Experience contributes important knowledge in many procedures. From the experience gained in tuning the Multiple Virtual Storage (MVS) operating system, a set of guidelines to help MVS installations avoid performance problems are suggested, along with an approach to tuning an MVS system. The guidelines can help a performance analyst isolate the cause of a performance problem. Not all possible problems that might be encountered are included. It is found that most MVS performance problems are a result of poor workload management and are often related to I/O activities.

An MVS tuning approach

by R. M. Schardt

Effective tuning of the Multiple Virtual Storage (MVS) operating system (the later releases of OS/VS2)[1] requires knowledge of MVS operations, knowledge of the particular system to be tuned, and tuning experience. Earlier papers have dealt with different aspects of MVS tuning,[2,3] but this paper addresses the issue of experience by sharing with the reader the MVS tuning experience of IBM's Washington Systems Center staff. Some knowledge of MVS and its associated terminology is assumed.

In some respects this paper could be considered a primer for the performance analyst who has been trained on MVS but may not have had much "hands-on" experience. This information is also useful to managers responsible for system performance. Included are suggestions as to the kinds of activities that are needed to manage performance in a data processing installation. The paper addresses those situations that tend to be common or reoccurring performance issues.

Included in this paper are "rules of thumb," or guidelines, that can be used to help pinpoint the cause of actual or potential performance problems. The primary source for these guidelines is the experience gained from studies and from solving actual performance problems, and for that reason, the guidelines are subject to change with increasing experience. These guidelines should be used along with knowledge of other system factors to determine the probable source of a performance problem.

The rules of thumb are based on reasonable performance. In a given environment, a guideline may be violated, and yet the resulting performance is acceptable to a particular user in that environment. For example, we know from experience that TSO (Time Sharing Option) response will suffer once channel utilization gets above 35 percent. Therefore, the rule of thumb states that utilization of channels servicing TSO volumes should be kept below 35 percent. However, "reasonable" response for a specific system may be less critical, and a channel utilization of about 40 percent may be perfectly acceptable.

The real value of the guidelines presented in this paper is to assist in localizing the source of an actual or potential performance problem. Although the discussion is directed primarily toward MVS, many of the guidelines may apply to any operating system because they are related to I/O characteristics that are the same regardless of the software.

Categories of performance problems

The problems encountered in customer installations generally fall into three categories:

1. Performance management.
2. Saturated CPU, where the CPU is running at or near 100 percent, and all of the work is not being done.
3. Contention, where real storage is overcommitted and/or there is DASD (direct access storage device) contention at the channel, control unit, or spindle level.

Of the three types of problems, causes for the last two are usually easier to find because they are, for the most part, purely technical in nature. Also, the experience level is such that most problems encountered today are not unique. However, the ease of implementing a solution can vary substantially. The fix could be as simple as changing a system parameter or as involved as a hardware upgrade.

The more difficult of all situations to correct are those involving performance management. In many cases, these situations are in environments where no priorities are set for the work, no performance objectives exist, and no monitoring of the system is done. These conditions must be corrected before performance problems can be properly addressed.

Each of the problem areas is discussed below. The discussions include symptoms of the problem, along with some suggestions as to how the problem may be solved.

Performance management

Certain standards should be established to properly manage a data processing installation. In terms of system performance, several areas need to be addressed. First, performance objectives must be set for the various kinds of work the system will process. In addition to providing users of the system with realistic performance expectations, the objectives establish a goal for system tuning. Next, a workload priority must be established so that the system can determine which work to service first, especially during peak periods. Finally, the system must be measured to determine if the performance objectives are being met.

Experience clearly indicates that the root cause of most performance crises is an inadequate performance management system. Unfortunately, the need for a good performance management system is not always understood until it is too late. Furthermore, trying to define an adequate system under the pressure of a crisis situation seldom produces the desired results. Therefore, if an installation does not have documented performance objectives, has not set priorities for its workloads, and is not monitoring system performance, it is more often than not heading for a performance crisis.

The *MVS Performance Notebook*[4] contains useful information on defining and measuring performance objectives. One activity it mentions is setting work priorities. There is a troublesome area for many users. The difficulty seems to be an inability or an unwillingness to decide what work is most important. The results of this indecision are particularly troublesome in those systems that are running at or near capacity.

There have been extreme cases where all work (TSO and batch) is run in a single domain or multiple domains with no discrimination between them. In this situation, the system appears to perform properly as long as there are sufficient resources. However, as the workload increases and the system reaches its capacity, performance problems begin to appear.

Generally speaking, attempting to tune a system that has reached the limit of a resource, such as the CPU, is an exercise in trade-offs. For example, TSO can be made more responsive, but it will probably affect batch throughput. Before these trade-off decisions can be made, priorities must be set for the work.

performance monitoring

Typically, data processing installations perform two kinds of measurements usually referred to as "long-term" and "short-term." Long-term measurements are those associated with performance-monitoring functions. These measurements provide information about general system performance and workload trends and are run continuously.

Table 1 RMF parameters and data to be tracked†

Long-term measurements		*Short-term measurements*
RMF parameters	*Recommended data to track*	*RMF parameters*
CHAN	CPU utilization	CHAN
CPU	Channel utilization	CPU
CYCLE (1000)	DASD device utilization	CYCLE (250), (333) IF MVS/SE**
DEVICE (DASD)	(critical devices)	DEVICE (DASD)
ENQUEUE (Summary)	Workloads (by domain or	INTERVAL (30m)
INTERVAL (60M)	performance group)	PAGESP
PAGESP	Response time	PAGING
PAGING	Number of transactions	RECORD
RECORD	(transaction rate)	REPORT*
NOREPORT	Paging activity	NOSTOP
NOSTOP	Rate fault	TRACE*
NOTRACE	Demand	WKLD (Period)
WKLD (Period)	Swap	ENQUEUE (DETAIL)*
ARDJ (Major subsystems	VIO	
and components)	Page data sets	
	Address space data	
	CPU utilization	
	Storage	
	I/O activity	
	Page faults	

†*For more information on the function of these parameters, see Reference 5.*

*These parameters would be used according to the kind of data needed to solve a specific problem.

**Sixty samples (one sample per RMF cycle) of trace data are collected and displayed on a single line. The default cycle time is 250 milliseconds. For SU7 systems this default was fine since the SRM routine was invoked every 30 seconds. However, in MVS/SE the RMR routine is invoked every 20 seconds. Therefore, a cycle of (333) is more appropriate since this results in a line of trace data every 20 seconds.

Short-term measurements are associated with solving a specific performance problem or system tuning activity. These measurements produce more specific and detailed information. For example, suppose that the long-term measurements showed TSO response time was increasing and additional measurements were needed to help identify the source of the problem. The additional set would be referred to as short-term measurements.

The Resource Measurement Facility (RMF)[5] is flexible enough in its options that it can often be used both as a long- and short-term measurement tool. Table 1 contains suggestions for RMF parameters and the kind of data that should be tracked.

The major difference between the long- and short-term RMF parameters is the cycle time and interval. More frequent sampling and a shorter interval are required to capture more accurate data. A 30-minute interval for short-term measurements works well for most situations. It is short enough to show shifts in workload and long enough to smooth out spikes that could be misleading.

Obviously the long-term measurements with less frequent sampling and a 60-minute interval will put a smaller load on the system than the short-term measurements. For example, sample

measurements show that long-term measurements driven by the RMF parameters described here impacted an IBM 3032 CPU only about one to two percent. The impact of short-term measurements will be greater and will depend on how many variables are being traced.

Performance monitoring and tuning should not be considered as one-time activities, but rather as ongoing ones. Most systems are not static. Workloads are constantly changing, and new applications are being implemented. This kind of change cannot but help affect system performance. A good monitoring system should help to anticipate and avoid problems rather than react to them.

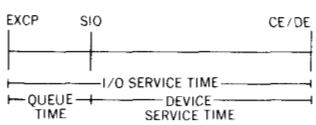
I/O contention

I/O contention, which in many cases is independent of the operating system in use, accounts for about 75 percent of the problems reported to the Washington Systems Center as poor MVS performance. Channel loading, control unit or device contention, data set placement, paging configurations, and shared DASD are often the major culprits.

service time

The key to finding and fixing I/O-related performance problems is service time. The length of time it takes to complete an I/O operation can have a dramatic effect on performance, particularly for on-line and interactive subsystems such as TSO, IMS (Information Management System), CICS (Customer Information Control System), etc. The following discussion addresses the various service time elements and the factors that may affect their duration.

Figure 1 Service time



For the purpose of this discussion, service time will be defined as shown in Figure 1. Actually, two service times will be referred to. One, I/O service time, is the elapsed time from the execution of the EXCP (Execute Channel Program) macro to the completion of the data transfer (Channel End/Device Interrupt). It includes any queue time plus the actual I/O operation. The other, device service time, is the elapsed time from the successful SIO (Start I/O) instruction to completion of the data transfer. It includes seek time and any rotation delays, plus data transfer. Device service time plus queue time equals I/O service time.

There are a number of factors that can affect service time. High utilization of a channel, control unit, and/or device can reduce the chances for a successful SIO instruction, thereby causing the request to be queued. High channel utilization, control unit contention, and poor device seek patterns can further increase service time once the I/O operation has begun. Each of these factors will be examined in detail, but first some general service time rules of thumb are presented.

Average queue length is reported by RMF. Generally, if this queue length is 0.1 or greater, further investigation is needed. High queue length values are a result of some condition that prevents the I/O operation from starting, thus causing the request to be queued. Probable causes for this condition are high device and/or channel utilization, control unit contention, etc.

Some components (e.g., the Auxiliary Storage Manager, or ASM, and the Telecommunications Access Method, or TCAM) keep track of the active I/O operations to the data sets they control. Thus, average queue length may not indicate all queuing delays since the queuing takes place within the component and not IOS (I/O supervisor). Generally, the average queue length reported by RMF for these components is zero. However, if significant average queue length is reported, path contention or other active data sets on the volume should be considered.

Approximate average queue time in milliseconds can be calculated by dividing "average queue length" by "device activity rate." Both values are reported by RMF.

Device service time for nonpaging volumes of IBM 3330 and 3350 storage devices should be somewhere between 25 and 40 milliseconds (paging volumes have a different set of guidelines and are discussed later). If these values are exceeded, control unit contention, channel utilization, and/or arm contention may be the cause. The version of RMF that supports MVS/SE (Multiple Virtual Storage/System Extensions) provides device service time. For other versions of RMF, device service time can be calculated by dividing "percent device busy" by "device activity rate."

Having established goals for queue time and device service time, let us examine some of the conditions that can cause these guidelines to be violated.

channel utilization

Generally, DASD channel utilization should not exceed 35 percent. However, this rule can be refined based on the subsystems that are active. For example, in some cases it appears that IMS begins to suffer measurable response degradation whenever its channels exceed 30 percent utilization. TSO response, however, appears adequate until 35 percent channel utilization is exceeded.

Response for batch is not as critical as for interactive and on-line subsystems. Therefore, higher channel utilizations are usually more tolerable. Channel utilizations of 40 percent or more are not necessarily bad.

Channel utilization affects I/O responsiveness. Higher channel utilization increases the length of time it takes to complete an I/O operation (i.e., reduces responsiveness). Each subsystem has its

own response requirements. On-line subsystems need to be highly responsive, whereas batch subsystems have less critical requirements. This difference suggests that in order to meet various response requirements, the I/O operation must be configured so that it can be tuned for the unique requirements of a given subsystem. It means that the I/O required for a given subsystem should be isolated to its own channels, control units, and devices. Recommendations on how this might be accomplished are presented later with the discussion on shared DASD. It also suggests that the concept of balancing channel utilization is not valid. In most environments, channel utilization should not be the same for on-line use as it is for batch, and the system should be tuned to the individual requirements of each one.

Before concluding this discussion of channels, it is appropriate for us to address channel service time. Channel service time, as reported by RMF, is essentially the time required for data transfer. It is calculated by dividing the "percent channel busy" by "activity per second."

Because average channel service time is a reflection of block size, this number could be used as a gross indicator of the blocking factor of an installation. For example, a channel servicing only IBM 3300 storage devices with an average block size of 6K would probably show an average channel service time of eight to nine milliseconds. For IBM 3350 devices, the service time would be in the five to six millisecond range. Whether these numbers are good or bad depends upon the kind of data being transferred. Generally, higher block sizes and, therefore, high channel service time are good for sequential data, but are not necessarily good for randomly accessed data.

Randomly accessed data, such as in data bases, usually have smaller block sizes than sequential data. Mixing the two kinds of data on a single channel can affect data base response. For example, a high-activity, full-track, blocked program library on the same channel with data base volumes could interfere with data base I/O requests. An indication of this situation would be high channel service time on data base channels.

This discussion of channel service time assumes that either SAM-E (Selectable Unit 9, or SU9) is installed or OPTCD=Z (Option Code) has been specified for the access methods, BSAM (Basic Sequential Access Method) and QSAM (Queued Sequential Access Method). If neither is true, then the channel service time could indicate twice the actual block size since "search previous" would be used rather than "search direct."

device utilization

The traditional rules of thumb for device utilization have been similar to those for channels (the 35 percent rule). However, high

Table 2 Determination of condition codes

SIO instruction condition code	*CSW*	*UCB busy*	*UCB control unit busy*	*RMF*
0	0	1	0	Device busy
1	Busy & SM	0	1	Control unit
1	Busy & no SM	1	1	Reserve delay

device utilization, in and of itself, is not necessarily a problem. It can be a problem when it causes excessive queue lengths, particularly for response-oriented subsystems like IMS and TSO. Therefore, the measure of device "goodness" or "badness" is not utilization but queue time plus device service time, which together equal I/O service time. High device utilization can be one of the causes for excessive queue time. The higher the device utilization, the less likely the SIO instruction will be successful and, therefore, the request queued. High device utilization does not necessarily imply high device service time. In fact, it is not uncommon to find a DASD device with high utilization but reasonable device service time.

control unit contention

DASD control unit contention, like channel utilization, affects both queue time and device service time. It can cause busy conditions that delay a successful SIO instruction (condition code 0) and delay reconnects once the seek and set sector operations are complete.

One way to determine the level of contention within a control unit is to do an analysis of condition codes resulting from an SIO instruction. If the number of condition codes of 1 is greater than 20 to 30 percent of the total number of SIO instructions, control unit contention may be too high.

Because condition code 1 is an indication of contention, a review of the circumstances under which it is set is appropriate. Table 2 summarizes the possible conditions under which condition codes 0 and 1 are set. Also shown are the contents of the CSW (Channel Status Word), which bits in the UCB (Unit Control Block) are set, and what conditions are reported by RMF.

There are essentially two conditions that can cause a condition code 1 on an SIO instruction: control unit busy, which is indicated by the busy and status modifier bits in the CSW, and device, or "head of string" busy, which is indicated by the busy bit and no status modifier in the CSW. Let us examine control unit busy conditions first.

The control unit is busy during data transfer time. While data are being transferred, the device, control unit, and channel are essentially tied together for the duration of the data transfer. With only one channel path to the control unit, the probability of encountering a control unit busy condition is small unless there is a high level of Format Write activity to IBM 3330 Model 1 drives. (The Model 1 drive does not release from the control unit while erasing the remainder of the track. Therefore, the control unit and device are busy during this time but the channel is not.)

An excessive number of control unit busy conditions is usually caused by one of two conditions: too many devices on a single control unit or one dominant volume. The latter condition is relatively easy to spot because a volume that monopolizes a control unit usually has a higher number of SIO instructions with few condition codes of 1 while the other devices on that control unit have a relatively high percentage of condition codes of 1. What needs to be done in this situation is to determine why the device is dominating the control unit. Perhaps, if multiple data sets are involved, some could be moved to devices on other, less active control units.

Condition codes of 1 that result from a device busy condition usually indicate some level of shared DASD contention. A device busy condition means that an SIO instruction was issued and the channel and control unit were available, but the device was busy, or, when string switching is installed, the "head of string" was busy. For more detailed information on the causes of a device busy condition, see Reference 6.

To summarize, condition codes of 1 are caused by two situations: control unit busy and device busy. When the number of condition codes of 1 becomes too high (20 to 30 percent of total SIO instructions), remedial action is usually necessary.

arm contention

Arm contention affects device service time and is caused by two or more active data sets on the same volume. Delays result because the arm is moved from one data set to the next. Generally, any nonpaging volume (paging configurations are discussed later) with an average seek distance of more than 40 to 50 cylinders should be examined.

In any case, excessive arm movement indicates that some sort of data set placement activity is needed. It may require moving high-activity data sets to low-use volumes or reallocating data sets closer to one another to reduce arm movement.

Another condition that can cause excessive device service times is having volumes with large volume tables of contents (VTOCs) that are researched frequently. Generally, this type of volume has

many small data sets. It may require moving some data sets to a different volume to reduce VTOC search time. The performance exposure here is not necessarily to the jobs accessing data on this volume but to other more critical applications that cannot access their data because the channel and control unit are busy doing a VTOC search.

A similar condition can occur with large program libraries and their directory searches. Such situations may require splitting the library across several volumes.

Paging configuration tuning

A properly configured paging subsystem is critical to MVS performance. Because paging is a global service, it can affect the performance of everything running in the system. Therefore, special attention must be paid to its specification.

One very critical key to system performance is ensuring that the right number of address spaces are in storage. Too many can drive the paging rate to excessive levels. On-line and interactive subsystems, the most critical in the system, are usually the first to feel the impact of an excessive paging rate. Thus, the goal of the system tuner should be to ensure that the paging rate is kept to an acceptable level.

The multiprogramming level (MPL) adjustments that MVS provides act as the mechanism used to keep paging reasonable. In essence, the procedure for tuning a paging configuration is a two-step process. First, control the MPL to ensure that paging does not exceed acceptable levels. Second, define a paging configuration to handle the paging rate dictated by the MPL.

Let us discuss controlling the MPL first. Because there are some differences between the ways in which an MVS/SE and a Selectable Unit 7 (SU7) (non-MVS/SE) system control MPLs, each system will be addressed separately.

SU7 system

Experience clearly indicates one of the better ways to control MPL adjustment is via page fault rate. However, the page fault rate (page transfer rate) constants are essentially disabled in the MPL decision algorithm of the System Resources Manager (SRM).[7] Therefore, it is necessary to modify the SRM constants to activate the page transfer rate and deactivate the current controlling constants: ASM queue, UIC (Unreferenced Interval Counter), and CPU.

One procedure is as follows:

1. Disable the ASM queue, UIC, and CPU constants by setting the SRM constants to the values shown below:

	High	*Low*
ASM queue	100	100
UIC	0	0
CPU	100	100

2. Set the SRM page fault rate constants (page transfer rate, PTR) as follows:

Processor	*Page transfer rate high*	*Page transfer rate low*
3031/158	25	20
3032/168	35	30
3033/168MP	55	50

These are initial values and may need adjustment depending upon the performance objectives of the system. Most installations determine their final page transfer rate values by measuring page fault rate (i.e., operations such as NON-SWAP, NON-VIO PAGE INS plus RECLAIMS) against on-line or interactive responses. Generally, there is a correlation between page fault rates and response time. As the page fault rate increases, so does the response time. The point at which the response exceeds the performance objectives defines the page transfer rate high value. The low value should be 5 to 10 below the high.

Another way to determine proper paging levels is to measure the page faults per second that a subsystem is encountering. For example, IMS response may begin to degrade when the IMS control region encounters more than two to three page faults per second or a message region exceeds three to four page faults per second. JES3 (Job Entry Subsystem 3) appears to degrade when it incurs more than eight to ten page faults per second. These values assume that there is a properly tuned paging configuration.

3. Monitor utilization. Paging channel utilization should be kept below 35 percent, and control unit contention should be kept to a minimum. Device service time for IBM 3330 and 3350 paging devices should fall into a range of 50 to 70 milliseconds. This is because ASM builds channel programs to achieve a 50-millisecond service time. In systems where paging is light, the service time for the local, Pageable Link Pack Area (PLPA), and common data sets may be less than 50 milliseconds. However, because requests to the swap data set involve the transfer of five to seven pages, a service time of less than 50 milliseconds is unlikely.

Because ASM does have its own internal queuing, queue time for paging requests cannot be determined. In this situation, device utilization can be used as an indicator of queue depth. Generally, paging volumes should be added when the page device utilization exceeds 35 percent. Also, the following should be considered when configuring page data sets:

a. Paging volumes should be dedicated, that is, no other active data sets on the same volume.
b. Use IBM 2305 fixed head storage devices for local and PLPA data sets. Heavily loaded environments with subsystems such as IMS, VTAM, JES3, etc., that make extensive use of a common storage area (CSA) might also consider the common area on such a storage device.
c. Swap data sets are generally less sensitive to arm movement because the arm is required to move a maximum of one time to access a swap set. Therefore, the IBM 3330 and 3350 devices perform very well as swap devices.

MVS/SE system

A slightly different approach may have to be taken to control MPL when running under MVS/SE because changes have been made to the decision algorithm. It is possible the default values of the MVS/SE algorithm may allow a demand paging rate higher than was acceptable under the SU7 system. If this is the case, there are at least two possible approaches. The first is to set the page transfer rate values in the MVS/SE system to the same values used in the SU7 system. The second approach is to change the demand paging rate and/or the milliseconds-per-page (MSPP) in the MVS/SE algorithm until the proper demand paging rate is achieved. More detailed information on how these constants affect MPL adjustment can be found in Reference 6.

Problem analysis approach

Having discussed the various rules of thumb associated with DASD tuning, we now briefly discuss how one might approach solving a DASD performance problem. Our goal is to determine I/O service times for the DASD devices. To that end, the following procedure is offered as one approach that has been used successfully:

1. Start with the RMF Device Activity Report. Scan the data and make note of all the devices that violate the rule of thumb for average queue time and device service time.
2. For each device noted in Step 1, calculate I/O service time (average queue time plus device service time).
3. Order the devices by I/O service time (highest first). Devices with the highest I/O service time probably require attention first.

Table 3 RMF measured device I/O activity

CUADDR	*BUSY* (percent)	*IORATE* (I/O per second)	*AVQUEUELEN*	*AVGDEVSERV* (milliseconds)	*AVGQUETIME* (milliseconds)	*AVGIOSERV* (milliseconds)
130	1.70	0.174	0.0003	98.2	1.6	99.9
122	0.72	0.082	0.0010	87.8	11.9	99.7
10F	1.09	0.116	0.0002	94.1	2.1	96.2
111	3.20	0.374	0.0019	85.7	5.0	90.7
134	0.27	0.033	0.0000	82.4	1.3	83.7
104	0.72	0.103	0.0010	69.5	9.6	79.1
120	0.62	0.089	0.0001	70.1	0.8	70.9
271	19.54	3.820	0.0613	51.2	16.1	67.2
275	9.98	1.982	0.0320	50.4	16.1	66.5
13D	0.34	0.057	0.0001	60.3	1.4	61.7
27F	0.88	0.153	0.0002	57.1	1.6	58.7
250	2.92	0.531	0.0011	54.9	2.1	57.0
252	7.99	1.776	0.0210	45.0	11.8	56.9
126	1.55	0.287	0.0006	53.8	2.0	55.8
272	14.82	3.078	0.0208	48.1	6.8	54.9
132	0.70	0.134	0.0001	52.5	1.1	53.5
102	0.58	0.115	0.0002	50.8	1.3	52.1
151	15.59	3.429	0.0186	45.5	5.4	50.9

4. Take additional measurements if necessary to further isolate the cause of the problem. Where control unit contention is suspected, the Generalized Trace Facility Performance Analysis Reporting System (GTFPARS)[8] is recommended.

The Service Level Reporter Program Product[9] can be a useful tool for analyzing I/O data. This program product provides the capability of manipulating SMF and RMF data and then producing user-defined reports.

Table 3 is an example of a report in which average queue time (AVGQUETIME), device service time (AVGDEVSERV), and I/O service time (AVGIOSERV) were calculated from RMF data and a report printed in I/O service time sequence. This report was produced by Service Level Reporter, Release 1 Modification Level 2, installed at the IBM Washington Systems Center.

A tool such as the Service Level Reporter can be extremely useful in assisting the system analyst in very quickly identifying trouble spots in an I/O configuration. For more information on how to approach solving DASD performance problems see Reference 6.

Other I/O-related topics

over-initiation

Over-initiation has a negative effect on performance. The impact comes from the CPU cycles required to manage additional address spaces, greater demand on the spool and work packs, and a greater demand for real storage. The result is higher utilization for spool, paging, and work packs along with higher utilization for the channels that service them.

The solution to this problem is not to over-initiate. However, this is not always possible. Take, for example, the RJE (remote job entry) system in which each RJE user insists on his own initiator. Therefore, the next best approach is to reduce the impact of over-initiation.

As it turns out, the cost, in CPU cycles, of managing the additional address spaces is not as great as expected. A recent study on a System/370 Model 168 showed the cost of over-initiating by six initiators was about three percent. The real cost in over-initiating comes from the result of the greater demands placed on the spool and work packs and the paging subsystem. If this could be controlled, the impact of managing the additional address spaces is acceptable.

The solution to the over-initiation problem is controlling the MPL level. Such control can be done by tuning the paging configuration. Even though extra initiators have been started, a tuned paging configuration, as described earlier, will keep the MPL at a proper level, thus allowing only the appropriate number of address spaces in storage. The paging, spool, and work packs should then be defined in sufficient quantity to handle that level of work, using the previously stated guidelines. In this situation, exchange swapping should be monitored. If the number of exchange swaps appears excessive, it may be necessary to increase the Interval Service Value (ISV) in the Installation Performance Specifications (IPS).

shared DASD

Shared DASD is a vital functional facility. However, its use must be carefully examined because of the potential impact on performance. The implications of shared DASD are discussed in Reference 10.

The problem of managing a shared DASD complex has perplexed many users. One solution that several installations have adopted appears to be working well. The basic concept is to isolate subsystems to channels, control units, and devices. For example, one installation assigned I/O units to each subsystem (i.e., TSO, IMS, batch, etc.). Each I/O unit consists of two IBM 3830 control units and two strings of string-switched spindles. Any given subsystem may require one or more "I/O units," depending upon

subsystem performance requirements, number of DASD spindles, I/O rate, etc. Only those I/O units that support subsystems that run on multiple processors are shared. The remaining I/O units are isolated to individual processors.

The advantages of this technique are significant. First, path delays can be tuned to the specific requirements of a given subsystem. For example, channel utilization for IMS data bases should be kept lower than channel utilization for batch jobs.

Second, I/O performance at the subsystem level can be measured. Also, when required, corrective action can be taken at the subsystem level without affecting the other subsystems.

Some examples of the subsystems that could be isolated are: (a) system packs—SYSRES, paging, etc., (b) batch—work packs, (c) IMS—data base packs, and (d) TSO—TSO spindles. System and work packs should never be shared. Other subsystem packs may have to be shared if those subsystems are run on multiple systems.

I/O block size

An evaluation of the block sizes of an installation is unquestionably the first thing to do when trying to reduce CPU utilization. No other activity has anywhere near the potential for saving CPU cycles. To illustrate that point, the data in Table 4 show the effect on a file maintenance program when going from a 200-byte to 6K block size. Four QSAM files were involved for a total of about 100 000 EXCPs at the 200-byte block size (number of buffers equals 5 in all cases). In this example, increasing the block size from 200 bytes to 6K reduced CPU utilization by more than 87 percent. In addition, elapsed time dropped by more than 90 percent, and channel time dropped by 65 percent.

The preparation and execution of a plan to evaluate and make changes to the data sets of an installation will take time and effort to complete. But the potential for reducing CPU cycles is substantial. There are, however, some relatively easy things that can be done initially. VIO (virtual input/output) for temporary data sets, whose block sizes are less than 2K, can usually provide savings in CPU cycles. Also, an examination of procedure libraries often turns up data sets whose block sizes are less than optimum.

As far as recommended block sizes are concerned, generally no block size for either tape or disk should be less than 4K. A more practical block size for disk is 6K, primarily because it provides compatibility between IBM 3330 and 3350 type devices. However, full track blocking for program libraries is still recommended. For

Table 4 Block size effect on a file maintenance program

Block size	200	1,000	2,000	4,000	6,000
EXCPs	105,000	21,000	10,500	5,250	3,500
SIO instructions	51,319	10,468	5,277	2,728	1,800
Elapsed seconds	600	150	103	71	56
CPU seconds	73.6	21.6	14.3	10.6	9.3
Channel seconds	95.2	46.9	38.4	36.5	33.3

tapes, block sizes of 8K or 12K are quite reasonable. For some applications, when 6250 bits-per-inch are available, 32K block sizes may be appropriate.

A common concern to increasing block size is that it will increase the paging rate to the point where performance will be adversely affected. If the paging configuration is assumed to be properly tuned, the evidence clearly indicates increased block sizes have a positive effect on system performance. For example, one test case in which multiple copies of a job with high I/O activity were run and block sizes increased from 2K to 12K (full track) showed that although paging increased 43 percent (24.6 pages per second to 35.2 pages per second), elapsed time dropped by 9 percent, and CPU seconds dropped almost 12 percent.

A recent survey of 265 614 user data sets representing 47 MVS installations showed the following:

- 65 to 70 percent of the data sets were sequential.
- The distribution of sequential block size was: 85 percent of sequential data blocks are 4K or less, 70 percent are 2K or less, and 40 percent are 500 bytes or less.

It would appear from this study that there are quite a number of data sets that could be reblocked for better performance.

Storage contention

One characteristic of a system with a storage contention problem is the inability to fully utilize the processor. In some cases it may not be possible to get CPU utilization above 60 percent.

The basic solution to a storage-constrained system is more real storage. If you have a four-megabyte IMS system and only three megabytes of storage to run it, no amount of parameter adjusting, System Resource Monitor modifications, or system zapping will make it run well. What will make it run well is four megabytes of storage, assuming the buffers have been tuned for system components such as TCAM, VTAM, VSAM, IMS, etc. However, in many cases, where multiple subsystems are involved, the system can

be tuned to run optimally with the storage it has by tuning the paging configuration as discussed earlier.

When tuning a system with insufficient real storage, a series of trade-offs must usually be dealt with. For example, it may be necessary to reduce the number of batch initiators in order to achieve reasonable TSO or IMS response. Therefore, it is extremely important that the issue of workload priorities has been addressed and resolved.

Reference 11 describes the relationship between real storage and system capacity. Included are measurements from various user environments showing the effect on performance of adding more real storage. One example of two MVS users increasing their five-megabyte System/370 Model 168 to six megabytes showed the following results:

- Paging decreased 43 percent (from 43.3 to 24.8 pages per second).
- Batch throughput increased 25 percent (from 53 to 66.5 jobs per hour).
- Average TSO response decreased 33 percent (from 4.4 to 2.9 seconds).
- Average IMS response decreased 20 percent (from 5 to 4 seconds).

For more information on how additional storage can affect system capacity, see Reference 11.

Conclusion

This paper has highlighted some of the MVS tuning experience of the IBM Washington Systems Center. Two points can be concluded from that experience. First, the predominant reason an installation finds itself in the midst of a performance crisis is because its performance management system is inadequate. Many performance problems can be avoided if objectives are set and an effective monitoring system is in place.

Second, I/O contention of some sort is usually the cause of MVS performance problems. Approximately three-quarters of the problems the Washington Systems Center has been involved in were caused by I/O bottlenecks.

This experience strongly suggests that many installations should devote more resources to analyzing, measuring, and tuning their I/O configurations. It would appear that in the quest for more processing power, sufficient attention has not been given to I/O requirements. Performance is a function of three basic elements: processor, storage, and I/O operations. All three must be exam-

ined, tracked, and modified if the system is to provide maximum efficiency in today's changing environment. A key to the solution is a performance management system where objectives are set, performance monitored, and changes made before a crisis arises.

ACKNOWLEDGMENTS

The author is grateful to all of the Large Systems Support staff of the IBM Washington Systems Center, namely, Don Boos, Dick Clark, Brian Kasch, and Jim Tavenner, who helped in the review and critique of this paper. Their help was invaluable.

Special thanks goes to Ron Clark, who provided most of the MVS/SE information, Siebo Friesenborg, and Kurt Ziegler, who provided much advice and counsel in the DASD tuning area, and Jerry Sly for his RMF expertise. Their contributions are greatly appreciated.

CITED REFERENCES AND NOTES

1. *OS/VS2 System Logic Library*, Volumes 1-7, SY28-0713 to SY28-0720, IBM Corporation; available through the local IBM branch office.
2. T. Beretvas, "Performance tuning in OS/VS2 MVS," *IBM Systems Journal* **17**, No. 3, 290-313 (1978).
3. W. W. Chiu and W. Chow, "A performance model of MVS," *IBM Systems Journal* **17**, No. 4, 444-462 (1978).
4. *MVS Performance Notebook*, GC28-0886, IBM Corporation; available through the local IBM branch office.
5. *OS/VS MVS Resource Measurement Facility (RMF) Reference and User's Guide*, SC28-0922, IBM Corporation; available through the local IBM branch office.
6. R. M. Schardt, *An MVS Tuning Perspective*, Technical Bulletin, GG22-9023, IBM Corporation (August 1978); available through the local IBM branch office.
7. H. W. Lynch and J. B. Page, "The OS/VS2 Release 2 System Resources Manager," *IBM Systems Journal* **13**, No. 4, 274-291 (1974).
8. GTFPARS, a Generalized Trace Facility reduction program, is available as IBM Program Number 5798-CQQ through the local IBM branch office.
9. The Service Level Reporter Program Product is available as SLR Program Product 5740-DC3 through the local IBM branch office.
10. K. Ziegler, Jr., *DASD Configuration and Sharing Considerations*, Technical Bulletin, GG22-9052, IBM Corporation (August 1978); available through the local IBM branch office.
11. C. C. Burns, *Memory Can Increase Your Capacity*, Technical Bulletin, GG22-9053, IBM Corporation (July 1978); available through the local IBM branch office.

GENERAL REFERENCES

Initialization and Tuning Guide, GC28-0681, IBM Corporation; available through the local IBM branch office.

R. M. Armstrong, *IBM 3031, 3032, 3033 Processor Complex Channel Configuration Guidelines*, Technical Bulletin, GG22-9020, IBM Corporation (September 1978); available through the local IBM branch office.

The author is located at the IBM Washington Systems Center, 18100 Frederick Pike, Bldg. 2, Gaithersburg, MD 20760.

Reprint Order No. G321-5118.