DISK ISSUES

Poor performing file I/O is the most predominant cause for program delay that I have encountered. With the current fast, highly cached disk controllers, poor I/O would be thought to be of almost no concern. But there is much opportunity for improvement even beyond the significant improvements provided by larger block sizes and increased file buffering.

Specify disk allocations in CYLINDERS. While this is not technically necessary, it is easy to remember and can improve the performance of VSAM I/O and of I/O that accesses the older uncached 3880 disk controllers. For VSAM KSDS clusters, DFP will convert IDCAMS DEFINE space requests into cylinder allocation when the control area (CA) size is one cylinder, otherwise the cluster will be allocated in tracks. To check a cluster's allocation, run an IDCAMS LISTCAT and validate the DATA component's ALLOCATION SPACE-TYPE value. Since the sequence set of a VSAM index contains one control interval (CI) for each data CA, 15 one-track CAs will cause 15 times more index sequence set entries than will a one-cylinder CA, and as a result, track allocation may cause an increase in the number of index levels. For a given cluster size, the number of index entries does not affect a cluster's sequential performance¹. However, random service time will improve with an increase in the percentage of index buffer hits. Index buffer hits will increase with an increase in the ratio of index buffers divided by the number of index records. Therefore, the reduced index sequence set size of cylinder allocation should improve the buffer hit ratio when compared to track allocation.

QSAM block size is very important to the performance of batch jobs. The MVS operating system's SMS facility named System Determined Blocksize (SDB) can be used to assign QSAM output file block sizes according to the device type being used. SDB is typically set up to assign half-track block sizes for disk files and 32KB block sizes for tape files. If SMS is available at your site, use it. Invoke SDB by omitting the output file's block size from the JCL DCB parameters. Initially, after the file is created, verify that the SDB block size used was the block size expected.

As mentioned earlier, block sizes and the number of buffers are critical to a program's execution speed. Manually specifying buffering parameters in JCL can achieve the same level of performance as an automated buffering product, but it requires considerable manual effort to implement and maintain. Increased file buffering is highly effective because the IBM defaults are set at very low buffering levels. QSAM defaults to reading and writing five blocks of data.

There are two types of VSAM file access, Non-Shared Resource (NSR) and Local-Shared Resource (LSR). Sequential processing is best performed with NSR. Random processing is best performed with LSR. NSR provides the read-ahead processing that sequential access needs; LSR does not. LSR provides improved buffer lookaside processing

¹ May 1996: personal conversation with Ron Ferguson, Software Information Services, Bellevue, Wash.
that random access needs. NSR sequential VSAM KSDS defaults to one index and two data CIs. NSR random VSAM KSDS defaults to two index and one data CF. These NSR defaults are the bare minimum and will result in poor access performance. Use the following JCL buffering guidelines when an automated buffering product is not available:

- For QSAM files, use BUFNO=20.
- For batch random access to VSAM KSDS clusters, it is very important to utilize LSR rather than NSR processing. The Batch LSR Facility is documented in IBM’s GC26-1672 Batch Local Shared Resources. For batch LSR clusters, set the number of index buffers (BUFNI) as high as the region size will allow, but do not exceed the number of index entries (including the sequence set) plus one, as more buffers than this will be wasted. Set the number of data buffers (BUFND) in the range of 20 to 30. It is more important to retain the index buffers in memory than to retain the data buffers.

For batch random access, NSR processing is an alternative to BLSR, but NSR is not as beneficial in terms of performance. NSR processing does not perform BLSR’s buffer looksaside, nor its deferred writes. NSR processing is invoked with the BUFND and BUFNI AMP parameters. For NSR random access, set BUFND at 2 (or more to assist in CA split processing). Set BUFNI to a maximum of the number of index set records (excluding the sequence set records) plus one.

- For batch sequential access to VSAM KSDS clusters, it is very important to utilize NSR processing. VSAM NSR invokes the read-ahead processing that BLSR does not. To invoke NSR processing, use the BUFND and BUFNI AMP parameters in the JCL. Set BUFNI to 1. Set BUFND to a minimum of twice the number of data CIs per track plus 3 to activate alternating buffer support and anticipatory buffering.

Textbooks often advise setting VSAM data CI sizes small for random access and large for sequential access. This is true in theory, but VSAM KSDS files are usually accessed both sequentially and randomly. This small/large CI size rule of thumb was developed in the days of slower I/O subsystems and when buffering products were not in common use. I recommend larger and more practical CI sizes than the old rule of thumb would suggest. The larger CI sizes do not significantly impair random performance but do improve sequential performance. For initial CI size recommendations, use 4KB for the index and 8KB for the data. These provide good performance for the blend of random and sequential processing. Specify the CI sizes in the IDCAMS DEFINE, or the defaults will be applied. In many instances, the defaults will result in CI sizes that are too small, may create unaddressable space in the data CI, and often result in poor performing clusters.

When increasing the CI size for an existing VSAM dataset, exercise caution. There are a few additional elements to check.

First, if the cluster’s DEFINE included a BUFFERSPACE parameter, then it too may need to be increased. In reality it is better to not use the BUFFERSPACE option. Remove it from the DEFINE. Instead, control VSAM buffering with either an automated buffering product or through the buffering JCL parameters.

Second, if the cluster is defined to a CICS LSR pool, then notify the CICS specialist. The LSR memory allocation may need to be increased due to the increase in the CI size. LSR buffers are defined by either the File Control Table (FCT) macro or by the Resource Definition Online (RDO) facility. With an FCT-defined LSR pool, the index CI sizes should differ from data CI sizes by at least 2KB so that index and data CIs will be assigned to different sized buffer subpools within the buffer pool. When index and data control intervals are assigned to the same buffer subpool, the index buffers are subject to flushing as BROWSE I/O loads sequential read-ahead data buffers into the buffer subpool. Since random VSAM performance is very dependent upon the index buffer hit ratio, better random performance can be expected with improved index retention in the buffers. RDO bufferpools are designed to provide improved index retention. RDO can segregate index buffers from data buffers even when they are the same CI size.

Third, if the VSAM cluster is a frequently accessed cluster, then suggest that the CICS specialist place the cluster into a Shared Data Table. Shared Data Tables are a data-in-memory technique that provides excellent cluster performance. Shared Data Tables now support CICS BROWSE in addition to GET.

VSAM clusters have been in existence for nearly 20 years. Some options, such as SPEED, are still recommended, but other options are not. Do not use BUFFERSPACE, IMBED, REPLICATE, ERASE, or WRITECH. BUFFERSPACE indirectly defines the numbers of buffers for NSR use, but it is better to use automated or JCL buffering. IMBED and REPLICATE are two options that diminish the benefits of disk caching, and they increase the data component's space requirements.

ERASE is a data security option that causes the cluster’s data to be erased when the file is deleted. Data security is better controlled with a site’s security package. WRITECH invokes immediate verification (reread) of every block of data written to the cluster. WRITECH may have been important in the days of the IBM 2314 disk drive, but it is not necessary today.

VSAM clusters are defined with various levels of SHAREOPTIONS, which provide different levels of concurrent cluster access. Of these levels, SHAREOPTIONS 4 is the least restrictive in that it permits concurrent cluster update; however, it results in the poorest performance and requires the user to design the most complex of recovery procedures. To allow concurrent update access on a cluster, VSAM must flush the in-memory buffers of a SHAREOPTIONS 4 cluster with every read and write. Yes, read too! Do not use SHAREOPTIONS 4 unless absolutely necessary. With a little luck, it is possible to improve the performance of a SHAREOPTIONS 4 cluster. During periods when no more than one program will be updating a cluster, access can be greatly improved by temporarily assigning SHAREOPTIONS 1 (or 2) to the cluster with an IDCAMS ALTER, executing the read-only or update program, and then resetting the SHAREOPTIONS back to 4 with a second IDCAMS ALTER.

A dummy key is often used to initialize a cluster. Use a low-values key rather than a high-values key. A high-values key will cause continuous CI and CA splitting because future inserts will be at a key lower than the dummy key.

Believe it or not, VSAM CI and CA splits are not necessarily bad. Consequently, it is more beneficial to invest time improving VSAM buffering than to invest time reducing splits. Two ‘split’ concerns are the sizing of the freespace for a cluster and the scheduling of the cluster reorganization. CI splits occur very quickly, and their duration is not significant. The duration of a CA split has traditionally been a concern, but a CA split can be completed in about one second with a fast disk subsystem. The one minute 30 seconds of extra elapsed time that is required to process 90 CA splits is not a significant time penalty for a batch program. After CI and CA splits have occurred, their existence does not significantly impair the performance of future sequential or random access on the cluster.

Clusters should not be frequently reorganized simply to remove splits. CI freespace is seldom needed, and CA freespace is needed in only about 50 percent of the clusters. Consider the following points:

- Typically, splits do not occur uniformly throughout the cluster.

---

Most splitting occurs during batch updating when the loss of CPU cycles and incurred delays are least critical.

The split impact occurs at the instant of the split, and once a split occurs, the lost CPU and the delay caused by the split cannot be recovered.

Splitting places the most possible free space at the point of the split, which is the most likely location for future inserts (a hot spot).

From these points it can be argued that frequent cluster reorganization may not be good. Reorganization would:

- reduce the free space at the location of the splits (hot spots), possibly leading to re-splitting in the future;
- require the cluster to be out of service during the reorganization; and
- consume CPU cycles to perform the reorganization. Therefore, consider setting free space parameters to zero. Zero free space sets a cluster at its minimum size and maximizes performance.

### TAPE ISSUES

Current channel speeds, robotic drives, low transport ratio per controller, Improved Data Recording Capability (IDRC), and cartridge capacity make tape processing issues less critical than in the past. But there are still a few methods to improve tape performance. Confirm that IDRC is active by checking for the COMP indicator in joblog message IEC705I. If available, use SMS’s System Determined Blocksize feature (SDB) by omitting the blocksize from the output file’s DCB parameters. While it can be expected that very small blocksizes will impair a program’s I/O performance, a small blocksize can greatly impact the transfer rates for other files on the same tape controller. Increased buffering, whether by an automated product or by DCB parameters, can improve the overall data transfer rate. While I recommend BUFNO=20 for DCB controlled buffering, automated buffering products may utilize a greater number of buffers. I suggest exercising caution when implementing increased buffering, because increased buffering causes each I/O to contain a greater number of bytes and each of these larger I/Os will tie up the controller for a longer period of time. This may cause a delay for the I/Os directed to other drives on the same controller. Therefore, I recommend increasing tape buffering only for critical or important jobs. If a batch job encounters long delays due to waits for manual mounting of multi-volume tape datasets, consider using UNIT=(,2). UNIT=(,2) calls for pre-mounting of successive volumes for the dataset. For output tape files, the last mounted cartridge is never opened and it returns to the scratch pool. Use UNIT=(,2) only if sufficient tape transports are available when the job starts execution.

The concluding article in this series will examine DB2 and sorting issues.

Special thanks to Mr. Ron Ferguson, founder and president of Software Informational Services, Bellevue, Wash., for his excellent suggestions and technical review of the disk issues segment of this article.

---

Neil Ervin is a performance and capacity planning specialist with the Huntington National Bank in Columbus, Ohio. Neil’s articles “MVS Performance Tuning Tips” and “Getting Started With Performance Tuning and Capacity Planning” have appeared in Technical Support and have been presented at local, regional, and national conferences. He is active on the Internet’s MVS newsgroups and can be contacted at ervineis@infinet.com.

©1996 Technical Enterprises, Inc. Reprinted with permission of Technical Support magazine. For subscription information, email mbrship@naspa.net or call 414-768-8000, Ext. 116.