994). *Scalability Study of Parallel Spatial Direct Numerical Simulation Code on IBM SP1 Parallel Supercomputer.* ICASE Report No. 94-80. Submitted to *J Sci. Comp.*

[6] IBM Parallel Programming Reference, *AIX Parallel Environment, Release 1.0.* SH26-7228-00, (1993).

[7] IBM Guide and Reference, *Engineering and Scientific Subroutine Library, Version 2.* SH23-0526-00, (1992).

[8] Joslin, R.D., and Zubair, M. (1993). *Parallel Spatial Direct Numerical Simulations on the Intel iPSC/860 Hypercube.* ICASE Report No. 93-53. *J. Sci. Comp.*, Vol. 9, 1994.

[9] Joslin, R.D., Street, C., and Chang, C.-L. (1993). Spatial DNS of Boundary-Layer Transition Mechanisms: Validation of PSE Theory. *Theor. and Comp. Fluid Dyn.* **4**(6), 271–288.

[10] Joslin, R.D., and Street, C. (1994). The Role of Stationary Crossflow Vortices in Boundary-Layer Transition on Swept Wings. *Phys. Fluids A*, Vol. 6, No. 10, 1994.

[11] Saini, S. (1994). *The IBM SP2: Hardware, Software, Porting and Optimization Overiew.* Numerical Aerodynamics Simulation Program, NASA Ames Research Center, NAS User Seminar, July 27, 1994.