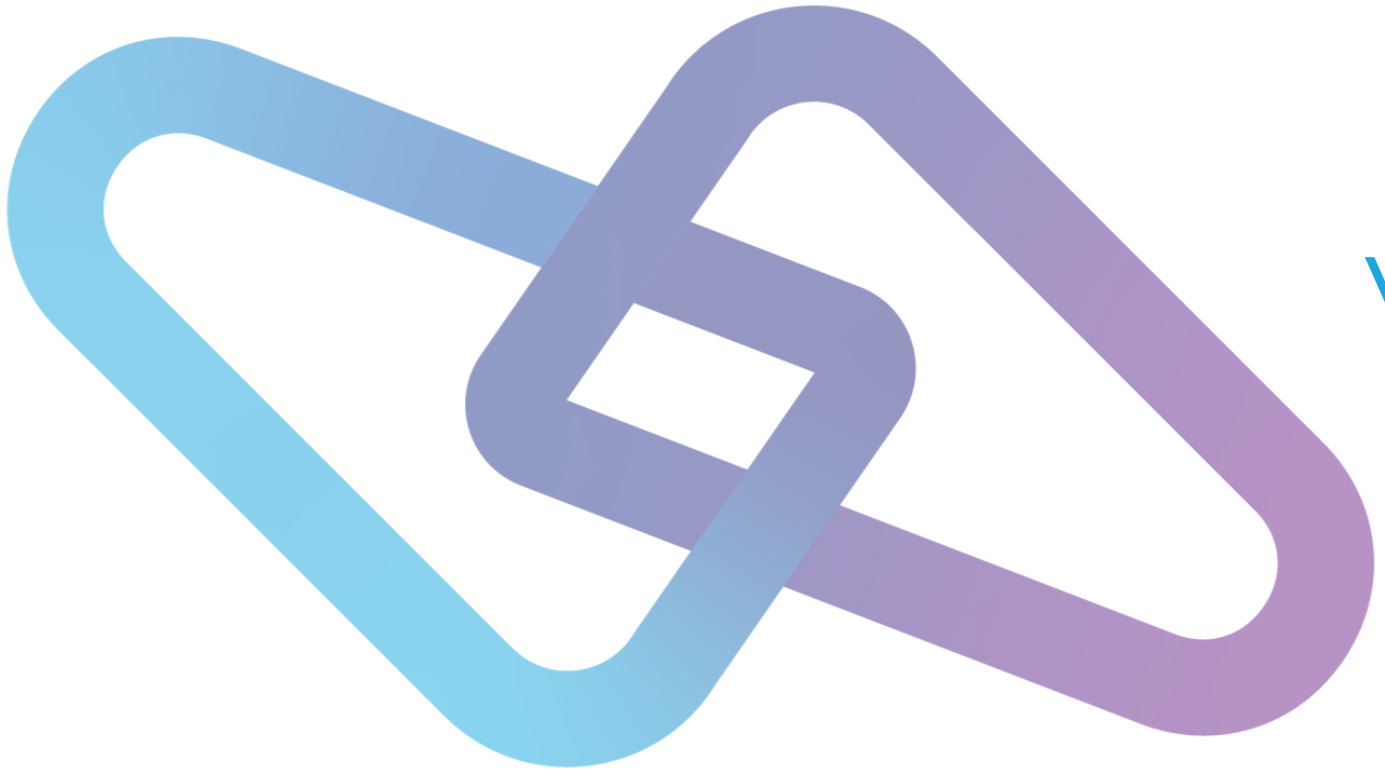

21CS VSE^n VSAM Tuning and Troubleshooting

Edmund Wilhelm
VSE^n Core Senior Software Engineer

Agenda

- Performance Enhancements
- Excessive VSAM File Reorganization
- “Gobi Desert” Problem
- Rebuild a User Catalog
- Virtual Disk
- Recommendations

Performance Enhancements

24-bit Constraint Relief (2010 z/VSE 4.3)

- In z/VSE 4.3, VSAM modules were moved from SVA-24 to SVA-31. **24-bit SVA Savings: 473K, over z/VSE 4.2.**

- All VSAM control blocks, with a few exceptions, were moved from 24-bit to 31-bit Partition GETVIS. **Savings: about 1.2M for 300 files. For non-shared (non-LSR) files, the savings are much greater.**

- These benefits require no action on the part of the customer.

Performance Enhancements cont.

- Partition GETVIS requests by VSAM are now acquired from GETVIS pool IJBVSM. This enables better tracking of GETVIS usage by component.

- VSAM supports user-generated control blocks (ACB, RPL, EXLST) as well as action macros (OPEN, CLOSE, GET, PUT, ERASE, TESTCB, SHOWCB, GENCB) to be in and executed from 31-bit Partition GETVIS.

  Note: When a 31-bit ACB is passed to either OPEN or CLOSE macro, only one ACB may be passed at a time. 24-bit ACBs may be passed in a list, delimited at the end with x'0A'.

Performance Enhancements cont.

- An option (“BUFDAT=RMODE31”) has been added to the DLBL to allow a legacy application to move the VSAM data buffers to 31-bit Partition GETVIS.

  Before this option can be used, ensure that the application does not access VSAM records in “Locate” mode. If so, the application needs to be running in AMODE(31) in order to “see” records residing in buffers 31-bit space.

  See description for RPL OPTCD=(LOC) in Chapter 12 “Descriptions of VSE^n/VSAM Macros” in “VSE^n/VSAM User’s Guide and Application Programming”

Performance Enhancements cont.

Index and Upgrade Set (Alternate Index) Buffers

- May always reside above 16M (CICS and Batch), even for 24-bit applications, since index buffer location is transparent to application.
- LSR only if also the data reside above 16M, since data and index of an LSR file are always in same LSR pool.

Performance Enhancements cont.

Multiple VSAM LSR Pools Above the Line

- Up to 15 LSR pools for OLTP (and 16 for batch partitions).
- Separation of heavily used files with same CI sizes (data, index). Such files may be 'unfriendly' to each other (stealing buffers, dominating a subpool during BROWSE etc...).
- Group/Separate files by application, work shift, usage frequency etc.
- Separation of data and index-CIs (of different files) so that data and index CIs do not compete in the same subpool.
- Full freedom to select optimal data and index-CI sizes without regarding CI-sizes of other LSR files).

Performance Enhancements cont.

- Smaller subpools for faster searches are possible. This avoids long subpool searches (CPU-time) with low chance of a hit ratio increase if a subpool is shared with too many other files.

Overall:

- Reduction of VSAM I/Os (with the same subpool sizes)

  or

- Even higher reduction of VSAM I/Os (with larger subpool sizes) at cost of some CPU-time.

Performance Enhancements cont.

z/VSE customers can exploit very large LSR buffer pools through VSE/VSAM buffer hashing.

Performance Enhancements cont.

Tuning of Multiple LSR Pools

- LSR pools: Define only as many as required.
- STRNO: Do not largely oversize STRNOs.
  For a single LSR pool, the maximum STRNO was and remains 255.
  Assuming the same workload, you may require only a few more STRNOs for all your pools (since they are shared less) compared to the number you had before for a single pool.
- KEYLEN: Specify correctly.
  If you specify as KEYLEN the maximum of all the KSDS files in the pertinent pool, CICS need not determine that value at startup time.

Performance Enhancements cont.

- Use the provided OLTP LSR hit ratio statistics with the I/Os and the number of hits per CI-size (subpool) for each pool to optimize the number of buffers per subpool as follows:
  - Increase number until hit ratio no longer increases significantly.
  - Decrease number until hit ratio no longer decreases significantly.

Performance Enhancements cont.

Reserving a subpool for a single file

This may be considered for very specific tuning situations or special treatment/isolation of an individual file.

It may also be a temporary step, just to find out how a specific file behaves on its own, before sharing a subpool.

Performance Enhancements cont.

- **Larger VSAM Physical Blocksizes** (up to 30K vs 8K)

  3390 track utilization will increase from 87,2% ta 98,2% (CI-size= physical block size= 18K, as an example)

  CI-sizes: 8K and above

  - Faster I/Os for sequential access
  - Fewer index I/Os through larger max-CAs
  - Less CCWs cause faster CCW translation

Performance Enhancements cont.

- OLTP Usage of VSAM Dataset Name Sharing

  For VSAM NSR and LSR base and alternate index files, opened multiple times (via multiple ACBs) within the same partition. Mostly an integrity benefit.

  – Slightly more efficient buffer usage

  or

  – Slight virtual storage savings, mostly for VSAM NSR

Performance Enhancements cont.

- VSAM B/R Functions Extended
  - Backup/Restore from/to Disk (e,g, automatic operation)
  - Fixing buffers in storage with the UPSI Job Control command: // UPSI 1
  - COMPACT/NOCOMPACT parameter for Backup

Excessive VSAM File Reorganization

- VSAM add on enhancement products may decrease performance,
- When defining a file, you can specify a percentage of each Control Interval (CI) to be reserved as free space, and a percentage of CIs in each Control Area (CA) can also be kept free,
- Splitting CIs and CAs to create additional free space where needed can be very useful,
- The benefits from splits are negated by reorganizing files more often than necessary.

Excessive VSAM File Reorganization cont.

- A CI split is inexpensive, as far as computer processing goes (approx. 4 I/O operations).

- Suppose the file is reorganized and all free space is restored to the initial load configuration. All the extra space that was created through the CI split process is removed and future record insertions may have to re-create the free space over a period of time.

Excessive VSAM File Reorganization cont.

- If there is no free CI available, VSAM needs to create free CIs and a CA split is used to create the needed free space.
- This is like a CI split but is much more time-consuming.
- It is often thought that shorter CIs are critical to direct performance. With today's DASD subsystems, FICON and ESCON channels, and the ability to cache large number of records in main storage buffers, the gain in I/O response time from shorter CI sizes is much less important than it was 30 years ago.
- Unlike CI splits, CA splits can take a lot of processing and I/O resources. May require hundreds of I/Os when many small Data Cis exits in each CA. Larger Data CIs (and fewer Data CIs per CA) will reduce the cost of CA splits.

Excessive VSAM File Reorganization cont.

- A completed split has only a small impact on subsequent processing.
- Most important is the positive effect on additional insert activity in the same key value vicinity.
- The number of CI or CA splits that have occurred should NOT be used as a trigger for reorganization.

Excessive VSAM File Reorganization cont.

- VSAM file reorganization will:
  - Squeeze out any additional free space that was created in the CIs and CAs due to split activity.
  - Move records from CAs moved to the end of the file by CA splits back into physical sequence.
  - Repopulate the file with the initial free space defined in the DEFINE CLUSTER command.
- If your file has any clustering of insert activity, reorganization may do more harm than good.

“Gobi Desert” Problem

.

- Can affect any KSDS file (including VSAM catalogs)
- Add at end, delete from beginning
- Index High Key not changed by delete
- Empty data CIs are never reused
- Impact:
  - Performance degradation
  - Cluster (catalog) growth
- Resolution:
  - Define keys (or cluster names) so that they are random
  - Frequent reorganization of file (or catalog).

“Gobi Desert” Problem cont.

.

JCL to reorganize catalog to correct “Gobi Desert” problem:

```
// JOB REPRO FILE TO DISK
// DLBL OFILE,'REPRO.OUT',0,SD
// EXTENT SYS005,SYSWK2,1,0,1515,75
// ASSGN SYS005,DISK,VOL=SYSWK2,SHR
// DLBL IFILE,'VSESP.USER.CATALOG',,VSAM,CAT=VSESPUC
// EXEC IDCAMS,SIZE=AUTO
REPRO INFILE (IFILE) -
OUTFILE (OFILE -
ENV ( BLOCKSIZE(18432) -
RECFM(V)))
IF LASTCC = 0 THEN -
DO
REPRO INFILE (OFILE -
ENV ( BLOCKSIZE(18432) -
RECFM(V))) -
OUTFILE (IFILE)
END
/*
/&
```

Rebuild a User Catalog

Preparatory Steps

1. Verify that MINI startup is working (required for step 8).
2. Run a LISTCAT and VTOC Display for step 9.

Rebuild a User Catalog cont.

```
// JOB LISTCAT SHOW CATALOG SPACE
// SETPARM JOBCAT='<Catalog ID>'
// DLBL IJSYSUC,&JOBCAT
// EXEC IDCAMS,SIZE=AUTO
         LISTCAT SPACE ALL
/*
/&
```

Rebuild a User Catalog cont.

```
// JOB DITTODVT PRINT VTOC FOR DOSRES AND SYSWK1
// UPSI 1
// EXEC DITTO
$$DITTO DVT INPUT=VDOSRES,DSNAME=Z999999%.**,SORTBY=EXTENT
$$DITTO DVT INPUT=VSYSWK1,DSNAME=Z999999%.**,SORTBY=EXTENT
$$DITTO EOJ
/*
* OTHER DISKS AS IN LISTCAT SPACE ALL ???
/&
```

Rebuild a User Catalog cont.

1. Stop any applications (for example CICS, ICCF, VTAM, …) which access VSAM files resident in the VSAM User Catalog. This may include applications that access VSE/AF libraries in VSAM or DL/I databases in VSAM.

2. Back up all VSAM files belonging to the VSAM user catalog. Run IDCAMS BACKUP(*) in order to back up all the data sets from your user catalog. Use the parameter EXCLUDE to bypass corrupted files.

3. Back up any VSE libraries from the User Catalog using the Librarian. Since IDCAMS BACKUP ignores NOCIFMT files, you need to back up your VSE libraries separately. Please use LIBR BACKUP for that.

Rebuild a User Catalog cont.

4. Prepare **DEFINE** jobs for the User Catalog and related dataspaces as well as **RESTORE** jobs for VSAM files and VSE libraries.
5. Add the DEFINE job for the Compression Control Data Set (CCDS) if using compression and for Default Models if needed.
6. Submit all the prepared jobs to the POWER Reader **with the disposition L** (* $$ JOB JNM=DEFINE,**DISP=L**,CLASS=0) in order to keep them in the queue.
7. Disconnect the User Catalog from the Master Catalog. Run **IDCAMS EXPORT DISCONNECT** in order to remove the User Catalog pointer from the Master Catalog.

Rebuild a User Catalog cont.

8. IPL VSE with only POWER (“MINI” configuration). To do this use the Load Parameter LOADP=…P…. In order to be prompted for a startup mode.
9. Delete VTOC entries of the User Catalog using **IKQVDU**. The entry names must be prepared in advance, e.g., comparing VTOC listing with **IDCAMS LIST SPACE ALL**.
10. Release the jobs to define a new User Catalog, its dataspaces and restore data. Release the jobs to define the CCDS, default models and VSE Libraries if necessary. Restore any libraries that were in the User Catalog.

Virtual Disk

- Data in Memory for unchanged applications
  - Compiler work files
  - Sort work files
  - BLDINDEX work files
- Moving well buffered VSAM files to virtual disk may not be beneficial. The I/Os are already minimized. If lower buffering is used, more CPU-time may be required in I/0 paths and their interceptions.
- For VSAM files under OLTP, before using Virtual Disk, try to use OLTP Data Tables or large VSAM LSR/NSR buffers.
- VSAM space on virtual disks can only be defined in catalogs residing also on virtual disks.

Recommendations

See „*VSE/VSAM User's Guide and Application Programming*“, Chapter 7, Optimizing the Performance of VSE/VSAM.

Addresses:

- Number of Files Defined in a Catalog
- Data Space Classification
- Control Area (CA) Size
- Control Interval (CI) Size
- NSR Buffers
- LSR Buffers
- Multiple Volumes
- Space Allocation
- Data Protection and Integrity
- Free Space
- Indexes

Recommendations cont.

General

- Maximize size of Control Area
- Use reasonably large data Control Intervals
- Let index Control Interval size default.
- Compression will save I/Os, but will cost CPU
- Additional buffering will save I/Os
- For sequential processing, use largest possible data CIs, and multiple data buffers.
- For direct processing, use smallest possible index CIs, and multiple index buffers.

Recommendations cont.

Catalog Considerations

- Place static (once defined, multi-access) and dynamic (Reusable) files in separate catalogs (on separate volumes).
- Place batch vs on-line files in separate catalogs.
- Do not put all your eggs in one basket. (Do not keep large number of files in a single Catalog). Many files in a single catalog (for example, a thousand files) can significantly increase the run time for most IDCAMS functions. This includes DEFINE, DELETE, and LISTCAT functions. It also impacts open and close performance.

Recommendations cont.

Space Considerations

- Do NOT use small (less than cylinder or MAX-CA) allocations
- Check results of DEFINE and load using LISTCAT
  - Data CI size (bigger = better)
  - Physical block size (= CI size ideally)
  - Index CI size (small as possible, rounded up to the next LSR buffer size for files used by OLTP)
  - Number of CIs per CA
  - Size of CA (One CA for small files, cylinder size for larger files)

Recommendations cont.

CI Split Considerations

- CI splits are not very costly in terms of system overhead (four I/Os, a bit of CPU processing overhead).
- Do not specify CI free space. Do not reorganize files just based on CI split numbers.
- If an error occurs (split is interrupted), original CI with Split-in-Process bit is kept.
  - Next time this CI is read, in keyed update mode, the split will be completed.
  - If the access is not keyed and not update, rc x'00' with feedback x'1C' is returned.
  - If the access is not keyed, but get-for-update, rc x'08' with feedback x'9C' is returned.

Recommendations cont.

CI Size Considerations

- Cached DASD
  - Bigger Data CIs means smaller Index CIs
  - Bigger Data CIs may mean fewer CI splits with a given free space percentage
- LSR Considerations
  - Set Data CI to 4K, 8K, 12K…(8K good start)
  - Set Index CI to smallest possible LSR pool size

Recommendations cont.

CA Split Considerations

- CA splits are quite expensive (up to a thousand I/Os) when they occur, but do not substantially impact future processing.
- If inserts are heavily clustered, CA splits may be more efficient than CA Free Space.
- Reorganization will consolidate the cluster, removing free space created by splits, which may have to be added back in.
- Do not reorganize dataset after a certain number of CA splits.
- Define CA free space (at least 20%) for on-line files. FREESPACE(0,20)

Recommendations cont.

CA Size Considerations

- CA Size should be as big as possible -- it is the largest of:
  - Cylinder or MAX-CA size
  - Primary allocation amount
  - Secondary allocation amount
- Make primary and secondary amounts at least a cylinder (MAX-CA)

Recommendations cont.

File Load Considerations

- SPEED vs. RECOVERY
  - RECOVERY is the default
    - CAs are pre-formatted before records are loaded
    - Permits restart of user written load program
  - SPEED is the preferred option
    - Only affects initial load after DEFINE
    - Bypasses preformat during initial load
    - No effect after initial load
    - Saves 1/3 to 1/2 of I/O operations during load

Recommendations cont.

Freespace Considerations

- Only applicable during sequential insert. Initial load, sequential add of new records

- Excessive free space ("bubbles" in the data)
  - Extra I/O operations
  - Less efficient buffering and caching
  - Slower normal (sequential and direct) processing

- Too little free space
  - Extra split activity
  - Slower insert processing
  - Slower normal (sequential and direct) processing

Recommendations cont.

Sharing Considerations

- Significant performance degradation going from:
  - SHR(1) or SHR(2) going to SHR(4)
  - SHR(4) going to SHR(4 4)
- Degradation comes from:
  - Additional CPU (processor) time
  - Additional I/O operations

Recommendations cont.

Alternate Index (AIX) Considerations

- Consider NOUPDATE and rebuild of AIX instead of dynamic update
- Consider sequential extract and sort of base cluster records (without using AIX) if many records are touched (e.g. 1/10 or more); AIX access if few are touched (1/1000 or less). In between -- try it and see what works best.

Recommendations cont.

Buffer Considerations

- Non-Shared Resources (NSR)
  - Each string must have adequate index buffers
  - Requirements PER STRING…
    - Unacceptable -- one buffer (old default)
    - Acceptable -- one buffer per index level (new default)
    - Good -- enough buffers to hold all high-level index plus one
    - Best -- enough buffers to hold entire index

Recommendations cont.

Buffer Considerations cont.

- Local Shared Resources (LSR)
  - The pool must have adequate index buffers
    - See above -- Requirements PER STRING becomes IN POOL
  - Monitor VSAM LSR statistics to ensure sufficient buffers are provided in pool to get high probability of finding desired record in the pool (high hit ratio)
  - Data buffers monitored for high hit ratios as well

Recommendations cont.

Backup/Restore Considerations

- **Compaction** (“COMPACT” option)
- Software compaction of backup data via “**COMPACT**” option during Backup.
- More efficient to use hardware
- Do not use to backup compressed data
- “BLOCKSIZE(65535)”
- “BUFFERS(4)” (maximum 8)
- “// UPSI 1” for PFIXing buffers sometimes helps.
- Multiple concurrent backups very efficient.

Recommendations cont.

Performance Measurement

VSAM keeps statistical information about a file in its catalog record. Some statistics, such as number of extents in a file, number of records retrieved, added, deleted, and updated, and number of CI splits, can help you decide when to take action to improve performance. Appropriate actions could be, for example, reorganizing a file or altering the type of processing.

You can list the entire catalog record, the statistics, and the parameters selected when the file was defined, by using the LISTCAT command.

Recommendations cont.

Performance Measurement cont.

Statistics and parameters include:

- CI size
- Percentage of free CIs per CA
- Number of bytes of available space at the end of the file
- Length and displacement of the key
- Maximum record length
- Number of buffers
- Usage of LSR buffer pools
- Number of records
- Number of levels in the index
- Number of extents
- Number of records retrieved, inserted, deleted, and updated.
- Number of CI splits in the data and in the sequence set of the index
- Number of EXCPs that VSAM has issued for access to a file
- A time stamp that indicates if either the data or the index has been processed separately

Conclusion

Thank you for your attention ;-)