PERFORMANCE OF AQM APPLICATIONS ON COMMODITY LINUX PLATFORMS

George Delic *
HiPERiSM Consulting, LLC, Durham, NC
e-mail: **george@hiperism.com**
Web address: **http://www.hiperism.com**
Voice (919) 484-9803 Fax (919) 806-2813

1.0 INTRODUCTION

This is a progress report on a project to evaluate industry standard fortran 90/95 compilers for IA-32 Linux™ commodity platforms when applied to Air Quality Models (AQM). The goal is to determine the optimal performance and workload though-put achievable with commodity hardware. New results are presented for selected benchmarks on Intel Pentium 4 Xeon processors.

2.0 CHOICE OF HARDWARE, OPERATING SYSTEM, AND COMPILERS

The hardware used for the results reported here is the Intel Pentium 4 Xeon (P4) processor (3GHz, 1MB L3 cache) in a dual configuration on a Super Micro SuperServer 6023P-i platform with the Intel E7501 chipset. This configuration has 4GB of memory and a 533MHz System Bus shared by both processors. Storage capacity is 1.5 TB on six ATA/IDE hard drives. The operating system (OS) is HiPERiSM Consulting, LLC's modification of the Red Hat (RH) Linux™ 9.0 SMP kernel. When combined with the Xeon HT technology the RH SMP kernel shows four CPU's on the dual Xeon processors. Four compilers are now available for this combination of hardware and OS from Absoft (8.0), Intel (8.0), Lahey (6.2), and The Portland Group, STMicroelectronics (5.1). The Xeon architecture offers Streaming Single-Instruction-Multiple-Data Extensions, SSE2, (SSE) to enable vectorization of loops operating on multiple elements in a data set with a single operation. Three compilers specifically enable SSE through a compiler switch and this has been used in some tests. The wall clock time reported here is obtained either from the Linux time command or from calls to the Fortran 90/95 system_clock routine.

* *Corresponding author address:* George Delic, HiPERiSM Consulting, LLC, P.O. Box 569, Chapel Hill, NC 27514-0569

3.0 CHOICE OF BENCHMARKS

The choice of benchmarks includes three real-world models: the Princeton Ocean Model (POM), the MM5 modeling system Version 3 (MM5), and the Comprehensive Air Quality Model with Extensions (CAMx). The consequences for performance as problem size scales is also investigated for the POM.

For each compiler listed in the previous section four sets of compiler switches are used: no opt (all optimization disabled), opt (the default switches with some optimization), vect (vectorization enabled), and SSE (SSE2 instructions enabled). The exact description, and the list of compiler switches used, is to be found at HiPERiSM's URL on Technical Reports pages HCTR-2004-3 to HCTR-2004-6. While it is expected that different compilers will deliver different performance (as will different choices of compiler switches), it is less clear what the consequences of memory bandwidth limitations may be. For this reason the analysis begins with the STREAM memory bandwidth benchmark.

3.1 STREAM memory benchmark

It has been known for some time that multiprocessor commodity hardware encounters memory bandwidth bottle-necks when more than one processor is active. Although only serial applications are discussed here, this memory bandwidth limitation affects both OpenMP and MPI parallel applications. In the case of the hardware described in the previous section, this limitation is due to both processors sharing the memory bandwidth of the common system bus. To measure performance of memory bandwidth the STREAM (Sustainable Memory Bandwidth) benchmark has been used (for more information see http://www.cs.virginia.edu/stream). Here the results of the serial version of the STREAM benchmark are presented with a view to discovering differences in performance of this

group of compilers. The STREAM benchmark consists of multiple repetitions of the four Kernels in Table 3.1 and the best results of typically ten trials are chosen. The iteration range of the loop is chosen to range from 1 to $20x10^6$ data points with unit stride. Only the memory bandwidth is reported here (in units of MB/second) and results are presented without and with compiler optimizations.

Table 3.1 Compute kernels of the STREAM benchmark (referenced by number in what follows).

No.	Name	Kernel	Bytes / iterate	Flops / iterate
1	COPY	a(i)=b(i)	16	0
2	SCALE	a(i)=q*b(i)	16	1
3	ADD	a(i)=b(i)+c(i)	24	1
4	TRIAD	a(i)=b(i)+q*c(i)	24	2

3.2 Princeton Ocean Model (POM)

The Princeton Ocean Model (POM) is a legacy Fortran 77 code with compute kernels consisting of over three hundred vectorizable loops. Typically these are triple-nested loops (i,j,k) that perform operations over a three-dimensional finite difference grid. The vertical zones over the k range form the outermost loop in the nest. The number of iterations varies with the choice of data set as shown in Table 3.2. The inner loop structure presents compilers with good prospects for vectorization.

Table 3.2 Problem sizes and scaling for the POM.

GRID	i_{max}	j_{max}	k_{max}	Scaling
1	100	40	15	1
2	128	128	16	4.37
3	256	256	16	17.47

3.3 MM5

The MM5 Community Model has been executed on a wide variety of platforms and Version 3 is used here in serial mode for the Storm-of-the-Century (SOC) benchmark with the Portland and Intel compilers. The optimizations applied are different for the two compilers so the equivalence is not precise. For, example, the equivalence in the vector group does not exist. Whereas the pgf90 compiler allows separation of vector and SSE instruction implementation, the Intel compiler does not and the Intel vect entry is left blank in what follows.

3.4 CAMx

The CAMx code developed by ENVIRON (http://www.camx.com) is a Fortran 77 code for an Eulerian photochemical model that is widely used in the AQM community. This benchmark analysis is concluded with some selected results for the 2000 episode from 8/22 to 8/31 in the Houston Greater Metro area with the base5a.regular.GOES base case for 2000 using the reported non-egu emissions and the GOES-satellite correct meteorology. Modeling files are obtainable from the TCEQ site at
http://www.tnrcc.state.tx.us/air/aqp/airquality_photomod.html#section4>
http://www.tnrcc.state.tx.us/air/aqp/airquality_photomod.html#camx>

Results are presented for five groups of compiler switches with the pgf90 compiler to investigate the behavior of CAMx on the P4 Xeon platform with this compiler. There is a requirement for many simulations with CAMx and thus some prior motivation exists for reducing wall clock time.

4.0 RESULTS OF BENCHMARKS

A fuller discussion of the benchmarks is found at HiPERiSM's URL on Technical Reports pages HCTR-2004-3 to HCTR-2004-6. What follows is a summary.

4.1 STREAM results

Tables 4.1 and 4.2, respectively, show memory bandwidth results without and with optimization.

Table 4.1 Memory bandwidth (MB/second) for the STREAM benchmarks with four compilers on the Pentium 4 Xeon (3.06 GHz, 1MB L3 cache) without optimization.

N	Absoft	Intel	Lahey	Portland
Copy	1252.27	1289.8	1077.44	1300.81
Scale	1252.32	1314.17	1138.79	1316.87
Add	1616.75	1651.76	1230.76	1655.17
Triad	1649.41	1650.05	1230.76	1666.67

In both tables Kernels 3 (Add) and 4 (Triad) show the higher values because each has two memory loads and one store compared to one of each for the first two kernels. Different compilers can produce different memory performance on the same code. Particularly striking is the boost in performance for the Intel compiler as

demonstrated in Figure 4.1 showing the ratio of optimized to non-optimized bandwidth. Note that since this is serial benchmark one process on a dual Xeon node has a large performance boost from the Intel compiler with optimization enabled.

Table 4.2 Memory bandwidth (MB/second) for the STREAM benchmarks with four compilers on the Pentium 4 Xeon (3.06 GHz, 1MB L3 cache) with optimization.

N	Absoft	Intel	Lahey	Portland
Copy	1356.58	2677.82	1138.79	1322.31
Scale	1351.96	2675.59	1207.55	1327.8
Add	1660.76	2843.6	1227.62	1678.32
Triad	1662.69	2802.1	1230.77	1684.21

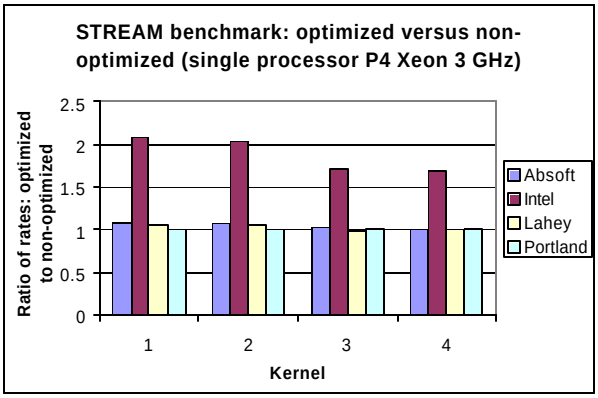

Fig. 4.1 Ratio of memory bandwidth of four different compilers on the Pentium 4 Xeon for results with and without compiler optimization for the STREAM benchmark.

4.2 POM Results

For the Princeton Ocean Model whole code execution was measured with calls to the Fortran 90/95 system_clock routine for all compilers. Results with no SSE enabled are shown in Table 4.3 for the three problem sizes of Table 3.2. For the largest problem size the Lahey compiler is noticeably less efficient than the others and this is due to the requirement of the --long option for large integers.

The gain in performance when SSE is enabled is shown in Figure 4.2 for the three compilers that have switches for SSE. The precise performance gain for each compiler is shown in Table 4.4. This shows the relative gain (SSE versus no SSE) as a percentage, for each of the three problem sizes.

Table 4.3 Execution times (seconds) for the POM algorithm with four compilers on the Pentium 4 Xeon (3.06 MHz, 1MB L3 cache) without SSE enabled.

GRID	Absoft	Intel	Lahey	Portland
1	167.7	156.1	189.5	190.1
2	1925.9	1518.9	2809.7	1756.7
3	8685.2	7432.3	12731.8	8764.8

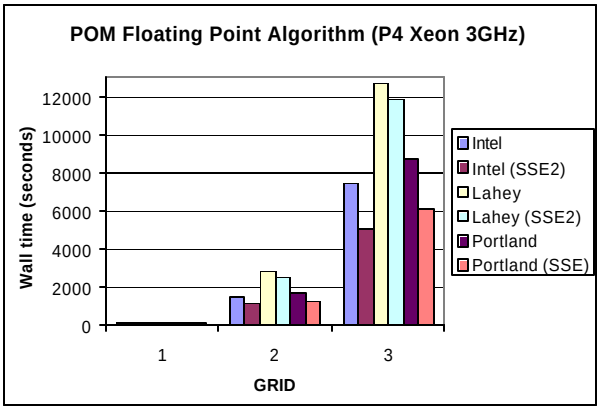

Fig. 4.2 Execution times of three compilers for the POM floating point algorithm without and with SSE enabled on the Pentium 4 Xeon.

Table 4.4 Percentage gain in execution times for the POM algorithm with three compilers when SSE/SSE2 is enabled.

GRID	Intel	Lahey	Portland
1	1.2	-0.3	11.2
2	25.5	10.3	27.6
3	31.5	6.7	29.8

The largest gains in performance are for the largest problem size, where it is clear from Fig.4.2, that the Intel compiler delivers the best performance once SSE is enabled.

4.3 MM5 Results

For MM5 whole code execution was measured with the Linux time command for both the Intel and Portland compilers and relative performance is compared in Figure 4.3. As a base-line the noopt group disables all optimization for either compiler and there is a clear improvement in performance when the opt group of switches is used (with –O2). However, there is no exact equivalence between the two compilers for the vector group of switches because the pgf90 compiler allows separation of

vector and SSE instruction implementation, whereas the Intel compiler does not (and therefore the Intel entry is left blank). For the baseline, with the noopt choice, the Intel compiler lags pgf90, but the situation is reversed for the opt group of switches where the Intel compiler delivers a 20.5% smaller elapsed time. However, the largest difference between the two compilers is for the SSE group of switches, where the Intel compiler delivers a 33.8% smaller elapsed time. It is clear that the Intel compiler delivers the best performance once SSE is enabled.

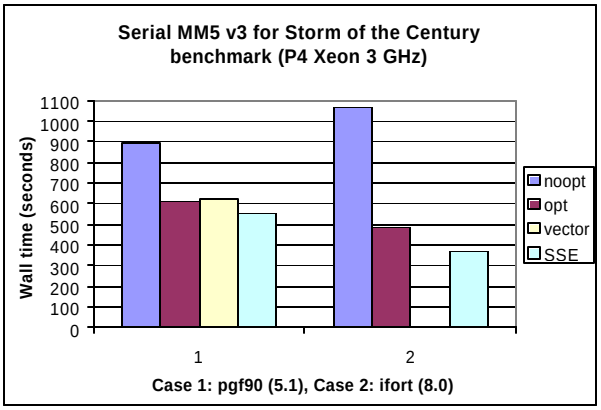

Fig. 4.3 Elapsed time for MM5 with the SOC benchmark for the Portland 5.1 (Case 1), and Intel 8.0 (Case 2) compilers.

4.4 CAMx Results

Only the pgf90 (5.1) compiler was used in this benchmark with five groups of optimization switches similarly to what was done in the POM and MM5 cases. Whole code execution was measured with the time command for the results of the TCEQ 2000 Mid-course Review Scenario for the SIP with an episode from 8/22 to 8/31 in the Houston Greater Metro area. Results are presented in Figure 4.4. The grouping of switches is: noopt (-O0), opt (-O2), vect (-fast –Mvect), SSE (-fast –Mvect=sse), and FSSE (-fastsse). The remaining switches used where: -tp p7 –pc 64 –Mdalign –Mextend -Mnoframe –Mlfs –byteswapio –Wl, -Bstatic. The results of this analysis are remarkable in that this increasing hierarchy of optimizations resulted in no noticeable reduction in wall clock time. This suggests that CAMx is a scalar dominated code that received none of the benefits of this architecture such as vector pipelining, cache, and SSE2 instruction sets with a compiler that has been shown to enable such benefits on vectorizable code (e.g. POM, MM5).

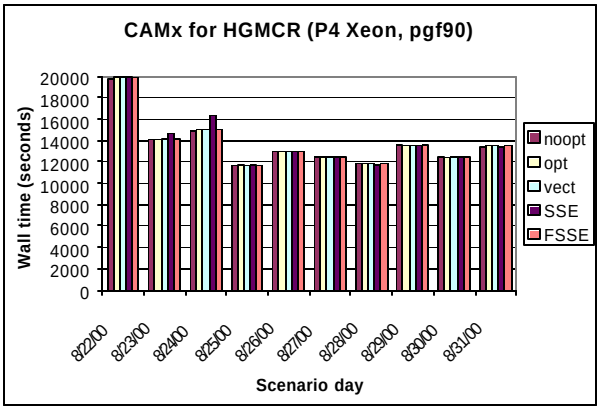

Fig. 4.4 Elapsed time for the CAMx benchmark with the Portland 5.1 compiler for five compiler optimization choices.

5.0 CONCLUSIONS

Performance results of four fortran compilers on the Intel Pentium 4 Xeon were presented. For such commodity hardware memory bandwidth is known to be one of the principle performance bottle-necks with consequences for large problem sizes. The STREAM benchmark showed an exceptional bandwidth enhancement for the Intel compiler when optimizations are enabled.

Wall clock time was also measured for three fortran codes used in ocean, atmospheric, and environmental models: POM, MM5 and CAMx. A fundamental result is that models developed to effectively utilize vector register architectures, also receive very significant performance boosts from optimization and SSE2 instructions on cache-based commodity hardware. This was observed to be the case for all compilers with switches to enable SSE2 instructions. Performance gains (with SSE2 enabled) increased steadily with increasing problem size, to reach ~30% (POM), and differences ~34% were observed in performance between compilers (MM5). The exceptionally poor performance of CAMx suggests a redesign of the code is advisable if performance improvement on commodity hardware is desired.

The author gratefully acknowledges the willingness of Dr. Harvey Jeffries and his Ph.D. student, Byeong-Uk Kim, to share the UT/UNC version of CAMx and for help on how to use it.