HP-UX PERFORMANCE MONITORING FUNDAMENTALS

Donna Fountain
Technical Consultant
Hewlett-Packard Company
19055 Pruneridge Avenue, Mailstop 46T-U2
Cupertino, CA 95014
email: donnaf@cup.hp.com

Paper Objective and Design

Designed for the beginning to intermediate level System Manager, this paper explores performance monitoring fundamentals on HP-UX. We'll examine each of the four major categories affecting system performance: CPU, disk, memory and other resource contention. For each of these primary resources, we will discuss bottleneck symptoms, performance metrics and immediate actions you can take to help alleviate bottlenecks on your system.

All of these actions are summarized in Appendix A, *First Steps in Alleviating System Bottlenecks*. Take this checklist back to your shop. You can use it as an *at-a-glance* reference for reducing immediate resource consumption and bottlenecks on your system.

For the purpose of this paper, Hewlett-Packard's GlancePlus/UX is cited, although these basic performance analysis concepts can be applied using other diagnostic system performance monitoring utilities as well.

Introduction -- Pro-active and Reactive Troubleshooting

Solving performance problems is a valuable skill. However, a more impressive accomplishment is to prevent problems from happening in the first place. Once you find the system bottlenecks via a performance monitoring utility, you must follow through

with corrective actions to reduce their effect on system performance. The key to effective performance management is to monitor the system regularly.

Through pro-active performance management you can prevent potential performance problems from happening on the system. Identify bottleneck symptoms in the early stages and take immediate corrective steps. Regularly monitor the resource utilization levels for CPU, memory and disk. The sooner you unveil a problem or potential problem, the more lead-time you'll have to put a solution in place. For example, you can plan for a CPU upgrade or redistribute workloads as your computer approaches capacity rather than when the CPU is saturated.

A performance monitoring tool can also be used for reactive troubleshooting. When slowdowns occur, activate your performance monitoring tool to assess workload imbalances or saturation of critical system resources. Is one particular group of users monopolizing system resources? Are certain types of transactions taking inordinately long to complete? Is disk activity to certain drives delayed?

Real-time performance utilities are not intended to be used for long-term trend analysis or as logging utilities. They cannot be used to address all matters regarding application performance nor can online tools be expected to meet all of your performance needs. For more indepth performance analysis and additional functionality, you will need to supplement your performance toolbox with additional utilities (e.g., tools whose primary function is longer-term data collection and analysis, utilities designed for analyzing application programs, etc.)

A Quick Look at Glance

HP GlancePlus/UX is Hewlett-Packards's online performance monitoring and diagnostic utility for HP9000 HP-UX based computers. It is an add-on application that can be installed using the swinstall utility on 10.X and the update utility on 9.X-based systems.

There are two user interfaces of GlancePlus/UX -- *glance* is character-based, and *gpm* is motif-based. Each contains graphical and tabular displays that depict how primary system resources are being utilized. *Glance* incurs less overhead, but *gpm* features additional graphical embellishments. *FOR THE PURPOSE OF THIS PAPER, WE WILL REFER TO GLANCE, SINCE IT CAN BE RUN USING JUST ABOUT ANY EQUIPMENT YOU MAY HAVE AT YOUR SITE.*

When running Glance, *Global Bars* followed by a *Process List* is displayed. The *Global Bars* contain information regarding current system usage of CPU, disk, memory and swap. The data are depicted in both bar and tabular formats and are locked at the top of

the Glance display above the *Process List*. When moving between screens in Glance, this important global utilization information will continue to be immediately available.

When Glance is running, a user can change from one screen to another by typing the letter which corresponds to a respective screen. For example, type *c* to bring up the *CPU Detail Screen* and *m* to access the *Memory Detail* display.

GlancePlus/UX consists of several screen displays including:

- A global utilization overview that is the first screen displayed when GlancePlus/UX is initially invoked.
- Several screens for each of the primary system resources -- CPU, disk, memory and swap. One change in HP-UX 10.X was that *swapping* was replaced with a nominally less expensive method called *activation/deactivation*.
- Detailed data screens reflecting process-specific data.
- System-level information including: system table utilization, networking (e.g., LAN, NFS) and diskless server.

For more information regarding these screens, please consult Glance's online help facility. If you have access to gpm, you can view gpm's *Self-Guided Tour*.

HP-UX Performance Monitoring Fundamentals

CPU BOTTLENECK INDICATORS.

The Central Processing Unit (CPU) is responsible for executing the instructions that do the work required by user and system processes. A CPU bottleneck is a situation where demand for CPU exceeds the supply. Processes on the system are requesting more CPU than the system can provide.

We'll start with CPU bottlenecks since they are the easiest to identify. Bottlenecks that originate in other parts of the system (e.g., disks, controllers, memory or networking) often manifest themselves as "CPU problems". Thus, it is important that we complete our analysis of the other system resources before declaring that a CPU bottleneck exists.

CPU UTILIZATION, CPU Global Bar

When probing the system, the system manager should check the level of CPU utilization. Although 100% busy is not a certain indicator of a CPU bottleneck, it does confirm that a lot of activity is taking place on the system. A bottleneck exists only when the resource is consumed *and* tasks waiting for that resource are being slowed down. Occasional *spikes*, or short periods of activity, do not constitute a system bottleneck.

CPU Busy % is reflected in the first global bar appearing at the top of each core GlancePlus/UX screen. Be sure to note the composition of the *Global CPU Busy Bar*. User activity can be comprised solely of online user processes, solely of batch, or a combination of both.

Examples of batch processing include background reporting jobs, program compilations and other processing which could be run at a reduced priority or during non-peak hours. Batch processing can easily cause a CPU bottleneck, if run at an inappropriate time.

Let's take two opposite examples that both reflect 100% CPU Busy. The first situation features only interactive users with no batch activity taking place on the system. The performance of online users' tasks is slow. In this case, there is probability of a CPU bottleneck.

In the second situation lots of batch activities are occuring on the system. Although the system is 100% CPU busy, batch activities are consuming surplus CPU, since minimal demand is being placed on the system for CPU by the online users. This situation is fine as long as the online users are getting good response time. However, if online response time is affected, the batch activities should be redirected to a less used system or rescheduled during non-peak hours.

You'll want to begin evaluating your current environment when the *Global CPU Busy Bar* consistently hits 60%. This is also a good time to consider future computing needs. It is always best to plan for a CPU upgrade well in advance of the actual timeframe when you'll be needing it.

SYSTEM TIME VERSUS USER TIME, CPU Detail Screen (c)

Generally speaking, the greater the time spent in User Time, the better the throughput is on a system. User time indicates that real work is getting done. On the other hand, System Time -- time spent in the HP-UX kernel -- is overhead and reduces the overall amount of resources available to the applications.

Problems can arise when user processes spend too much time inside kernel code. That is, the user processes cause an excessive number of system calls to be executed. Investigate

applications that spend 50% or more of their time in system code. Remedies will usually involve redesign and programming changes (e.g., reduce the number of process creates to reduce the amount of time spent in system code). This is beyond the scope of a System Manager, but you can help by informing your MIS or Programming Manager.

Applications that spend large amounts of time in User Code (i.e., 25% or less System Time) should scale well on multi-processor (MP) systems. System mode activity does not scale as well as user process throughput on MP systems.

LOAD AVERAGE, CPU Detail Screen (c)

Load Average can be a measure of contention for the CPU. It reflects user processes that are ready to run but must wait their turn for the CPU to become available. *Current* Load Average is a measure of contention for the current interval only. *Average* Load Average depicts the average number of processes waiting for CPU resources since Glance was last reset. *High* Load Average reflects the peak number of users waiting for CPU resources since Glance was last reset.

The Load Average can also be influenced by processes which are *short-waited*, such as processes waiting on disk I/O. That is why there can be times when the Load Average is high yet the CPU is not fully utilized. Usually the Load Average is a good indicator of CPU queuing but keep in mind that intensive disk workloads can also contribute to a high Load Average.

Note any queuing for CPU resources. Investigate why processes are waiting for CPU resources. If the queued processes are doing CPU-intensive activities (e.g., computational, sorting), the demand for CPU exceeds the supply and could be indicative of a CPU bottleneck. However, other situations, such as heavy disk I/O activity, could cause queuing for CPU resources. In the latter case, the real solution would be to address the disk I/O bottleneck. When the disk bottleneck is alleviated, queuing for CPU resources will go away.

TOP CPU USER, CPU Detail Screen (c)

To recoup the greatest gain, it is important to concentrate on the process(es) that are consuming the greatest amount of CPU. Go to the CPU Detail Screen (c), and look at the bottom of the display to find *Top CPU User*. Is it a user process, a system process or even a batch process? Is it a legitimate or *runaway* process (e.g., a shell script caught in an infinite loop)?

To get detailed information on that process, hit the *Select Process function key* and specify the PID number for the process that you wish to analyze. If several processes are dominating the CPU, be sure to investigate each of them. Here are some of the key metrics that you will want to examine:

- *CPU Usage*, total CPU time consumed by the process;
- *Priority*, the dispatch priority of a process; and,
- *Blocked On*, the reason a process has stopped executing.

If the process has been running for a very long time and is a mission-critical process, you may want to have a programmer examine its code for efficiency. Perhaps the code can be restructured so that it will use less resources (e.g., convert semaphores into spinlocks where appropriate). Do you set optimization levels when compiling your code? Often 20-25% performance improvement can be gained by merely recompiling the program at Level 1 optimization (+O1).Was the process running at higher than its scheduled priority? If so, investigate why.

The *Blocked On* reason will lend good clues to what is contributing to the bottleneck. For example, if a process is spending most of its time blocked on *PRI*, you can investigate probable causes relating to system calls. Hit the *Wait States function key* for more details on the amount of time being spent in various wait states for that particular process.

Some of the more significant *Blocked On* reasons include:

- *CACHE*, waiting for a CACHE buffer to become available;
- *DISK*, waiting in the disk device driver for an I/O operation to complete (does not include raw I/O);
- *IO*, waiting for I/O to local printers, instruments, tapes or non-file system (raw) disks to complete;
- *IPC*, waiting for an inter-process communication (remote) operation;
- *LAN*, waiting for a LAN operation to complete;
- *MBUF*, waiting for a memory buffer (for cluster traffic);
- *PRI*, waiting for a higher priority process to give up the CPU;
- *SYSTM*, waiting for system resources;
- *VM*, wating for a virtual memory operation to complete.

Investigate which files the process has open by hitting the *Open Files function key*. This may provide additional understanding into what the process is doing and how it is spending its time.

FIRST STEPS IN ALLEVIATING A CPU BOTTLENECK

Here's a quick checklist of ways to reduce CPU usage on your system:

- Identify and kill any runaway processes.
- Redistribute workloads by diverting some processing onto other systems with spare CPU cycles.
- Offload graphics, spreadsheets, etc. to personal computers or workstations.
- Recompile programs with higher optimization levels set. Optimized code usually runs faster than unoptimized code and uses less CPU.
- Identify any binaries on your system that were created using HP-UX 9.04 or earlier. Recompiling on 10.01 can yield an instant 5-25% payback. For even larger returns, recompile with higher optimization.
- Restrict batch processes to non-peak usage hours. Batch processes that must be run during production hours should be *NICE*d to run at lesser priority.
- Allocate CPU to various groups of users based on your business needs using the *Process Resource Manager*. Four scheduling queues are provided with the HP-UX operating system and additional functionality can be added on. There are also various vendors' products on the market which will enable you to customize CPU allocation.
- Eliminate all unnecessary process forks. When using process handling, recycle processes rather than spawning additional processes.
- Streamline applications to eliminate inefficiencies. This can include rewriting or replacing portions or even an entire application. Be sure to concentrate on programs, library routines or portions thereof that are run the most often for maximum payback.
- Use a sorting utility to presort large data files. For very small data files, an internal sort within the program would be more efficient.
- Optimize PATH environmental variables to reduce overhead when executing and accessing files.
- Continually educate the users of the system. Empower them to use the system and its resources more efficiently.

DISK BOTTLENECK INDICATORS.

Disk I/O access is one of the slowest components of system performance. Thus, we want to be sure to eliminate any unnecessary I/Os and balance the entire I/O workload across the disk hardware. For optimal system performance, I/Os must be balanced across file systems, disk spindles and disk controllers to reduce uneven queuing & delays.

When discussing disk bottlenecks, we refer only to physical I/Os. Physical disk I/Os are those that require some type of interaction with an external disk device. One physical I/O often consists of multiple logical I/Os. That is, several logical I/Os are completed in a single disk access which speeds up performance tremendously.

The objective when diagnosing disk bottlenecks is to isolate the source of the problem. GlancePlus/UX will enable you to trace the intense disk I/O to its origin. The *Disk I/O By File System Screen* (*i*) identifies the process that is initiating the highest I/O rate on the system. This may be a good place to start when sleuthing a disk bottleneck on your system.

DISK UTILIZATION, Disk Global Bar

The *Disk Global Bar* reveals the percentage of time that the busiest physical disk was being used during the past interval. This statistic is for local disks only and does not include NFS or DUX devices. Concentrate on the average disk utilization which reflects the average usage of the busiest disks from each interval. A rule of thumb is that 50% average measurement for peak disk utilization is an I/O bottleneck indicator. If that utilization average is greater than or equal to 70%, a severe I/O bottleneck is likely on your system.

BALANCED DISK UTILIZATION, Disk Queue Lengths By Device Screen (u)

There are two key metrics to check. First, check to see whether or not any particular disk is being utilized beyond the 50% level for a sustained period. Second, check to see that the utilization levels of the drives are fairly well-balanced. Let's say, for example, that there are three disks on your system. The disk utilization levels are 29%, 40% and 43%. In this case, disk utilization is fairly well distributed.

If I/O throughput is slow on your system, an unbalanced workload may indicate an opportunity for improving performance. Let's take another example where utilization levels are 15%, 65% and 25%. The disk load is not balanced. Redistributing the workload more evenly across the three disks may yield I/O performance gains, since some of the queuing on the disk with 65% utilization would be alleviated. However,

sometimes there is no choice regarding moving data around. For example, many databases access disks in raw mode, and balancing the load is very non-trivial.

DISK QUEUE LENGTH, Disk Queue Lengths By Device Screen (u)

Proceed to the *Disk Queue Lengths By Device Screen*. Check the disk utilization level for each individual drive. Identify all disks with a sustained utilization level of >= 50%. To the right you'll see a breakdown of how long the disk queues were when I/Os were being serviced. Look for longer queue lengths, such as 4 or greater, that are sustained.

After you have identified the heavily hit disks, proceed to the *Disk I/O File System Screen*. You will be able to identify file systems with the highest I/O usage. What disk drives are they located on? Are several of the heavily accessed file systems located on the same disk? You may want to consider redistributing files and/or workload to spread out the disk I/Os more evenly.

FIRST STEPS IN ALLEVIATING A DISK BOTTLENECK

Here's a checklist of various ways to reduce disk I/O bottlenecks on your system:

- Load balance by redistributing applications among different systems to achieve optimum I/O balance. You may also choose to offload read/write intensive applications, such as editing and graphics to PCs.
- File placement can help alleviate a disk bottleneck by load balancing across disks. Spread out frequently accessed files and swap areas over different file systems, spindles and controllers.
- Increase the size of the file system buffer cache if the I/O rate is high. This pertains to file system I/O only. Raw I/O does not use the buffer cache, so this will not help speed up most database I/O, for example.
- Use disk striping and *Logical Volume Manager* (LVM) to further spread out I/Os across multiple disks.
- Add more disk controllers on the system to reduce the chance of an I/O bottleneck occurring at the controller level. You can then spread out the existing disks so that there are fewer disks per controller.
- Add more disks. If you add more disks to the system, I/Os can be spread across multiple disks.
- Practice good housekeeping! Insure that sufficient freespace is available on all drives. Next, archive files that have not been accessed for a specific period of time,

and purge them off the system. Minimize unnecessary logons, file opens and file closes.

- Your MIS staff can decide to redesign the application(s) to eliminate unnecessary disk I/Os. File opens should provide a good clue to where excessive disk I/Os may be occuring. An example of a design change they may institute is to move a frequently used file, such as a config file, to shared memory to eliminate unnecessary file opens. The *Top Disk I/O User* is identified at the bottom of the *Disk I/O By File System Screen.* From there, proceed to the *Process Resource* display for that particular process and finally to the *Open Files* display for further clues.

MEMORY BOTTLENECK INDICATORS.

There are three types of memory: cache memory, random access memory (RAM) and swap. The fastest type is cache memory which is located on the same chip as the CPU processor. Information may be contained in cache memory, if it has been recently accessed or is contained on a page that was recently accessed.

RAM, commonly referred to as *memory*, is fast but not as instanteous as cache memory. Resources that a program needs to run must be contained in memory before the program may execute. To the extent possible, a program's pages are retained in memory to facilitate their reuse, should the application need them again. Much of the HP-UX kernel is contained in memory.

Swap space resides on disk which makes access time much slower. When a program executes, swap space is reserved on its behalf. A program usually uses only a portion of its reserved swap space depending largely upon what modules of the code are executed. The used portion is referred to as active swap.

When the demand for memory is high, memory management uses swap space as temporary storage for processes that are active but are not currently executing. During these peak periods, memory management has to swap out or deactivate/page out entire processes or parts of entire processes to the swap area on disk to make room in memory for code and data needed by the currently executing process.

The concept of *swapping* applies to HP-UX 9.X. In HP-UX 10.X swapping has been replaced by process *deactivation.* Instead of swapping an entire process to a swap area, it is marked as *deactivated* and may or may not end up being paged out to disk. The process may be reactivated before paging occurs. Either way, paging/swapping are expensive operations since they involve disk access.

When the requests for memory outstrip supply, memory pressure occurs on the system. Objects residing in main memory are swapped out/deactivated to make room for objects needed by the executing process. When your process again becomes active, its needed pages are swapped in/reactivated before the process can run. This excessive memory management overhead causes system performance to degrade.

Memory pressure may also cause CPU or disk I/O bottleneck symptoms. For example, if excessive paging/swapping is taking place, a large increase in system CPU utilization may result. Excessive swapping/paging means lots of disk writes/reads. You may see some disk bottleneck symptoms.

When the causes of the memory bottleneck are alleviated, the CPU and disk I/O symptoms will also be eliminated. Thus, you should check for and eliminate memory pressure first, the disk I/O bottleneck second and CPU bottleneck third. Practically speaking, however, CPU bottlenecks are the easiest to detect, so they are often identified and examined first.

MEMORY UTILIZATION, Memory Global Bar

The third global bar reveals the overall contents of main memory. Components of the *Memory Global Bar* show how much memory is being allocated for the system kernel, user code & data and the filesystem buffer cache.

Ideally, you will see lots of memory going to user processes. When more of memory goes to user processes, more work gets done on the system. The opposite is also true. A high level of memory utilization by system processes means that less system resources are available to user processes.

Memory utilization often shows up as quite high even though little work is being done on the system. Often you'll see memory utilization in the 60th and above percentiles. This is not a problem! Since ample memory is available, the system will continue to bring in needed code and data with minimal memory management overhead.

If memory utilization is in the 90th percentile or higher, look to see how much swapping and paging is occurring on the system. If paging and swapping levels are low, there is not a memory bottleneck. However, if you observe high paging/swapping levels, a memory bottleneck is probable.

SWAP UTILIZATION, Swap Global Bar

Swap space resides on disk. It is used to extend the logical boundaries of memory beyond the RAM's physical size. Memory management can move data from memory to the swap area on disk and later retrieve the data back into memory.

When a program executes, swap space is reserved on its behalf. The *Swap Utilization Global Bar* reveals on a system-wide level what percentage of swap space has been reserved and is, therefore, not available to other processes. If the system is out of swap space (i.e., all swap space is reserved), no new processes can be created on the system.

Since disk space is relatively inexpensive, a good rule of thumb is to configure your system's swap space generously.

PRIVATE RSS/TOTAL VSS, Global Summary Screen (g)

To understand why excessive memory is being consumed, you'll need to identify the process(es) that are using the most memory. Go to the *Global Summary Screen*, and look for processes that have large RSS (Resident Segment Size) and VSS (Virtual Segment Size) values. These processes are the ones to check first when sleuthing memory pressure. You can go into the *Select Process Screen* (s) to obtain detailed information regarding an individual process.

Virtual memory management allows parts of a user's program to be spread freely throughout the memory systems. Processes with large VSS values are consuming high amounts of virtual memory which includes swap space out on disk. Processes with large RSS values are consuming large amounts of RAM. To diagnose memory pressure on your system, investigate processes that are consuming large amounts of resident and/or virtual memory.

PAGE FAULTS, Memory Detail Screen (m)

A page fault occurs when a program tries to reference data or code (e.g., flat file or executable) that is not present in memory. The program suspends execution, and a physical I/O to disk must be done to retrieve the needed page(s) into memory. Page Fault statistics will lend clues as to how often needed code and data are absent from memory and therefore must be retrieved from disk.

CACHE HITS, Memory Detail Screen (m)

If lots of file system I/O is taking place on the system, a very high cache hit rate in the 90th percentile is good news. This indicates that needed code and data are present in the buffer cache. These transfers take place almost instantaneously since the buffer cache is located in immediate proximity of the CPU processor. In this case, a high cache hit rate is indicative of a good performing system.

When I/Os do not use the HP-UX file system, the cache hit rate is not a good performance barometer. For example, the cache hits metric does not apply for raw I/O database environments.

FIRST STEPS IN ALLEVIATING A MEMORY BOTTLENECK

- Weigh the cost of engineering hours for application redesign vs. the cost of purchasing additional memory for your system. The latter will sometimes win out as the more cost-effective solution.
- If the MIS staff agrees to redesign, rearchitect or fix memory hungry programs, they will be analyzing in terms of data locality, reorganizing/reloading shared libraries, identifying/alleviating memory leaks, etc. There are several good 3rd party tools available for detecting and eliminating memory leaks.
- You sometimes have the option of replacing older programs with newer, more memory-efficient applications.
- When swap utilization exceeds 70%, convert unused disk space into swap space.
- If your system's workload makes use of the buffer cache and the buffer cache hit read ratio decreases to below 90%, use SAM to increase the size of the buffer cache.

OTHER CONTENTION.

What if CPU, memory and disk look okay, yet system performance is still slow? Other factors on the system could be affecting performance. Factors such as lock contention, context switching, table overflows and networking could be slowing down the system.

Many of the contention issues you'll encounter originate from application design choices and are usually beyond the control of a system manager. Additional utilities may be necessary to identify and isolate the problem. Remedial actions may involve modifying or even replacing existing applications.

Networking is a very comprehensive topic and is beyond the scope of this paper. It will be only minimally addressed by directing you to look for high volumes of network activity.

System table size is sometimes a factor that could affect system throughput. For example, if a system table is underconfigured, a table could run out of entries. A process ready to run would not be able to execute. If a table is overconfigured, resources such as memory may be wasted.

SYSTEM TABLE UTILIZATION, System Table Utilization Screen (t)

Go to the *System Table Utilization Screen* (t) in Glance. Check that table's utilization %. If a system table is consistently over or under utilized, you may need to increase or decrease the table's size at your next convenience. Some tables will require a kernel regeneration and system reboot in order to increase their size. Full system tables can result in various consequences on the system. For example, new programs cannot execute if the *nproc* table is full. New files cannot be opened if the *nfile* table is full.

A good rule of thumb is to configure the tables large enough so you don't have to worry about approaching their configured values. This applies in most cases, but there are some exceptions.

Note that some HP-UX tables behave differently. Some tables, like the *Inode Cache*, cache unreferenced entries which makes them show up as nearly always being full. These tables are harder to size correctly, and it is best not to reduce their size. Also, when the system is memory bottlenecked, you may consider reducing table sizes to make the kernel smaller which keeps more memory available to user processes.

FIRST STEPS IN ALLEVIATING OTHER CONTENTION

Here are some approaches you can take to help reduce contention, locking and table overflows on your system:

- You can alleviate contention on your system via load balancing. Distribute problem applications evenly among several systems. That way system performance is effected less on several systems rather than to a large degree on one system.
- If table overflows occur on the system, increase the appropriate table sizes by using SAM.
- If you uncover issues with database locking, your database vendor can help. There are many database utilities available that can assist with monitoring and tuning databases.

- If heavy network activity is hurting the system's performance, add more networking processing power. For example, an additional LAN card can help to redistribute the workload. You can also load balance by moving applications to less used systems.

CONCLUSION

Rules of thumb are a great place to start, but they are not intended as absolute. With performance monitoring, just as with life, guidelines should be tempered by practical experience and common sense.

For those times when crisis management prevails, a performance monitoring tool is essential for immediate, reactive troubleshooting. The preferred mode for performance management, of course, is pro-active. With regular monitoring and planning, you will be using your performance utility to *maintain* control of your system rather than to *gain* control.

The *bottom line* is to actively and regularly manage your system to avert performance problems. By examining your computer system's resource usage levels, you will be able to pro-actively recognize and alleviate bottleneck symptoms before they become full-blown problems. Your reward will be longer lead-times for handling many performance issues.

QUICK REFERENCE CHECKLIST:

FIRST STEPS IN ALLEVIATING SYSTEM BOTTLENECKS

FIRST STEPS IN ALLEVIATING A CPU BOTTLENECK

- Identify and kill any runaway processes.
- Redistribute workloads by diverting some processing onto other systems with spare CPU cycles.
- Offload graphics, spreadsheets, etc. to personal computers or workstations.
- Recompile programs with higher optimization levels set. Optimized code usually runs faster than unoptimized code and uses less CPU.
- Identify any binaries on your system that were created using HP-UX 9.04 or earlier. Recompiling on 10.01 can yield an instant 5-25% payback. For even larger returns, recompile with higher optimization.
- Restrict batch processes to non-peak usage hours. Batch processes that must be run during production hours should be *NICE*d to run at lesser priority.

- Allocate CPU to various groups of users based on your business needs using the *Process Resource Manager*. Four scheduling queues are provided with the HP-UX operating system and additional functionality can be added on. There are also various vendors' products on the market which will enable you to customize CPU allocation.
- Eliminate all unnecessary process forks. When using process handling, recycle processes rather than spawning additional processes.
- Streamline applications to eliminate inefficiencies. This can include rewriting or replacing portions or even an entire application. Be sure to concentrate on programs, library routines or portions thereof that are run the most often for maximum payback.
- Use a sorting utility to presort large data files. For very small data files, an internal sort within the program would be more efficient.
- Optimize PATH environmental variables to reduce overhead when executing and accessing files.
- Continually educate the users of the system. Empower them to use the system and its resources more efficiently.

FIRST STEPS IN ALLEVIATING A DISK BOTTLENECK

- Load balance by redistributing applications among different systems to achieve optimum I/O balance. You may also choose to offload read/write intensive applications, such as editing and graphics to PCs.
- File placement can help alleviate a disk bottleneck by load balancing across disks. Spread out frequently accessed files and swap areas over different file systems, spindles and controllers.
- Increase the size of the file system buffer cache if the I/O rate is high. This pertains to file system I/O only. Raw I/O does not use the buffer cache, so this will not help speed up most database I/O, for example.
- Use disk striping and *Logical Volume Manager* (LVM) to further spread out I/Os across multiple disks.
- Add more disk controllers on the system to reduce the chance of an I/O bottleneck occurring at the controller level. You can then spread out the existing disks so that there are fewer disks per controller.
- Add more disks. If you add more disks to the system, I/Os can be spread across multiple disks.
- Practice good housekeeping! Insure that sufficient freespace is available on all drives. Next, archive files that have not been accessed for a specific period of time, and purge them off the system. Minimize unnecessary logons, file opens and file closes.
- Your MIS staff can decide to redesign the application(s) to eliminate unnecessary disk I/Os. File opens should provide a good clue to where excessive disk I/Os may

be occuring. An example of a design change they may institute is to move a frequently used file, such as a config file, to shared memory to eliminate unnecessary file opens. The *Top Disk I/O User* is identified at the bottom of the *Disk I/O By File System Screen*. From there, proceed to the *Process Resource* display for that particular process and finally to the *Open Files* display for further clues.

FIRST STEPS IN ALLEVIATING A MEMORY BOTTLENECK

- Weigh the cost of engineering hours for application redesign vs. the cost of purchasing additional memory for your system. The latter will sometimes win out as the more cost-effective solution.
- If the MIS staff agrees to redesign, rearchitect or fix memory hungry programs, they will be analyzing in terms of data locality, reorganizing/reloading shared libraries, identifying/alleviating memory leaks, etc. There are several good 3rd party tools available for detecting and eliminating memory leaks.
- You sometimes have the option of replacing older programs with newer, more memory-efficient applications.
- When swap utilization exceeds 70%, convert unused disk space into swap space.
- If your system's workload makes use of the buffer cache and the buffer cache hit read ratio decreases to below 90%, use SAM to increase the size of the buffer cache.

FIRST STEPS IN ALLEVIATING OTHER CONTENTION

- You can alleviate contention on your system via load balancing. Distribute problem applications evenly among several systems. That way system performance is effected less on several systems rather than to a large degree on one system.
- If table overflows occur on the system, increase the appropriate table sizes by using SAM.
- If you uncover issues with database locking, your database vendor can help. There are many database utilities available that can assist with monitoring and tuning databases.
- If heavy network activity is hurting the system's performance, add more networking processing power. For example, an additional LAN card can help to redistribute the workload. You can also load balance by moving applications to less used systems.