

# *Some Practical Suggestions for Performing Gaussian Benchmarks on a pSeries™ 690 System*

Carlos P. Sosa, IBM, pSeries High-Performance Computing, Chemistry and Life Sciences  
Development Solutions, Minneapolis, MN

Stefan Andersson, IBM, Webserver Sales Technical Support, Germany

February 6, 2002

©2002 International Business Machines Corporation, all rights reserved

## Abstract

*Gaussian98* is a connected series of programs from Gaussian, Inc., that can perform a variety of semi-empirical, *ab initio*, and density functional theory calculations. For more than 20 years, the *Gaussian* program has been extensively used at universities, and in the pharmaceutical and chemical industries to carry out basic research in the simulation and elucidation of new pharmaceuticals or materials. *Gaussian* has introduced new algorithmic improvements that leverage the latest IBM® eServer architectures, in particular, the pSeries™. In this study, we present a series of benchmarks to analyze the performance of *Gaussian* as a function of multiple input parameters. This study also provides a guide to optimize system resources management on IBM pSeries.

## Introduction

The IBM systems based on the POWER processor were introduced in February of 1990. It was based on a multiple chip implementation of the POWER architecture [1,2,4]. This technology is now commonly referred to as POWER1. The model introduced included an eight KB instruction cache (I-cache) and either a 32 KB or 64 KB data cache (D-Cache). They had single floating point unit (FPU) capable of issuing one compound floating-point multiply add (FMA) operation each cycle, with a latency of only two cycles. Therefore, the peak MFLOPS rate was equal to twice the MHz rate.

Announced in September 1993, the Model 590 was the first RS/6000 machine based on the POWER2 architecture [1,2,4]. The most significant improvement introduced with the POWER2 architecture was that the FPU contained two 64-bit execution units, so that two FMAs instruction could be executed in each cycle. In addition, a second fixed-point execution unit and several new hardware instruction were added:

- Quad-word storage instructions: The quad-word load instruction moves two adjacent double-precision values into two adjacent floating-point registers.
- Hardware square root instruction
- Floating-point to integer conversion instruction

Although the Model 590 ran only with a marginally faster clock than the POWER1-based Model 580, the architectural improvements listed above, combined with a larger 256 KB D-Cache, enabled it to achieve far greater level of performance. As an example, the MFLOPS achieved in the Linpack TPP benchmark went from 104 to 237.

In October 1996, IBM announced the RS/6000 Model 595 [1,2,4]. This was the first machine to be based on the P2SC (POWER2 Super Chip) processor. This is a single chip implementation of the POWER2 architecture, enabling the clock speed to be increased further. The Model 595 runs at 135 MHz, and the fastest P2SC processors, found in Model 397 workstation and RS/6000 SP Thin4 nodes, run at 160 MHz, with a theoretical peak performance of 640 MFLOPS.

The POWER3 [3] microprocessor introduced a new generation of 64-bit processors especially designed for high performance and visual computing applications. POWER processors are the replacement for the POWER2 and P2SC in high-end RS/6000 workstations and technical servers.

The first RS64 processor was introduced in September of 1997 and was the first step into 64-bit computing for RS/6000 [1,2,4]. While the POWER2 product had strong floating-point performance, this series of products emphasized strong commercial server performance. It ran at 125 MHz with a 2-way associative, 4 MB L2 cache and had a 64 KB L1 instruction cache, a 64 KB L1 data cache, one floating-point unit, one load-store unit, and one integer unit. Systems were designed to use up to 12 processors. pSeries products using the RS64 were the first pSeries products to have the same processor and memory system as iSeries™ products.

In September 1998, the RS64-II was introduced. It was a different design from the RS64 and increased the clock frequency to 262 MHz. The L2 cache became 4-way set associative with an increase in size to 8 MB. It had a 64 KB L1 instruction cache, a 64 KB L1 data cache, one floating-point unit, one load-store unit, two integer units, and a short in-order pipeline optimized for conditional branches.

With the introduction of the RS64-III in the fall of 1999 [4], this design was modified to use copper technology, achieving a clock frequency of 450 MHz, with a L1 instruction and data cache increased to 128 KB each. This product also introduced hardware multithreading for use by AIX®. Systems were designed to use up to 24 processors.

In the fall of 2000, this design was enhanced to use silicon on insulator (SOI) technology, enabling the clock frequency to be increased to 600 MHz. The L2 cache size was increased to 16 MB on some models. Continued development of this design provided processors running at 750 MHz. The most recent version of this microprocessor was called the RS64-IV.

The new POWER4 processor continues the evolution [4]. The POWER4 processor chip contains two microprocessor cores, chip and system pervasive functions, core interface logic, a 1.41 MB level-2 (L2) cache and controls, the level-3 (L3) cache directory and controls, and the fabric controller that controls the flow of information and control data between the L2 and L3 and between chips. Table I provides a processor comparison.

**Table 1. Processor Comparison**

	<b>POWER3</b>	<b>RS64-III</b>	<b>POWER4</b>
Frequency [MHz]	375	450	1,300
Fixed Point Units	3	2	2
Floating Point Units	2	1	2
Load/Store Units	2	1	2
Branch/Other Units	1	1	2
Dispatch Width	4	4	5
Branch Prediction	Dynamic	Static	Dynamic
I-cache size [kb]	32	128	64
D-cache size [kb]	128	128	32
L2-cache size [mb]	8	16	1.41 <sup>a</sup>
L3-cache size [mb]	N/A	N/A	512 <sup>b</sup>

Data Prefetch	Yes	No	Yes
---------------	-----	----	-----

<sup>a</sup> Shared between two cores

<sup>b</sup> Shared between 32 cores

Each microprocessor contains a 64 KB level-1 instruction cache, a 32 KB level-1 data cache, two fixed-point execution units, two floating-point execution units, two load/store execution units, one branch execution unit, and one execution unit to perform logical operations on the condition. Instructions dispatched in program order in groups are issued out of program order to the execution units, with a bias towards oldest operations first. Groups can consist of up to five instructions, and are always terminated by a branch instruction. The processors on the first IBM POWER4-equipped servers, the IBM pSeries 690 Model 681 servers, operate at either 1100 MHz or 1300 MHz. Table 2 compares SPECint2000 and SPECfp2000 for the latest POWER-based processors.

**Table 2. Comparative Metrics<sup>a</sup>**

Metric	POWER3 450 MHz	RS64-III 450 MHz	POWER4 1300 MHz
SPECint2000	335	234	814
SPECfp2000	433	210	1,169

<sup>a</sup> <http://www.spec.org>

The combined effort between software and hardware is what makes possible to carry out calculations that just a few years ago were unthinkable. An RHF/6-31G\*\* single point energy calculation on triamino-trinitro-benzene (TATB) using *Gaussian88* on an IBM RS/6000 model 550 used to take 4.5 hours [5]. The same calculation using *Gaussian92* took about 1 hour [5]. Today this calculation takes less than 2 minutes on the latest IBM POWER3 systems.

In this study, we consider a series of benchmarks, not only to compare the POWER3 systems against the POWER4, but also to look at the performance of POWER4 HPC (code name Regatta-HPC) versus the POWER4 Turbo (code name Regatta-H). In the next section we summarize the design features of the systems employed in this study. We also devote a section to look at how *Gaussian* has been parallelized. In the last four sections we describe the benchmarks and discuss the results in terms of parallel speedup and efficiency. A summary is provided that we hope will serve as a guidance for running or selecting appropriate parallel benchmarks.

## Design Features

Three different types of IBM pSeries servers were used in this study: 375 MHz POWER3-II multiprocessor (SMP) and two pSeries 690 servers. One corresponds to the 1.3 GHz POWER4 pSeries 690 HPC (High Performance Computing) and the other system was 1.3 GHz POWER4 pSeries 690 Turbo. For simplicity, we will refer to them as the HPC system and Turbo system.

The pSeries 690 server is the latest UNIX<sup>®</sup> server from IBM and provides a new architecture [4,6]. At the core of its architecture is the POWER4 Multi-chip Module (MCM). The building blocks for the systems utilized here are: an 8-way MCM Turbo running at 1.3 GHz and a 4-way MCM HPC running at 1.3 GHz. One of the differences of the HPC system (4-way MCM) is that it is optimized for data intensive applications that require larger memory bandwidth per core. Each POWER4 HPC processor chip contains one rather than two cores; the L2 cache is dedicated for each core. On the other hand, the Turbo system (8-way MCM) has two cores per L2 cache, hence, one MCM is 8-way. A full description of the POWER4 architecture is beyond the scope of this work. For further details, see refs. [4] and [6]. However, in this section we will provide an overview of the most important features of this architecture.

Each processor chip on the pSeries 690 consists of either one or two cores, an L2 cache that runs at the same speed of the microprocessor, the microprocessor interface unit which is the interface for each microprocessor to the rest of the system, the directory and cache controller for the L3 cache, the fabric bus controller, and a GX bus controller that enable I/O devices to connect to the Central Electronic Complex (CEC). The L3 cache is a new component not available on the POWER3 architecture. The L3 caches are mounted on a separate module.

The 375 MHz POWER3-II system consisted of 16 processors, 24 GB of memory, and 8 MB L2 cache. The Turbo system 1.3 GHz consisted of 32 processors and 8X32 GB memory cards for a total of 256 GB of memory. The HPC system 1.3 GHz had a total of 8X32 GB memory cards for a total of 256 GB of memory.

All the timings in this work correspond to elapsed time. Timings were measured by using the time printed by each link using the #p option. *Gaussian* prints the clock time and CPU time. This information was used as input for a utility program that tabulates timings and speedups.

## Parallel Gaussian

*Gaussian* [7], a connected series of programs, can be used for performing different kind of electronic structure calculations, for example semi-empirical, *ab initio*, and density functional theory calculations. *Gaussian* consists of a collection of programs commonly known as links; each link communicates through disk files. Links are grouped into overlays [8]. Links are independent executables located in the g98 directory and labeled as lxxx.exe; where xxx is the unique number of each link. In general, overlay zero is responsible to start the program, this includes reading of the input file. After the input file is read the route card (keywords and options that specify all the *Gaussian* parameters) is translated into a sequence of links. Overlay 99 (l9999.exe) terminates the run, in most cases l9999.exe finishes with an archive entry (brief summary of the calculation).

As previously pointed out, the *Gaussian* architecture on shared-addressable or distributed memory systems is basically the same [9]. Each link is responsible for continuing the sequence of links by invoking `exec()` system call to run the next link. The links that have been parallelized will run on multiple processors. The links that run sequentially are mainly links that are responsible for setting up the calculation and assigning symmetry. Although in previous publications we have summarized all the links that run in parallel [9], it is important to note that, the use of multiple processors benefit from calculations that make use of the PRISM routines [10].

In a self-consistent field (SCF) scheme, the two-electron integrals are part of the Fock matrix[11]:

$$F_{\mu\nu} = H_{\mu\nu} + \sum_{\lambda} \sum_{\sigma} P_{\lambda\sigma} [(\mu\nu|\lambda\sigma) - 1/2(\mu\lambda|\nu\sigma)] \quad (1)$$

where  $H$  represents the core Hamiltonian.  $m, n, l, s$  are atomic orbital indices. The quantities  $(mn|ls)$  are two-electron repulsion integrals. In *Gaussian*, these quantities are computed once and stored or recomputed many times as needed, depending on the memory available and the algorithm chosen.

In density functional theory (DFT), eq. (1) can be rewritten by replacing the last term by the well-known exchange-correlation term  $F^{XC}$  [12].

$$F_{\mu\nu} = H_{\mu\nu} + \sum_{\lambda} \sum_{\sigma} P_{\lambda\sigma} (\mu\nu|\lambda\sigma) + F_{\mu\nu}^{XC} \quad (2)$$

In *Gaussian*, the two-electron integrals are parallelized by distributing batches of integrals among all the available processors (normally selected using the `%nproc` keyword in the input file). This procedure is illustrated in Figure 1. The main loop over  $N_{\text{PROCS}}$  (number of available processors) determines the number of integrals that will be distributed among the processors. Each task looks at  $1/N_{\text{PROCS}}$  of the shell quartets, discarding those that do not need to be done due to symmetry, cutoffs, or the fact that they have already been done. All the integrations are carried out in these loops. Within these series of loops and still in the main  $N_{\text{PROCS}}$  loop, the last loop sums up the contributions to the Fock matrix, derivative matrices, or density matrices, depending on the types of integrals computed. The next loop outside the first  $N_{\text{PROCS}}$  loop, is another  $N_{\text{PROCS}}$  loop that adds all the contributions to the Fock matrix together in a serial block of code. This scheme can be exploited to compute first and second two-electron integral derivatives, first and second one-electron integral derivatives and electrostatic potential integrals.

```

loop over N_PROCS
  loop over total angular momentum
    loop over degrees of bra and ket contraction
      do integrals for 1/N_PROCS of shell quartets
    endloop
  endloop
  add integral contributions to partial Fock matrix
endloop
loop over N_PROCS (sequential code)
  add 1/N_PROCS Fock matrix contributions
endloop

```

**Figure 1.** Parallel loops to compute two-electron integrals in *Gaussian*.

The parallelization of the Fock matrix in *Gaussian98* on IBM shared-memory architectures has been accomplished by using standard UNIX `fork` and `wait` commands [13]. `fork()` creates a new process. Upon successful completion, the `fork` subroutine returns a value of 0 to the child process and returns the process ID of the child process to the parent process. Otherwise, a value of -1 is returned to the parent process, no child process is created. In *Gaussian98* `fork()` is called within the  $N_{\text{PROCS}}$  loop and each new child executes a batch of integrals. All the children that have completed their task wait prior to entering the sequential `do` loop to add all their contributions.

## Selected Benchmarks

Similarly as in previous studies [9,13], we consider four major characteristics when studying parallel performance on the IBM SP or selecting benchmarks: job type, theoretical method, basis set size, and molecular size. The job type corresponds to a single point energy, a gradient calculation, or calculation of second derivatives. Normally single point calculations are used to compute accurate energies at a level of theory that is too expensive to carry out a full geometry optimization for any medium-to-large size systems. Due to the importance of molecular structure in chemistry, a large majority of calculations are geometry optimizations using Hartree-Fock or Density Functional Theory. This type of calculations are normally followed by a frequency calculation. To a lesser extent, geometries are also carried out at the MP2 level of theory.

There are many options for carrying out calculations using *Gaussian*. In this study we tried to select job types that reflect how users are currently running *Gaussian*. This by no means represent the only options used in the program but they represent a large percentage of calculations carried out at computer centers. They also correspond to many of the benchmarks carried out for hardware procurements.

In this study we have chosen the following types of calculations: single-point energy (SP) calculations at different levels of theory, FORCE, this type of calculation corresponds to an SP calculation followed by the calculation of the first derivatives of the energy with respect to the position of the atoms in the molecule. The time required for a geometry optimization is a multiple of the time needed for a FORCE calculation. We also carried out frequency as the third type of calculations. *Rather than doing a full geometry optimization, a FORCE calculation is recommended for benchmarks that involve hardware performance.* It is equivalent as doing one cycle of the optimization and should provide a good approximation for the performance of an optimization calculation.

A large number of approximate theoretical methods have been reported in the literature [11]. These methods range in accuracy and computational cost. Since *Gaussian* provides most of these methods it is important to understand how is that they perform as a function of system resources. In this study we refer to system resources as the number of processors, memory and disk space needed for optimal performance. The theoretical methods chosen in this study have been extensively discussed in the literature [11] and it is beyond the scope of this work to describe these methods. The approximation used in this work correspond to Hartree-Fock [11], the three parameter density functional theory of Becke (B3-LYP) [14], and Configuration Interaction Singles (CIS) energy and gradients [15].

The cases used in this study correspond to a subset selected from a previous study [9] augmented with a FORCE calculation using Valinomycin. The system used for cases I and III is a-pinene. Case I is an SP calculation at the HF level of theory using 6-311G(df,p) basis sets. Case II corresponds to a SP FORCE calculation on a fairly large system (Valinomycin). The third case tested is a-pinene frequency calculation at the B3-LYP/6-31G(d) level of theory. Cases II and III exercise several of the links that run in parallel. To better understand which links run in parallel Table 3 shows the links that are executed during these calculations and the ones that run in parallel. Although each of these calculations execute several links, it is important to note that most of the CPU time is spent in the parallel links. This ensures good scalability.

**Table 3. List of Links Executed in Cases I-IV<sup>a</sup>**

Case I <sup>b</sup>	Case II <sup>c</sup>	Case III <sup>d</sup>	Case IV <sup>e</sup>
l1.exe	l1.exe	l1.exe	l1.exe
l101.exe	l101.exe	l101.exe	l101.exe
l202.exe	l103.exe	l103.exe	l103.exe
l301.exe	l202.exe	l202.exe	l202.exe
l302.exe	l301.exe	l301.exe	l301.exe
l303.exe	l302.exe	l302.exe	l302.exe
l401.exe	l303.exe	l303.exe	l308.exe
<i>l502.exe</i>	l401.exe	l401.exe	l303.exe
l601.exe	<i>l502.exe</i>	<i>l502.exe</i>	l401.exe
l9999.exe	l601.exe	l801.exe	<i>l502.exe</i>
	l701.exe	l1101.exe	l801.exe
	l702.exe	l1102.exe	<i>l914.exe</i>
	<i>l703.exe</i>	<i>l1110.exe</i>	<i>l1002.exe</i>
	l716.exe	<i>l1002.exe</i>	l601.exe
	l103.exe	l601.exe	l701.exe
	l9999.exe	l701.exe	l702.exe
		l702.exe	<i>l703.exe</i>
		<i>l703.exe</i>	l716.exe
		l716.exe	l103.exe
		l103.exe	l9999.exe
		l9999.exe	

<sup>a</sup> Parallel link are in *italics*.

<sup>b</sup> HF/6-311G(df,p).

<sup>c</sup> B3-LYP/3-21G FORCE.

<sup>d</sup> B3-LYP/6-31G(d) FREQ.

<sup>e</sup> CIS/6-31++G FORCE.

Cases IV computes excited states using CI single excitations FORCE calculation with the 6-31++G basis sets [15] on acetyl phenol. This set of benchmarks represent small-large systems to test speedup and efficiency for most parallel links. All the geometries are available from Ref. [16].

## Results and Discussion

In this study, as in our previous reports, we look at performance in terms of speedup [9]. Speedup ( $S$ ) is defined as the ratio of the serial run time (elapsed time,  $t_s$ ) over the time that it takes to do the same problem in parallel (elapsed time,  $t_p$ ).

$$S = \frac{t_s}{t_p} \quad (3)$$

Efficiency ( $e$ ) is the fraction of time that a processor is doing useful work. This measurement also indicates or provides an indirect indicator for the percentage of parallel code needed for linear or nearly linear scalability.

$$e = \frac{S}{N_{PROCS}} \quad (4)$$

To compare scalability, we look at the measured speedup against ideal speedup, rather than using an extrapolated speedup as we have done in the past. We have taken this approach because we want to compare how these two machines perform, rather than analyzing *Gaussian's* scalability. Although it is important to note that clock speed of the POWER4 is significantly faster than the POWER3 and this may affect scalability.

Table 4 illustrates the performance between the POWER3 and the POWER4 (Turbo and HPC) for a Hartree-Fock single-point energy calculation using medium size basis sets (Case I). Parallelization is achieved mainly in l502.exe (almost all of the CPU is spent in l502.exe). Table 4 looks at the scalability of l502.exe (or SCF procedure) and of the entire calculation (Total time or time to solution). The column under the label "Total" includes the overhead introduced by the links that run sequentially. In the case of single processor, the improvement in going from the POWER3, 375 MHz to the POWER4, 1.3 GHz (Turbo) is a factor of approximately 3.34 for l502.exe and the total time. In this particular case and for single processor performance, the improvement of going to the HPC architecture is less 1%, this is not surprising.

**Table 4. Hartree-Fock single-point energy calculations on a-pinene(C<sub>10</sub>H<sub>16</sub>)<sup>a</sup>**

Number of Processors	l502.exe <sup>b</sup>	S <sup>c</sup>	e <sup>c</sup>	Total <sup>b,d</sup>	S <sup>c</sup>	e <sup>c</sup>
1						
POWER3	2819	1.00	1.00	2827	1.00	1.00
Turbo	843	1.00	1.00	847	1.00	1.00
HPC	841	1.00	1.00	845	1.00	1.00
2						
POWER3	1385	2.04	1.00	1394	2.03	1.00
Turbo	439	1.92	0.96	444	1.91	0.96
HPC	447	1.88	0.94	451	1.87	0.94
4						
POWER3	702	4.02	1.00	716	3.95	0.99
Turbo	282	2.99	0.75	288	2.94	0.74
HPC	253	3.32	0.83	259	3.26	0.82
8						
POWER3	364	7.74	0.97	385	7.34	0.92
Turbo	164	5.14	0.64	173	4.90	0.61
HPC	145	5.80	0.73	152	5.56	0.70
16						
POWER3	196	14.38	0.90	232	12.19	0.76
Turbo	97	8.69	0.54	111	7.63	0.48
HPC	84	10.01	0.63	97	8.71	0.54
32						
Turbo	63	13.38	0.42	85	9.96	0.31

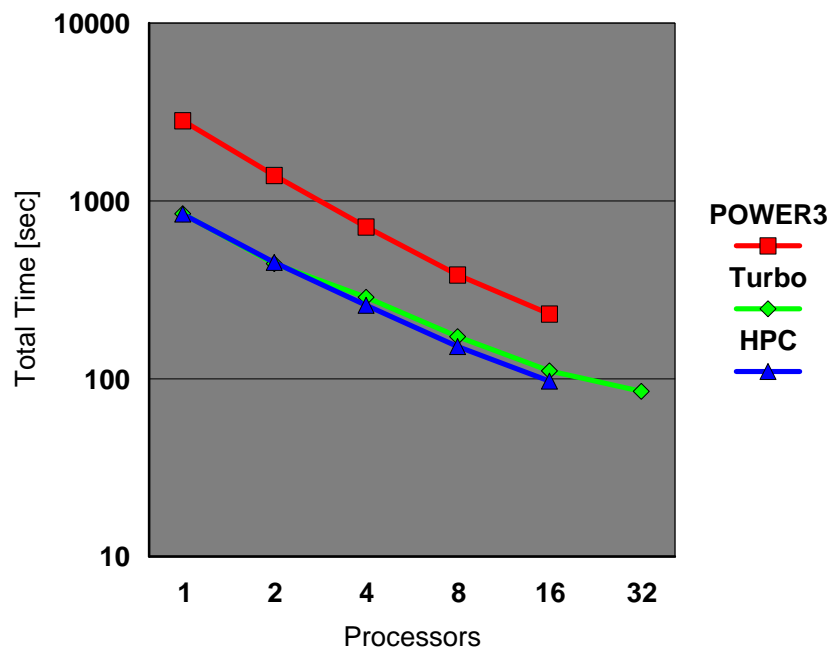
<sup>a</sup> Total of 346 basis functions; basis sets are 6-311G(df,p); C1 symmetry.

<sup>b</sup> All timings are in seconds and correspond to elapsed time.

<sup>c</sup> Speedup and efficiency.

<sup>d</sup> Total time to complete the run.

The scalability observed in this case is what one might expect (see Figure 2 for scalability of the entire run). With 16 processors, the POWER3 shows an efficiency of 0.90 (or 90%) for l502.exe. Part of the 10% overhead may be attributed to the diagonalization that is done serially. On the other hand, the efficiencies for the Turbo and HPC machines are 0.54 and 0.63, respectively. The lower speedups may indicate that additional code or system tuning might be required due to the substantially faster processors. As the number of processors increases, the advantage of a dedicated secondary-cache (L2 cache) becomes evident with the higher efficiency on the HPC machine. In this case, a dedicated L2-cache reduces contention and increases the efficiency when using 16 processors to almost 20%. Also, it is interesting to note that as the number of processors increases and the parallel regions are completed quicker, the serial overhead begins to dominate the calculation. From Table 4 we see that the efficiency of 0.90 for l502.exe goes down to 0.76 for the entire calculation. Larger basis sets are required to maintain a very high efficiency with large number of processors. In other words, if the molecule remains the same (number and type of atoms), to maintain a high efficiency we need to increase the basis sets to increase the number of integrals that are computed by each processor.



**Figure 2. Total time as a function of processors for the Hartree-Fock single point energy calculation on a-pinene.**

Table 5 summarizes a FORCE calculation. In addition to l502.exe, this case also illustrates the scalability for l703.exe and of course, the scalability for the complete run when using DFT methods. Although these systems (and basis sets) are not the same, an approximate comparison may be obtained between l502.exe in Table 4 and Table 5.

**Table 5. B3-LYP FORCE Calculation on Valinomycin (C<sub>54</sub>H<sub>90</sub>N<sub>6</sub>O<sub>18</sub>)<sup>a</sup>**

Number of Processors	l502.exe <sup>b</sup>	S <sup>c</sup>	l703.exe <sup>b</sup>	S <sup>c</sup>	Total <sup>b,d</sup>	S <sup>c</sup>
1						
POWER3	36,597	1.00	10,343	1.00	47,084	1.00
Turbo	12,437	1.00	3,039	1.00	15,541	1.00
HPC	12,156	1.00	2,997	1.00	15,220	1.00
2						
POWER3	19,845	1.84	4,947	2.09	24,933	1.89
Turbo	6,509	1.91	1,651	1.84	8,222	1.89
HPC	6,486	1.87	1,634	1.83	8,183	1.86
4						
POWER3	11,160	3.28	2,758	3.75	14,069	3.35
Turbo	4,421	2.81	1,095	2.78	5,583	2.78
HPC	4,128	2.94	1,025	2.92	5,220	2.92
8						
POWER3	6,319	5.79	1,589	6.51	8,073	5.83
Turbo	2,692	4.62	626	4.85	3,389	4.59
HPC	2,474	4.91	565	5.30	3,111	4.89
16						
POWER3	4,044	9.05	995	10.39	5,249	8.97
Turbo	1,788	6.96	398	7.64	2,273	6.84
HPC	1,646	7.39	356	8.42	2,088	7.29
32						
Turbo	1,370	9.08	294	10.34	1,777	8.75

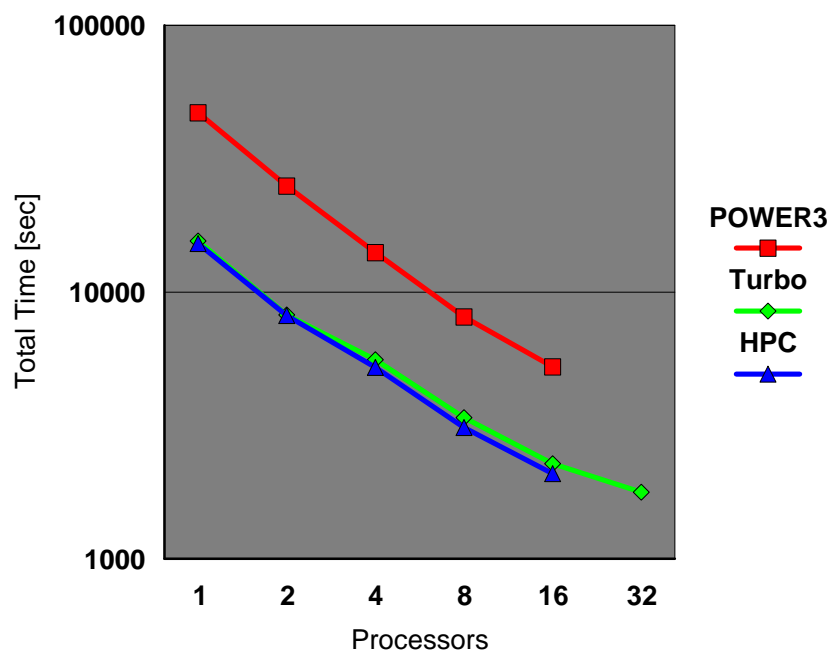
<sup>a</sup> Total of 882 basis functions; basis sets are 3-21G; C1 symmetry.

<sup>b</sup> All timings are in seconds and correspond to elapsed time.

<sup>c</sup> Speedup.

<sup>d</sup> Total time to complete the run.

Similar to the previous case, we look at the single processor performance. The Turbo system shows speedups of 2.94, 3.40, and 3.03 for l502.exe, l703.exe, and for the total time, respectively. The HPC in this case is faster than the Turbo by 2.26%, 1.38%, and 2.07% for l502.exe, l703.exe, and the total time, respectively. The results between the Turbo and HPC are consistent with the previous case. Figure 3 shows scalability for the entire run.



**Figure 3. Total time as a function of processors for the B3-LYP FORCE calculation on Valinomycin.**

In the case of l502.exe the parallel results are similar to a-pinene, although, this is a larger system and has more basis functions but the basis sets are only 3-21G (smaller). This case shows slightly lower scalability than the previous case which may be attributed to less integrals per batch; either due to the small basis sets or cutoff used for the integrals for large systems (loops sizes to compute integrals are also shorter). The HPC system shows an improvement of approximately 7% in scalability with 16 processors when compared to the Turbo.

l703.exe computes the first and second-derivatives of the two-electron integrals. In this case we see that the POWER3 shows superlinear scalability with two processors and very good scalability up to 16 processors. The efficiency on the POWER3 with 16 processors is 65%. On the other hand, the efficiency on the Turbo or HPC is about 50%.

Table 6 summarizes timings for a frequency calculation. A simple inspection of Table 3 reveals that in Case III there are several parallel links that are exercised in this example: l502.exe, l1110.exe, l1002.exe, and l703.exe. l502.exe and l703.exe have been discussed previously. l1002.exe solves the CPHF equations to produce the derivatives of the molecular orbital coefficients. And l1110.exe computes the two-electron contribution to the Fock matrix derivatives with respect to nuclear coordinates [8]. Since only the direct scheme of the atomic orbitals (AO) production integrals is parallelized, it is not surprising that l1002.exe does not show the same type of scalability as l703.exe not l1110.exe [9].

**Table 6. B3-LYP Frequency Calculation on a-pinene (C<sub>10</sub>H<sub>16</sub>)<sup>a</sup>**

Number of Processors	l502.exe <sup>b</sup>	l1110.exe <sup>b</sup>	l1002.exe <sup>b</sup>	l703.exe <sup>b</sup>	Total <sup>b,d</sup>	S <sup>c</sup>
1						
POWER3	1040	2862	3747	3494	11156	1.00
Turbo	355	1244	1469	1531	4603	1.00
HPC	355	1242	1463	1529	4594	1.00
2						
POWER3	510	1426	1956	1746	5654	1.97
Turbo	182	631	777	807	2403	1.92
HPC	187	642	790	803	2428	1.89
4						
POWER3	259	776	1080	959	3095	3.60
Turbo	108	369	477	454	1416	3.25
HPC	99	353	443	446	1350	3.40
8						
POWER3	140	420	656	550	1802	6.19
Turbo	63	202	313	267	857	5.37
HPC	56	191	285	260	804	5.71
16						
POWER3	80	218	452	294	1109	10.06
Turbo	37	109	227	145	541	8.51
HPC	32	100	207	138	501	9.17
32						
Turbo	25	62	192	86	406	11.34

<sup>a</sup> Total of 346 basis functions; basis sets are 6-31G(d); C1 symmetry.

<sup>b</sup> All timings are in seconds and correspond to elapsed time.

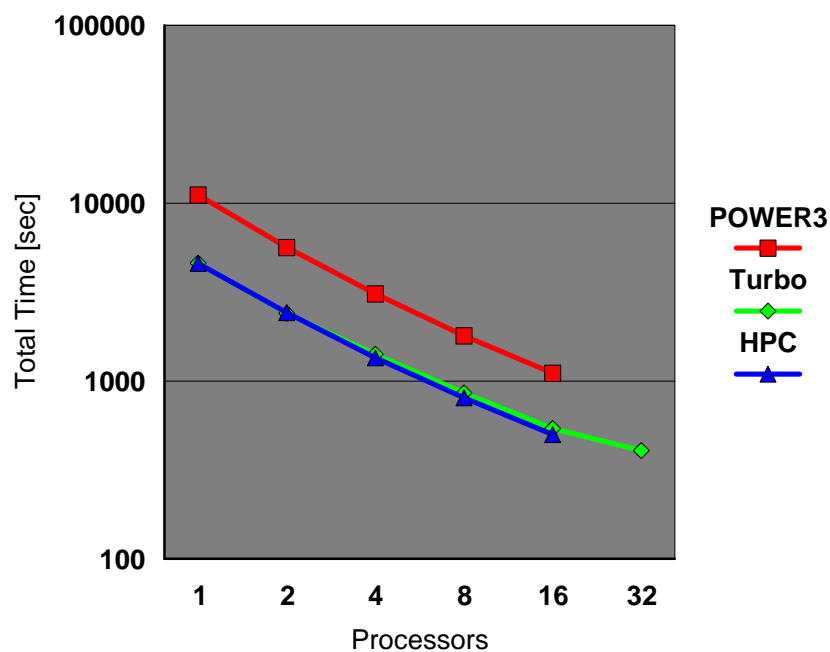
<sup>c</sup> Speedup.

<sup>d</sup> Total time to complete the run.

In the case of the POWER3, while l502.exe and l1110.exe show superlinear scalability with 2 processors, the speedup for l1002.exe is 1.92. On the Turbo and HPC, although, we do not see superlinear scalability the trend is the same with 2 processors as on the POWER3. If we do the same type of analysis with 16 processors, we will see that scalability of l1002.exe has decreased approximately 36%, 43%, and 36% when compared to l502.exe, l1110.exe, and l703.exe, respectively. Parallelization of additional terms in the solution of CPHF equations will help this situation [17].

On the other hand, l1110.exe shows excellent scalability, similar to l703.exe. l1110.exe shows an almost linear scalability up to 16 processors. S is 13.13, 11.41, and 12.42 for the POWER3, Turbo, and HPC, respectively.

Finally, Figure 4 summarizes the performance of the time to solution, that is, the time that it takes to complete the entire calculation. The figure also illustrates the difference in performance between the POWER3 and the POWER4. The difference between Turbo and HPC becomes more significant as the number of processors increases.



**Figure 4. Total time as a function of processors for the B3-LYP frequency calculation on a-pinene.**

Table 7 summarizes elapsed timings and total speedups for acetyl-phenol (Case IV). This benchmark consists of a CI singles energy and FORCE (gradients) calculation. This example illustrates the scalability of l914.exe. l914.exe computes excited states using CI singles excitations [15]. This type of calculation runs in parallel since the repulsion two-electron integrals contributing to the CI singles can be computed using the PRISM algorithm. The PRISM algorithm runs in parallel as shown in the previous sections.

**Table 7. CI-Singles Energy and Force Calculation on Acetyl Phenol (C<sub>8</sub>H<sub>8</sub>O<sub>2</sub>)**

Number of Processors	l502.exe <sup>b</sup>	l914.exe <sup>b</sup>	l1002.exe <sup>b</sup>	l703.exe <sup>b</sup>	Total <sup>b,d</sup>	S <sup>c</sup>
1						
POWER3	780	1323	694	177	2983	1.00
Turbo	231	432	205	52	924	1.00
HPC	229	430	203	52	917	1.00
2						
POWER3	347	655	339	87	1441	2.07
Turbo	110	225	107	27	476	1.94
HPC	112	231	108	28	484	1.89
4						
POWER3	201	343	175	44	783	3.81
Turbo	79	147	68	17	319	2.90
HPC	71	135	61	17	291	3.15
8						
POWER3	110	184	90	24	439	6.79
Turbo	49	92	40	11	203	4.55
HPC	43	81	36	10	181	5.07
16						
POWER3	67	106	48	14	290	10.29
Turbo	30	58	21	6	135	6.84
HPC	30	51	21	5	126	7.28
32						
Turbo	30	39	21	4	130	7.11

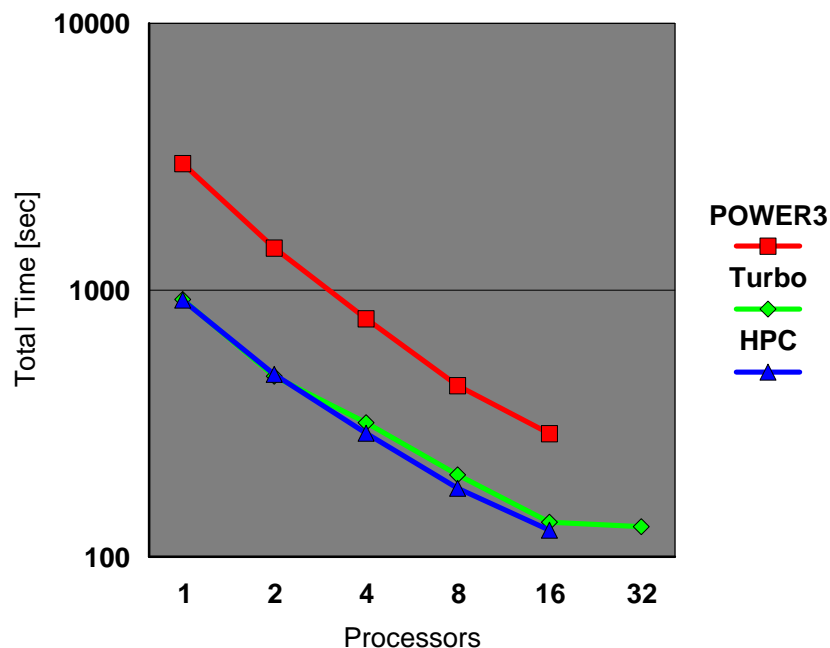
<sup>a</sup> Total of 154 basis functions; basis sets are 6-31++G; C1 symmetry.

<sup>b</sup> All timings are in seconds and correspond to elapsed time.

<sup>c</sup> Speedup.

<sup>d</sup> Total time to complete the run.

This link shows (see Table 7) good scalability consistent with l502.exe. Clearly, this case is becoming too small for the POWER4. All the links with 16 processors take less than 60 seconds. l703.exe takes only 5 seconds. Furthermore, this is illustrated in Figure 5, the scalability in going from 16 to 32 processors is almost flat. Figure 5 illustrates the total time as a function of processors for the CI-Singles energy and gradients.



**Figure 5. Total time as a function of processors for the CI-singles energy and FORCE calculation on acetyl phenol.**

## System Resource Management

Although system resource management is indirectly related to performance, the optimal use of the computer resources will affect the performance of a calculation, either by enhancing or decreasing it. Thus, we feel that it is important to include this section in this type of study, specially since *Gaussian* can make optimal use of memory, disk space, and multiple processors. In this study we shall concentrate in looking at memory optimization with respect to the number of processors. Memory allocation when using only 1 processor has been discussed in refs. [7,19] and shall not be covered here.

*Gaussian* uses a replicated data algorithm for the parallelization of the two-electron integrals [18]. On shared-memory machines this means that allocation of additional memory is required to run parallel calculations. The amount of memory required is directly proportional to the number of processors. In a previous study we have successfully used eq. (5) to optimally allocate memory when using multiple processors [13].

$$\text{Memory} (N_{\text{procs}} \text{ run}) = \text{Memory} (1_{\text{proc}} \text{ run}) + 0.75*(N_{\text{procs}}-1)*\text{Memory} (1_{\text{processor}} \text{ run}) \quad (5)$$

where memory ( $N_{\text{procs}}$  run) is the final amount of memory (either in MW or MB) that will be allocated to %mem in the input file. Memory ( $1_{\text{proc}}$  run) is the *minimal* amount of memory required to run a particular calculation on a single processor.  $N_{\text{procs}}$  is the total number of processors that will be used in a particular calculation. For instance, in case I, we allocated 8 MW for a single processor run and 98 MW for the 16 processors [16]. Prior to computing the two-electron integrals and its derivatives (if applicable) in parallel, *Gaussian* checks if there is enough memory to run in parallel. In cases where not enough memory has been allocated, the number of processors requested will be reduced until they fit in the provided amount of memory.

## Summary

In this study we have tried to provide information on the performance of several options in the *Gaussian* program. These options are commonly used at many supercomputer centers but by no means is an exhaustive set of benchmarks. Also, in this first exploratory work we have not looked at post-Hartree-Fock options in *Gaussian*. The levels of theory tested here correspond to Hartree-Fock, Density Functional Theory and CI Singles excitations. In part, they tend to show larger scalability with larger number of processors than post-Hartree-Fock methods. As first approximations to many higher-order methods, this set of methods are extensively used and certainly can provide valuable information when comparing the newest POWER architecture.

In general, SCF and DFT calculations with and without first and/or second derivatives have shown to run efficiently on the machines tested in this study. However, when designing benchmarks for scalability it is important to consider the system and basis sets. The first case that we studied here, a-pinene, has been used as a system that scales well with large number of processors [9]. A comparison of the scalability of l502.ee for cases I-III gives an approximate indication (it is approximate because each case uses different level of theory with slight different options and different convergence thresholds) that is order to achieve better scalability, basis set size is more important than either the level of theory or size of the molecule.

In terms of the three computer systems tested in this study, the speedup in going from the POWER3 to the POWER4 is very impressive. For these cases (cases I-IV) is between 2.4 and 3.4. Although the parallel scalability between these systems is similar, the POWER3 tends to show for 16 processors parallel speedups that are between 14 - 29 % higher than the POWER4. Although this difference may be attributed to faster processors, we are currently carrying out additional studies to address this difference [20].

Finally, the last case, case IV provides a direct method of computing excited states with minimal or no disk storage required. Due to its simplicity, this method provides a good approximation for large systems. This and other studies have shown that this is a good method to take advantage of multiple processors. l914.exe shows very good parallel speedups.

## Acknowledgements

We would like to thank S. Behling for reading this manuscript and making valuable comments. We also thank R. Panda, S. Selzo, and S. Qualters for all their help with the Regatta systems. One of us, CPS would like to thank Bruce Hurley for encouraging and making this type of studies possible.

## References

- [ 1] S. Andersson, R. Bell, J. Hague, H. Holthoff, P. Mayes, J. Nakano, D. Shieh, and J. Tuccillo *RS/6000 Scientific and Technical Computing: POWER3 Introduction and Tuning Guide*, IBM Corporation, International Technical Support Organization, Austin TX, 1998; and references therein.
- [ 2] J. DeRoest *AIX Version 4 System and Administration Guide*, McGraw-Hill, New York, NY, 1997.
- [ 3] G. Mojtabaezamani *RS/6000 SP 375 MHz POWER3 SMP Thin and Wide Node Architecture*, IBM Corporation, Poughkeepsie, NY, 2000.
- [ 4] S. Behling, R. Bell, P. Farrell, A. Holthoff, F. O'Connell, and W. Weir *The POWER4 Processor Introduction and Tuning Guide*, IBM Corporation, International Technical Support Organization, Austin TX, 2001; and references therein.
- [ 5] *Gaussian News*, **4**, 1(1993), Gaussian, Inc., Pittsburgh, PA.
- [ 6] H. M. Matis, J. D. McCalpin, M-C, Chiang, F. P. O'Connell, P. Buckland *IBM pSeries 690 Configuring for Performance*, IBM Corporation, Austin, TX, 2001.
- [ 7] AE. Frisch and M. J. Frisch, *Gaussian98 User's Reference*, 2nd Edition, Gaussian Inc., Pittsburgh, PA; <http://www.gaussian.com>
- [ 8] M. J. Frisch, A. B. Nielsen, and AE. Frisch, *Gaussian98 Programmer's Reference*, Gaussian Inc., Pittsburgh, PA; <http://www.gaussian.com>
- [ 9] C. P. Sosa, J. Ochterski, J. Carpenter, and M. J. Frisch, *J. Comp. Chem.* **19**, 1053(1998).
- [10] P. M. Gill, M. Head-Gordon, and J. A. Pople, *J. Phys. Chem.*, **94**, 5564(1990).
- [11] W. J. Hehre, L. Radom, P. V. R. Schleyer, and J. A. Pople, *Ab Initio Molecular Orbital Theory*, John Wiley & Sons, New York, NY, 1985.
- [12] B. G. Johnson, P. M. W. Gill, and J. A. Pople, *J. Chem. Phys.*, **98**, 5612(1993).
- [13] C. P. Sosa, G. Scalmani, R. Gomperts, and M. J. Frisch, *Parallel Computing* **26**, 843(2000).
- [14] A. D. Becke, *J. Chem. Phys.*, **98**, 5648(1993).
- [15] J. B. Foresman, M. Head-Gordon, J. A. Pople, and M. J. Frisch, *J. Phys. Chem.* **96**, 135(1992).
- [16] To obtain all the input files used in this study send an e-mail message to: [cpsosa@msi.umn.edu](mailto:cpsosa@msi.umn.edu)
- [17] C. P. Sosa and M. J. Frisch, in preparation.
- [18] R. J. Harrison and R. Shepard, *Annu. Rev. Phys. Chem.* **45**, 623(1994).
- [19] J. B. Foresman and AE. Frisch, *Exploring Chemistry with Electronic Structure Methods*, 2nd Edition, Gaussian Inc., Pittsburgh, PA, 1993.
- [20] C. P. Sosa and J. McCalpin, in preparation.

## Special Notices

This document was produced in the United States. IBM may not offer the products, programs, services or features discussed herein in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the products, programs, services, and features available in your area. Any reference to an IBM product, program, service or feature is not intended to state or imply that only IBM's product, program, service or feature may be used. Any functionally equivalent product, program, service or feature that does not infringe on any of IBM's intellectual property rights may be used instead of the IBM product, program, service or feature. IBM makes no representation or warranty regarding third-party products or services.

Information in this document concerning non-IBM products was obtained from the suppliers of these products, published announcement material or other publicly available sources. Sources for non-IBM list prices and performance numbers are taken from publicly available information including D.H. Brown, vendor announcements,

vendor WWW Home Pages, SPEC Home Page, GPC (Graphics Processing Council) Home Page and TPC (Transaction Processing Performance Council) Home Page. IBM has not tested these products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquires, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of a specific Statement of General Direction.

The information contained in this document has not been submitted to any formal IBM test and is distributed "AS IS". While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. The use of this information or the implementation of any techniques described herein is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. Customers attempting to adapt these techniques to their own environments do so at their own risk.

IBM is not responsible for printing errors in this publication that result in pricing or information inaccuracies.

The information contained in this document represents the current views of IBM on the issues discussed as of the date of publication. IBM cannot guarantee the accuracy of any information presented after the date of publication.

All prices shown are IBM's suggested list prices; dealer prices may vary.

IBM products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Information provided in this document and information contained on IBM's past and present Year 2000 Internet Web site pages regarding products and services offered by IBM and its subsidiaries are "Year 2000 Readiness Disclosures" under the Year 2000 Information and Readiness Disclosure Act of 1998, a U.S. statute enacted on October 19, 1998. IBM's Year 2000 Internet Web site pages have been and will continue to be our primary mechanism for communicating year 2000 information. Please see the "legal" icon on IBM's Year 2000 Web site (<http://www.ibm.com/year2000>) for further information regarding this statute and its applicability to IBM.

Any performance data contained in this document was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

## **Trademarks**

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM, e(logo), AIX, iSeries, pSeries

IBM Trademarks information can be found at: <http://www.ibm.com/legal/copytrade.shtml>.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SPEC, SPECjbb, SPECint, SPECfp, SPECweb, and SPECsfs are trademarks of the Standard Performance Evaluation Corporation and information can be found at: <http://www.spec.org>.

TPC, TPC-R, TPC-H, TPC-C, and TPC-W are trademarks of the Transaction Processing Performance Council. Information can be found at: <http://www.tpc.org>.

Other trademarks are the property of their respective owners.

## Notes on Benchmarks and Values

The benchmarks and values shown here were derived using particular, well configured, development-level computer systems. Unless otherwise indicated for a system, the values were derived using 32-bit applications and external cache, if external cache is supported on the system. All benchmark values are provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Actual system performance may vary and is dependent upon many factors including system hardware configuration and software design and configuration. Buyers should consult other sources of information to evaluate the performance of systems they are considering buying and should consider conducting application oriented testing. For additional information about the benchmarks, values and systems tested, contact your local IBM office or IBM authorized reseller or access the following on the Web:

TPC	<a href="http://www.tpc.org">http://www.tpc.org</a>
GPC	<a href="http://www.spec.org/gpc">http://www.spec.org/gpc</a>
SPEC	<a href="http://www.spec.org">http://www.spec.org</a>
Pro/E	<a href="http://www.proe.com">http://www.proe.com</a>
Linpack	<a href="http://www.netlib.no/netlib/benchmark/performance.ps">http://www.netlib.no/netlib/benchmark/performance.ps</a>
Notesbench Mail	<a href="http://www.notesbench.org">http://www.notesbench.org</a>
VolanoMark	<a href="http://www.volano.com">http://www.volano.com</a>
Fluent	<a href="http://www.fluent.com">http://www.fluent.com</a>
Gaussian	<a href="http://www.gaussian.com">http://www.gaussian.com</a>

Unless otherwise indicated for a system, the performance benchmarks were conducted using AIX V4.2.1 or 4.3, IBM C Set++ for AIX/6000 V4.1.0.1, and AIX XL FORTRAN V5.1.0.0 with optimization where the compilers were used in the benchmark tests. The preprocessors used in the benchmark tests include KAP 3.2 for FORTRAN and KAP/C 1.4.2 from Kuck & Associates and VAST-2 v4.01X8 from Pacific-Sierra Research. The preprocessors were purchased separately from these vendors.

The following SPEC and Linpack benchmarks reflect the performance of the microprocessor, memory architecture, and compiler of the tested system:

- SPECint95 - SPEC component-level benchmark that measures integer performance. Result is the geometric mean of eight tests that comprise the CINT95 benchmark suite. All of these are written in the C language. SPECint\_base95 is the result of the same tests as CINT95 with a maximum of four compiler flags that must be used in all eight tests.
- SPECint\_rate95 - Geometric average of the eight SPEC rates from the SPEC integer tests (CINT95). SPECint\_base\_rate95 is the result of the same tests as CINT95 with a maximum of four compiler flags that must be used in all eight tests.
- SPECfp95 - SPEC component-level benchmark that measures floating-point performance. Result is the geometric mean of ten tests, all written in FORTRAN, that are included in the CFP95 benchmark suite. SPECfp\_base95 is the result of the same tests as CFP95 with a maximum of four compiler flags that must be used in all ten tests.
- SPECfp\_rate95 - Geometric average of the ten SPEC rates from SPEC floating-point tests (CFP95). SPECfp\_base\_rate95 is the result of the same tests as CFP95 with a maximum of four compiler flags that must be used in all ten tests.
- SPECint2000 - New SPEC component-level benchmark that measures integer performance. Result is the geometric mean of twelve tests that comprise the CINT2000 benchmark suite. All of these are written in C language except for one which is in C++. SPECint\_base2000 is the result of the same tests as CINT2000 with a maximum of four compiler options that must be used in all twelve tests.
- SPECint\_rate2000 - Geometric average of the twelve SPEC rates from the SPEC integer tests (CINT2000). SPECint\_base\_rate2000 is the result of the same tests as CINT2000 with a maximum of four compiler options that must be used in all twelve tests.
- SPECfp2000 - New SPEC component-level benchmark that measures floating-point performance. Result is the geometric mean of fourteen tests, all written in FORTRAN and C languages, that are included in the CFP2000 benchmark suite. SPECfp\_base2000 is the result of the same tests as CFP2000 with a maximum of four compiler options that must be used in all fourteen tests.

- SPECfp\_rate2000 - Geometric average of the fourteen SPEC rates from SPEC floating-point tests (CFP2000). SPEC\_base\_rate2000 is the result of the same tests as CFP2000 with a maximum of four compiler options that must be used in all fourteen tests.
- SPECweb96 - Maximum number of Hypertext Transfer Protocol (HTTP) operations per second achieved on the SPECweb96 benchmark without significant degradation of response time. The Web server software is ZEUS v.1.1 from Zeus Technology Ltd.
- SPECweb99 - Number of conforming, simultaneous connections the Web server can support using a predefined workload. The SPECweb99 test harness emulates clients sending the HTTP requests in the workload over slow Internet connections to the Web server. The Web server software is Zeus from Zeus Technology Ltd.
- LINPACK DP (Double Precision) - n=100 is the array size. The results are measured in megaflops (MFLOPS).
- LINPACK SP (Single Precision) - n=100 is the array size. The results are measured in MFLOPS.
- LINPACK TPP (Toward Peak Performance) - n=1,000 is the array size. The results are measured in MFLOPS.
- LINPACK HPC (Highly Parallel Computing) - solve largest system of linear equations possible. The results are measured in GFLOPS.

VolanoMark is a 100% Pure Java™ server benchmark characterized by long-lasting network connections and high thread counts. In this context, long-lasting means the connections last several minutes or longer, rather than just a few seconds. The VolanoMark benchmark creates client connections in groups of 20 and measures how long it takes for the clients to take turns broadcasting their messages to the group. At the end of the test, it reports a score as the average number of messages transferred by the server per second.

VolanoMark 2.1.2 local performance test measures throughput in messages per second. The final score is the average of the best two out of three results.

The following SPEC benchmark reflects the performance of the microprocessor, memory subsystem, disk subsystem, network subsystem:

- SPECsfs97\_R1 - the SPECsfs97\_R1 (or SPEC SFS 3.0) benchmark consists of two separate workloads, one for NFS V2 and one for NFS V3, which report two distinct metrics, SPECsfs97\_R1.v2 and SPECsfs97\_R1.v3, respectively. The metrics consist of a throughput component and an overall response time measure. The throughput (measured in operations per second) is the primary component used when comparing SFS performance between systems. The overall response time (average response time per operation) is a measure of how quickly the server responds to NFS operation requests over the range of tested throughput loads.

The following Transaction Processing Performance Council (TPC) benchmarks reflect the performance of the microprocessor, memory subsystem, disk subsystem, and some portions of the network:

- tpmC - TPC Benchmark C throughput measured as the average number of transactions processed per minute during a valid TPC-C configuration run of at least twenty minutes.
- \$/tpmC - TPC Benchmark C price/performance ratio reflects the estimated five year total cost of ownership for system hardware, software, and maintenance and is determined by dividing such estimated total cost by the tpmC for the system.
- QppH is the power metric of TPC-H and is based on a geometric mean of the 17 TPC-H queries, the insert test, and the delete test. It measures the ability of the system to give a single user the best possible response time by harnessing all available resources. QppH is scaled based on database size from 30GB to 1TB.
- QthH is the throughput metric of TPC-H and is a classical throughput measurement characterizing the ability of the system to support a multiuser workload in a balanced way. A number of query users is chosen, each of which must execute the full set of 17 queries in a different order. In the background, there is an update stream running a series of insert/delete operations. QthH is scaled based on the database size from 30GB to 1TB.
- \$/QphH is the price/performance metric for the TPC-H benchmark where QphD is the geometric mean of QppH and QthH. The price is the five-year cost of ownership for the tested configuration and includes maintenance and software support.

The following graphics benchmarks reflect the performance of the microprocessor, memory subsystem, and graphics adapter:

- SPECxpc results - Xmark93 is the weighted geometric mean of 447 tests executed in the x11perf suite and is an indicator of 2D graphics performance in an X environment. Larger values indicate better performance.
- SPECplb results (graPHIGS) - PLBwire93 and PLBsurf93 are geometric means of literal and optimized Picture Level Benchmark (PLB) tests for 3D wireframe and 3D surface tests, respectively. The benchmark and tests were developed by the Graphics Performance Characterization (GPC) Committee. The results shown used the graPHIGS API. Larger values indicate better performance.

- SPECopc results - CDRS-03, CDRS-04, DX-03, DX-04, DX-05, DRV-04, DRV-05, DRV-06, Light-01, Light-02, Light-02, AWadvs-01, AWadvs-02, Awadvs-03, and ProCDRS-02 are weighted geometric means of individual viewset metrics. The viewsets were developed by ISVs (independent software vendors) with the assistance of OPC (OpenGL Performance Characterization) member companies. Larger values indicate better performance.

The following graphics benchmarks reflect the performance of the microprocessor, memory subsystem, graphics adapter, and disk subsystem:

Bench95 and Bench97 Pro/E results - Bench95 and Bench97 Pro/E benchmarks have been developed by Texas Instruments to measure UNIX and Windows NT<sup>®</sup> workstations in a comparable real-world environment. Results shown are in minutes. Lower numbers indicate better performance.

The NotesBench Mail workload simulates users reading and sending mail. A simulated user will execute a prescribed set of functions 4 times per hour and will generate mail traffic about every 90 minutes. Performance metrics are:

- NotesMark - transactions/minute (TPM).
- NotesBench users - number of client (user) sessions being simulated by the NotesBench workload.
- $\$/\text{NotesMark}$  - ratio of total system cost divided by the NotesMark (TPM) achieved on the Mail workload.
- $\$/\text{User}$  - ratio of total system cost divided by the number of client sessions successfully simulated for the Mail NotesBench workload measured.

Total system cost is the price of the server under test to the customer, including hardware, operating system, and Domino Server licenses.