Performance of MC2 and the ECMWF IFS forecast model on the Fujitsu VPP700 and NEC SX-4M

Michel Desgagné, Stephen Thomas[1] and Michel Valin
Recherche en prévision numérique (RPN), Environment Canada, 2121, route Transcanadienne, Dorval, Québec, Canada H9P 1J3
Tel.: + 1 514 421 4661; Fax: + 1 514 421 2106; E-mail: {michel.desgagne, steve.thomas, michel.valin}@ec.gc.ca

The NEC SX-4M cluster and Fujitsu VPP700 supercomputers are both based on custom vector processors using low-power CMOS technology. Their basic architectures and programming models are however somewhat different. A multi-node SX-4M cluster contains up to 32 processors per shared memory node, with a maximum of 16 nodes connected via the proprietary NEC IXS fibre channel crossbar network. A hybrid combination of inter-node MPI message-passing with intra-node tasking or threads is possible. The Fujitsu VPP700 is a fully distributed-memory vector machine with a crossbar interconnect which also supports MPI. The parallel performance of the MC2 model for high-resolution mesoscale forecasting over large domains and of the IFS RAPS 4.0 benchmark are presented for several different machine configurations. These include an SX-4/32, an SX-4/32M cluster and up to 100 PE's of the VPP700. Our results indicate that performance degradation for both models on a single SX-4 node is primarily due to memory contention within the internal crossbar switch. Multinode SX-4 performance is slightly better than single node. Longer vector lengths and SDRAM memory on the VPP700 result in lower per processor execution rates. Both models achieve close to ideal scaling on the VPP700.

[1]Also at: Computational Sciences Section, Scientific Computing Division, National Center for Atmospheric Research (NCAR), 1850 Table Mesa Drive, Boulder, CO 80303, USA.

Scientific Programming 8 (2000) 23–30
ISSN 1058-9244 / $8.00

1. Introduction

John Hennessy, professor of computer science, dean of the Stanford University School of Engineering and co-inventor of the MIPS RISC microprocessor recently speculated during the Supercomputing 97 conference in San Jose that vector processors would disappear from high-performance computing within five to ten years [5]. Given the impressive sustained floating point execution rates of the NEC SX-4 and Fujitsu VPP700 vector processors, these two Japanese computer vendors could easily argue that 'reports of their demise are greatly exaggerated'. Despite the fact that the peak execution rates of pipelined RISC microprocessors continue to double every eighteen months, highly optimized codes can usually sustain no more than 15 to 20% of peak. This situation may change as larger secondary cache memories become available. However, the SX-4 vector processor can routinely achieve 1 Gflops/sec or higher on representative atmosphere, ocean and climate codes [3]. Both SX-4 and VPP700 processors can sustain in the range of 30 to 50% of their rated peak performance levels. NEC and Fujitsu build parallel architectures based on these processors with existing or planned customer installations capable of 100 Gflops/sec or higher sustained performance.

Cluster type architectures are becoming prevalent in high-performance computing and current designs can trace their roots back to the pioneering work of Paul Woodward who demonstrated the capabilities of symmetric multiprocessor (SMP) cluster supercomputing in 1993 [13]. The US Department of Energy's Accelerated Strategic Computing Initiative (ASCI) has also led to the announcement of cluster type computers from several US manufacturers. Individual nodes contain from 1 to 128 cache or vector processors. Typically, shared or distributed-shared memory (DSM) is used within a node and additional cache-coherence mechanisms are often present. Low-latency, high-bandwidth interconnection networks then link these

nodes together. NEC SX-4M clusters and the Fujitsu VPP700 perhaps represent opposite ends of the design spectrum. SX-4 nodes contain up to 32 vector processors and 8 Gbytes of fast SSRAM main memory, whereas the VPP700 is a fully distributed-memory machine. Each VPP700 processing element contains a vector processor along with up to 2 Gbytes of slower SDRAM memory. The two machines are compared in this paper by using benchmarks of two decidely different atmospheric models. The ECMWF IFS forecast model is a global weather prediction model based on the spectral transform method. The Canadian MC2 is a nonhydrostatic, fully compressible limited area atmospheric model designed for high-resolution mesoscale forecasting. A fully 3D semi-implicit scheme is implemented with second-order finite differences in space. Both models implement semi-Lagrangian advection with overlaps.

2. The NEC SX-4M and Fujitsu VPP700

The multi-node NEC SX-4M is an SMP cluster type architecture with up to 32 processors per node and a maximum of 16 nodes interconnected via the proprietary NEC IXS crossbar network with fibre channel interface. Each node executes an enhanced version of UNIX System V with features such as resource sharing groups (RSG) to dedicate resources to single or multi-node jobs. The total 8 Gbytes/sec IXS (bi-directional) bandwidth is augmented by a direct memory-mapped addressing scheme between nodes [4]. An SX-4 CPU contains a 100 Mflops/sec scalar unit and a vector unit. The vector processor is based on low-power CMOS with a clock cycle time of 8ns (125 MHz). Three floating point formats are supported: IEEE 754, Cray, and IBM. The vector unit of each processor consists of 8 parallel sets of 4 vector pipelines, 1 add/shift, 1 multiply, 1 divide, and 1 logical. For each vector unit there are 8 64-bit vector arithmetic registers and 64 64-bit vector data registers used as temporary space. The peak performance of a concurrent vector add and vector multiply is 2 Gflops/sec and atmospheric codes can sustain 1 Gflops/sec or higher. Main Memory Unit (MMU) configurations for a node range from 512 Mbytes to 8 Gbytes of 15 ns Synchronous Static Random Access Memory (SSRAM). The maximum 8 Gbytes configuration comprises 32 banks of 256 Mbytes each, providing memory bandwidths of 16 Gbytes/sec per processor. Supplementing main memory is 16 or 32 Gbytes of eXtended Memory Unit (XMU) built with 60ns Dynamic Random Access Memory (DRAM) and having a 4 Gbyte/sec bandwidth. MPI/SX is based on a port of the MPICH package by NEC's C & C European Lab with the assistance of Rusty Lusk and Bill Gropp from Argonne National Laboratory [4].

A processing element of the Fujitsu VPP700 also contains both a scalar and vector unit. The vector unit consists of 8 functional units which can operate in parallel. The peak performance of the vector unit is 2.2 Gflops/sec, whereas the scalar unit is a 100 Mflops/sec processor. Both 32 and 64-bit IEEE floating point formats are supported. Each PE can be configured with up to 2 Gbytes of Synchronous Dynamic Random Access Memory (SDRAM). A full copy of the 32-bit UNIX operating system kernel is executed by each processor with 1.7 Gbytes available for programs and data. A 64-bit operating system is planned for the next generation VPP architecture with up to 8 Gbytes of memory per PE. Processing elements are interconnected with a switching network, capable of 570 Mbytes/sec (bi-directional) point-to-point transfer rates. MPI is implemented on top of the proprietary VPP message-passing layer. Any processor can make I/O requests but only 11 of the 116 VPP700 PE's at the ECMWF (the so-called I/O processors) are configured with disks.

3. Parallel programming models

Climate and ocean modeling groups at NCAR [7] and the University of Minnesota [9] have identified and tested hybrid programming models for SMP architectures. Shared-memory tasking mechanisms or threads can be applied for intra-node parallelism, whereas inter-node communication is implemented with MPI. Coarse-grain tasks on an SX-4 node are created with the `pt_fork` and `pt_join` primitives and loop-level parallelism in the form of micro-tasking is specified through the inline compiler directive *vdir pardo*. A POSIX threads compliant library `pt_thread` is also available. With the recent acceptance of an OpenMP standard for shared-memory parallelism, it should now be possible to build portable codes employing both MPI and tasks. The MC2 model is discretised on a $N_X \times N_Y \times N_Z$ grid, where the number of points in the vertical direction is typically one order of magnitude less than in the horizontal. A distributed-memory model of computation is based on a domain decomposition across a $P_X \times P_Y$ processor mesh. All vertical loops in the dynamics and physics code are micro-tasked, allowing for a hybrid combination with bound-

ary exchanges implemented using MPI. The elliptic solver in MC2 is a minimal residual Krylov iteration with line relaxation preconditioners (see Skamarock et al. [11] and Thomas et al. [12]). To handle global data dependencies, a data transposition strategy is implemented using MPI all-to-all communication. Fixed-size halos are implemented for semi-Lagrangian advection.

The IFS forecast model is a global spectral model which can use either a full or reduced Gaussian grid. In the case of a reduced grid, the number of grid points along a latitude line decreases near the poles. Both Eulerian and semi-Lagrangian advection schemes are available. A parallel domain decomposition is based on a latitude by longitude decomposition in grid point, Fourier and spectral space where $\mathrm{NPROC} = \mathrm{NPROCA} \times \mathrm{NPROCB}$. A data transposition strategy is implemented between each computational phase of a time-step. A fixed overlap strategy is also implemented for the distributed-memory implementation of semi-Lagrangian advection where the global maximum wind-speed determines the halo size (see Dent and Mozdzynski [2]). The shared-memory version of the model is still retained and was not sacrificed in order to build a distributed-memory implementation. In fact, the IFS model can be run in a hybrid shared/distributed configuration. FFT's are computed on all processors and are independent in both the vertical and longitudinal directions. Likewise, the Legendre transforms are also executed on all processors and are independent in the vertical and over spectral waves. Finally, the IFS has been coded to perform effectively on vector architectures by supporting a runtime parameter NPROMA which controls the optimal vector length.

4. Benchmark results

We have benchmarked the full forecast configurations of MC2 (adiabatic kernel + RPN physics version 3.5) and IFS (RAPS 4.0 version) at the CMC in Montreal and at the ECMWF in Reading. The current CMC configuration consists of the operational machine 'hiru', an SX-4/32 with 8 Gbyte MMU along with 'yonaka' (SX-4/16 + 4 GB MMU) and 'asa' (SX-4/16 + 8 GB MMU). The two SX-4/16 nodes can operate as an SX-4/32M cluster and all three machines can be connected to the IXS crossbar. Four full nodes in an SX-4/128M cluster should be in place by the year 2000 or 2001, with a peak performance of 256 Gflops/sec. Given our results to date, it is reasonable to expect that 50% of peak is possible on such a machine. The ECMWF VPP700 is currently configured with 116 PE's, each containing 2 Gbytes of memory or 232 Gbytes in total.

The MC2 model is written in Fortran 77 with Cray `POINTER` extensions for dynamic memory allocation. The code was compiled using 32-bit arithmetic on both the SX-4 and VPP700. Whereas the IBM floating point format was specified on the SX-4, 32-bit IEEE arithmetic was used on the VPP700. The only compiler options specified to assist in vectorisation were `-pvctl noassume loopcnt=1000000`. Extensive inline compiler directives such as *vdir nodep* are specified in the physics library due to dynamic memory allocation. The SX-4 compiler is conservative and assumes both aliasing and recurrences are present unless otherwise indicated. The vectorisation level on the SX-4 (scalar versus vector instructions) then usually exceeds 98%. Similar directives were specified to the VPP700 Fortran 90 compiler `frt`. Multi-node SX-4 runs require a `mpi.hosts` file containing the number of processes to launch on each node. In particular, the order of processes launched from this file determines their rank in `MPI_COMM_WORLD`.

The IFS forecast model code is written in a subset of Fortran 90 with extensive use of `ALLOCATABLE` arrays. The model code was compiled for 64-bit IEEE arithmetic on both the SX-4 and VPP700 machines. In fact, this was our first experience at RPN/CMC with the NEC Fortran 90 compiler. It was found to be far too slow for production usage and would likely perform better as a cross-compiler similar to Fujitsu's `frtpx` run on a SGI/Cray Origin 2000 at the ECMWF. Vectorisation and performance of the IFS code are largely determined by the `NAMELIST` parameters `NRPROMA` for the radiation package and `NPROMA` in the dynamics. In all tests we varied `NPROCA` and set `NPROCB=1` [1]. Performance data for the IFS RAPS 4.0 benchmark (T106L19, T213L31) and an MC2 run at 10km resolution using a $512 \times 432 \times 41$ grid ($10 \times \Delta t = 180$sec) are presented at the end of the paper.

Performance data for the SX-4 was collected using hardware counters made available to the operating system via the environment variable `setenv PROGINF=detail`. More accurate timings were obtained by directly reading hardware registers from the application software. Hardware counters were also queried on the VPP700 to obtain timings and flop counts. The performance of the IFS model on the VPP700 and SX-4M cluster is summarized in Tables 1 and 2 along with Fig. 1. Results in Figure 1 on the

Table 1
IFS T106L19 12 hr forecast timings (secs) on SX-4/32M cluster. SX-4/16, 4 GB MMU (yonaka) + SX-4/16, 8 GB MMU (asa). Semi-Lagrangian (y/n). Processing Elements (PEs): $n_1 + n_2$, indicating number of processors on each SX-4 node. Elapsed wall-clock (real) time. Average CPU time (user) per process. Total CPU time (cp) charged to all processes. Total vector time (vector) for vector instructions. Vectorization ratio (% vec) of scalar to vector instructions issued. Estimated parallel (par) time with I/O amortized for a longer run

sl	PEs	real	user	cp	vector	% vec	par
y	1+1	80	78	155	99	97.5	73
y	2+2	71	52	203	129	97.0	65
y	4+4	42	29	227	140	96.5	36
y	12+4	28	16	255	150	96.0	18

Table 2
IFS T213L31 12 hr forecast timings (secs) on SX-4/32, 8 GB MMU (hiru) and SX-4/32M cluster. SX-4/16, 4 GB MMU (yonaka) + SX-4/16, 8 GB MMU (asa)

sl	PEs	real	user	cp	vector	% vec	par
y	4	200	144	566	326	96.0	183
y	8	101	74	580	331	96.0	88
y	16	63	42	665	373	96.0	47
n	2	276	236	465	161	90.0	249
n	4	137	107	420	166	90.0	120
n	6+2	78	55	437	183	90.0	63
n	8	80	57	450	192	90.0	67
n	16	52	33	514	223	89.0	35

SX-4 are reported for the highest level of compiler optimisation recommended by NEC. The IFS sustains between 750 and 800 Mflops/sec per processor on the VPP700 [2] and SX-4M performance is slightly higher. The MC2 model sustains 750 Mflops/sec with less than 3% degradation from 8 to 32 processors on the VPP700 with a vector length of 512 and $P_X = 1$ (see Fig. 2). For $P_X = 2$ and a vector length of 256, the SX-4M multi-node execution rate of MC2 is higher than on a single SX-4/32 node from 8 up to 32 PE's as illustrated in Fig. 3. We attribute the slightly faster drop-off in the single-node sustained execution rate to the behaviour of the SX-4 inter-node processor to memory crossbar switch under increasing load. To justify our assertion, a performance model is presented in the next section.

5. Performance model

In this section we develop a simple performance model for the degradation of per processor execution rates observed within a single SX-4/32 node. It will be assumed that a drop in the Mflops/sec per processor rate R is directly related to a decrease in the effective bandwidth of the 32×32 multi-port packet-switched crossbar network between processor network units (PNU) and main main units (MMU). A single packet contains an 8-byte word and so a 256 element vector would require 256 separate packet requests. All other effects such as message-passing latencies are ignored. The SX-4 crossbar network was designed to support a 1:1 operation to load/store ratio when a single processor is operating at 2 Gflops/sec (i.e. 16 Gbytes/sec = 2 Gwords/sec).

Following Appendix C of Siegel [10], it will be assumed that:

1. each source PNU generates a request with probability $p \leqslant 1$ each cycle.
2. each request is sent with equal probability to a destination MMU.

A cycle is defined as the time it takes for a request to propagate through the network plus the time needed to access a memory word plus the time used to return through the network to the source [6]. The packet *rate* $p \leqslant 1$ is the number of packet requests issued per PE per cycle. The network *bandwidth* is the average number of requests accepted per cycle.

Consider an $M \times N$ packet based crossbar interconnection network as described in Siegel [10], where M PNU's are connected to N MMU's. The probability p that a PNU makes a memory reference during a cycle is defined to be the average number of requests generated per cycle by each processor. Patel [8] has shown that the expected bandwidth of a crossbar network (accurate to 1% for $N \geqslant 32$) is given by

$$B(M,N) = \left(1 - e^{-p\,M/N}\right) N$$

where the bandwidth of an individual channel is $(1 - e^{-p\,M/N})$. Moreover, the ratio of expected bandwidth to the expected number of requests pM generated per cycle is defined to be the probability of acceptance.

$$P_A = \frac{N}{pM}\left(1 - e^{-p\,M/N}\right)$$

To model per processor performance degradation, let the number of active processors making memory requests increase from $M = 1$ to $M = 32$ PNU's and assume the following.

1. each active PE has a request rate of $p = 0.45$ per cycle.
2. a sustained execution rate of 900 Mflops/sec/PE, representing 45% of peak, implies $p = 0.45$.

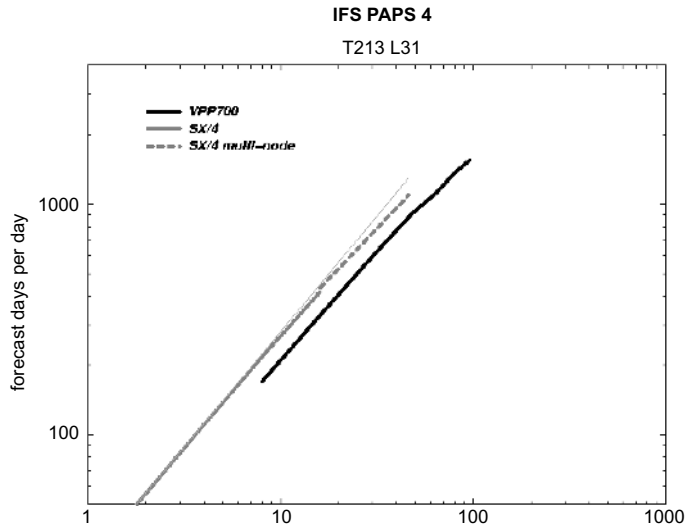

Fig. 1. IFS RAPS 4.0 T213L31 Benchmark. Semi-Lagrangian. Thin line represents ideal scaling.

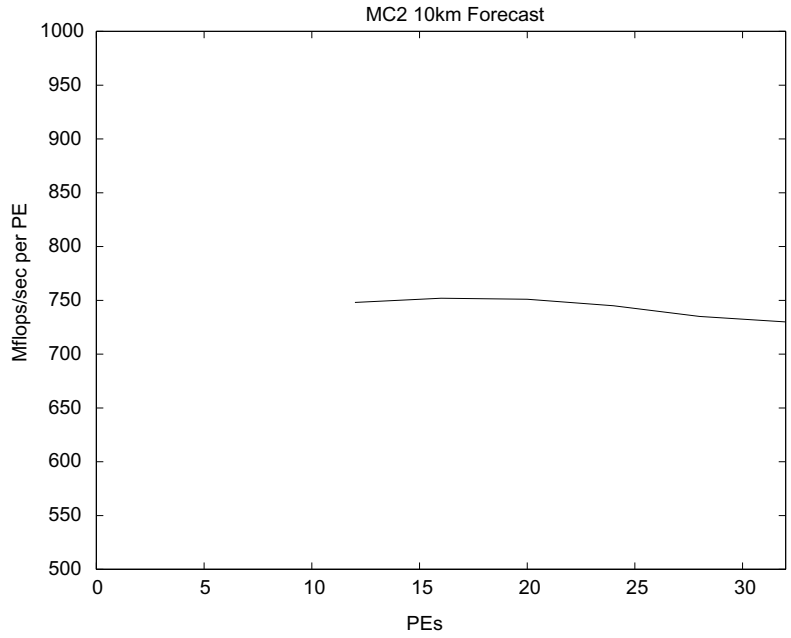

Fig. 2. MC2 Performance on VPP700: Runs: 12, 16, 20, 24, 28, 32 PE's.

The average request rate for M active processors is therefore $0.45 \times (M/N)$. It is implicitly assumed in 2. that the Mflops/sec rate R is directly related to the memory performance. P_A is plotted in Fig. 4 for $M = 1$ to $M = 32$ active processors and $N = 32$ MMU's. For comparison, the linear approximation

$$P_A = 1 - \frac{1}{2}\left(\frac{0.45M}{N}\right)$$

is also plotted in the same figure.

Since the sustained execution rate of a processor depends directly on the rate at which memory requests can be serviced, we model the degradation of performance as the maximum single processor Mflops/sec rate R for a single active processor multiplied by the probability of acceptance P_A. Predicted and observed performance degradation due to crossbar contention $R \times P_A$ is plotted in Fig. 5. The model and experimental results are in good agreement.

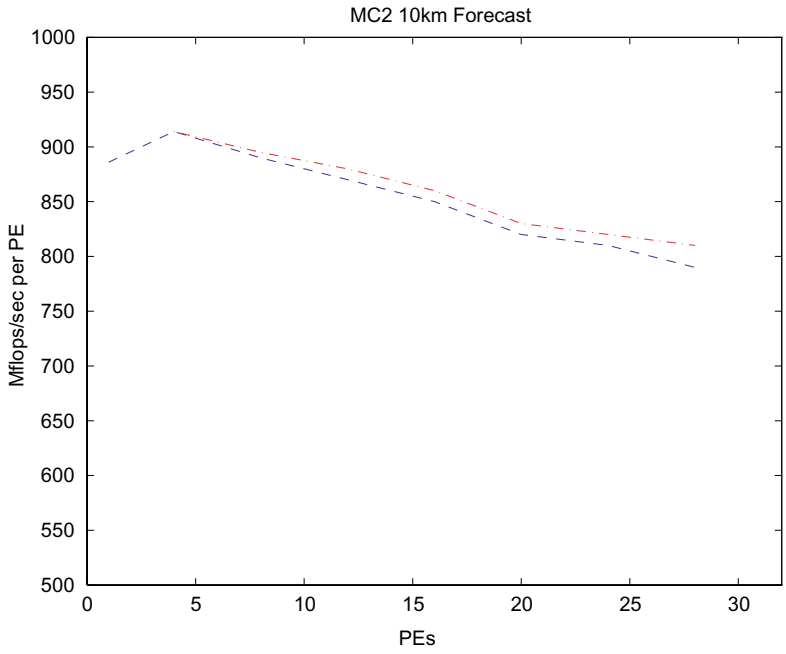

Fig. 3. MC2 Performance on SX-4M: Single (bottom) versus multi-node (top).

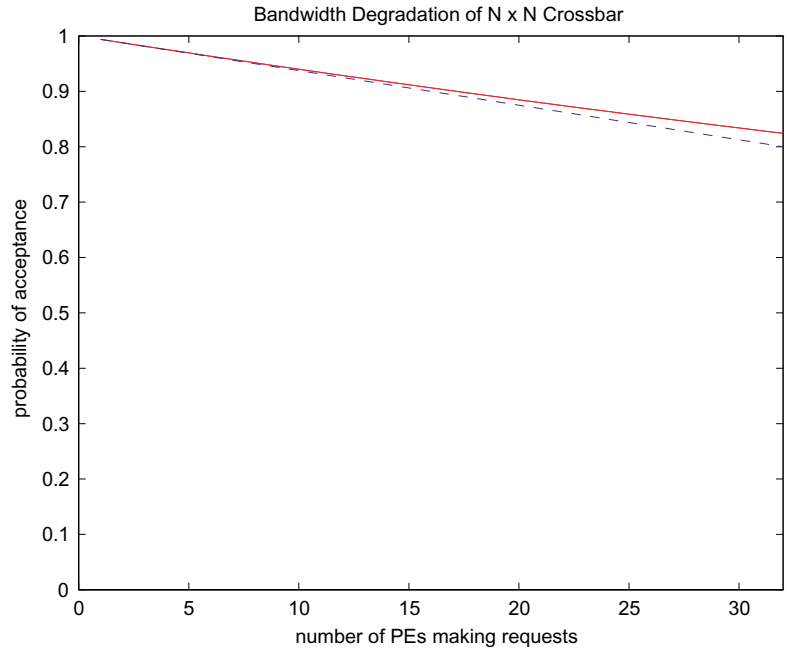

Fig. 4. NEC SX-4/32 32×32 crossbar probability P_A that memory request is accepted during a machine cycle. Top: Linear approximation of memory request acceptance rate P_A. Bottom: Acceptance rate P_A from Patel [8].

6. Discussion and conclusions

For both the MC2 and IFS models, we encountered what might be best characterized as a problem with 'memory starved' nodes. The SX-4 has 128 Mbytes of SSRAM memory per Gflop of computing power, whereas the VPP700 has over 900 Mbytes of SDRAM per Gflop, a factor of 7 more in terms of memory size.

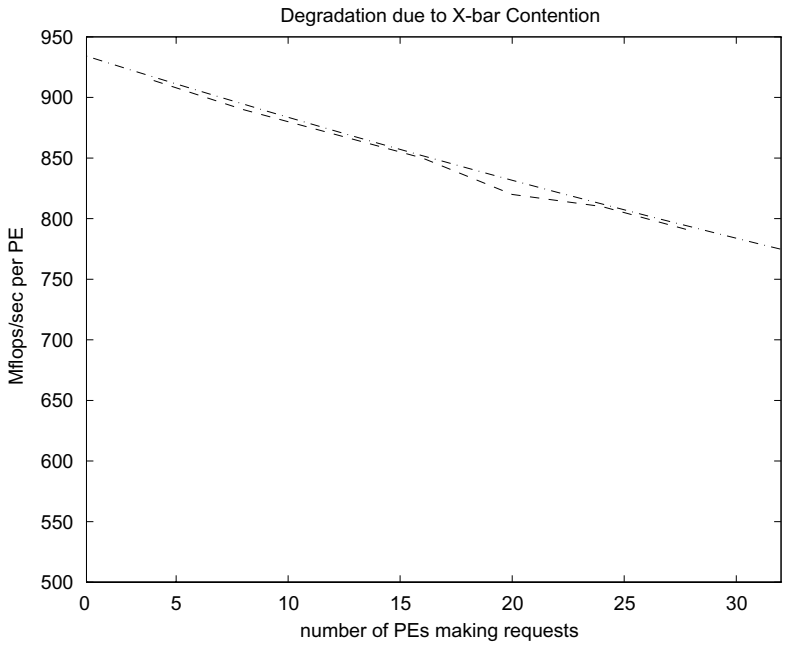

Fig. 5. NEC SX-4/32 performance degradation due to crossbar. Top: Predicted Mflops/sec rate from $R \times P_A$ assuming $R = 900 M flops/sec$. Bottom: Observed Mflops/sec on an SX-4/32 node.

In the case of the SX-4, it appears that 8 Gbytes of fast SSRAM may not be sufficient for 32 processors, each operating at 1 Gflop/sec, in a single distributed-memory program. Since the SX-4 is a 'transition' machine, designed to support both a traditional computing mix of single threaded jobs and multi-node applications, certain design compromises were required. Future designs such as the follow-on SX-5 from NEC, or for that matter any SMP cluster type architecture, must strike the right balance between the number of processors per node and providing a memory hierarchy that supports the highest possible sustained execution rate within a node. Shared-memory tasking mechanisms tend to quickly saturate within a node unless very large grain tasks are used. For example, given a grid size of $238 \times 243 \times 30$, the shared-memory parallel efficiency drops rapidly from 75% using four processors to 50% at eight processors. For such small problem sizes, a hybrid mix of sub-domain boundary exchanges using MPI combined with micro-tasking in the vertical direction can be more efficient. However, we have always found that a distributed-memory model of computation for both inter and intra-node parallelism yields the highest performance and the transition from single to multi-node is seamless across the NEC IXS crossbar switch with no degradation in performance. Moreover, the performance across nodes was better than on a single node. The correlation between experimental results and our performance model confirm that the degradation is due to memory contention.

Since the scalar units on both the SX-4 and VPP 700 are 20 to 100 times slower (50 to 100 Mflops/sec versus 1 Gflops/sec the SX-4) than the vector units, scalar code is to be avoided at all costs. With 2 Gbytes of SDRAM available per PE and likely 8 Gbytes in the next generation machine, memory on the VPP 700 is not a major issue. The slower SDRAM may affect the sustainable floating-point execution rate of some scientific codes. Both the SX-4 and VPP700 processors have an abundance of vector registers which the compiler can exploit to reduce memory traffic. We have found in our benchmarks that the SX-4 processor performs slightly better on short vector lengths than the VPP700. The performance of the VPP700 crossbar interconnect for the IFS spectral model is now well documented, but also the particular communication patterns of a grid point model (such as halo exchanges) are also well handled. The overall performance of the IFS forecast model is slightly better on the SX-4M than the VPP700 (both single and multi-node) for the T213L31 benchmark as can be seen from Fig. 1. However, the performance is very close and we believe that the gap could be bridged with a modest tuning effort. The observed differences may be attributed to the slower SDRAM memory and longer vector lengths required by the VPP700.

Acknowledgements

David Dent and George Mozdzynski were our original co-authors for an earlier version of this article. They have asked not to be included in order to speed-up the review process for publication. Both David and George contributed to preparing this article by performing all IFS and MC2 benchmark runs at ECMWF. They also provided technical advice and interpretation of the results.

References

[1] S. Barros, D. Dent, L. Isaksen, G. Robinson, G. Mozdzynski and F. Wollenweber, The IFS Model: A parallel production weather code, *Parallel Computing* **21** (1995), 1621–1638.

[2] D. Dent and G. Mozdzynski, ECMWF operational forecasting on a distributed memory platform: Forecast model, in: *Proceedings of the Seventh ECMWF Workshop on the Use of Parallel Processors in Meteorology,* G.-R. Hoffmann and N. Kreitz, eds., World Scientific, Singapore, 1997, pp. 36–51.

[3] S. Hammond, R. Loft and P. Tannenbaum, Architecture and Application: The Performance of the NEC SX-4 on the NCAR Benchmark Suite, *Supercomputing 96 Proceedings*, San Jose, CA.

[4] R. Hempel, H. Ritzdorf and F. Zimmermann, Implementation of MPI on NEC's SX-4 multi-node architecture. Proceedings of the 4th European PVM–MPI User's Group Meeting, 1997.

[5] J.L. Hennessy. Perspectives on the architecture of scalable multiprocessors: Recent developments and prospects for the future. Presentation, Supercomputing 97, San Jose, November 1997.

[6] K. Hwang and F.A. Briggs, Computer Architecture and Parallel Processing. McGraw–Hill Publishing, New York, 1984.

[7] R.D. Loft, A modular 3-D dynamical core testbed, in: *Proceedings of the Seventh ECMWF Workshop on the Use of Parallel Processors in Meteorology,* G.-R. Hoffmann and N. Kreitz, eds., World Scientific, Singapore, 1997, pp. 270–283.

[8] J.H. Patel, Performance of processor-memory interconnections for multiprocessors, *IEEE Transactions on Computers* **C–30**(12) (1981), 771–780.

[9] A.C. Sawdey, M.T. O'Keefe and W.R. Jones, A general programming model for developing scalable ocean circulation applications, in: *Proceedings of the Seventh ECMWF Workshop on the Use of Parallel Processors in Meteorology,* G.-R. Hoffmann and N. Kreitz, eds., World Scientific, Singapore, 1997, pp. 209–225.

[10] H.J. Siegel, Interconnection Networks for Large-Scale Parallel Processing. Theory and Case Studies. 2nd Edition, McGraw-Hill, New York. 1990.

[11] W.C. Skamarock, P.K. Smolarkiewicz and J.B. Klemp, Preconditioned conjugate-residual solvers for Helmholtz equations in nonhydrostatic models, *Mon. Wea. Rev.* **125** (1997), 587–599.

[12] S.J. Thomas, A.V. Malevsky, M. Desgagné, R. Benoit, P. Pellerin and M. Valin, Massively parallel implementation of the mesoscale compressible community model, *Parallel Computing* **23** (1997), 2143–2160.

[13] P.R. Woodward, Perspectives on supercomputing: Three decades of change, *IEEE Computer* **29** (1996), 99–111. (special 50th anniversary issue).