Scaling - Up or Out

Al Talkington (Alt1@us.ibm.com)
Kaivalya Dixit (dixit@us.ibm.com)

(512) 838-2530
(512) 838-2061

February 2, 2002

©2002 International Business Machines Corporation, all rights reserved

Abstract

Insatiable demands for performance and the availability of inexpensive microprocessors have led to the development of multiprocessor-based systems. Passionate debates amongst architects, designers, and vendors continue to rage regarding cost effective multiprocessor design alternatives. Hypes and claims of performance and scalability are confusing and often misleading. Increasing performance and reliability by replication of subsystems and/or full systems are old concepts. Even the definition of what constitutes a "large" number of processors has changed over time, and will continue to evolve. This paper addresses performance scalability issues on Uniform Memory Access (UMA) Symmetric Multiprocessor (SMP) systems up to 128 processors.

Executive Summary

Scalability is significantly influenced by the hardware configuration, software configuration, and workload. There are a very large number of workloads and their characteristics are continually changing as new technologies and applications are developed. Scalable systems (Scale Up or Scale Out) should scale on more than one application or benchmark. Increasing performance by adding more processors has come to be commonly referred to as "scaling up". Increasing performance by adding additional complete systems is referred to as "scaling out".

At the hardware level, SMP systems replicate processors and caches , share global memory and IO, and also share the connections between these devices. Our observations will show that efficiency, scalability, reliability, and cost effectiveness of Scale Up systems degrade beyond 32 modern high performance microprocessors. The primary reasons for this degradation are bottlenecks caused by the ever widening gap between processor and memory speeds and contention of shared resources (e.g., bandwidth, memory, operating system). Clusters, or Scale Out systems, replicate complete systems interconnected via a variety of interconnect mechanisms. In Scale Out systems, the performance and scalability are limited by both the speed and efficiency of the intra-node communication and workload management. For the fast changing computing dynamics and long term investment in large enterprise systems, it is imperative to configure a judicious combination of efficient Scale Up and easy to manage Scale Out systems to solve both current and future computing needs of customers.

Introduction

The evolution of processor technologies and architectures have validated Moore's law of doubling the CPU clock rates every12 to18 months for the last 20 years. The speed of DRAMs are following a very different pace. Memory technology development has focused on increasing the density and reducing the cost of memory. Since 1997 CPU clock rates have jumped nearly an order of magnitude (from 300 MHz to 2000 MHz). During this same period memory chip speeds have only managed to double (from 100 MHz to 266 MHz (DDR)). Interconnect (CPU $\leftrightarrow$ CPU and CPU $\leftrightarrow$ Memory) speeds and interconnect bandwidth have also followed a much slower growth rate than the processor clock speed. As a result, the clock rate gap between the CPU and the main memory is increasing at a rapid rate. Computer architects, memory system designers, and computer system designers have been exploiting a myriad of techniques and exotic mechanisms (i.e. super-scaling, pipe-lining, adding multiple levels of cache, additional load/store units, speculative scheduling, hardware multithreading (HMT), separate and wide memory buses, double DRAM, and exotic interconnect technologies) to keep the CPU performing useful work during long waits (many CPU cycles) for data from memory.

At the same time customers have been given the perception that a high clock-rate means proportionately higher performance. So the customer expects a 1000 MHz system to deliver 2X the performance of a 500 MHz system. Unfortunately, this is simply not the case in most real world environments. In fact, processor clock rate improvements may even reduce a specific application's performance, depending on the demands placed on the other system resources.

To meet the performance demands of modern enterprise systems, vendors are offering a variety of multiprocessor based systems. Entry, midrange, and some high-end systems are frequently offered as shared memory based systems, also known as Scale Up systems. To attain higher performance and ease the burden on programming these systems are typically designed as Symmetric Multiprocessors (SMPs), such that all CPUs uniformly access memory and other system resources (I/O, disk, etc) through a bus, crossbar switch, or backplane. This means that only the CPU and caches are replicated. Other resources including the interconnect mechanism, main memory, and operating system, are shared amongst the processors. On the workloads that fit in caches and require little or no access to memory or operating system resources the user may attain almost linearly scaled performance increases as processors are added. Real applications are rarely that simple. By definition sharing means contention. And contention means sub-linear scalability and decrease in performance.

An alternate architecture that can add performance is "Share Nothing Systems", also known as Clusters or Scale Out systems. The Clusters are made up of interconnected nodes, which can be either uniprocessor or SMPs. For some workloads, a cluster may be a cost effective approach to scale on a benchmark or an application. Given a workload that can be partitioned into multiple nodes, the trick to optimizing performance in this environment is in the workload balancing and system management. Balancing and managing clusters was very much an art rather than an exact science but significant progress has been made in useable tools. With clusters you may or may not attain higher performance on a given application but you may attain better reliability with appropriate high availability software. Replication of the hardware and software eliminates many common points of failure.

There is a spectrum of architectures that span between the classic SMP and Clustered systems. Broadly these include NUMA systems, Massively Parallel Processor systems (MPP), tightly coupled cluster systems, and highly available clustered systems. Further there are many variations around these basic approaches. All of these exploit features of the basic SMP/Clustered designs, while attempting to mitigate the limitations, to address specific needs in final solutions. Discussions in this paper are limited to UMA Shared Memory Processors systems.

Hardware is not the only component to be considered in scaling. While to most buyers performance and scalability are mainly associated with clock rates, number of CPUs , and the number of disks, scaling is also largely defined by other system components and software performance. Obviously once an architecture is implemented in hardware, further scaling must be done by the software. This scaling is addressed by programming models, application tuning, operating system tuning, compiler libraries, database software tuning, and other techniques. While we focus on hardware scaling in this paper, it is impossible to completely eliminate the effects of the software.

Scalability

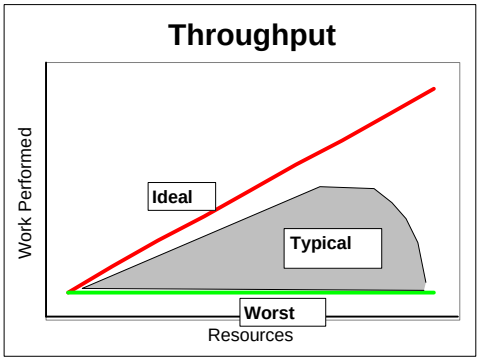
The simplest definition of "perfect scalability" is: No performance limitation due to hardware, software, or the size of computing problem. Unfortunately this is similar to flying at the speed of light. No one can get there. Scalability is basically a metric that indicates the performance benefits of multiple processor based Scale Up systems and Scale Out systems. While it is defined in many ways, it is not a precise metric and is often confused with speed, throughput, and system configuration. In general a good

scalable system should scale the performance of your application in a *predictable* manner when you increase or decrease the configuration.

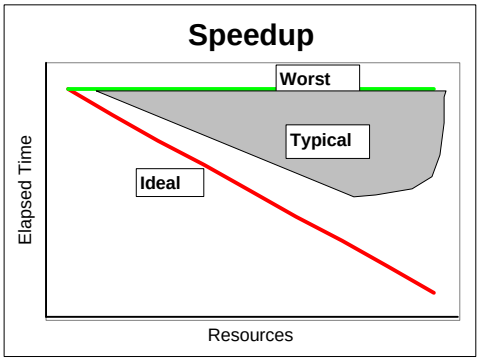
There are numerous components involved in scalability but, broadly, processor architectures exploit instruction level parallelism while minimizing latency effects. Multiprocessors architectures exploit application and system software level parallelism while minimizing contention between resources. The combination is intended to improve both speedup and throughput.

Scale Up and Scale Out systems provide scalability in different domains (workloads). Additionally, there are two distinct types of scalability issues : throughput and response time (how much work) and speedup (how much faster).

Throughput and response time scalability is defined as the proportionate increase in throughput of work with a constant response time (e.g., number of transactions) by adding processors, disks and other devices that permit increasing the work that can be completed (e.g., TPC -C, TPC-H, SPECjbb) within a specified time. This measure is mostly used for characterization of commercial and integer workloads. Scalability implies that the resultant improvement should be proportional to the increase in system resources. Ideally each added processor enables an equal amount of additional work to be completed in the same time. In the worst case, additional processors enable no additional work, or may even reduce the amount of work that can be completed. In reality, a system will usually fall into the "typical" area between these curves.



Speed up scalability is defined as completing the same amount of work in less time by adding additional processors, disks, and other devices for a given problem size (e.g., Fluent, SPEComp). Again, scalability implies that the resultant improvement should be proportional to the increase in system resources. In an ideally scalable solution, the time required to complete a task decreases linearly as processors are added. Worst case scalability, of course, means that added resources do nothing to improve the elapsed time. In reality, a system will usually fall into the "typical" area between these curves. In fact in some cases the addition of resources may increase the elapsed time due to the lack of adequate parallelism in the workload.



All vendors claim that their system is scalable and cite a benchmark result or an application result that shows almost an ideal (linear) scalability behavior. Many customers mistakenly equate SMP scalability with execution speed. In fact some popular benchmarks (i.e. Dhrystone and SPECint_rate) frequently do demonstrate almost linear scalability because the entire benchmark will fit in the cache of most modern systems. Unfortunately, the real world rarely fits in the cache. The footprints (working set size) of most applications, particularly commercial applications, are very large and expected increase over time. When the footprint of the program is larger than the cache, processors go to main memory to fetch data. Unless

the processors can find some other useful work to do while the data is retrieved, the system simply idles and both performance and scalability degrade. So plotting performance with number of processors against run-time and claiming either super-linear or linear scalability is very misleading.

Many benchmark configurations are tuned (software and hardware) for a maximum performance results. Such configurations do not generally reflect a typical customer configuration. A super linear or linear speed up indicates that either the benchmark or the application did not stress shared resources on that configuration.

As discussed earlier, a system is comprised of all its hardware and software components. Scalability depends on the efficiency of shared components under stress. Further, a solution tuned for large systems runs suboptimally on smaller systems and vice versa. Even small inefficiencies per processor can quickly degrade scalability as the number of processors increase or decrease. The efficiency problem is exacerbated by the increasing disparity between clock speeds and other shared components.

Software also plays a major role in system scalability. Software scalability (operating system, application and database) can be observed by running different versions of software on a constant hardware configuration. In the following paragraphs we will see that improvements of 300 percent were observed on a specific workload just by changing combination of JavaVirtual Machine (JVM) and Just-in-time-compilers (JITs). Further scalability is affected by running 32 bit applications on a 64 bit architecture. Clearly, workload has a major influence on scalability.

Scalability is a very complex issue but two facts are clear. First, if a system shows linear scalability on a benchmark and/or application then it is only useful information if your livelihood depends on running that benchmark or tuning that application. Second, it is very difficult to find and isolate scalability problems on a large two or 3 tier enterprise systems. Running a diverse mix of orthogonal workloads (CPU, I/O, interactive, and batch) on different applications and problem sizes will give you a better understanding of scalability that no single scalable benchmark or scalable application can provide.

Observations

When we started to investigate this issue we found that a complete set of results which could include results for a wide range of CPU configurations was generally not available. Since the intent was to understand the practical efficiency of large SMP systems, our sample size and selection of both systems and workloads was limited by dearth of published results. Overall we found results that could be used on:

- **SPECjbb2000** is a CPU intensive benchmark that executes business logic and is not cluster scalable. The benchmark is written in Java so the data was available on many different platforms. Major performance and scalability influencers are Java Virtual Machine (JVM) implementation, Just-in-time-Compiler (JIT), thread management and Garbage Collection. This benchmark and problem size were designed for small to midrange SMPs and 32 bit JVMs but is now used with both 32 bit and 64 bit JVM implementation. This benchmark was selected to show both JVM scalability characteristic (purely software) and also SMP scalability.

- **SPECfp_rate** is a CPU intensive benchmark that executes N (N = number of CPUs) copies of 14 floating-point (engineering and scientific) workloads. The size of data on some of the workloads cause data cache misses thereby creating some bus contention access to main memory. Some of the details of its characteristics can be obtained from reading John McCalpin's presentation on Benchmark vs. Applications.

- **SPEComp** is also a CPU intensive benchmark suite (eleven workloads) that uses Open MP standards. It measures compute intensive parallel workloads. This suite is used to demonstrate speed up scalability of SMPs and is cluster scalable. The most significant influence on scalability on this benchmark are the parallelizing compiler and the workload.

- **Fluent** is a compute intensive benchmark owned by Fluent Inc that models flow dynamics on various size problems. We used one of the medium class benchmarks in this suite - FL5M1 - primarily because of the larger sample of results available.

- **TPC-C** is a transaction processing workload that represents large transaction applications in a commercial environment.

Observation Approach

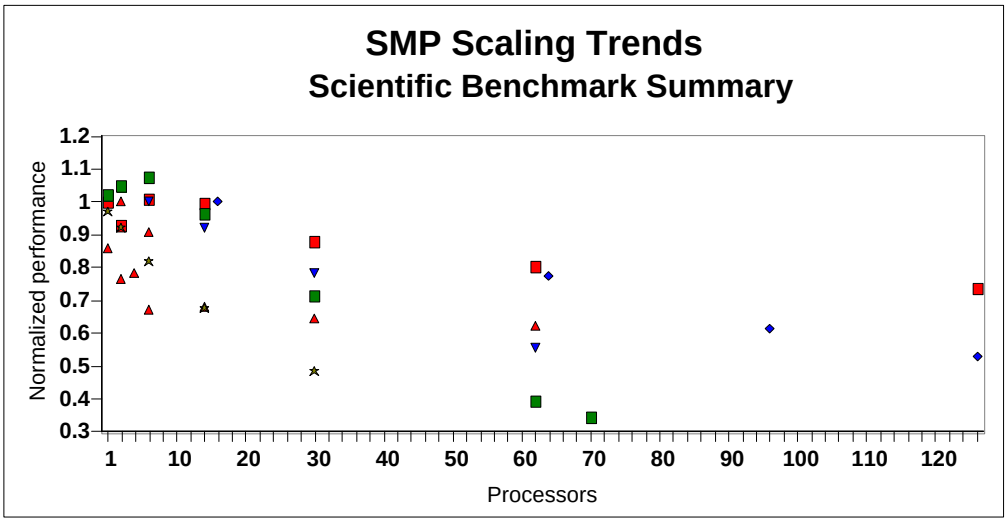
In order to evaluate the scaling of SMP architectures it was necessary to factor out the effects of absolute performance and clock rate variations within a specific system/CPU architecture. By doing this we could analyze and compare just the scalability. To achieve this we normalized the results from each benchmark by grouping and plotting the results by individual CPU/System architecture. Frequency difference effects were minimized by dividing each result by the processor clock frequency. Each data point was then established by dividing the result/frequency by the number of processors and plotting this point against the number of processors. Finally the data points were normalized by referencing each architecture group to the data point containing the fewest number of processors reported on a given benchmark. Typically this data point represented the highest result. In some cases additional software tuning (see above) was done that allowed a few points on the final plot to exceed "1". This approach provides a reasonable approach to analyzing the data but also has limitations that must be understood.

- The approach assumes that a given System/CPU architecture will scale linearly with frequency. This is a "best case" assumption that, as indicated in the introduction of this paper, is optimistic at best.
- The approach "weights" systems which report results starting at a large number of processors (16 or 24 rather than 2 or 8 way) more scalable as the processors increase than might be the case. The result of this effect is that large SMP systems may appear to scale better than they actually do. Regardless of some claims, one size system does not fit all.

We intentionally did not identify the specific systems used in this analysis but included information from at least four different vendors. Further we were interested in finding trends in the collection of data points. As a result the following graphs intentionally do not identify specific vendors nor do they highlight specific benchmarks. The information was gathered from publicly available sources including SPEC (http://www.spec.org), TPC (http://www.tpc.org), and Fluent web pages (http://www.fluent.com).

Scientific analysis and commercial workloads are classically different in how they use system resources. Regardless the typical system sold today may be used in either environment. However, since users of systems are typically either interested in running scientific analysis or commercial workloads, we then broke the results into these two categories.

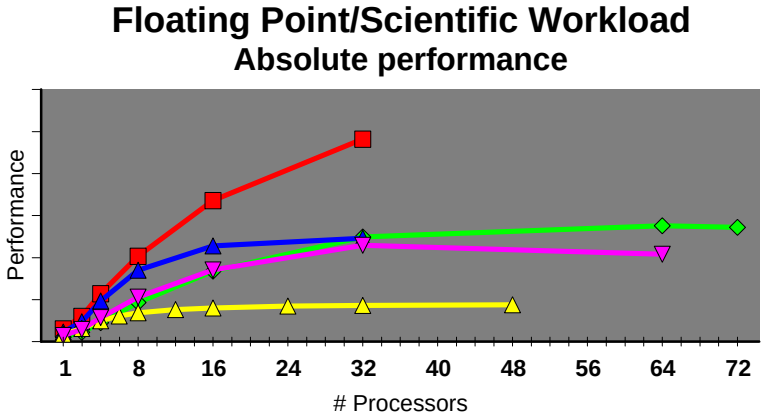
Observed scalability on Scientific workloads (SPECfp_rate, SPEComp, Fluent)



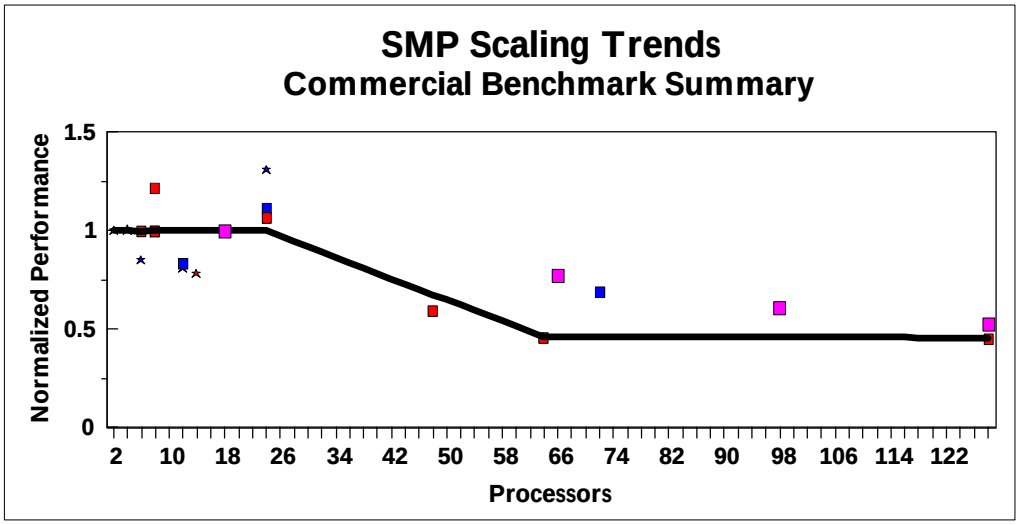
In this plot, several vendors' results were observed on 3 different benchmarks associated with scientific applications. Ideal scalability would result in all of the points along a line corresponding to "1". On this set of benchmarks the efficiency of each processor unsurprisingly tends to fall off as processors are added. However, as can be seen by the scatter of data points on the graph, the rate in which this performance drops off is highly dependent on the type of workload. Also of note are several points that exceed "1", implying "super linear behavior". These are either the result of the normalization approach or of configuration tuning (larger caches, software tuning, or compiler improvements) and do not seriously affect the overall conclusions.

To better observe the characteristics of the scientific workloads, more information can be gleaned by looking at a specific workload and the absolute performance differences. We have chosen a scientific benchmark that shows a distinct scaling limit. While each workload exhibits different characteristics, a scaling limit can usually be identified.

All of these graph points use the same benchmark and plot actual performance, as opposed to normalized performance. As can be seen, the performance tends to increase through a small number of processors and then drop off dramatically. In fact, except for the vendor in red, all systems demonstrated virtually no increase in performance as processors were added above 32 and some began to fall off after 16 processors were added. These limits may be due to limited processor capabilities, a lack of parallelism in the workload, saturation of the hardware interconnect paths, ineffective memory management, poor parallelizing compilers or a combination of all of these factors.



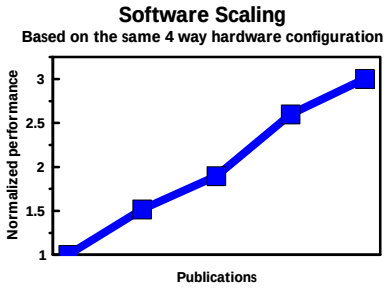
Observed scalability on Commercial workloads (OLTP, SPECjbb)



The trend in the commercial workloads is much more distinct. As processors are added, scalability seems to be maintained until somewhere between 24 and 32 processors. In that area, the scalability drops off dramatically such that processors added beyond 32 tend to only yield a 50% efficiency. As with the scientific benchmarks, the data points that exceed "1" and likely due to either the normalization approach or specific tuning by a vendor.

Observed Software Scalability

A major fallacy in most scalability discussions is the absence of the effects of software. In reality, once a hardware architecture is implemented, full scalability is purely a matter of tuning the software components. If any of the software components used by either the benchmark or an application fails to scale, an overall scalability measure can be very misleading. To illustrate this, we include the following graph.



Over a 1 year period, the hardware used to report on this particular benchmark, SPECjbb2000, remained constant and only the JVM and JIT changed. The performance improved 3 fold. Earlier we were discussing throughput improvement with the addition of processors and disks. This improvement in performance was purely due to tuning of software. This indicates that the earliest versions of the JVM and JIT were not very scalable. So, during the time frame of the first set of results, a system analyst who was trying to size his application server needs might have ordered 3X number of processors needed just a year later. This is a valuable lesson in the confusing issue of scalability. Analysts are always tuning system software (e.g., kernels, operating systems, firmware, compilers, database, drivers), and application designers are also tuning applications. For better or worse given a constant hardware configuration, software actually shapes the scalability.

Workload also provides scaling challenges and opportunities. For example, the SPEC SFS 3.0 benchmark measures both throughput and response time for 4 very different workloads. NFS Version 2

(TCP and UDP) and NFS Version 3 (TCP and UDP). The performance differences are significant for the same hardware.

Scalability and Future Workloads

Large SMPs and Clusters are used for both scientific and commercial applications. Many enterprises will be consolidating servers and applications on both types of these systems with both static and dynamic partitioning schemes and heterogeneous workloads managed by intelligent workload managers. All of these schemes create an overhead that could seriously affect the scalability of less optimal systems. The scalability and performance will have to be redefined for emerging deep computing, grid computing, and eLiza™ environments. Scalability metrics for multi-tier systems that can withstand sustained overloads without crashing and maintain acceptable performance will emerge over time. These are the interesting challenges of the future.

Conclusion

The needs of the computer industry are both dynamic and diverse. While ideally a customer would like a system with an infinitely fast processor and system components, this simply does not exist. To compensate, customers need both Scale Up and Scale Out solutions for their mission critical and business equivocal growth needs. IBM® has many years of experience in architecting and implementing a spectrum of eServer products that meet reliability, availability, absolute performance, and price/performance on both approaches. The analysis of many diverse sets of customer needs, from the mainframe to the PC, real-life workloads, and industry standard benchmarks has indicated that scalability efficiency of Scale Up approach is good for up to around 32 processors. The law of diminishing returns takes over after this due to memory, bandwidth, and other system bottlenecks. Our observations indicate that there is a natural efficiency limit for SMPs in the 24 to 32 processor range. While, on some heavily tuned scalable benchmark, 100+ processor SMP may show good results, this does not appear to be the general case. In fact, some of the "fat" SMPs run as partitioned clusters just to minimize bottleneck problems. While running in partitions can bring into play some of the cluster advantages, it cannot address the higher failure rates (single points of failure) and maintenance problems associated with a single system.

In general, the findings tend to support the concept that the most optimum approach is to develop the highest performance processors possible, enable them to be used in an SMP configuration (Scale Up) up to 32 processors, and then cluster (Scale Out) beyond this to meet more demanding requirements

References

[1] http://www.usenix.org/publications/library/proceedings/als2000/bryantscale.html

[2] Intel Consumer Desktop PC Microprocessor Timeline: http://www.intel.com/pressroom/archive/backgrnd/30thann_timeline.pdf

[3] Memory: Evolution or Revolution?; http://www.dewassoc.com/performance/memory/intro_2.htm

[4] Performance Workloads in a Hardware Multi-Threaded Environment by Bret Olszewski & Octavian F. Herescu; Presented at the Computer Architecture Evaluation using Commercial Workloads (CAECW-02) - March 2002.

[5] Computer Architecture A Quantative Approach: John L Hennessy & David A Patterson (page 734).

[6] SPEC SFS Performance on eServer pSeries™ Systems: Agustin Mena III; 2001 Performance Technical Journal

[7] http://www.usenix.org/publications/library/proceedings/als2000/bryantscale.html

[8] SPEC SFS Performance on eServer pSeries Systems: Agustin Mena III; 2001 Performance Technical Journal

[9] LPAR heterogeneous workloads on IBM eServer pSeries 690 system: Antonio Garcia, Et.Al.; 2001 Performance Technical Journal

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, eLiza, 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	http://www.tpc.org
GPC	http://www.spec.org/gpc
SPEC	http://www.spec.org
Pro/E	http://www.proe.com
Linpack	http://www.netlib.no/netlib/benchmark/performance.ps
Notesbench Mail	http://www.notesbench.org
VolanoMark	http://www.volano.com
Fluent	http://www.fluent.com

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 30 GB to 1 TB.

- 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 30 GB to 1 TB.

- $/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® 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.
- $/NotesMark - ratio of total system cost divided by the NotesMark (TPM) achieved on the Mail workload.
- $/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.