Paper 271-26

Findings and Sizing Considerations of an Integrated BI Software Solution in an AIX Environment and a Windows NT Environment

E. Hayes-Hall, IBM Corporation, Austin, TX

ABSTRACT

This paper presents the first in a number of discussions concerning the implementation and capacity management of Business Intelligence software solutions. This paper discusses capacity management implemented on two distinct architectures. The architectures are 64 bit AIX running on IBM's line of pSeries servers and of 32 bit Windows NT running on IBM's xSeries servers. The software environment consisted of the SAS 8.2 core application. The paper identifies some of the distinct properties associated with these very different architectures, the performance characteristics of these configurations and the relative advantages of the configurations for various business intelligence workloads.

INTRODUCTION

By their very nature, Business Intelligence workloads are varied, unpredictable and constantly changing. Business Intelligence workloads can vary from the generation and manipulation of a data cube through the unpredictability of an ad-hoc query, to the handling of statistical analysis and generation of results for visualization and reporting. Coupled to this variety of scenarios, there are considerable performance characteristics of different operating systems and architectures that affect the overall performance of a total business solution. Future architectural advances, particularly in the area of 64 bit chip sets, will also affect the performance and capabilities of these solutions.

A Business Intelligence solution typically includes a variety of software and hardware ranging from large server systems for back-end data warehousing, to intermediate server systems for ETL and analytical processing, to client based systems for visualization and reporting systems. Many commercial Business Intelligence solutions use a combination of UNIX based or NT based servers to fulfil all or some of these roles, but the question always remains, "which one will work for me?"

This paper discusses aspects of the performance of a Business Intelligence software solution in both the NT and AIX environments. The paper does not seek to directly compare the performance of one configuration against the other. Moreover, this paper looks at aspects of the performance of each configuration and discusses the relative merits of each architecture based on the workload characteristics. The paper discusses which architecture is the best candidate for a particular Business Intelligence workload.

WORKLOAD CHARACTERIZATION

Most Business Intelligence data access is concerned with answering a specific question or set of questions concerning the business process under consideration. This, in turn, means that the Business Intelligence workload is driven predominantly through the generation of complex and non-repetitive ad-hoc queries. Complex, large queries of this nature are very I/O intensive, and a major performance bottleneck of a Business Intelligence solution is the I/O subsystem. Additionally, such workloads typically require complex data cleansing and analysis before the results can be visualized in a meaningful way. Data cleansing is vital to the process as erroneous data will be passed directly back to the requestor and ultimately will affect the decision making process. Such data cleansing and analysis is very CPU and memory intensive. From a performance and tuning perspective, then, Business Intelligence workloads exercise the entire resources of a architected solution from the I/O subsystem, through the memory and caching architecture to the throughput and performance of the CPUs themselves.

The validity of the workload used in a test is also important. Capacity planning and capacity management theory suggests that workloads can often be modeled. However, within the Business Intelligence environment, query workload requirements are often poorly understood by designers and modeling techniques are not yet sufficiently sophisticated to model a Business Intelligence workload with a great degree of precision. Often, it is wiser to use example workloads from a commercial environment to exercise a Business Intelligence solution. Example workloads, however, have their own set of problems and complexities based on the relevance of the data.

Another aspect of workload characterization is the choice of metric used to determine capacity management. There are a number of standard benchmarks available and quotable that pertain to a Business Intelligence solution: OLAP Council benchmarks (APB-1), TPC-H, SPEC (integer, floating point). Each of these benchmarks represents a specific functional part of a total Business Intelligence solution at either a component or system level (Table 1). However, no single benchmark represents the entire solution. The OLAP APB-1 benchmark defines a metric called the Analytical Query Time (AQT) which is essentially a ratio of the total server processing time to the number of queries processed. This is the closest to a "real-world" measure of the performance of the Business Intelligence solution from an end-user's perspective.

Table 1

	pSeries 680	RS/6000 S80	RS/6000 H70	RS/6000 F50
ROLTP	716.6	452.7	57.1	32.8
tpmC	N/A	135815	17133	9853
SPECweb 96	N/A	40161	11774	6716

However, from a capacity management perspective, these metrics do not provide clear information concerning the performance of the individual resources and functional parts of the total Business Intelligence solution. Essentially, component-level and system level benchmarks offer excellent tools for comparing systems rather than acting as an accurate capacity planning tool for applications on a specific system [4]. Component level workload modeling and performance modeling is vital to understanding how the underlying resources (CPU, Memory, I/O, Network) perform and are enhanced. It is vital to efficient capacity management of Business Intelligence solutions involving database servers, application servers and front-end visualization services through web or client based reporting.

With this in mind, examination of aspects of the CPU, memory and I/O characteristics of an example workload on different architectures can shed significant light on the performance and tuning of enterprise servers to improve capacity management. Understanding of these resource usage characteristics allows designers to determine the optimal architectures, both in terms of size and type, for the specific functional parts of a solution.

ARCHITECTURAL DESIGN

This section gives a very brief introduction into memory architecture and the two architectural environments used for testing.

WINDOWS NT ENVIRONMENT

Windows NT memory is managed by the Virtual Memory Manager and the Cache Manager. However, I/O performance and memory management are linked by the interaction of File System Cache with these two resources.

The File System Cache is a dynamically allocated area of physical memory, where data is cached from I/O subsystem read and write operations. The File System Cache resides in upper memory or kernel space. The Cache Manager attempts to optimize I/O requests using techniques such as lazy writes, deferred commits, etc.

However, under very heavy I/O loads, the I/O can flood the File System Cache forcing physical I/O to the device. Additionally, on busy servers, when memory is at a premium, the Virtual Memory Manager will trim processes by paging out memory to make room for paging in new processes. However, cache thrashing occurs as the VMM has to essentially “rob Peter to pay Paul” and significant resources as expended to manage this situation.

Additionally, as the File System Cache is resources from kernel memory, on heavily memory loaded systems (in-core solvers, sorts, binary searches, etc.) the File System Cache can compete with the memory resources available to run the applications. In situations like this it is often advantageous to reduce the File System Cache size considerably.

WINDOWS NT CONFIGURATION

The baseline system used for the tests was a Netfinity 7000M10 with four 550MHz Pentium Xeon processors, 2MB L2 cache. The operating system was Windows NT 4.0, SP6a. The system contained 2G of 100 MHz SDRAM based memory. The system has 33 MHz PCI bus technology. The disk subsystem consisted of a ServeRaid 3H adapter connected to an EXP-15 external SCSI enclosure.

The EXP-15 was in a split bus configuration, with each of the two external RAID adapter connections attached to separate hot swap backplanes each containing five 9.1GB hard drives. The 9.1GB SCSI hard drive containing the operating system was attached to the system’s internal hot swap drive backplane which, in turn, was connected to one of the system’s integral Adaptec SCSI controllers.

The IBM ServeRaid 3H adapter, with its attached 9.1GB hard drives, was initially configured with two RAID 0 arrays. The first array consisted of two physical hard drives on SCSI channel 1 and three physical drives on SCSI channel 2. The second array consisted of three physical hard drives on SCSI channel 1 and two physical drives on SCSI channel 2. These arrays were presented to the operating system as two logical drives, X: and Y:, of approximately 45GB each. The stripe unit size for the adapter was set to 8, write cache was set to ‘write through’, and read ahead cache was set to Adaptive.

Striping, or RAID 0, minimizes the disk seek time which is a function of the rotational latency associated with the underlying member disks of a disk array. Striping allows the write function to divide the I/O into track-sized chunks, based on the stripe unit size, and distribute the I/O to individual member drives. [3]

This baseline configuration was altered in the following ways during testing. Configuration changes were not cumulative, after each series of tests, the test harness was returned to the baseline configuration before new changes were made.

Configuration A

The ServeRAID adapter read-ahead cache was disabled to see what effect this would have on the performance of the test workload.

Configuration B

The number of physical hard drives in each RAID 0 logical drive was reduced from five to two in order to see the limiting effects of the individual drive transfer rates. This reduces the I/O bandwidth to the striped device.

Configuration C

The location of the saswork directory was changed from a RAID 0 logical drive attached to the ServeRAID adapter to a single 9.1GB SCSI 2 drive attached to the Netfinity integrated SCSI adapter.

AIX ENVIRONMENT

In the same manner that I/O and memory are inter-related in Windows NT, the same is true of the fundamental relationship in AIX.

AIX memory is managed by the AIX Virtual Memory Manager who essentially makes real memory appear larger than the actual physical memory of the system. Virtual memory is made of real memory and disk space. Real memory is, in turn, made up of three segments (as shown in Figure 1)

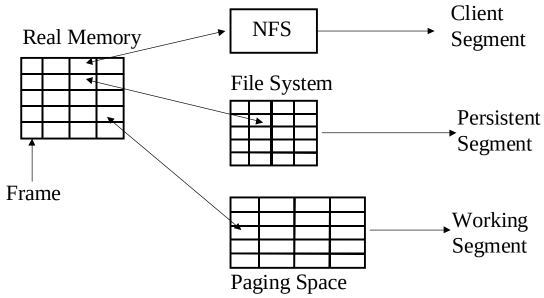


Figure 1

The persistent storage segment is where data files are mapped into persistent storage (on disk). On execution of a process only a few pages are resident in core memory with extra pages passed in to virtual memory on demand. AIX 4.3.2 onwards uses a late allocation algorithm which approves the page request when a page is touched. The VMM will search the data cache, instruction cache, TLB and Page Table Frames before going out to pages on disk in an effort to satisfy page requests as quickly as possible.

AIX CONFIGURATION

The AIX environment comprised the following four separate systems:

pSeries 680

A pSeries 680 system with twenty-four PowerPC RS64-III 600 MHz CPUs, 48 GB of system RAM, 16 MB of L2 cache, 53 GB of directly attached disk storage defined as two stripped logical volumes across seven 9.1 GB SCSI attached disks. The stripe size was set to 32 Kbytes. Two file systems were created – a data file system of 9 GB and a data repository of 44 GB. The system ran AIX 4.3.3

RS/6000 S80

RS/6000 S80 system with six PowerPC RS64-III 450 MHz CPUs, 32 GB of system RAM, 8 MB of L2 cache, 36 GB of SCSI attached disk storage. The system ran AIX 4.3.3

RS/6000 H70

RS/6000 H70 system with four PowerPC RS64-II 340 MHz CPUs, 3 GB of system RAM, 4 MB of L2 cache, 77 GB of SCSI attached disk storage. The system ran AIX 4.3.2

RS/6000 F50

RS/6000 F50 system with two PowerPC 604e 332 MHz CPUs, 1 GB of system RAM, 256 KB of L2 cache, 77 GB of SCSI attached disk storage. The system ran AIX 4.3.2

WORKLOAD TEST SUITE

The US Census Bureau makes available to the public its Public Use Microdata Sample as flat text files. The SAS CTC scripts utilize the I/O and analytical functions of SAS 8 core product and are single threaded applications.

The SAS core product has a number of tunable options available

1. MEMSIZE: The total amount of memory SAS may use for any single process.
2. SORTSIZE: Total amount of memory allocated for the SORT procedure (defaults to 16Mb)
3. BUFSIZE: Size of I/O buffer used in read/write operations from SAS to the operating system. Default of 0 causes SAS to use an internal algorithm to optimize I/O based on the host systems' characteristics.
4. BUFNO: Number of buffers of BUFSIZE to allocate for data moves. Defaults to 1.

Unless specified, the defaults were used for all these variables. An independent study [2] showed that maintaining the default values especially for BUFSIZE and BUFNO, thereby using the SAS optimization algorithm, provided the best resource utilization of the system.

The SAS CTC suite of test scripts manipulates this data in various ways.

Test1: Reads data from 5 raw Census Bureau files to create 2 SAS data sets and then validates the information in each SAS data set to make sure there is no incorrect information. This test is very I/O intensive. The test creates the HRECS data set which is ~1.1 gigabytes in size (5,527,406 records) and the PRECS data set which is ~2.8 gigabytes in size (12,501,046 records).

Test2: This test sorts the HRECS file - and then creates several indices. These two features are used often within SAS datawarehouse applications. This test is very CPU and memory intensive in nature. The HRECS data set used in this test program is ~1.1 gigabytes in size (5,527,406 records). During the SORT the algorithm maintains an growing sorted copy of the original dataset. MEMSIZE=608M and SORTSIZE=512M.

Test3: Summarize and collects frequency information on the two tables - HRECS and PRECS. These frequency tests are I/O and memory intensive. The test continues by doing some summarization of the files. The HRECS data set used in this test program is ~1.1 gigabytes in size (5,527,406 records) and it will create the PRECS data set which is ~2.8 gigabytes in size (12,501,406 records).

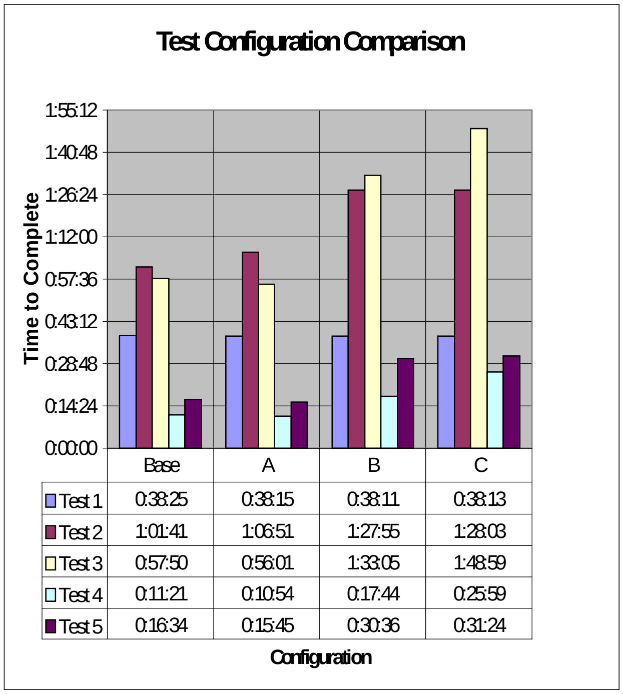
Test4: This test collects statistics from the HRECS file. This test is CPU intensive. The HRECS data set used in this test program is ~1.1 gigabytes in size (5,527,406 records) and it will create the PRECS data set which is ~2.8 gigabytes in size (12,501,406 records). There are two statistical passes at the data in addition to a sort.

Test5: Creates several MDDBs for use with various SAS interactive procedures. This test is memory intensive. The HRECS data set used in this test program is ~1.1 gigabytes in size (5,527,406 records) and it will create the PRECS data set which is ~2.8 gigabytes in size (12,501,406 records). MEMSIZE=1200M since the MDDB is created in memory.

RESULTS

WINDOWS NT ENVIRONMENT

Table 2 shows the elapse time taken for each workload test across the baseline configuration. The table also shows the associated elapse time for completion of the workload tests for each of the three altered configurations of the Netfinity 7000. Test 1 shows no significant change in elapse time despite changes to the I/O configuration. Statistically, the variation of Test 1 results across the configurations is less than 0.3%. All the remaining tests show significant changes in elapse time across the different configurations. Removing read-ahead cache (configuration A) increases the elapse time of the memory intensive sort (Test 2). Significant changes in elapse time occur with changes to the RAID 0 layout or the placement of the filesystems across all tests except for Test 1.



	Base	A	B	C
🟦 Test 1	0:38:25	0:38:15	0:38:11	0:38:13
🟥 Test 2	1:01:41	1:06:51	1:27:55	1:28:03
🟨 Test 3	0:57:50	0:56:01	1:33:05	1:48:59
🟦 Test 4	0:11:21	0:10:54	0:17:44	0:25:59
🟪 Test 5	0:16:34	0:15:45	0:30:36	0:31:24

Table 2

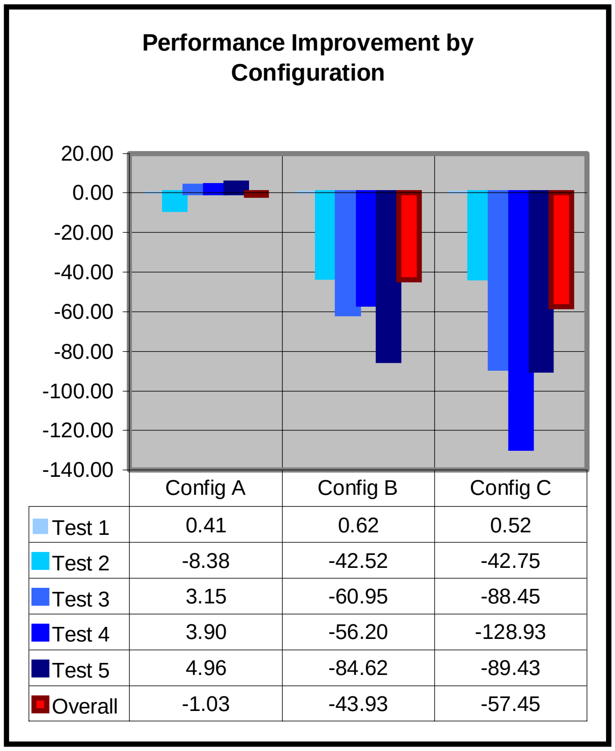
Table 3 shows the relative performance change (%) of each of the Windows NT configurations as compared to the baseline system configuration.

For configuration A – disabling read-ahead cache – showed slight performance improvements for all workload tests with the exception of Test 2 which involves the sorting and indexing of the workload data files. In this case the performance of the system was degraded by 8.4%. Tests 1 and 3 are heavily I/O intensive, in the case of Test 4, are CPU intensive. These showed a slight improvement in performance when read-ahead cache was disabled.

For configuration B – reducing the number of RAID 0 member drives from 5 to 2 per array – showed significant performance degradation across all tests except Test 1. Test 1 is an I/O intensive sequential read of two flat files. There was a slight but not significant improvement in performance of 0.62%. For the memory and complex I/O intensive workloads the performance degradation ranges from 42% (test 2 – sort in memory) to 85% (Test 5) for the memory intensive MDDB creations.

For configuration C – moving saswork directory to a non-striped single SCSI attached disk – showed the most dominant performance degradation across all tests except Test 1. There was a slight but not significant improvement in performance of 0.52% for the sequential read test (Test 1). The memory intensive sort test (Test 2) showed a similar performance degradation to that of reducing the number of RAID 0 member drives (46%) and the remaining tests showed very strong performance degradation.

Table 3



	Config A	Config B	Config C
Test 1	0.41	0.62	0.52
Test 2	-8.38	-42.52	-42.75
Test 3	3.15	-60.95	-88.45
Test 4	3.90	-56.20	-128.93
Test 5	4.96	-84.62	-89.43
Overall	-1.03	-43.93	-57.45

Table 4 shows the elapse time for Tests 1 through 4 on the four AIX based servers and the NT baseline server extracted from table 2.

Table 5 attempts to show how the different architectures compare for specific tests. The NT baseline configuration from table 2 is used to compare test result characteristics in table 4. Given that the results for test 1 across all different NT configurations do not show much variation, the result of test 1 for all results is used as a index. The data in table 5 for instance shows that test 2 executed on the NT configuration ran 1.6 times longer than test 1 on the NT configuration. For the pSeries 680 executing test 4 on this system ran in one-tenth of the time it took to execute test 1 on the pSeries 680. By using test 1 on each architecture as an milestone index for other tests on that same architecture it is possible to determine some of the difference in how the systems handle I/O and memory intensive workloads.

Table 4

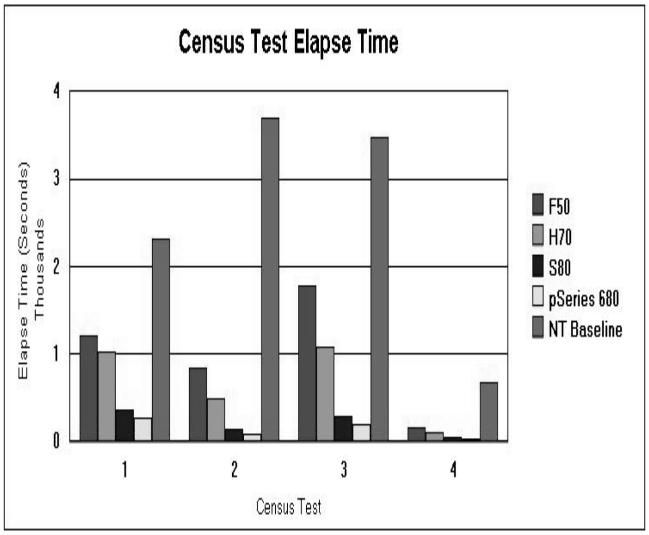
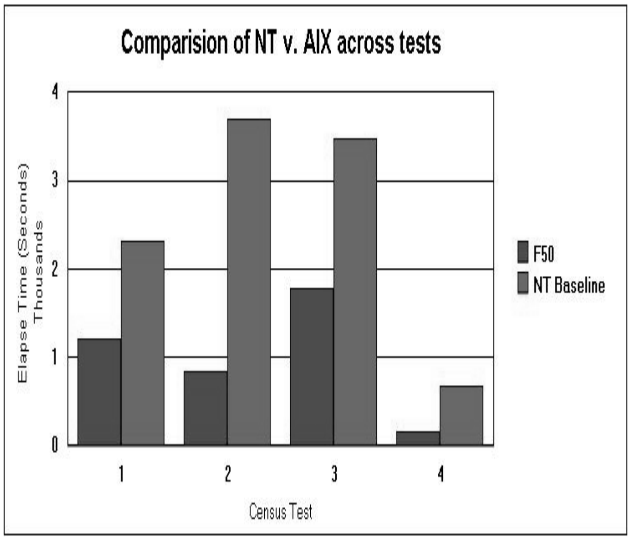


Table 5

Test	NT	F50	H70	S80	pSeries 680
1	1.0	1.0	1.0	1.0	1.0
2	1.6	0.7	0.5	0.4	0.3
3	1.5	1.5	1.1	0.8	0.7
4	0.3	0.1	0.1	0.1	0.1

These workload differences can also be viewed by comparing the relative differences of the NT baseline configuration against an AIX configuration. The F50 has been used in Table 6 as the time differential on the Y-axis is sufficiently small to make the scales readable.

Table 6



The overall differences can be summarized as:

1. The overall elapse time of each test running on AIX is significantly shorter than for the tests running on Windows NT. This is not surprising given the RAM associated with each of the AIX servers.
2. The memory intensive sort (Test 2) completes much faster on AIX relative to its base indices.
3. The I/O and memory intensive workload test (Test 3) which comprises the execution of four multiple field frequency procedures, four summations and a sort is benefited particularly by the newer AIX server hardware (S80 and pSeries 680). On both NT and the F50, this test took approximately 1.5 times the base index for the respective hardware. On the S80 and pSeries 680 this test took approximately 70-80% of the index baseline for the relevant system.
4. The CPU intensive workload test (Test 4) which comprises the execution of two univariate statistical functions with an associative sort again clearly benefits from the Power chipset. The workload tests are essentially single-threaded and a comparison of pure clock speed would suggest that the 550 MHz Xeon processors in the NT system should execute this test faster than the 332 MHz and 340 MHz Power processors in the RS6000 F50 and H70.

CONCLUSION

For single threaded applications the effects of memory and disk resource usage far outweigh the effects of processor speed.

Adaptive read-ahead cache, under the examined workloads, has negligible effect on the performance of raw data reads from disk. This suggests that the SAS I/O optimization algorithms are efficient for handling sequential read I/O.

The effects of filesystem placement and the configuration of the I/O subsystem have extremely significant effects on the performance of a system.

The effects of constrained available memory are considerable and have a cascading effect on disk I/O performance. Both the Windows NT and F50 systems have available RAM which is smaller than the data sets being processed. On the NT system, the File System Cache does not appear to be being utilized efficiently. The SORT of a 1.1 GB file with a SORTSIZE of 512 MB on NT appears to be causing performance degradation predominantly through lack of available memory and a corresponding increase in excessive paging. On the F50 system the available memory is also constrained causing excessive page-fault activity in persistent storage and corresponding I/O contention. The advances in the Power memory and I/O technology appears to be compensating somewhat for this extra activity. On the other AIX systems the data files can be resident in memory which reduces the paging activity to a minimum.

In AIX the persistent storage segment, by default, gives 20% (minimum) to 80% (maximum) of memory to the handling of buffers for I/O. Under heavy I/O loads, for instance with SAS applications, the persistent storage area will fill up due to excessive buffer usage. As the persistent storage segment fills so the Virtual Memory Manager can easily get into a situation where there are excessive page referencing and associative page faults.

Under loaded circumstances the Virtual Memory Manager will swap out a page to make room for the newly referenced page and thrashing in persistent storage takes place with multiple page-faults associated with a single I/O operation. This leads to significant performance degradation. Within the Scientific and Technical server segment the limits have been successfully reduced to 5%(minimum) to 10%(maximum). This has the effect of providing more memory available for processes to execute with higher throughput even though I/O activity may increase.

This same principle holds for the possibility of looking to alter the Windows NT LargeSystemCache value to maximize throughput for network applications. Setting this value to favor network workloads has the effect of turning off the Windows NT file system cache. This may be appropriate under certain memory intensive workloads on heavily loaded NT servers.

There are a number of other recommendations for maximizing I/O performance. These include:

1. Maintain the paging space (persistent storage or pagefile) on separate devices from user filesystems.
2. Spread the filesystems around the I/O subsystem. If disk space is at a premium then mix lightly used filesystems with heavily accessed filesystems to reduce I/O contention.
3. Place heavily accessed I/O filesystems across wider stripe volumes (disk members) and controllers as possible to maintain maximum bandwidth.
4. Keep paging to a minimum. In heavily loaded, large memory workload situations possibly reduce the buffer cache to facilitate keeping processes in memory for longer which should improve throughput. If this still does not alleviate the problem then the alternative in most cases will be obtaining more memory or offloading work onto another system.

The 32 bit Windows NT system with its constraint on maximum available memory means that multi-user environments with processes executing in-core algorithms are going to be at a disadvantage. Clearly it is unfair to compare an NT workstation with 4 CPUs and 2 GB of RAM with a pSeries 680 server with 24 CPUs and 48 GB of RAM. However, Windows NT does have some issues concerning the maximum memory available and the ability to accurately tune the memory sub-system. This plus Windows NT scaling issues preclude the Windows NT server from enterprise style Business Intelligence scenarios.

In situations where there are either multiple tiers of a BI solution on a single server or there are a large number of concurrent users executing memory intensive workloads then the NT server is not going to have the memory, I/O or processor scaling to be able to perform effectively. Where the NT server becomes a viable solution is when cost becomes an issue. The NT server solution would be a viable option in small to medium business situations where there are relatively small numbers of users making relatively small demands on the Business Intelligence applications. For larger enterprise situations then the larger AIX based servers provide the flexibility, scalability and capacity to execute large memory, I/O intensive workloads across larger numbers of users.

The question still remains though; How does one size a Business Intelligence solution? The answer to this is difficult. There is no single answer or model that can accurately determine the size of a system to run a particular solution. Sizing comprises techniques and tools from capacity planning, capacity management and performance and tuning of application and system resources.

In essence there are three basic steps in the sizing methodology:

1. For a given solution, size each of the individual processes/applications in terms of CPU, memory, network and I/O resource usage. This involves a detailed workload analysis and performance and tuning exercise of either an existing or a planned system. Thought has to be given to such

issues as multi-function user of the system, user access and resource usage policy.

2. Sum all the resources determined in step 1 and then add the resources (CPU, memory, network, I/O) for the operating system. Then adjust for head-room. Given that throughput and response times will degrade considerably as total system utilization approaches 100%, it is suggested that the final sizing should represent 70% of the capable processor capacity and 40-50% of the disk I/O capable capacity. This will also allow for growth of the workload over time.
3. The final step is to define resource usage policies and them enforce them as the system is being used. In AIX 4.3.3 then CPU and memory resource can be managed through the Workload Manager. With the release of AIX 5L this capability is being enhanced to include disk I/O policy management. Within Windows NT there are a number of tools available to allow appropriate monitoring however, the tuning of the system is somewhat more rudimentary and inflexible.

There is still much work to be completed on understanding and predicting accurately the sizing of a given Business Intelligence solution. Future work that is currently under consideration involves;

1. Experimenting more fully with the VMM on AIX under heavy paging conditions.
2. More thorough disk I/O performance and tuning especially looking at how cache sizes in various parts of the subsystem (buffer cache, logical volume, disk controller) affect the overall performance of an I/O
3. The effects of different filesystem technologies on Business Intelligence applications
4. Mid-tier Business Intelligence sizing methodologies
5. The performance characteristics of clustered Business Intelligence solutions.

It is envisioned that as these pieces of technical information become available they will be made public, and be useful, to the Business Intelligence community.

REFERENCES

[1] Aubley, C. (1998) *Tuning and Sizing NT Server*, Prentice Hall, NJ

[2] Conradsen, O., et al (2000) *Implementing SAS on RS/6000,* IBM ITSO

[3] Field, G., et al (2000) *The Book of SCSI*, No Starch Press, CA

[4] Menasce, D.A. & Almeida, V.A.F. (1998) *Capacity Planning for Web Performance*, Prentice Hall, NJ

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Edward Hayes-Hall
IBM Corporation
11400 Burnett Road
Austin TX 78758
ehayesha@us.ibm.com