IBM

Washington
Systems
Center

Technical
Bulletin

VSAM Performance Study
Foil Presentation with Text

By S. E. J. Friesenborg
T. R. Mitchell
Washington Systems Center

DAPS Code 0894
GG22-9022-00
April 1979

Washington Systems Center
Gaithersburg, Maryland
Technical Bulletin

VSAM Performance Study
Foil Presentation with Text

S. E. J. Friesenborg
T. R. Mitchell

VSAM using the ISAM Interface Program was measured and compared to native ISAM performance on 3330-1 disks and 158-1 CPU. The major findings are part of this presentation. The text explains the contents of each foil.

The prupose of the measurement runs was to quantify the affect on performance of the various VSAM and ISAM options and to provide current VSAM measurement information.

This Technical Bulletin is being made available to IBM and customer personnel. It has not been subject to any formal review and may not be a total solution. The results obtained are valid only for the configuration and environment in which they were obtained. Similar results may not be achieved on other configurations and environments.

A form is provided in the back for comments, criticisms, new data, and suggestions for future studies, etc. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever.

DAPS Code 0894
GG22-9022-00
April, 1979

TABLE OF CONTENTS

VSAM

(ISAM)

PERFORMANCE STUDY

OBJECTIVES

- SCOPE
  - BATCH
  - IIP
  - COBOL
- QUANTIFY
- CONTROL

DESIGN

- MEASUREMENT
  - SMF
  - SMI
  - (RMF)
- ISOLATION
- INVESTIGATION
- STAND-ALONE

DESIGN - PROGRAM EXAMPLE

```
IDENTIFICATION DIVISION.                                                00010002
PROGRAM-ID.     RR.                                                     00020002
AUTHOR.         SIEBO FRIESENBORG.                                      00030002

INPUT-OUTPUT SECTION.                                                   00130002
FILE-CONTROL.                                                           00140002
    SELECT ISAM-FILE,                                                   00150002
    ASSIGN TO DA-I-ISAM,                                                00160002
    ACCESS MODE IS RANDOM,                                              00170002
    NOMINAL KEY IS N-KEY,                                               00180002
    RECORD KEY IS R-KEY.                                                00190002
DATA DIVISION.                                                          00200002
FILE SECTION.                                                           00210002
FD  ISAM-FILE,                                                          00220002
    BLOCK CONTAINS 12 RECORDS,                                          00230013
    RECORD CONTAINS 200 CHARACTERS,                                     00240006
    RECORDING MODE IS F,                                                00250002
    LABEL RECORD IS STANDARD,                                           00260002
    DATA RECORD IS ISAM-RECORD.                                         00270002
01  ISAM-RECORD.                                                        00280002
    05 FILLER      PIC       A(1).                                      00290002
    05 R-KEY       PIC       9(8).                                      00300002
    05 FILLER      PIC     A(192).                                      00310002
WORKING-STORAGE SECTION.                                                00320002
77  N-KEY          PIC       9(8).                                      00330002

PROCEDURE DIVISION.                                                     00460002
    OPEN INPUT ISAM-FILE.                                               00470002
    ACCEPT CARD-IMAGE FROM SYSIN.                                       00480008
    COMPUTE HIAMT = HIAMT / INCREMENT.                                  00490009
    CALL 'SPIKE'.                                                       00500002
LOOP.                                                                   00510002
    COMPUTE IZN2 = IZN1 * 65539.                                        00520003
    IF IZN2 IS GREATER THAN ZERO GO TO STATEMENT6.                      00530005
    COMPUTE IZN2 = IZN2 + 2147483647 + 1.                               00540003
STATEMENT6.                                                             00550003
    COMPUTE YFL = IZN2.                                                 00560003
    COMPUTE IZN1 = IZN2.                                                00570003
    COMPUTE YFL = YFL * .4656613E-9.                                    00580003
    COMPUTE RNDU = YFL.                                                 00590003
    COMPUTE TEMP = RNDU * HIAMT.                                        00600009
    COMPUTE N-KEY = INCREMENT * TEMP.                                   00610009
    READ ISAM-FILE.                                                     00620005
    ADD 1 TO X-KEY.                                                     00630006
    IF X-KEY IS LESS THAN OPERATNS GO TO LOOP.                          00640006
END-OF-FILE.                                                            00650004
    CLOSE ISAM-FILE.                                                    00660002
    MOVE 1 TO RETURN-CODE.                                              00670012
    STOP RUN.                                                           00680002
```

DESIGN - STEPS

IIPSW	SEQUENTIAL WRITE (LOAD)
IIPSR	SEQUENTIAL READ
IIPRR	RANDOM READ
IIPRU	RANDOM UPDATE
IIPRW	RANDOM WRITE (INSERT)

- ISAM VARIANTS
  CYLINDER INDEX
  TRACK AREA
  BLOCKSIZE
  OPTIMAL
- VSAM VARIANTS
  SPLITS

DESIGN - RUNS

ISAM1	VSAM1	BASE CASE
ISAMB	VSAMB	BUFFERING
ISAMC	VSAMC	RESIDENT INDEX
ISAMT	VSAMT	FULL TRACK
ISAMW	VSAMW	WRITE CHECK
ISAMD	VSAMD	DUMMY
ISAMR	VSAMR	ADDRSPC=REAL
ISAMU		OPT=U
ISAMA		TRACK AREA
	VSAMA	CA SPLITS
	VSAMI	CI SPLITS
ISAMOR		'OPTIMAL' REAL
ISAMO	VSAMO	'OPTIMAL'

DESIGN - DATA SETS

PARAMETER	VSAM	ISAM
RECORDS	86763	86763
LRECL	200	200
BLKSIZE	2048	2400
KEYL	8	8
FREE REC/CYL	264	64
CYLINDERS	117	103
IMBED	YES	TRKX
BUFNO	2	5

RESULTS - SEQUENTIAL WRITE

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1 ISAMB	203.90 98.16	36.58 21.69	11.94 5.48	86763 86763	7708 1979	53.2 29.3
ISAM1 ISAMT	203.90 104.38	36.58 17.45	11.94 4.04	86763 86763	7708 1534	53.2 27.1
ISAM1 ISAMU	203.90 195.38	36.58 34.85	11.94 12.25	86763 86763	7708 7797	53.2 51.3

VSAM1 VSAMB	370.01 143.66	77.73 57.98	10.76 4.40	86763 86763	11713 976	78.3 66.8
VSAM1 VSAMT	370.01 126.88	77.73 50.11	10.76 4.11	86763 86763	11713 2631	78.3 40.8
VSAM1 VSAMRC	370.01 311.68	77.73 80.37	10.76 12.02	86763 86763	11713 13626	78.3 84.5

ISAMOR ISAMO VSAMO	63.86 75.96 134.92	11.92 20.13 51.43	1.52 5.83 3.77	86763 86763 86763	1979 2067 1143	28.0 27.7 66.1

RESULTS - SEQUENTIAL READ

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1	184.11	32.37	10.78	86763	7232	40.2
ISAMB	90.56	17.82	4.46	86763	1588	35.4
ISAM1	184.11	32.37	10.78	86763	7232	40.2
ISAMT	83.00	13.07	3.65	86763	1356	34.4

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
VSAM1	222.79	57.86	11.44	86763	10954	40.8
VSAMB	85.89	41.53	3.65	86763	215	30.8
VSAM1	22.79	57.86	11.44	86763	10954	40.8
VSAMT	95.92	30.86	3.79	86763	1878	29.9

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAMOR	57.43	9.29	1.27	86763	1588	35.7
ISAMOR	110.69	12.01	3.83	86763	4298	39.3
ISAMO	89.10	17.22	4.34	86763	1588	36.7
ISAMO	125.44	21.33	9.66	86763	4298	39.0
VSAMO	74.98	30.76	3.08	86763	428	30.6
VSAMO	80.33	31.44	3.09	86763	428	35.7

RESULTS - FULL TRACK

RUN	STEP	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1	SW	203.90	36.58	11.94	86763	7708	53.2
ISAMT	SW	104.38	17.45	4.04	86763	1534	27.1
ISAM1	SR	184.11	32.37	10.78	86763	7232	40.2
ISAMT	SR	83.00	13.07	3.65	86763	1356	34.4
VSAMC	SW	369.68	77.25	10.81	86763	11713	78.2
VSAMT	SW	126.88	50.11	4.11	86763	2631	40.8
VSAMC	SR	222.64	57.57	11.35	86763	10954	41.4
VSAMT	SR	95.92	30.86	3.79	86763	1878	29.9

ISAM1	RR	395.20	54.45	28.29	4520	4520	129.3
ISAMT	RR	425.67	54.56	30.64	4520	4520	157.4
ISAM1	RU	335.44	59.98	25.71	3180	6360	104.8
ISAMT	RU	408.08	58.99	25.72	3180	6360	165.9
ISAM1	RW	880.48	66.33	86.13	2710	26560	233.8
ISAMT	RW	727.24	65.54	56.22	2710	13529	254.5
VSAMC	RR	189.40	32.66	8.49	4520	8999	31.5
VSAMT	RR	483.30	56.02	15.59	4520	18079	100.2
VSAMC	RU	188.63	32.61	8.87	3180	9508	34.0
VSAMT	RU	447.22	50.38	14.22	3180	15900	123.4
VSAMC	RW	215.82	39.68	10.25	2710	11215	41.4
VSAMT	RW	667.06	75.71	20.70	2710	24441	165.6

RESULTS - BUFFERING (OPTIMAL)

- AMOUNT OF PAGING
- AMOUNT OF STORAGE

  'WORKING SET'

  ELAPSED TIME

JOB	WSET	ET
A	50K	5 MIN
B	200K	1 MIN

WHICH TAKES MORE ?

RESULTS - BUFFERING

ISAM - WRITE R+1
- READ 2(R+1)
- DEFAULT BUFNO=5

VSAM - READ = WRITE
- DEFAULT BUFNO=2

```
IF PLHBFRNO<=TWO THEN    /* IF LESS THAN 3 BFRS,             */
  PLHRMIN=PLHBFRNO;      /* OVERLAP UNDESIRABLE.             */
ELSE                     /* SOME OVERLAP DESIRABLE           */
  DO;
    RWORK1=((AMDCINV/AMDLRECL)+EIGHT)/NINE;/* ROUND UP       */
    RWORK3=(PLHBFRNO+ONE)/TWO;/* SET MINIMUM VALUE          */
    IF RWORK1¬=ZERO THEN/* IF NON-SPANNED RECORDS,           */
      PLHRMIN=RWORK3+(((RWORK1-ONE)*(PLHBFRNO-RWORK3))/ RWORK1);
    ELSE                 /* SPANNED RECORDS                  */
      PLHRMIN=RWORK3;    /* USE 1/2 THE BUFFERS              */
    IF AMBSPEED=ON THEN  /* IN SPEED CREATE?                 */
      DO;                /* YES, ADJUST SCHED VAL            */
        /* CODE TO DROP DOWN TO TRACK BOUNDERY */
      END;               /* END OF SPEED CREATE              */
  END;                   /* END OF CALCULATION               */
```

THUS - OPTIMAL RUN TO 4K

RESULTS - BUFFERING

	READ			WRITE		
BUF	ET	CPU	EXCP	ET	CPU	EXCP
2	370.01	88.49	11713	222.79	69.30	10954
10	93.90	42.26	1703	127.22	67.35	2571
20	82.87	40.42	853	127.29	64.39	1720
30	83.71	40.72	640	122.38	62.70	1401
40	81.32	41.30	533	125.66	62.52	1295
50	81.41	41.65	427	138.93	61.31	1082
60	81.45	41.58	321	135.73	61.52	1082
70	82.92	42.72	321	139.80	62.33	1082
80	84.11	44.04	321	139.43	60.65	976
90	86.86	46.02	321	142.67	62.30	976
108	85.89	45.18	215	143.66	62.38	976

RESULTS - WRITECHECK

RUN	STEP	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1	SW	203.90	36.58	11.94	86763	7708	53.2
ISAMW	SW	384.06	41.45	13.05	86763	8094	109.4
ISAM1	RU	335.44	59.98	25.71	3180	6360	104.8
ISAMW	RU	387.17	58.68	25.07	3180	6360	116.3
ISAM1	RW	880.48	66.33	86.13	2710	26560	233.8
ISAMW	RW	1162.80	67.11	87.20	2710	26560	255.6

VSAM1	SW	370.01	77.73	10.76	86763	11713	78.3
VSAMW	SW	471.72	78.60	10.76	86763	11713	223.8
VSAM1	RU	303.27	39.70	11.45	3180	12720	40.7
VSAMW	RU	357.19	39.85	11.34	3180	12720	93.7
VSAM1	RW	473.34	57.79	16.18	2710	19364	60.3
VSAMW	RW	519.74	57.70	16.31	2710	19364	106.3

RESULTS - RANDOM READ

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1	395.20	54.45	28.29	4520	4520*	129.3
ISAMC	263.92	39.18	25.27	4520	9044	79.4
VSAM1	353.71	42.77	11.93	4520	12720	40.5
VSAMC	190.11	32.86	8.65	4520	8999	31.5

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAMOR	242.31	27.46	2.61	4520	4523*	78.4
ISAMO	261.22	37.84	23.64	4520	9043	78.2
VSAMO	197.74	33.30	8.6	4520	9002	36.9

RESULTS - RANDOM UPDATE

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1	335.44	59.98	25.71	3180	6360*	104.8
ISAMC	242.26	47.97	23.18	3180	9542	68.7
VSAM1	303.27	39.70	11.45	3180	12720	40.7
VSAMC	188.63	32.61	8.87	3180	9508	34.0

ISAMOR	223.86	33.90	3.65	3180	6362*	68.3
ISAMO	240.92	46.93	22.28	3180	9542	68.2
VSAMO	194.64	33.02	9.00	3180	9511	46.5

RESULTS - RANDOM INSERT

RUN	ET	TCB	SRB	XACT	EXCP	CHNL
ISAM1 ISAMC	880.48 683.76	66.33 59.82	86.13 78.17	2710 2710	26560 26562	233.8 177.0
ISAM1 ISAMA	880.48 730.21	66.33 74.85	86.13 62.25	2710 2710	26560 7797	233.8 215.8
VSAM1 VSAMC	473.34 219.23	57.79 45.51	16.18 10.29	2710 2710	19364 11215	60.2 39.5
ISAMOR ISAMO VSAMO	512.10 538.73 220.10	45.87 69.19 41.18	13.81 55.35 10.46	2710 2710 2710	8189 13533 10999	158.6 158.2 56.9

VSAM - SPLITTING

CASE	ET	TCB	SRB	XACT	EXCP
NO SPLITTING	.1747	.0213	.0060	2710	7.15
CI SPLITS	.1955	.0312	.0105	500	9.00
CA SPLITS	2.4922	.3722	.1374	23	41.09
CA SPLIT + EXTENT	5.5539	.8692	.2348	23	92.09

NOTES OF USE

- NO V=R VSAM IMPACT
- ISAM EXCP COUNTS
- INDEX SET STRATEGY
- BLOCKING

VSAM/ISAM CONCLUSIONS

BOTH: NOT SELF OPTIMIZING

VSAM SEQUENTIAL DEPENDS

- WRITE USES CPU
- READ IS 'ALRIGHT'
- ET IS GOOD

VSAM RANDOM IS VERY GOOD

- READ, UPDATE, WRITE
- RECOURSE ON SPLITS
- DEGRADATION ON INSERTS

VSAM

(ISAM)

FOIL 1

A performance study was done at the Washington Systems Center comparing ISAM to VSAM through the ISAM Interface Program (IIP). This presentation will discuss the results of those measurements and suggest which options help VSAM or ISAM performance.

FOIL 2

The objective of the study was to quantify the affect on performance of the various VSAM and ISAM options. In order to do this a specific environment was chosen which would permit those changes in performance to be readily measured. A jobstream of single-thread batch COBOL programs written for ISAM provided the environment desired. It allowed the control necessary to isolate the results of each change. The use of COBOL programs was thought to be typical of the customer batch environment.

FOIL 3

The design of the measurement technique included several elements to assure consistent and repeatable results. The tools used to capture performance information were SMF, RMF, and the hardware monitor SMI. The jobstream was run stand-alone single-thread on a S/370 158 model 1.

The base cases were run with all of the above tools as well as a test with a full GTF trace. The GTF trace showed the actual flow of activity which allowed us to understand several unpredicted results.

Isolation was achieved by using single purpose programs running single-thread batch in a stand-alone environment. Only one change at a time was made to the VSAM or ISAM options.

FOIL 4

The COBOL programs used were single purpose as this example shows for the random read case. There was no heavy logic to distort the results. The programs OPENed the file, called SPIKE which; did a GETMAIN for all of storage, touched each page, then did a FREEMAIN, to allow us to measure the actual working set of the program instead of that of VSAM OPEN. The program then accessed the file using a random number generator, and when done, CLOSEd the file. To show successful completion based on an expected path in the program, a user return code of one was set.

FOIL 5

There were basically five programs used.The IIP prefix stands for ISAM Interface Program which was used for all the VSAM runs. The suffix indicates the type of file access being done. The programs were run in the sequence you see here; sequential write (load),sequential read of the entire file, random read, random update, random write (insert). The optimal runs used a combination of options which were found to be beneficial in the isolated runs. In addition, the optimal runs included a second pass of sequential read, random read, random update, and random write programs in order to study the impact of degradation caused by insertion.

Several variants of the base programs were required to implement the ISAM options of: cylinder index in storage, a main storage work area, blocksize, and combinations of the preceeding. It is of note that just to change the blocksize in COBOL required a recompile of all the random processing programs. Thirteen additional programs were created to accomodate this inflexibility.

A special variant was written for VSAM to measure the affects of CA and CI splits.

FOIL 6

The jobname indicates if the run is ISAM or VSAM. The suffix indicates the option being measured for this series of steps. ISAM1 and VSAM1 are the base cases, neither being particularly tuned. They are the starting point for all the options, thus the measured change is from this base. There is no reason to believe that installed ISAM or VSAM users are well tuned.

Encoding of the jobnames is explained here.

ISAMB	A buffered run using a cylinder's worth of buffers.BUFNO=85
VSAMB	A buffered run using a cylinder's worth of buffers.BUFND=109
ISAMC	The APPLY-CORE-INDEX clause in the I-O-CONTROL section of the COBOL program brings the cylinder index into main storage
VSAMC	Emulates cylinder index in storage by providing enough index buffers to hold the index set, not including the sequence set
ISAMT	Is a full track buffering run achieved by specifying DCB BLKSIZE=12800 and changing the number of records per block in the program from 12 to 64. This is to test the hypothesis that big blocks in a virtual environment are better for performance.
VSAMT	Uses a data CISIZE of 12288 bytes to provide a comparable measurement to ISAMT.
ISAMW AND VSAMW	Implements the WRITECHECK option which causes an extra rotational delay to reread the data written and check the ECC for a good compare.It should be noted that this does not read the data into the host nor does it compare the data sent with the data read from DASD.
ISAMD AND VSAMD	Called "DUMMY" runs since they did no file accesses. The program was executed, issued the OPEN, ran through the random number generator, and issued a CLOSE without doing any file access. This was done to measure the basic cost of executing a VSAM program.
ISAMR AND VSAMR	These jobs were run with ADDRSPC=REAL. Could VSAM run real and would it perform better as ISAM does ?
ISAMU	This applies only to ISAM's option during load to use OPTCD=U which tells the system to accumulate and write track index records as a group for each track of the track index. There is no comparable VSAM option.
ISAMA	Implements the TRACK-AREA clause in the FILE-CONTROL section of the program. It causes a full track to be accumulated before any writes are done to the file. This option is not valid for load (QISAM) processing.
VSAMA	It measures the additional cost of a VSAM control area split. Two variations were run: 1) a CA split with no additional extents required and 2) a CA split with additional extents required.
VSAMI	It measures the cost of a CI split 1) not causing a CA split and 2) causing a CA split.
ISAMO and VSAMO	These runs were a selection of options in the previous runs which had improved performance notably.
ISAMOR	This is the ISAM optimal run executing V=R which is of particular benefit to ISAM but, not for VSAM.

FOIL 7

Here are the base case parameters either selected or defaulted for both ISAM and VSAM. You will notice that ISAM under MVS now defaults to 5 buffers, not 2. VSAM defaults to 1 index and 2 data buffers. One of the data buffers is set aside for record inserts. The size of the file is 86,763 records. Free space in VSAM was allocated to avoid splits for the base runs. This caused a larger physical file for VSAM. The VSAM option, IMBED, puts the sequence set portion of the index on the same cylinder as the data (assuming cylinder CA's) it references which is most like the track index structure that ISAM forces on the user.

FOIL 8

The measurement results are depicted by comparing the base run results against each of the several options selected. The optimal ISAM and VSAM runs are then compared for that type of processing. The headings from left to right describe the run name, the total elapsed time, the TCB time, the SRB time, the number of file requests made from the program, the number of EXCP's, and channel seconds. These numbers are taken from SMF except, obviously, the number of program requests.

This first chart shows results for the sequential write or load runs. The first comparison is that of the base with the highly buffered, BUFNO=85, ISAM run. All factors show improvement. As we move on to the full track comparison a further improvement is shown even though large buffering is not used. The OPTCD=U has only slight improvement over the base case.

The VSAM buffered run, BUFND=109, shows improvement over the base case although not as good as the ISAM runs. VSAM full track CI size helps a little, but running VSAM recovery creates a greater CPU burden while using less elapsed time. The buffered run did greatly reduce EXCP's. Depending on your shop's billing routine or largest bottleneck, this may be to your advantage.

From the optimal runs one can see that ISAM running real or virtual is a better sequential performer than VSAM. Thus one should not expect VSAM to be a good load performer.

If the ISAM file load is for reorganization only, we should still consider using VSAM since random and sequential processing do not suffer greatly as the level of insertion increases. If the the ISAM load is due to application design (new records added with merge logic) there will be significantly more CPU time required by VSAM which cannot be eliminated.

FOIL 9

Again for sequential read the buffered and track blocking were the most favorable runs. ISAM track blocking is more efficient than a cylinder's worth of buffers. VSAM did somewhat better reading than loading. The elapsed times are reasonable.

Looking at the optimal runs, ISAM running real is superior. One must consider the scheduling problems in MVS for an address space running real. How many concurrent jobs could one schedule and how would that affect the online systems? If the virtual runs are compared, the second pass of the file after a small number of inserts show a more rapid degradation for ISAM in all factors, especially EXCP's.

NOTE: VSAMO equals ISAMO in CPU time if 3808 records are added to the file. This is 4.39 % insert level. VSAMO equals ISAMOR if 13745 records are inserted, a 15.8 % insert level on the file. (This is extrapolated data.)

This prevents hard conclusions as to the relative performance of VSAM and ISAM sequential processing: each ISAM insert will result in an additional I/O during sequential processing. Pointer logic in ISAM overflow areas prevent any CCW chaining possibilities. This causes rapid performance degradation for ISAM.

VSAM sequential reads are clearly inferior to ISAM sequential reads against a "clean" file, but it is highly probable that applications are not reading "clean" files most of the time.

One of the design objectives of VSAM was to avoid the degradation of inserted records. The number of EXCP's is much lower than any of the ISAM runs. VSAM channel utilization time is lower in most of the runs shown here.

FOIL 10

Here are the results for full track runs. As mentioned before, full track blocking greatly helps performance factors both for ISAM and VSAM. The slower performance of VSAM prompted us to do some further investigation of sequential processing which will be discussed shortly.

The random results for full track blocking show the expected result in most cases. Random processing is looking for one record not a group of records, thus there is extra time required to read a larger DASD block for the DASD device reflected in longer elapsed time, yet there is little affect on CPU time for ISAM. Looking at channel utilization, both ISAM and VSAM are affected by the large data transfer time.

For VSAM both CPU and elapsed time increased. This says that no or small blocking for random processing is a better performer for VSAM. The EXCP's go up as well because of additional index reads.

FOIL 11

In our benchmark design we had to consider the trade-off of additional buffers versus storage and CPU utilization. The results shown in preliminary runs proved that the steps took less time with more buffers, thus tying up storage for a shorter time.

If one looks at the use of extra buffers and their affect on paging and CPU usage, what is the "working set" of this run over what period of time? If job A uses only 50k of storage (fewer data buffers) but takes 5 minutes elapsed time. It has taken the equivalent of 5 times 50 or 250 storage minutes. If JOB B on the other hand, takes 200k "working set" and runs for one minute, it has used 200 storage minutes. Which would you say takes more resource?

FOIL 12

The sequential performance of VSAM led us into a further investigation. Buffering for the two access methods is different. For ISAM the default in MVS is now 5 buffers. Prior to MVS 3.0 it was 2. The default for VSAM is 2 data buffers, one of which is used only for inserted records. How could we tune each? We needed to study the logic of buffer scheduling for each.

ISAM will use R plus one buffers for sequential writes, where R is the number of blocks per track specified. For read two times R plus one is the formula. Since five 2400 byte records fit on a 3330 track in the optimal runs, six buffers for file load and 12 buffers for sequential read performed as well as the full cylinder buffering runs.

VSAM uses the same number of buffers for read as for write. In the VSAM code is the algorithm which follows. It schedules at least half of the data buffers specified and adds to that quantity a factor depending on the number of records per CI.
As a result of this and other measurement runs, the data CI size was changed from 2k to 4k for the optimal runs realizing that we would hurt random performance but help sequential performance. Because of this some "optimal" run times will exceed those of earlier study runs. Please remember this on later foils.
An installation that understands the relative importance of online random performance versus sequential batch performance might not make the same decision.

FOIL 13

Here are some experiments with VSAM sequential buffering. The number of data buffers was increased up to a cylinder's worth, the maximum number that VSAM will chain together and schedule. As the number of buffers increases the number of EXCP's should fall and as you can see, they did. Now as we examine CPU and elapsed time we discover a strange phenomenon. The CPU initially decreases then begins to rise again as does elapsed time. Thus lots of buffers may not be the solution to a performance problem.

Back to the code. For sequential as well as any other processing buffer look-aside is being attempted for each BUFC with every data buffer. This appears to account for the results.

FOIL 14

WRITECHECK measurements had predictable results. The CPU was not affected, but the elapsed time was due to the extra wait time for an additional DASD rotational delay.

FOIL 15

Random results show VSAM's best side. Using in-storage cylinder index ISAM improves nicely. Note the number of EXCP's for ISAM1. It is highly suspect that some I/O's are not being reported to SMF since the number of file requests equals the number of EXCP's. ISAM must at least read the track index before the data. Note that with cylinder index in storage the count is more realistic.

By adding only a few (3) index buffers to VSAM: elapsed time, CPU time, and EXCP's are all reduced. This is a very cheap resource cost for a big benefit in performance.

Inserts did not degrade the random performance of ISAM or VSAM in this study because of their even distribution throughout the file. There was never more than first-in-overflow for the ISAM file. This is not thought to be typical.

The optimal runs show VSAM an elapsed time winner against ISAM running real and a definite winner against ISAM virtual's best options. VSAM channel usage is much more frugal than ISAM.

FOIL 16

Here are the random update runs. The results are very similar to the random read results. The VSAM optimal run CPU total is closer to the ISAM real CPU total. (SRB +TCB) Again EXCP count is off for ISAM. Note the VSAM channel time here as well.

FOIL 17

VSAM is again a very good performer during random inserts, even versus the ISAM optimal running real. It should be noted that the level of inserts is relatively small (3%) and that the random number generator uniformly distributed the records so evenly as to not cause more than first-in-overflow records. Since customers do reorganize their ISAM files frequently, we feel that our insert level is not typical.

FOIL 18

As one adds records to a VSAM file there exists the possibility of creating a CI split which in turn may require a CA split in order to complete. What does that cost? Should splits really be avoided? Our project design avoided splits because we did not know how many splits are "represenative".

The base run was set up to cause no splits. This is felt to be a reasonable approach. We think that in the customer environment which reruns the same file load many times, their experience can help them avoid CI/CA splits.

The next line shows the unit cost of doing only a CI split, ie. there was a free CI in the CA. There is some cost but it is minimal. The next run caused CA splits within existing file extents,ie. the file was defined with extra space. This is a much heavier cost. The worst case is a CA split causing VSAM to go to the catalog to acquire secondary suballocation. This is much worse than any of the previous tests. Notice the number of EXCP's that are used in this environment for a single request.

In summary, CI splits within a CA cost very little additional overhead but, CA splits, especially those requiring additional extents should be avoided. There is feedback available in the catalog and SMF records for a user to determine his split level and adjust the file freespace. These results show you how concerned you should be about each type of split.

FOIL 19

Some miscellaneous comments should be made about the runs.
VSAM running real has no advantage, but it does run real.
ISAM EXCP counts may not be valid so comparisons in the customer shop should note that fact as it applies.
It makes no significant CPU difference if the VSAM data set has 2 or 3 index levels if the index set is kept resident.
This is because of a very fast VSAM index search algorithm.
Blocking is not the answer to performance problems, especially in the online environment (random processing).

FOIL 20

After all these measurements and investigations there are some conclusions one can define. Neither ISAM or VSAM is self-optimizing, ie. defaults are not the best performance options. Design is required. It is hard to come to any other conclusion when runs on the same data produce double or triple run times depending on the options selected.

Sequential performance depends on usage. If the file is loaded frequently, the results for VSAM will be elongated. Read performance will pass but, the buffer look-aside hurts thruput. The elapsed time is good.

VSAM random performance is outstanding even when compared to a well-tuned ISAM running virtual=real. CI/CA splits are not desirable but, can be tuned to take the least costly option in terms of CPU and EXCP's. ISAM has increasing and rapid degradation as the level of inserts increases for all types of processing. VSAM was specifically designed to avoid this degradation.

FOIL 21

This study has shown many definite performance advantages of VSAM over ISAM even though there has been no discussion of functional capability differences. For an online response oriented system VSAM is the answer.

READER'S COMMENTS

Title: VSAM Performance Study Foil Presentation with Text
Washington Systems Center
Technical Bulletin GG22-9022-00

Please state your occupation: ________________________________

Comments:

Please mail to: T. R. Mitchell
IBM Corporation
18100 Frederick Pike
Gaithersburg, Md. 20760