SHARE

SHARE SESSION REPORT

61	B466	DASD Management & Measurement in TSO	350
SHARE NO.	SESSION NO.	SESSION TITLE	ATTENDANCE
MVSI		Sam Lepore	WFB
PROJECT		SESSION CHAIRMAN	INST. CODE

Wells Fargo Bank, 525 Market St., #8513, San Francisco, CA 94163, 415-396-4111

SESSION CHAIRMAN'S COMPANY, ADDRESS, AND PHONE NUMBER

DASD Management and Measurement in the TSO Environment

Bill Fairchild

National Education Association
1201 Sixteenth Street, NW
Room 701E
Washington, DC 20036

Installation Code: NEA

MVS Interactive Facilities Project (TSO)

Session Number B466

DASD Management and Measurement in the TSO Environment

Introduction

The two major areas in which problems can occur in the management of direct-access storage devices (DASD) are capacity and performance. This paper discusses some common types of problems that occur in both of these areas as well as some solutions to these problems. The solutions concentrate on available user education and software packages that address these problems.

Capacity Problems

Why do capacity problems exist in the first place? Frequently they are caused by someone trying to make infinite use of a finite resource. This is equivalent to trying to put ten pounds into a five-pound bag. Eventually someone tries to allocate a new data set and cannot because all disks are full. As we will see, the 5-pound bag will not even hold five pounds.

Capacity problems can be caused by a variety of conditions. Some of them are:

- temporary data sets
- data set over-allocation
- needless production files
- inactive data sets
- repeated data set names
- PDS gas
- VTOC size
- VSAM files
- volume fragmentation
- CKD architecture
- block size
- data compression
- allocation unit size

Temporary Data Sets

Temporary data sets are often overlooked as a possible source of capacity problems. Often a temporary data set is left behind if the system crashes. Temporary data sets are removed when a job ends, but if a job doesn't end, these data sets may remain on the system. The system often allocates a temporary data set that becomes a permanent data set if the system crashes before the job ends.

Data Set Over-Allocation

Another problem that reduces the available DASD space is over-allocation of a data set. The design of a new system should allow for a certain amount of growth in the files. Over-allocation of data sets prevents B37 ABENDs when the data sets do grow. An example of an extreme case of being over-allocated is a completely empty file; the file is allocated but is never opened for output.

Needless Production Files

Occasionally a job stream creates a permanent file that should be deleted by a subsequent job stream, but for some reason the file is left on the disk. At one installation, for example, job streams created intermediate files to be used if a job needed to be restarted. Most of these files were never needed, yet they were never deleted and occupied valuable space.

Inactive Data Sets

Inactive data sets can also be an important factor in wasted DASD space. They might have been heavily used a year ago but now are not actively used. To detect this type of problem, look at the date on which the data set was last used (now being maintained by Selectable Unit 60), and either delete any unnecessary data sets or move them to tape.

Repeated Data Set Names

This problem occurs when DASD files are allocated with JCL such as UNIT=SYSDA. This can cause the same data set name to appear on many different volumes; however, only one data set name is ever cataloged. It is difficult to determine which is the correct version of this file.

PDS Gas

Wasted space can occur inside a PDS. For example, as maintenance is performed on a PDS, a member might be replaced. The old member stays

where it is and the new member is written at the end of the file. The space that was taken up by the old version of the member is never re-used until the PDS is compressed. If many such replacements are made without compressing the PDS, wasted space within the PDS will increase. This may also increase the time needed to find the member.

VTOC Size

A VTOC itself can cause capacity problems. For example, the VTOC might be too small. In this case, all the space on the volume cannot be allocated because each new data set requires a minimum of one DSCB. Hundreds of cylinders may be available on a volume but if the VTOC is too small, DSCBs may not be available to allocate that space. A VTOC can also be too big. A VTOC with too many DSCBs means that the VTOC can never be completely used, which wastes space. In addition, a VTOC must be an unblocked data set with keys. This can cause performance problems, and is discussed later in this paper.

VSAM Files

VSAM files can be over-allocated, have unusable space, be left behind in production job streams, and can be inactive. VSAM is basically a black box. It is very difficult to find out how much wasted space these files contain.

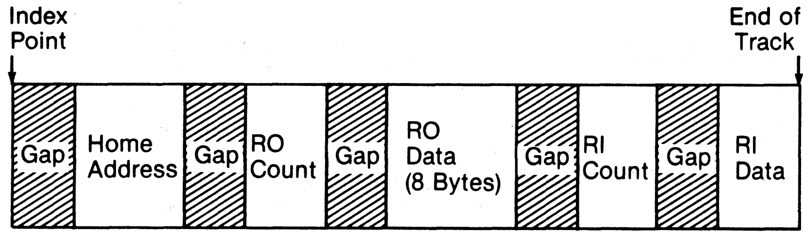
Volume Fragmentation

A volume can have enough total free space to service a request but still be unable to do so because the space is in small fragments. For example, if an operation requires ten contiguous cylinders, and the volume has five in one place, four in a second, and two in a third, the operation cannot be done. There is enough free space, but it is not contiguous. The operation must wait.

CKD Architecture

DASD architecture is itself the cause of many capacity and performance problems. The count, key, and data (CKD) architecture was designed twenty years ago, but today it wastes too much space.

Even a DASD track that is full has many gaps in it (see Figure 1). Each record is followed by a gap, and increasing the number of records increases the number of gaps. If the block size were smaller, more records could be stored on the track, but this would also increase the number of inter-record gaps.



- Known Uses of RO (Track Descriptor Record)
  - Count Field for Defective/Alternate
  - Data Field for BDAM Track Balance

Figure 1. CKD Track Architecture

There are several problems with this architecture:

- Inter-record gaps
- Record 0
- End-of-file record
- Keys

This architecture requires an inter-record gap between each pair of records and even between the various pieces of an individual record. This means much of the track is wasted.

What is Record 0 used for? It can be used to record a defective and an alternate track pair. The count field of Record 0 on the alternate track points back to the defective track (for which it is an alternate). Record 0 can also contain the track balance. (BDAM files only. For other types of access methods such as sequential and PDS, the track balance is redundant.)

Is Record 0 needed? It maintains compatibility with the past, but it serves no useful purpose in most modern cases. IBM is providing skip displacements on its newest devices, which will dispense with the defective and alternate track linkage. The purpose of skip displacements is to do away with the serious performance problem of moving the access arm away from a defective track to its alternate to read the data on that alternate, and then dragging the arm back to the next track after the defective one. On the other hand, Record 0 uses only about one percent of the track, so removing it saves little space. It is also possible to write data in Record 0, but the access methods discourage this practice.

A third requirement of this architecture is an end-of-file record. It is redundant in a sequential file because the end-of-file pointer is also kept in the Format 1 DSCB. There is a software pointer to end-of-file and a hardware end-of-file written after the last block of data. This seems necessary in the case of a PDS to know where a member ends, but the end of a member could also be stored in the directory. An EOF record has the smallest possible block size (zero), and this makes the least efficient use of the track.

The key also causes problems. Files are not typically created with keys, but twenty years ago IBM designed many critical system files this way. For example, the VTOC is an unblocked file that has a key length of 44 and a data length of 96. The SYSCTLG data set contains an 8-byte key and a 256-byte data field. PDS directory blocks also have an 8-byte key and a 256-byte data field. ISAM files all have keys. Any other type of user file could conceivably have keys as well. Keys take up a great deal of space on a track and, depending on the channel program that is generated to process this file, may cause serious performance problems when being searched. This paper discusses key searches in greater detail in the performance section.

Block Size

The block size, or length of a data record written on a track, can be inappropriate. The optimum size varies from device to device, so what may be an efficient size on one device may not be on another. This can cause problems; for example: when programs, JCL, and systems have been designed around a particular device and its optimum block size. If that device is replaced with one having a different optimum block size, the once efficient system may now waste a great deal of track space.

Figure 2 shows the percentage of a 3350 track that is used depending on how many blocks are written on the track.

The graph illustrates the percentage of the track used by the following special data sets which have fairly small block sizes:

Data Set	Percent Utilized
VTOC	50%
PDS Directory	65%
SYSCTLG	65%
VSAM Catalog	75%

212

Table 1 shows what can happen over a period of years. Twenty years ago, 2,311 tracks were optimized for a TSO card-image data set that had a logical record length of 80. A block size of 3,600 was good because it used 99% of the track. The 2314 device types with the same block size use only 49% of the track. Note in Table 1 that a block size of 3,520 on a 2314 utilizes 96.5% of the track.

The same block size (3,520) on a 3330 results in only 81% track utilization. Decreasing the block size to 3,120 increases the track utilization to almost 96%.

The optimal block size on a 3330 yields only 82% track utilization on a 3350. Increasing the block size to 6,160 on a 3350 increases track utilization to 97%.

Results for the 3375 and 3380 devices are similar.

Table 1 also shows how difficult it is to select a block size that is optimally efficient for any two device types. In addition, it demonstrates the effect of the number of blocks per track on the usage.

Careful analysis is required to select effective block sizes. That is why many users simply pick a block size at random or use unblocked files.

An analysis of the block sizes used in several data centers indicates, surprisingly enough, that the most common block size used is 80. For example, studying one volume on which 99% of the tracks were allocated showed that only 65% of the allocated space was actually used.

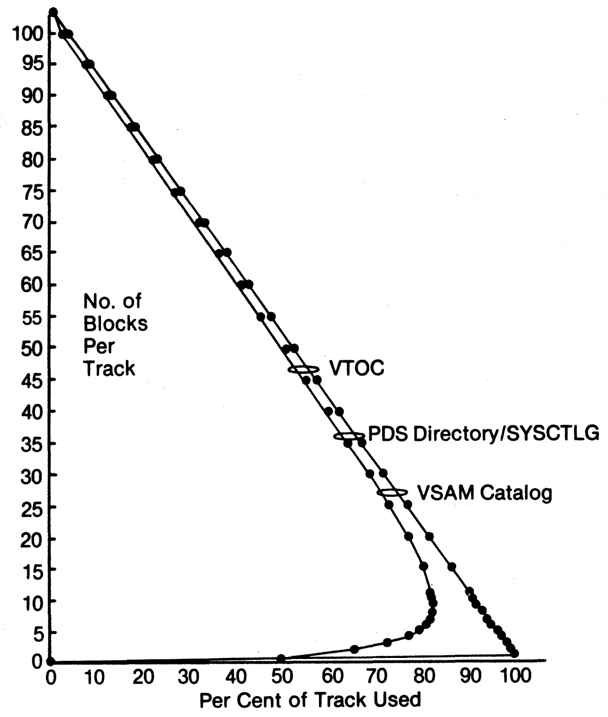


Figure 2. 3350 Track Use

Table 1. Percentage of Track Utilized

(LRECL = 80)

Block Size	2311	2314	3330	3350	3375	3380
3,600	99.3(1)*	49.4	82.9	94.4	91.0	83.4
3,520	97.1	96.5(2)	81.0	92.3	88.9	81.6
3,120	86.1	85.5	95.8(4)	81.8	87.6	85.4
6,160	—	84.5	94.6	96.9(3)	86.5	90.8
5,520	—	75.7	84.7	86.8	93.0(6)	81.4
5,440	—	74.6	83.5	85.6	91.6	91.7(8)
4K	—	56.2	94.3	85.9	92.0	86.3
6K	—	84.2	94.3	96.7	86.3	90.6
19,040				99.8(1)		80.2
9,440				99.0(2)		79.5
6,160				96.9(3)		90.8
4,560				95.7(4)		86.4
3,600				94.4(5)		83.4
47,440				—		99.9(1)
23,440				—		98.7(2)
15,440				81.0		97.6(3)
11,440				60.0		96.4(4)
9,040				94.8		95.2(5)
7,440				78.0		94.0(6)
6,320				66.3		93.2(7)
5,440				85.6		91.7(8)
4,800				75.5		91.0(9)
4,240				88.9		89.3(10)

*Number in Parentheses Indicates Number of Blocks Per Track

Furthermore, inter-record gaps occupied 11% of the tracks in use. A total of 43% of this pack was unused because of gaps, unused space at the end of allocated data sets, or a very small amount of unallocated space. Only 57% of this volume contained user data, but the volume was completely allocated.

Data Compression

To show how the lack of data compression can waste space, consider a TSO session which usually generates 80-byte card images. When a card image is saved in a sequential data set or a PDS, all 80 bytes of that logical record are written on a track even though many of those bytes are blank. Using an interactive text-editing system or one of the TSO enhancements to compress the card images and squeeze out blanks means that fewer bytes are written on a track, which saves space.

Allocation Unit Size

The smallest unit that can be allocated is now and has always been one track. On a 3380, one track equals 47,440 bytes. That is more storage than was available on many whole processors twenty years ago. This means, however, that storing only one 80-byte card image takes 47,360 bytes of overhead (59,200 percent).

Solving Capacity Problems

There are few good tools to report on DASD space usage. IBM offers IEHLIST, but it reports only the total number of cylinders and odd tracks that are available. It does not determine what is a good block size and what is not. A PDS is very deceptive. The last block pointer of a PDS may imply that the PDS is 90% full (or, rather, that 90% of it is used), but it cannot calculate how much of each track is being wasted. This is because there may be multiple members on one track, which means multiple end-of-file records and possibly multiple short blocks on that track. There may be whole members of unusable space until that PDS is compressed.

In 1978, the SHARE LSRAD report recognized all these problems. The report asked IBM to eliminate the need for users to worry about problems such as region size, block size, record format, number of buffers, and device capacity. To date, IBM has not removed any of these necessary evils. Today, end users can't be concerned with these problems. Can they be avoided?

Human Solutions

The first solution is to educate users to solve their own problems. Telling users what block size to use is an alternative, but a good size for one user may not be for another. Users can be encouraged to put small sequential files into a PDS, but then the PDS must be periodically compressed; if a user forgets the compression, he will suffer an ABEND. Also, users can be encouraged to delete their data sets when they are no longer needed, but the typical user reaction to this ranges from hilarity to hostility.

Software Solutions

The second solution is to have software manage DASD space automatically. Some products or programs that address capacity problems are shown in Table 2.

For the system-created temporary data sets, use IEHPROGM or a superscratch program that may be available on the SHARE MODS tape.

There are a number of packages available to solve the problem of being over-allocated, inactive, empty, not accessed for a long time, or not cataloged. These include ASM2, DF/DSS, DMS/OS, FDR/COMPAKTOR/ABR, HSM, PAC/MASTER, and UCC3.

There are three packages that can reduce wasted space in a PDS: ASM2, PAC/MASTER, and UCC3.

To solve the problem of a too small or too large VTOC, there is FDR/COMPAKTOR that makes it easy to change the size of a VTOC.

Performance Problems

The second major area of DASD management where problems can occur is performance. Some of these problems are.

- path contention
- spindle contention
- data set contention
- VTOC
- fragmentation
- keys
- VTOC - catalog interaction
- SYSCTLG
- block size
- Link-List and STEPLIB
- hardware reliability

Table 2. Software Solutions to Capacity Problems

Problem	Software Solutions
Systems-created temporary data sets	Superscratch, IEHPROGM
Over-allocated, inactive, empty not recently accessed, not cataloged	ASM2 (Cambridge Systems Group), DF/DSS,HSM (IBM) DMS/OS (SMM) FDR/CPK/ABR (Innovation Data Processing) PAC/Master (CGA Computer Associates) UCC3 (University Computing Company)
PDS gas	ASM2,PAC/Master,UCC3
VTOC Size	FDR/CPK

Path Contention

There are several types of path contention. One type is shared DASD, where two or more processors share DASD units. When the first processor wants to access a path, it may find that the device is busy because the other processor is doing I/O to that device. The other processor also may have reserved the device, in which case it is not busy, it is VERY busy. The device remains busy for an unpredictable amount of time depending on what program has done the reserve. For example, the Linkage Editor does not issue the release until the end of step (possibly several minutes later).

Other types of contention are channel contention, control unit contention, head of string, and/or end of string. Some new devices available from certain vendors allow I/O paths through either the lowest- or the highest-addressed device in a string.

Spindle Contention

Within the same volume, there will be contention if there are two or more data sets being accessed at the same time. That means the access arm will probably be moving, thus causing seek time. Another type of spindle contention is exemplified by the STC 8650 device. The STC 8650 contains two logical 3350s in one physical device with only one moveable access arm. If seek contention is considered in a case like this, one has to be aware of data sets that are being used on the other logical volume as well. One other cause of seek contention is when multiple jobs simultaneously access the same data set.

Data Set Contention

The types of contention described above may affect any data set that is needed by the TSO subsystem, and this impacts performance. The Master Catalog, user catalogs, swap data sets, and paging data sets may all be affected by contention, and that affects response time. The VTOC, Link List, STEPLIB, SYS1.UADS, BROADCAST, HELP, JES SPOOL volumes, user files, or even PROCLIBs may be affected.

An article was recently published that discusses CICS performance (Reference 1). It describes a problem that can exist on any interactive system. A paging file is on a volume that also contains a data set that is supposedly rarely used. It was found that this data set was frequently being used, which interfered with the paging system and resulted in much longer response times for all the CICS users. The same thing can happen to TSO; if a critical system data set is interfered with, TSO performance suffers.

Siebo Friesenborg, who works at IBM, Gaithersburg, has stated, "There is no such thing as a low-activity data set." There are only high-activity data sets that are frequently used and high-activity data sets that are infrequently used. No matter which type is accessed, the computer performs I/O operations on the data set as fast as it possibly can.

As an illustration, Figure 3 shows a typical high-performance 3350 with two cylinders of fixed heads. Under the fixed heads is a critical system data set SYS1.HOTFILE. Someone got the bright idea not to waste the other 552 cylinders, so there is another file on the same volume - SYS1.NEVER.USED. No wonder response is terrible. Analysis of the volume I/O shows that SYS1.NEVER.USED is being accessed, which means it is interfering with the ability of the system to get to that critical file. If the data set is never used, it should be moved to tape. If it is used, it ruins performance being on this volume, so it should be moved to another. To achieve the expected performance gain from these fixed heads, you must be prepared to waste the rest of the space of the volume.

VTOC

Another type of performance problem involves the VTOC. The VTOC is an unblocked file that has keys. What does being unblocked do to the size of the VTOC?

The last four major device types from IBM have so many tracks per volume that a huge VTOC may be needed, especially on a TSO storage volume where there could conceivably be thousands of very small data sets, each perhaps only one track long. It is possible, therefore, for every single track on that volume to be a separate one track data set. There ought to be a VTOC that would allow the entire pack to be allocated that way. A VTOC that big, however, will have to be 21 cylinders long on a dual-density 3330, 12 cylinders long on a 3350, and 17 cylinders long on a 3380 because the VTOC has a very small block size. Table 3 shows the maximum size of a VTOC for these four recent device types.

What is the performance impact of a huge VTOC? Whenever a new data set is allocated, the system must first search the VTOC to see if a data set already exists with the same name. It may have to scan all 12 cylinders of a 3350 VTOC with 30 tracks in each cylinder, or 360 revolutions. This will take six seconds. Once the system knows there is no duplicate data set name, then it must search the VTOC again to find the first available Format 0 DSCB to turn it into a Format 1 DSCB. On the average, this will take a search of one-half of the VTOC if you have a TSO storage volume where the VTOC is essentially full. If the storage volume is filled with very small data sets, then these timing estimates are reasonably close. Thus, it could easily take 9 seconds to allocate a new data set on a 3350. That is very poor response time. It is also poor for the rest of the system because during those 9 seconds DADSM is doing Multi-Track Search Key Equal operations on a whole cylinder at a time. This keeps the path busy and everyone else locked out of the path for most of those 9 seconds.

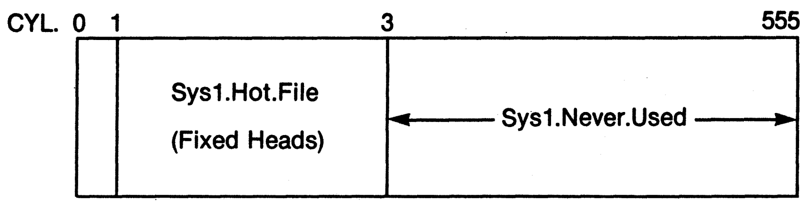


Figure 3. Typical High-Performance 3350

Table 3. VTOC Size on TSO Storage Volumes

Device Type	3330-1	3350	3375	3380
Tracks Per Cylinder	19	30	12	15
Cylinders Per Volume	808	555	959	885
Tracks Per Volume	15,352	16,650	11,508	13,275
Track Capacity (Max. R1)	13,030	19,069	35,616	47,476
DSCBs Per Track	39	47	51	53
Max. F1 DSCBs Needed	14,968	16,303	11,286	13,029
Max. VTOC Size (Tracks)	384	347	222	246
Max. VTOC Size (Cyls.)	21	12	19	17

Fragmentation

A TSO storage volume may contain many small fragments of available space. This causes long chains of Format 5 DSCBs that have to be updated whenever a new data set is allocated or an old data set is extended, scratched, or released.

Search Key Equal Multi-Track Operation

During the time the VTOC is being updated, a shared DASD will be both busy and reserved. Table 4 shows two channel programs. The only difference between these two channel programs is the third command. Channel Program A has a Search ID Equal command, which causes at most two revolutions of the track before either a record is found or a record not found condition is detected. Channel Program B, on the other hand, has a Search Key Equal Multi-Track operation because this file has been allocated on a cylinder boundary and the file has keys, such as a VTOC, SYSCTLG, or a PDS directory. Channel Program B might not end until the entire cylinder has been searched, which, in the case of a 3350, will mean 30 revolutions, or about one-half of a second.

VTOC - Catalog Interaction

On TSO storage volumes, there are typically only three types of data sets. There is always a VTOC, there may be a system catalog, and there will be user data sets almost all of which will be sequential or PDS.

If there is a catalog on this volume, there is usually a heavy interaction between the VTOC and the catalog. The reason for this interaction is that whenever a data set name is entered at a terminal, the TSO subsystem must do a catalog look-up to find out what volume that data set is on. First, the system looks in the master catalog and it finds the User ID as a CVOL pointer to a specific volume. Now the system must go to that volume and read through the catalog on that volume, but the system does not know where the catalog is on that volume. After all, you may have moved that catalog since the last TSO command was entered (something we do all the time!). Maybe someday IBM will keep such pointers in main storage like the UCB pointer to the VTOC. So first the system has to search the VTOC on that volume to obtain the Format 1 DSCB for the catalog. Once the extents of the catalog are known, the system must do I/Os to the catalog itself until it finds the block that describes the data set name.

Assuming the data set is on the same volume as the catalog just searched, even more interaction is required because next the system has to go back to the VTOC on this same volume to locate the Format 1 DSCB that describes the data set. This I/O used to be another Multi-Track Search Key Equal operation, but IBM has finally seen the wisdom of storing a pointer to this DSCB within the catalog entry. So at least the whole VTOC does not have to be searched now to find the

Table 4. Two Channel Programs

"Happy" Channel Program	"Unhappy" Channel Program
Seek	Seek
Set Sector	Set Sector
Search ID Equal	Search Key Equal Multi-Track
TIC *-8	TIC *-8
Read/Write Block	Read/Write Block

DSCB for a cataloged data set. Now the system can go directly to that DSCB but it must still go back to the VTOC and do one more I/O. Thus every time any TSO command is entered that involves one of your user data sets, there will likely be at least two interactions between the VTOC and the catalog on that volume.

Since the first of these interactions involves looking through the VTOC to find the DSCB for the catalog, the catalog should be the very first data set that is ever allocated on that volume. This causes its DSCB to be as close as possible to the beginning of the VTOC. Then when that Multi-Track Search Key Equal operation begins, it finds the DSCB of the catalog very quickly. All of this interaction between the VTOC and the catalog causes some seek time to occur unless the VTOC and the catalog can be placed within the same cylinder, or one of them under fixed heads. A VTOC, however, may be twelve cylinders long, all of which will not fit under fixed heads.

SYSCTLG

Another problem is that the SYSCTLG data set has hardware keys, so the problem of Multi-Track Search Key Equal commands still exists in an OS Catalog.

Block Size

Another performance problem mentioned before is block size. The VTOC must be unblocked, which causes performance problems. A SYSCTLG and a PDS directory are both blocked, but they have very small blocks.

Richard Schardt, who works for IBM at Gaithersburg, Maryland, wrote an IBM publication entitled "An MVS Tuning Perspective" in 1981. In it, he said, "Approximately seventy-five percent of the problems reported as poor MVS performance can be traced to some kind of I/O contention."

He also stated that a recent survey of over a quarter of a million user data sets from 47 different installations showed two-thirds of all data sets were sequential. He studied the sequential data sets and found 85% had block sizes of 4K or less, 70% of all those data sets were 2K or less, and 40% of all those sequential data sets had a block size of 500 bytes or less.

A non-optimum block size causes more EXCPs than should be necessary in order to read or write the whole file. The more EXCPs are done, the more CPU time will be used because each EXCP executes 5,000 instructions. The elapsed time to process that file is longer, and the path stays busy for a much longer time than necessary. Also, the file requires more DASD space than would be necessary with a more optimal block size.

Link List and STEPLIB

Another type of performance problem that can occur in a TSO environment is with Link List and/or STEPLIB. These have PDS directories that must be searched, and the normal channel program that is generated to search a PDS directory has a Multi-Track Search Key Equal command. If your Link List has many libraries concatenated, you can waste a lot of elapsed time and keep those I/O paths busy for too long if the libraries are in the wrong sequence!

Hardware Reliability

The final type of performance problem discussed here is hardware reliability. A soft data check may cause a DASD record to be re-read many times before the control unit decides that the error is correctable. Each re-read operation costs another revolution of the device, which means another 16.7 milliseconds. The tracks where soft errors occur must be identified and remedied.

Performance Solutions

What are some ways to address DASD performance problems? (See Table 5)

For shared DASD types of problems, there are three products available: GRS (Global Resource Serialization), MSI, and SDSI.

To solve the VTOC performance problems requires use of the indexed VTOC product (DF/DS) from IBM for your TSO storage volumes. Indexed VTOCs eliminate the Multi-Track Search Key operations by having a second copy of the VTOC which is in a pseudo-VSAM format and VSAM files do not have real hardware keys written on the tracks. This pseudo-VSAM file can be searched very quickly to determine if there is a duplicate data set name, and also the Format 0 and Format 5 chains are kept in a different form so they don't have to be searched and updated the same way they used to.

The multi-track key searches in catalogs can be eliminated by using VSAM Catalogs instead of OS Catalogs.

To address the problem of non-optimal block size or certain other types of data set attributes, there are some user exits available, such as the dynamic allocation exit, described in the Job Management Logic manual. Its name is IEFBD401. There are also three user exits now with the DF/DS product.

To reduce the problem of load library or Link List performance, there are five possible software solutions. PMO from Software Module Marketing dynamically manages Link List or a STEPLIB. TSOMON from Morino Associates describes which TSO commands are used most frequently; this can be misleading, however, because one TSO command might result in many different modules being loaded from Link List or

Table 5. Performance Solutions

- Shared DASD
  - GRS (IBM)
  - MSI (CGA Computer Associates)
  - SDSI (Duquesne Systems)
- Contention
  - Balance I/O Workload
  - Dedicate Channel Or String To Sub-System
- Seek Analysis - Determine Data Set Activity
- Indexed VTOC (IBM)
- VSAM Catalogs
- Data Set Attributes (Blk Size, Cyl. Alloc.)-User Exits
  - Dynamic Allocate (IEFBD401-Job Mgmnt. PLM)
  - DF/DS (IGGPRE00,IGGPOST0,IFG0EX0B)
- Load Libraries (Link List, STEPLIB)
  - PMO (Software Module Marketing)
  - TSO/MON (Morino Associates)
  - FastDASD (Software Corp. of America)
  - MVS/XA (IBM)
  - IMS Virtual Fetch (IBM)
- PROCLIB
  - FastDASD (Software Corp. of America)
  - EasyPROCLIB (Software Corp. of America)
- Reliability
  - Alarm (Tagg Associates)
  - UCCR+ (University Computing Co.)
  - DCR (Software Corp. of America)
- Fixed Heads on Storage Volumes
  - Not on Latest Devices
  - VTOC Won't Fit
  - Catalog, VTOC Index

from a STEPLIB. The third option is to use a seek analysis program, especially one that lists the load library members being accessed, because this can help determine which modules to make resident or put into a BLDL list. The only seek analyzer the author is aware of that can report on PDS member accesses is FastDASD. It works on any type of PDS -- load library, PROCLIB, source library, SMP, etc. It can also be used to determine the best sequence for all the concatenated libraries of the Link List. When going to MVS/XA, all the directories of the Link List will be in virtual storage (akin to a BLDL entry for every member). This will eliminate PDS directory searches to fetch a module at the expense of the paging subsystem. But still, the module must be fetched. There is only one way to keep all members of a library in storage, and that is with the new Virtual Fetch in IMS.

In addition to a load library, there might also be performance problems from a PROCLIB because it is also a PDS which may have a rather large directory that will need to be searched. PROCLIB members cannot be made resident or put in a BLDL list, but this type of PDS can be tuned if a seek analysis program is available that identifies the most actively used members. The PDS can be organized by copying it and moving the most actively used members together and as close as possible to the directory. There is also a program called EasyPROCLIB which allows as many different PROCLIBs as desired. Each TSO user can have his own private PROCLIB if he wants. This will cause most PROCLIBs to be very small, thus allowing them to be searched quickly.

In the area of reliability, two products will report problems with a device or a particular track on a device. There is ALARM and UCC Reliability Plus. There is also one product that will actually repair a hard data check error. This product, DCR, which stands for Data Check Recovery, can make an unreadable record readable once again.

Should you have fixed heads on a TSO storage volume? Perhaps, but they are not available now on the latest devices such as the 3375 and 3380. Also, the VTOC now will no longer fit in the fixed heads because a 3350 can have only two cylinders under the fixed heads, and a 3350 VTOC could be twelve cylinders long. Some possible categories for fixed heads on a storage volume would be an OS Catalog, a VSAM Catalog, or perhaps the VTOC Index data set.

A seek analysis (see Table 6) or an I/O analysis package could identify which particular data set on a storage volume ought to be under fixed heads. Generally, the most heavily accessed data set on a volume should be under fixed heads if it will fit.

To attack the channel, control unit, and string contention problems, there are two schools of thought. One idea is to balance the I/O workload, resulting in the same percentage of I/O operations on all channels, control units, and heads of string. But this assumes that all I/O is equally important and equally random. The other school of thought is to isolate each subsystem from all other subsystems, thus eliminating the possibility, for example, that IMS on one processor in a shared DASD complex can interfere with TSO on another processor. To do this, you have to dedicate a path to a particular subsystem. The

Table 6. Seek Analysis Programs

	Dapper (EDS)	FastDASD (Software Corporation of America)	Freebie (SHARE)	GTF PARS (IBM)	Seek Analysis Program (IBM)	VS1PT (IBM)
Cross Volume Reporting (essential for load balancing)	no	yes	no	no	no	no
Analyzes Multiple Volumes		yes			no	yes
PDS Analysis	no	yes	no	no	no	no
PDS Reorganization	no	yes	no	no	no	no
Graphs	no	yes*	no	yes**	no	no
History File	no	yes	no	no	no	no
Shared DASD		yes				
Reference Card	no	yes		no	no	no
Recommended Reorganization	no	yes		no	no	no
Extra Features						
Interfaces	no	yes***		no	no	no
TTR Reports	no	yes		no	no	no
Defective Track	no	yes		no	no	no

* Graphs can be generated using SAS
** Histogram of Seeks
*** FDR, COMPAKTOR, IBM Utilities

benefit of this approach is that response for that subsystem will be predictable and thus tunable no matter what happens to the files of other subsystems.

No matter which of these two schools of thought you subscribe to, you need some type of program to analyze your I/O activity, to tell you what the workload is on the different channels, control units, volumes, which data sets are being accessed, and so on.

What are some programs that can analyze I/O activity? There's DAPPER from EDS, DASD Seek Simulator from IBM, and FastDASD. There's even a free program on the SHARE MODs tape. IBM also has GTFPARS, MVS, and VS1PT for VS1.

Summary

Managing DASD in a TSO environment calls for attention to capacity as well as performance. There is no single vendor product that will do everything. A combination of products is needed: at least one to address capacity problems, and another to address performance problems.

References

1. Bloom, Jack, "NERDC CICS Performance Considerations," pp. 45-57 of CME Newsletter, December, 1982. See especially p. 51.

2. Johnston, Ted, "3350s and Small Data Sets," pp. 223-228, CME Selected Papers, Volume VI.

3. Kashmarek, Ken, "A Large Scale TSO System," pp. 631-643, Computer Management and Evaluation Selected Papers, Volume VI.

4. Reed, M. L., "Managing DASD Performance to Satisfy Workload Requirements," paper presented at CMG-XII Conference (December 1981).

5. Schardt, Richard, "An MVS Tuning Perspective," IBM Publication GG22-9023-01, 1981.