Optimizing Sybase ASE for IBM AIX 5L

A joint IBM and Sybase white paper on optimizing Sybase ASE and AIX 5L

Peter H. Barnett (IBM) and Stefan Karlsson (Sybase)
March 2006

Table of contents

Abstract

IBM and Sybase have jointly authored this white paper to assist database administrators (DBAs) and UNIX system administrators in working together. This paper will help these professionals achieve the best possible performance from the Sybase Adaptive Server Enterprise (ASE) on the IBM System p family running in the IBM AIX 5L operating environment.

The focus of this white paper is on Sybase ASE 12.5.1 ESD #1 and later releases that are running under IBM AIX 5L Version 5.2 and V5.3. Its recommendations apply to AIX 5L logical partitions (LPARs) with dedicated CPUs, memory, and devices. Administrators can increase, reduce, or modify these resources through dynamic logical partitioning (dynamic LPAR) in both AIX 5L V5.2 and V5.3. A future revision of this document might incorporate recommendations for Virtual I/O and shared CPUs that only AIX 5L V5.3 supports.

Introduction

The subject of this paper is particularly relevant to those businesses that are implementing new Sybase® Adaptive Server Enterprise® (ASE) instances on an IBM® System p™ family with the AIX 5L™ operating system. It is also relevant to those organizations that are either upgrading from earlier pSeries and AIX® installations or porting to a System p and AIX 5L environment from another version of the UNIX® operating system.

In each case (new implementations, upgrades, or migrations), administrators can encounter features of the AIX 5L operating system and the System p architecture (including its management of memory and CPU utilization, as well as network I/O and disk I/O operations) that are unfamiliar. Administrators might also have questions about how best to implement Sybase ASE in their environment. The high throughput and unique CPU dispatching mechanism of the IBM POWER5™ processor-based architecture can also require revisions of older configurations.

The scope of this white paper is limited to the interface between Sybase ASE and the AIX 5L operating system. Sybase ASE manuals address general ASE tuning, such as query optimization, indexing, or lock contention.

Put another way, the focus of this white paper is on the aspect of database administration in which the DBA and UNIX SA most closely interact. The reader will ideally have the following prerequisite knowledge:

- Competence in Sybase ASE (on the part of the DBA)
- Competence in the System p and AIX 5L environment (on the part of the UNIX SA)
- A reasonable acquaintance (on the part of both the DBA and the SA) for the other's specialty

This paper discusses some commands that require UNIX **root** privileges. Others require Sybase ASE **sa_role** authority. Close communication and cooperation between SA and DBA is essential in every aspect of tuning, including:

- Sizing and allocation of shared memory in the ASE engine
- Configuration of virtual memory in the operating system
- The relation of CPU resources to online Sybase engines
- Storage considerations, such as asynchronous I/O configuration

This paper's structure addresses these areas, with separate sections on virtual memory, processing, and storage. Following these three major sections, there are two short sections on networking and a final section on other configurable settings.

The recommendations have been gathered and compiled from a number of resources, including:

- Experience in the field with ASE in the AIX 5L environment (pertaining to development, testing, and production systems)
- Laboratory experiments that specialists within Sybase and IBM have conducted
- Close cooperation and discussions between IBM AIX and Sybase ASE performance and development teams

System and database administration might come close to being an exact science, but performance tuning is an art. The recommendations in this paper are a starting point. Administrators can apply these techniques in consultation with end users, business units, and application developers. Every enterprise has its unique characteristics, its mixture of batch and online transaction processing (OLTP), its daily ebb and flow of demand, and its evolution and growth.

Likewise, both Sybase and AIX specialists have changed performance recommendations over time. This will continue to be the case as Sybase, the AIX 5L operating environment, and the System p family evolve in parallel. IBM and Sybase have committed to continuing a dialogue to keep their recommendations current and relevant.

The authors want to close the introduction and continue with the major content of the paper after sharing an observation and a strong recommendation: In many organizations, a significant disconnect exists between database administration and system administration. This adversely affects system performance and availability. Therefore, it is important to establish close cooperation and direct communication between these units and roles to build stronger business benefits from the effective use of the system.

Considering virtual memory

Incorrect sizing or operating system configuration can severely degrade performance. This section provides a minimal background and focuses on relevant configuration advice in the area of virtual memory management.

All host applications consume memory, because of the code, static variables, and more. Certain applications have greater memory requirements than others. Typically, an administrator configures a relational database management system (RDBMS) to use the majority of the host's memory for data cache(s). However, other applications can use significant amounts as well. For example, memory must be available for the following:

- Operating system
- File system buffer cache
- Windowing environment (GUI)
- Batch operations, such as loading or unloading data (bulk copy)
- Maintenance operations, such as database backup and restore
- Other applications

If virtual memory approaches the point of over allocation (being overcommitted), then the operating system looks for memory pages to write to disk, in order to free up memory and make it available for other purposes. This process, typically referred to as paging, is only of concern when some memory area approaches its limits or the search for victim pages requires significant resources. The main performance hit stems from the fact that the application expects to find some piece of data in memory, yet the data has been paged to disk. Retrieving this paged data can result in throughput that is an order of magnitude slower.

Similar symptoms also occur when the administrator has not sufficiently configured the operating system. In such cases, unprioritized applications can use too much memory, or the operating system might choose the wrong type of memory to page to disk.

ASE memory usage

Sybase ASE uses memory for many purposes, including the following:

- Configuration (static and dynamic configurable parameters)
- Resources such as **number of open objects**
- User tasks (number of user connections)
- Data caches
- Procedure cache

During startup, the ASE engine will allocate shared memory to fit all these needs, up to the limit defined by the **max memory** value. If this is set too low, then the engine will not start. If it is set higher than the above items need, the ASE instance will only request the required amount of memory.

The classic ASE scheme for allocating shared memory was to request the total **max memory** amount at startup. Users can revert to the classical scheme by setting the **allocate max shared memory** parameter to **1**. This does not limit the administrator's ability to reconfigure the ASE engine dynamically, unless the **dynamic allocation on demand** value has changed from its default setting. Optionally, the ASE engine also allows administrators to lock its memory into physical memory. (See the **Locking ASE memory in physical memory** section.)

You can decrease the amount of memory that the ASE engine uses by increasing the **max memory** value during run time. However, for shrinking memory, there are restrictions. The reason for this is that the AIX 5L environment only releases memory from the high end (for example, the last memory allocated). If the free memory is not at the upper end, the administrator cannot release it.

AIX 5L virtual memory

For a dedicated ASE host, the two largest consumers of memory will likely be the ASE engine and the file-system buffer cache. When memory is close to being over allocated (overcommitted), then the AIX 5L operating system will page memory to disk. (This is also true for any other operating system in which ASE runs.) File buffer pages and pages that belong to executing processes, such as ASE instances, can be paged to disk. The latter case obviously causes far greater performance issues.

The size of the file system buffer cache is set as a percentage of physical memory (RAM) through the **minperm%** and **maxperm%** parameters. Within this cache, the journaled file system (JFS), as well as the client file systems, is managed. Common examples of client file systems are the enhanced journaled file

system (JFS2) and the network file system (NFS). The administrator can set the client file system buffer cache maximum size (through the **maxclient%** parameter) to a percentage of RAM.

One approach for tuning the file-system buffer cache size is to set a hard limit on the file system size. To do this, administrators set the **maxperm%** parameter to some low figure, then enforce this as a hard limit by setting the **strict_maxperm** parameter to **1**. (The **maxperm%** parameter defaults to a soft limit.) Enforcing a hard limit has two drawbacks:

- When workloads change, the administrator must change the configuration.
- When memory is overcommitted, the system does not behave gracefully.

The sizing and configuration guidelines below present an approach for ensuring performance and reliability without demanding regular reconfiguration.

Memory sizing guidelines

An administrator usually sizes a dedicated ASE host large enough to accommodate one or more ASE instances. The general guideline is to dedicate as much memory as possible to the ASE engine, while also saving space for other applications. The latter must be emphasized as proper sizing is imperative in the AIX 5L environment. The recommendations (below) strive to provide as robust a solution as possible.

Here is a simplified example of the sizing process:

- The operating system normally receives 1 gigabyte of physical memory. If a host has 16 gigabytes of RAM, then 15 gigabytes is now available.
- Assuming there are no other major applications and no windowing environment, the next item is the file system buffer cache. In this example, it is reasonable to assume a total of 3 gigabytes for the JFS and JFS2 file system buffer caches, leaving 12 gigabytes of memory available.

  Note: As stated in the **AIX 5L virtual memory** section, this does not imply that you limit file system buffer cache size to a maximum of 3 gigabytes. Instead, you can determine this maximum by examining empirical data from monitoring.
- The next consideration involves batch jobs and maintenance tasks. In this case, batch jobs do not run concurrently with maintenance tasks. The greater of these might consume 1 gigabyte of memory for the duration of the operation, which leaves 11 gigabytes available for allocation.
- Reserving 1 gigabyte of memory (for headroom and future increases in batch jobs) suggests configuring ASE for 10 gigabytes of memory.

  Note: In this example, if the 10 gigabytes to the ASE engine is insufficient, you can add more physical memory to the host system.

AIX 5L configuration guidelines

A versatile operating system that runs applications of different characteristics, such as the AIX 5L operating environment, requires that the administrator configure it specifically for its intended use. In the case of virtual memory, there are some areas of greater interest, where you need to ensure system performance and provide a robust configuration. Below is a list of settings to help you achieve these goals. The list includes details on individual parameters and what they accomplish together, followed by instructions on how to view and change their settings.

- **maxperm%** = **maxclient%** = <a high percentage for each>
  [Leave each of these values at their default settings of 80%. If possible, set these values so that the **maxclient%** parameter is always greater than the **numclient%** parameter. To determine the current **numclient%** value, use the AIX 5L **vmstat –v** command.]
- **minperm%** = <low percentage>
  [This value is usually between 5-10%, but you can set it to be **1%**. Set this value so that the **minperm%** parameter is always less than the **numperm%** parameter. To find the current **numperm%** values, use the **vmstat -v** command.]
- **strict_maxperm** = **0**
- **strict_maxclient** = **1**
- **lru_file_repage** = **0**
- **lru_poll_interval** = **10**

Notes:

- In contrast with many other UNIX operating systems, the AIX 5L environment does not require you to configure limits for semaphores or shared memory size. (For example, compare to the **kernel.shmmax** tunable (configuration parameter) on the Linux™ operating system.)
- To find the amount of physical memory, use the AIX 5L **prtconf** command (**/usr/sbin/prtconf**). The **vmstat -v** command also displays physical memory, but in 4-kilobyte pages, not in megabytes. (See the sample output of the **prtconf** command in the **Understanding processor considerations** section.)

Setting the file system buffer cache configuration

The **minperm%** and **maxperm%** parameters (which represent a percentage of physical memory) specify the AIX 5L file buffer size. Within this cache, The AIX 5L operating system also manages file pages for client file systems (for example, JFS2 and NFS). Administrators set the maximum size of this subset through the **maxclient%** parameter, again representing a percentage of physical memory.

If the **numperm%** parameter drops below the **minperm%** value, then the AIX 5L operating system might page computational pages instead (for example, pages belonging to the ASE processes). Hence, you must set the **minperm%** parameter low to prevent this.

By setting the **maxperm%** parameter to a high value, the file system buffer cache can grow, but not at the expense of computational pages, because the **lru_file_repage** value is set to **0**.

The AIX 5L operating system can page file system buffer pages as well as computational pages to disk. When memory requirements exceed the amount that is currently available, a certain amount of

paging occurs. At this time, the least recently used (LRU) algorithm comes into play. (You are probably already familiar with LRU because of its use in ASE data caches.) A good example of using the LRU algorithm involves burst loads where a large amount of file buffers are currently in use. Then, the AIX 5L page stealing daemon (**lrud**) examines memory pages to determine if a particular page has been paged recently; if not, the **lrud** process sends the page to disk. This happens regardless of whether the page belongs to a file or a process. The recommended **lru_file_repage = 0** setting prevents the **lrud** process from paging computational pages in these circumstances. Setting the **lru_poll_interval** value improves the responsiveness of the **lrud** process while it is running.

Note: The **lru_file_repage** parameter is tunable in AIX 5L V5.2.05 and above. For AIX 5L V5.2, this tunable parameter is available as a fix through the authorized problem analysis report (APAR) IY62224.

Viewing and setting virtual memory configuration parameters

Administrators can view and set parameters related to the AIX 5L Virtual Memory Manager (VMM) through the AIX 5L **vmo** command, which requires root privileges. Furthermore, you can see many VMM parameters by using the **vmstat -v** command, which, by default, does not require root privileges.

- To view all parameters and their respective settings, use the **vmo -a** command.
- To set the **maxperm%** parameter to 80%, use the **vmo -p -o maxperm%=80** command.

ASE memory locks in physical memory

Administrators can configure the ASE engine to lock (pin) its memory into physical memory. This requires both operating system and ASE configuration. If you decide to employ pinning, you can configure the ASE instance to allocate all configured shared memory during startup. The following demonstrates the configuration of the AIX 5L V5.2 (or V5.3) operating system and ASE for locking ASE memory:

1. To enable pinning from the operating-system level, use the **vmo –p –o v_pinshm=1** command.
2. To configure ASE to lock shared memory, use the ASE **EXEC sp_configure 'lock shared memory', 1** command.
3. To configure ASE to allocate all configured shared memory during startup, use the **EXEC sp_configure 'allocate max shared memory', 1** command.

Notes:

- For this to take effect, you will need to restart the ASE engine.
- By default, AIX 5L V5.3 allows 80% of physical memory to be pinned. To change this, use the **vmo-p –o maxpin%=<value>** command.
- Locking shared memory can cause problems for other applications if you have not properly sized the memory. Trying to lock too much memory can cause freezing on some systems.
- If ASE memory is increased dynamically, there will be no locking of the added memory.

Memory usage monitoring

The best way to understand ASE engine memory requirements is to check the size of its shared memory.

The ipcs command

See the example use of the AIX 5L **ipcs** investigator command, below:

```
# ipcs –b -m:
IPC status from /dev/mem as of Wed Dec 28 12:32:27 est 2005
T        ID     KEY         MODE          OWNER    GROUP     SEGSZ
Shared Memory:
m         8     0xd8009059  --rw-------   sybase   sybase    2147483648
m         9     0xd800905a  --rw-------   sybase   sybase    877723648
#
```

The combined byte counts in the SEGSZ column (shown in red, above) might be much less than the ASE configuration **max memory** parameter. This is an indication that ASE is configured not to allocate all memory at startup, but to allocate memory on demand. This is the default setting.

The AIX 5L VMM has a passive allocation algorithm for shared memory. This means that VMM reserves all the memory that an application requests; however, VMM does not actually allocate the memory until the application references it.

Applications that terminate abnormally, including ASE itself, can leave shared memory segments stranded. It is important that you explicitly remove these stranded memory segments by using the AIX 5L **ipcrm** command.

This is one of many reasons for close cooperation between DBAs and system administrators.

The vmstat command

Administrators can monitor virtual memory usage with the AIX 5L **vmstat -I** command (upper case i). The output of this command contains relevant information in the **avm**, **pi**, **po**, **fr** and **sr** columns.

- The **avm** column shows the amount of virtual memory in use (in 4-kilobyte memory pages).
- The **pi** column shows the number of cache misses that are paged in.
- The **po** column shows the number of pages that are paged out.
- The **fr** column shows the number of pages freed.
- The **sr** (scan rate) column shows the number of pages that were scanned to find eligible pages to page out.

```
# vmstat -I 10
System Configuration: lcpu=32 mem=191488MB
  kthr     memory              page              faults            cpu
 r  b  p      avm   fre  fi   fo  pi  po  fr   sr   in     sy     cs us sy id wa
 8  1  0 12945134  4213 151 7488   0  21 179  193  711 202851 203538 15  9 76  1
 9  1  0 12945157  3926  25 6423   0  23 179  277  453 116885 191633 14  8 78  1
 8  1  0 12945194  5759  15 9065   0  24 231  463 2008 125516 190439 14  9 76  1
 ...
```

Note: In contrast with other UNIX implementations, AIX 5L file system I/O operations are separate from the paging operations. You can see this by looking at the **fi** and **fo** columns from the **vmstat –I** command output (shown above). This helps in determining whether memory is actually running low.

Using large pages for the buffer cache

Large pages (16 megabytes) can make a noticeable improvement in performance for large memory configurations by decreasing the number of translation lookaside buffer (TLB) misses.

The steps for using large pages include the following:

1. Ensure that ASE memory is locked. (See the **Locking ASE memory in physical memory** section.)
2. Use the **lgpg_regions** and **lgpg_size** tunable parameters (in the **vmo** command) to specify how many large pages to pin. Then use the AIX 5L **bosboot** or **reboot** command to enable the large page variables you have specified.
3. Configure the application user ID with permission to use large pages, as follows:
   chuser capabilities=CAP_BYPASS_RAC_VMM,CAP_PROPAGATE <user id>
4. Verify the use of large pages with either the AIX 5L **vmstat -l** or the **svmon -G** (global view) commands.

Reclaiming swap space

AIX 5L V5.3 periodically reclaims swap space for reuse after the pages it contains have been **paged back** into physical memory. This is true even when long-running processes, such as ASE, are still running. In AIX 5L V5.3, if the initial paging space is of adequate size (usually 4.5 gigabytes for large memory systems), you do not have to worry about running out of swap space.

In AIX 5L V5.2, when paging occurs, occupied swap space is not released, even after the pages are paged back, until the owner process terminates. This creates a small risk for long-running processes that might run out of swap space. For the sake of performance, the best defense against this is to avoid paging altogether by using the virtual memory tuning recommendations offered above. Another option is to recalculate the memory needed for the ASE instance and the operating system and for related batch and maintenance processing. You can then pin this amount of memory to the ASE shared memory pool.

Processor considerations

Configuring the number of processors and ASE engines should be a straightforward task, but this is not always the case. From a performance point-of-view, it is highly desirable to keep ASE engines fairly busy (no lower than 50% and, preferably, upwards of 80 to 90%). Also, configure a maximum of one ASE engine per physical CPU or core. The best practice is to keep no more Sybase engines online than you can maintain more than 60% busy. However, the existence of varying workloads and intermittent-burst activity, as well as the anticipation of added users and application functions can take precedence over the recommended configuration. In short, you sacrifice performance configurations to provide ample headroom. When you also need to allocate CPU costs among departments, the allotment of processors becomes even more challenging.

ASE engines are peers; each is responsible for its own network and disk I/O processes. An engine that has started an I/O method must also manage it (from polling to I/O completion and delivering the I/O event to the sleeping task). As an ASE engine starts idling (for example, when there are no tasks on local or global run queues that the engine can run) it does not yield the CPU immediately. Instead, the ASE engine tries to stay on the CPU, the rationale being that it is beneficial for both throughput and latencies.

Therefore, instead of yielding, the engine looks for work to do by continuously checking for network I/O and completed disk I/O processes, and other tasks to run on the run queues. On the AIX 5L operating system, the completion of disk I/O methods does not require a system call, but polling the network does.

For a busy ASE engine, user time typically far outweighs system time, but an idling engine reverses this ratio; the data server process consumes a significant amount of system time and a marginal amount of user time. This is further aggravated by what appears to be a relative high cost of the AIX 5L **select()** system call. You can use the ASE **sp_sysmon** system procedure to troubleshoot the throughput issue and, in an extreme case, you might witness 100% CPU utilization at the operating-system level (from the output of the **vmstat** command) and only 25% average CPU busy at the Sybase level (from the output of the **sp_sysmon** procedure). If you improperly size the system, the system CPU time consumed in polling might actually degrade overall performance, even on a dedicated Sybase server.

Note: The ASE instance does not use the AIX 5L **select()** system call to poll for disk I/O completion; it only uses this **select()** system call to poll for network I/O completion.

Administrators can monitor the CPU through various means. As already mentioned, one common method is through the **vmstat** command, which displays **system**, **user**, **I/O wait**, and **idle** times. You can also identify the number of physical CPUs by issuing the AIX 5L **prtconf** command (**/usr/sbin/prtconf**) as shown below (in red):

```
# prtconf
System Model: IBM,9113-550
Machine Serial Number: 10BE8BD
Processor Type: PowerPC_POWER5
Number Of Processors: 1
Processor Clock Speed: 1656 MHz
CPU Type: 64-bit
Kernel Type: 64-bit
LPAR Info: 1 ibmspeed
Memory Size: 27776 MB
Good Memory Size: 27776 MB
Platform Firmware level: Not Available
…
```

Simultaneous multithreading with AIX V5.3

Simultaneous multithreading (SMT) is a POWER5 technology that goes beyond traditional symmetrical multiprocessing (SMP) in allowing two threads to execute the same type of operation (for example, integer or floating point) at the same time. Its implementation presents the appearance of two logical processors to the operating system for each physical processor that actually exists on the server. The benefit of SMT is that another process can use the CPU time when one process experiences latencies. In contrast with some technologies that are similar in appearance, this might help performance by as much as 30%. It is important to note that two logical processors do not have 200% of the capacity of a single physical processor, but might realistically have as much as 130%. Most RDBMS workloads enjoy the benefit of SMT, but certain highly uniform and repetitive processes do not. The AIX 5L V5.3 environment turns SMT **on** by default. However, the administrator can toggle SMT **on** or **off** without the need for a reboot.

You can identify whether SMT is enabled or disabled by comparing the output of two AIX 5L commands (**lsdev -Cc processor** and **bindprocessor –q**). Both are unprivileged commands. If SMT is enabled, the output of the **bindprocessor** command will be twice that of the **lsdev** command.

Logical partitions

Several POWER4™ processor-based systems and all POWER5 processor-based systems are LPAR-capable. That is, they support logical partitions, multiple operating system images of varying sizes and capacities on a single physical frame. Even if there is only one image on an applicable hardware platform, it is still an LPAR. All tuning recommendations contained in this paper apply to AIX 5L operating system images in the form of LPARs.

LPAR-capable POWER4 and POWER5 processor-based systems support the dynamic reallocation of memory and CPUs. On POWER5 processor-based systems, the system administrator can add or remove CPUs and memory in an LPAR without rebooting.

Sybase and IBM are still addressing two points in ongoing studies, regarding how Sybase ASE can best take advantage of virtualization technologies:

- Is it is possible to add or remove CPUs and memory programmatically from an ASE LPAR, according to a policy?
- Can a group of ASE LPARs share CPUs?

Sizing guidelines

Administrators are often tempted to keep the previously assigned CPU configuration and allocation, in particular when consolidating systems or migrating from slower CPUs to faster AIX 5L environments.. This often leads to idle Sybase engines and wasted CPUs. At most, one active ASE engine per physical CPU, or core, stands as a sizing guideline. Using more active ASE engines than the number of physical CPUs is typically detrimental to performance, throughput, and response times.

However, some environments benefit from the reverse: running fewer Sybase engines than there are CPUs. Other tasks (applications and operating system processes) require CPU resources. These include native threads (for example, Sybase Real-Time Data Service [RTDS]) and the asynchronous I/O servers for asynchronous I/O processes on the file systems. Administrators often forget the last process, which leads to poor performance in high workload situations. For this reason, administrators must take asynchronous I/O servers and other processes into account when allocating CPU resources. It is up to you to test and decide what is most beneficial to your organization.

Configuration recommendations

Even after an ASE engines starts up, an administrator can bring additional ASE engines online dynamically. The ASE **number of engines at startup** parameter defines the number of engines initially created during startup. If an administrator configures additional engines through the ASE **max online engines** parameter, these new ASE instances can come online by using the ASE **EXEC sp_engine 'online'** command.

Note: ASE supports taking engines offline through the ASE **sp_sysmon** procedure (**EXEC sp_engine 'offline' [, <engine ID>]**). However, this will not always succeed because the

designated engine might hold resources that cannot be released. You can use the **can_offline** argument to **sp_engine** procedure to find out whether it is possible to take an engine offline.

To decrease the amount of CPU time that idling ASE engines consume, set the ASE **runnable process search count** tuning parameter to **3**. This means an engine that cannot find a task to run in local or global queues will only loop three times looking for work, instead of the default 2,000 times.

There are two reasons for concern about decreasing the **runnable process search count** parameter:

- Doing so might increase latency because the operating system must schedule the engine onto a CPU before the engine can attend to incoming work.

  (This point is generally true for many UNIX installations. However, the joint conclusion between ASE and AIX 5L engineering teams is that a lower **runnable process search count** value does not adversely affect performance to the same extent that it has in the past. One reason for this is that ASE engines now have the ability to wake up other engines that have work to do. Another reason is that POWER5 processor-based systems are just so very fast; thus, the throughput benefit previously derived from looping many times diminishes.)

- Decreasing this value simply does not work in all circumstances.

  (This point is fully correct. An engine with at least one outstanding I/O process will not yield the CPU, even if the **runnable process search count** value is exhausted.)

Note: A trace flag allegedly helps an ASE engine yield, even though it still has outstanding I/O processes. However, it is better not to use this flag as it can have considerable detrimental effects in the AIX 5L environment.

To increase stability, set the ASE thread scope to system (**S**). When you set the thread scope to (**S**), the system uses native threads for RTDS and Lightweight Directory Access Protocol (LDAP) services. It also uses native threads when ASE reads the interfaces file to do a backup, performs a remote procedure call (RPC), or starts the Sybase Extended Procedure (XP) server. Set the ASE thread scope in the ASE **RUN_server** file before the ASE **dataserver** command (shown in red, below):

```
#!/bin/sh
#
# ASE page size (KB):   4k
# Master device path:   /sybase/devices/SYBASE.master
# Error log path:       /sybase/15.0/ASE-15_0/install/SYBASE.log
# Configuration file path:      /sybase/15.0/ASE-15_0/SYBASE.cfg
# Directory for shared memory files:    /sybase/15.0/ASE-15_0
# Adaptive Server name: SYBASE
#

AIXTHREAD_SCOPE=S
export AIXTHREAD_SCOPE

/sybase/15.0/ASE-15_0/bin/dataserver \
-d/sybase/devices/SYBASE.master \
-e/sybase/15.0/ASE-15_0/install/SYBASE.log \
-c/sybase/15.0/ASE-15_0/SYBASE.cfg \
-M/sybase/15.0/ASE-15_0 \
-sSYBASE \
```

Note: Although this example is with SYBASE ASE 15, the recommendation holds true for SYBASE ASE 12.5, as well. Lastly, for several reasons, it is best not to bind the ASE engine to a CPU for most applications.

Storage alternatives

There are several alternatives for storing data when using Sybase ASE on the AIX 5L operating system. These choices include raw devices and file system devices, as well as file system devices accessed through direct I/O and Quick I/O devices.

Raw devices

Using raw asynchronous disk I/O on character devices offers the best performance alternative and requires no particular configuration. Devices can be logical volumes on storage area networks (SANs) logical unit numbers (LUNs), or partitions on local disks or JBOD (just a bunch of disks). The performance improvement stems from the short code path and lack of double buffering. (See the information on file I/O, below.)

The main overhead related to raw asynchronous disk I/O involves the manual tasks of setting up and maintaining the devices themselves.

File system devices

Reading and writing to files is quite simple. Operating systems provide significant services to help developers be productive in coding these tasks. Utilizing these services reliably and efficiently is harder. In particular, there are two subjects of concern.

Buffered I/O writes

The application opens the files one time, then uses system calls to read from or write to the file. In the case of multiple processes that are writing to the same file, as with Sybase ASE, then access must be coordinated.

When using the AIX 5L **write()** call, writes to files are buffered by default. The data that is to be written is cached in the AIX 5L file system buffer cache and is actually written to disk at some later point in time. Applications can request that the operating system flush the written (pending) data to disk and acknowledge the successful write only after the disk I/O method has actually completed. To make this request, the AIX 5L operating system provides flags that applications can set when opening the file. The **dsync** attribute is the ASE default device setting that provides durable writes; it corresponds to the **O_DSYNC** flag for the AIX 5L **open()** system call. ASE defaults to requesting safe I/O writes from the operating system.

Therefore, I/O operations on a file will traverse a code path and a buffer cache that the operating system maintains.

Asynchronous I/O methods

Application developers must also concern themselves with the best method for processing asynchronous I/O methods on files. Some facility must manage the pending I/O to allow the

application to do other work while the disk I/O method is performed. In the case of the AIX 5L operating system, the application places I/O requests in a queue. Asynchronous I/O servers pick up the I/O methods from the queue and manage their associated asynchronous I/O processes. The AIX 5L operating system notifies the ASE engine of an I/O completion through a flag in the ASE memory space, the same as for asynchronous I/O methods on raw devices.

Quick I/O and direct I/O methods

Various techniques can help avoid the overhead that is associated with file system access. The two that ASE supports are:

- **Quick I/O method:** The proprietary VERITAS® I/O method that is available on the VERITAS File System™ (VxFS)
- **Direct I/O method:** The standard multiplatform I/O method that is available on JFS and JFS2 with
  Sybase ASE 15.0 (JFS2 is the recommendation for the reason stated below.)

Both techniques utilize optimized code paths and avoid the file-system buffer cache. This provides better performance and guarantees durable writes. Although results always vary between applications and workloads, in-house tests show a three-fold performance increase with the direct I/O method, as compared to regular file system devices with the **DSYNC** attribute set. The Quick I/O method on VxFS provides comparable performance benefits.

Considerations for temporary databases

ASE does not need to recover data in the **tempdb** database following a crash. Leverage this fact for performance benefits. Since the release of Sybase ASE 12.5.0.3, there has been an introduction of optimizations to decrease the amount of writes that are done in a temporary database (called lazy writes). Still, if the amount of data in a **tempdb** file exceeds the cache size, then the ASE engine obviously needs to write the data to disk. Also, ASE writes data to disk as part of truncating the log (as indicated by the ASE **trunc log on chckpt** value that is set on temporary databases). The ASE engine will also write data to disk during page splits. Nevertheless, with these two exceptions, the inline writes to disk are entirely gone.

With the **DSYNC** attribute set to **off**, file system devices provide fast writes. Because of this, each file system device that is used solely for the **tempdb** file must have this option set to **off**. Use the **sp_deviceattr** system procedure to do this:

EXEC sp_deviceattr '<device name', 'dsync', 'false'

If devices are up to 2 gigabytes in size, then you can use JFS, otherwise, use JFS2 without any mount options.

Storage recommendations

In general, raw devices provide the highest performing alternative, with no particular requirement on the AIX 5L operating system configuration apart from enabling the asynchronous I/O devices. For logical volumes, the devices that are associated with a logical volume have an **r** prefix (for example, **/dev/rmy_lv00** instead of **/dev/my_lv00**).

Use the AIX 5L System Management Interface Tool (SMIT) to enable asynchronous I/O, regardless of whether you use raw devices or any file system.

The Sybase user generally owns raw devices. If the user that brings up the ASE engine does not have read and write privileges on the raw device, the ASE engine will not start, or it will start with one or more devices and databases offline. Certain AIX 5L maintenance operations require that the root touch the AIX 5L **character special** file, which identifies a raw device (for example, **/dev/rmy_lv00**). This will change the ownership back to root. The system administrator must take care when performing maintenance, such as increasing the size of a logical volume, moving the logical volume to another physical device, exporting and importing a volume group containing ASE devices, and synchronizing the Object Data Manager (ODM). In IBM High-Availability Cluster Multi-Processing (IBM HACMP™) clusters, it is best to include commands for reowning the Sybase devices in the start script. These device-reowning commands have to precede the command that starts the ASE engine.

If the raw device is not a feasible choice, then use Quick I/O or direct I/O performing alternative. To mount a file system for direct I/O methods, use the **dio** option to the AIX 5L **mount** command: **mount –o dio <fsname>**. Typically, however, you do this through the SMIT command. Then, use either the appropriate option for the ASE **DISK INIT** command or the **sp_deviceattr** system procedure to set or change the existing status of a device:

EXEC sp_deviceattr '<device name>', 'directio', 'true'

Notes:

- The direct I/O method is only available in Sybase ASE 15.0 or later.
- The direct I/O method and the ASE **DSYNC** attribute are mutually exclusive. You must disable the **DSYNC** attribute before enabling the direct I/O method.
- If the I/O size is not a multiple of the file system block size do not match, the AIX operating system will silently demote the I/O method from a direct I/O method to a regular file system I/O method with the inherent performance penalty. The fact that the JFS block size is 128 kilobytes implies that you must use the JFS2 file system. It also implies that either the ASE engine must use a 4-kilobyte page size (the JFS2 default block size is 4 kilobytes) or the JFS2 block size is changed to match the ASE page size. Lastly, as long as the requested I/O size is a multiple of the block size, the direct I/O method is used. For example, a 16-kilobyte I/O request on a 4-kilobyte block-size file system will use the direct I/O method.

For the Quick I/O method, there is no specific ASE configuration requirement, as ASE will automatically use the Quick I/O method on the VxFS. (See the appropriate VERITAS manuals for this separately licensed product.)

For all file system-based devices, it is a good practice to increase certain configuration options from their respective default settings:

- **MINIMUM number of servers:** <leave at default>
- **MAXIMUM number of servers per cpu:** 300
- **Maximum number of REQUESTS:** 8192
- **Server PRIORITY:** 39
- **STATE to be configured at system restart:** available
- **State of fast path:** enable

You can view and set these options through the AIX 5L SMIT (**smitty aio)** command. (This command requires root privileges to set the parameters but not to view the parameters.) The two latter configuration options are necessary for asynchronous I/O methods to raw devices.

You can display the currently running asynchronous I/O servers with the AIX 5L **pstat** command or the standard AIX 5L **ps** command: **pstat -a | fgrep aio** or **ps -k | fgrep aio**

Notes:

- Asynchronous I/O servers are kernel processes. The **-k** parameter of the **ps** command is required to display them.
- Although the **pstat** command does not require root privileges, the resources that the command uses might.
- See the AIX 5L operating system documentation for a description of POSIX and legacy asynchronous I/O servers. The ASE engine uses legacy asynchronous I/O servers. Therefore, the asynchronous I/O servers are named **aioserver** and not **posix_aioserver**.

Setting the queue length to **8192** is sufficient for most environments but you might want to increase this number for high-end systems. The output from the ASE **sp_sysmon** command will indicate this by reporting that the operating system limit is delaying the I/O method.

It is worth repeating that asynchronous I/O servers require CPU resources and therefore you must consider this for proper CPU sizing. In addition, you must change the default ASE configuration:

disk i/o structures = 1024

This is a good starting point, but you should increase the disk I/O value only if the **sp_sysmon** output indicates I/O delays that result from disk I/O structures.

Examining network performance

Increased ASE network packet (tabular data stream or TDS packet) sizes usually provide a performance benefit and almost never hurt performance. Usually, a 2-kilobyte TDS packet size is a good starting point (by no coincidence, this is the new default in Sybase ASE 15.0). Certain operations, such as performing bulk loads, retrieving larger result sets, and operating with large objects (LOBs) benefit from a packet size that is 8 to 16 kilobytes. (The **Sybase System Administration Guide** [Volumes I and II] document client and ASE configuration. The **Resources** section of this paper provides a Web site listing for this documentation.)

A common objection to increasing the TDS packet size refers to underlying maximum transmission unit (MTU) and Ethernet frame sizes, both of which are in the 1,500-byte range. However, multiple TCP segments or Ethernet frames are significantly cheaper than multiple application-level protocol packets, (for example, TDS). Obviously, if network performance or utilization is of particular concern, then you must test the application and configuration changes.

As is true for all networks, some sanity checks are necessary (for example, to ensure consistent MTU across nodes). MTU mismatches en route between a client and the ASE engine can cause severe performance problems for OLTP applications as well as decision support systems (DSS). The default AIX network configuration is suitable for most topologies and host configurations. However, it is best to check the excellent IBM discussions on network tuning in the **AIX 5L Practical Performance Tools and**

Tuning Guide, which you can access through the IBM Redbooks™ Web site listed in the **Resources** section of this paper.. Some parameters to consider include **tcp_sendspace** and **tcp_recvspace**. For AIX 5L V5.3, the network section of the performance and tuning manual is located in the **Performance Management Guide** Web site listed in the **Resources** section.

You can view and set network tunable parameters through the root-privileged, AIX 5L **no** command. The syntax is consistent with the **vmo** command and other AIX tuning commands. The universal UNIX **netstat** command allows administrators to monitor network-level statistics. The AIX **entstat -d <ent#>** command is used to monitor the adapter level. Neither of these monitoring utilities requires root privileges. Check the output from the **netstat -ir** command for packet-level errors and for relative traffic. Check the output from the **entstat -d** command for collisions and overflows. You can also use the **netstat -a** command to check for hung sockets. Look at the values displayed on the output lines for **CLOSE_WAIT** or **FIN_WAIT**.

In a mixed-platform environment (for example, a Linux, Hewlett-Packard® HPUX, or Sun® Solaris™ client and an AIX 5L data server) it is important to look for differences in the TCP/IP stack parameter defaults among the various platforms.

Here are three examples that compare some default parameter values between Solaris and AIX 5L environments:

- In the AIX 5L environment, the **sack** (selective retransmission of dropped packets) value is set to **off** by default. Instead, there is a different retransmission algorithm, **tcp_newreno**. In the Solaris environment, a similar algorithm to enable sack is **tcp_sack_permitted=2**. There is no direct equivalent for the **tcp_newreno** algorithm.
- By default, the AIX 5L operating system enables the **RFC1323** parameter, but not for Solaris 8 and later releases. In Solaris 8, the equivalent parameter is **tcp_wscale_always**.
- The **tcp_sendspace** and **tcp_recvspace** parameters must agree between the client and server. In Solaris 8, the equivalent of the **tcp_sendspace** parameter is **tcp_recv_hiwat**.

Understanding other configurable limits

Occasionally, you might encounter challenges during backups or unloads. You can trace these to problems related to creating large files. The root cause is that the default file-size limit is 1 gigabyte.

In the AIX 5L operating system, the system administrator can manage configurable limits in the **/etc/security/limits** file. The administrator needs to create a stanza for the Sybase functional user and set the file size limit (**fsize**) to **-1**. The default behavior of the **fsize** token is to set the hard limit (**hard_fsize**) as well. It is recommended that the **nofiles** parameter in the **/etc/security/limits** file remain at its default.

Summary

Running Sybase ASE on the AIX 5L operating system is a common combination that performs very well, even for applications with very high performance requirements. As with all high-end systems, there is some necessary configuration effort to optimize the use of the platform.

It is important to size the platform to fit the intended use; this is true for memory as well as CPU requirements. Coupled with proper virtual memory configuration, correct sizing provides for a reliable and efficiently performing platform.

Resources

These Web sites provide useful references to supplement the information contained in this document:

- IBM System p5 servers
  ibm.com/systems/p
- IBM eServer pSeries LPAR specific information.

  ibm.com/servers/eserver/pseries/lpar
- IBM eServer pSeries Library

  ibm.com/servers/eserver/pseries/library
- IBM Redbooks
  ibm.com/redbooks
- Performance Management Guide
  http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.doc/aixbman/prftungd/netperf.htm
- Information about Sybase products
  Sybase.com
- Sybase ASE general information
  sybase.com/products/informationmanagement/adaptiveserverenterprise
- Sybase ASE manuals
  infocenter.sybase.com/help/topic/com.sybase.help.ase_15.0/title.htm
- Sybase ASE System Administration Guides
  infocenter.sybase.com/help/topic/com.sybase.help.ase_15.0.sag1/html/sag1/title.htm
- Sybase ASE Performance and Tuning Manuals
  infocenter.sybase.com/help/topic/com.sybase.help.ase_12.5.1.databases/html/databases/title.htm
- Sybase ASE 15.0 "What's New in Adaptive Server Enterprise"
  infocenter.sybase.com/help/topic/com.sybase.help.ase_15.0.whatsnew/html/whatsnew/title.htm

Acknowledgements

Peter H. Barnett (IBM Corporation) and Stefan Karlsson (Sybase, Inc.) authored this paper. It was built on research from the authors and IBM and Sybase colleagues, including (but not limited to): Matt Accapadi (IBM), Barry Saad (IBM). Ramesh Chitor (IBM), and David Wein (Sybase).

Special thanks to Matt Accapadi (IBM AIX engineering) and David Wein (ASE engineering).

The authors welcome readers to share their experiences and to provide feedback at: phbarn@us.ibm.com (Peter H Barnett) and Stefan.Karlsson@Sybase.com.

Trademarks and special notices

Sybase, Inc.
Worldwide Headquarters:
One Sybase Drive, Dublin, CA 94568-7902 USA
Telephone: +800 8 SYBASE
sybase.com

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

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

IBM	pSeries	POWER4	POWER5
ibm.com	AIX	AIX 5L	Redbooks
the IBM logo	System p		

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.