B73

The Top 10 IMS TM Performance Questions (and Answers?)

Dave Viguers

dviguers@us.ibm.com

IMS

Technical Conference

Sept. 27-30, 2004

Orlando, FL

RETURN TO INDEX

Session Description

- This session will explore the top 10 performance issues with the IMS Transaction Manager which have been observed over the past year or so and discuss how they show up, were diagnosed, and finally resolved. If a code defect or enhancement was involved that change will be discussed otherwise ways to monitor and tune to avoid these problems will be presented.

Topics

- SVSO
- WADS
- CTL CPU
- Tran Classes
- Log Buffers
- Traces
- Fixed Storage Mgr
- SMQ
- Latching
- Checkpoint

Why do SVSO pools have low hit rates?

- Where this might be noticed
  - /DIS POOL FPDB

```
IM2B     FPDB BUFFER POOL:
IM2B      AVAIL =   166  WRITING =     1  PGMUSE =     6  UNFIXED =   427
IM2B     POOLNAME CISIZE PBUF  SBUF   MAX CURRENT LK HITS VALID
IM2B     ITEMPOOL 04096  00200 00100 00500 00400  Y   045% 100%
IM2B     DISTPOOL 01024  00100 00025 00300 00100  Y   049% 091%
IM2B     WAREPOOL 00512  00020 00005 00030 00020  Y   100% 099%
```

  - DBFULTA0

```
SUMMARY OF VSO ACTIVITY

 SHR(2/3)     CF       CF  READ READ  DASD     DASD
  AREA       GETS     PUTS HIT  VALID GETS     PUTS
________ ________ ________ ___  ___ ________ ________

AREAWH01     1711     4310 047% 064%        0        1
AREAIT01    16453      502 019% 099%      251        0
AREADI01     3359     9674 024% 068%        0      200
```

  - IMSPA
    - FP Reports

What to do?

- Determine if it is a problem for you
  - Depends on amount of activity
- PBUF (and/or SBUF) just too small for number of CI's
  - Check number of CI's against PBUF/SBUF
  - Increase if necessary

```
/DIS FPV
AREANAME STRUCTURE         ENTRIES CHANGED AREA CI# POOLNAME OPTIONS
AREAWH01 IM0B_AREAWH01A    0000075 0000020 00000075 WAREPOOL PREO, PREL
AREAWH01 IM0B_AREAWH01B    0000075 0000020 00000075 WAREPOOL PREO, PREL
AREAIT01 IM0B_AREAIT01A    0002474 0000000 00002520 ITEMPOOL PREO, PREL
AREAIT01 IM0B_AREAIT01B    0002474 0000000 00002520 ITEMPOOL PREO, PREL
AREADI01 IM0B_AREADI01A    0000440 0000200 00000440 DISTPOOL PREO, PREL
AREADI01 IM0B_AREADI01B    0000440 0000200 00000440 DISTPOOL PREO, PREL
```

- PROCOPT=GOx
  - CI's are read from CF directly into private buffers
    - Does not populate the DEDB= pools
  - Possible bypass is to make the PBUF large enough and read with PROCOPT=G to populate pool

Why have my WADS rates increased

- Where rate is measured
  - RMF DASD report

STORAGE GROUP	DEV NUM	DEVICE TYPE	VOLUME SERIAL	LCU	ACTIVITY RATE	RESP TIME	IOSQ TIME	DPB DLY	CUB DLY	DB DLY	PEND TIME
	2083	3390	WAD001	0077	**27.169**	1	0	0.0	0.0	0.0	0.2

  - IMS logger statistics
    - Reported by IMSPA, Omegamon, IMF, etc.

LOGGER STATISTICS	#	#/SEC
LOG BLOCK SIZE	26,624	
NUMBER OF BUFFERS	10	
WADS TRACK GROUPS	4	
NUMBER OF RECORDS	54,447	465.35
CHECK WRITE REQUESTS	**5,357**	**45.78**
WAIT WRITE REQUESTS	497	4.24
WAIT 4 BUFF CKPT	0	.00
WAIT 4 BUFF NON-CKPT	0	.00
AWE SUBMITTED ON WRT	1,632	13.94
WADS EXCPVRS	**3,034**	**25.93**
2K SEGMENT WRITES	**7,735**	**66.11**
OLDS WRITES	400	3.41
OLDS READS	0	.00
INTERNAL CHKW REQ	13	.11
ACCUM WAIT TIME	30,782	263.09

What to do?

- Probably Nothing
  - Most common cause is new hardware
    - Normally 2K segment writes will decrease
    - WADS writes are finishing faster therefore less data per write
    - Should improve system thruput (at least internally)
  - If using dual WADs
    - Make sure both are on new hardware
      - Only as fast as the slowest
- If not new hardware
  - Other config changes
    - Additional MSC or ISC?
    - SMQ
    - Input/output changes

Why is my CTL CPU usage so high?

- Where noticed
  - Work just backing up
  - RMF workload activity report

```
SUBSYS = STC          TRXCLASS =              ACCTINFO = NO
 USERID =               TRXNAME = IMS1IMSC    SRVCLASS =
00701 ALL ALL ALL  AVG      0.99   ACTUAL             0  SSCHRT  30.9  IOC    139.1K  ABSRPTN  2,152   SINGLE   0.59   AVERAGE  1,178
                   MPL      0.99   EXECUT             0  RESP     1.5  CPU     1227K  TRX SERV 2,152   BLOCK    2.54
                   ENDED       0   QUEUED             0  CONN     1.2  MSO    32840  TCB       21.7   SHARED   0.00   TOTAL    1,177
                   END/S    0.00   R/S AFF            0  DISC     0.1  SRB    538.0K  SRB       9.5   HSP      0.00   CENTRAL  1,129
                   #SWAPS      0   INELIG             0  Q+PEND   0.2  TOT     1937K  RCT       0.0   HSP MISS 0.00   EXPAND   47.88
                                   CONV               0  IOSQ     0.0  /SEC   2,151  IIT       0.5   EXP SNGL 0.27   SHARED    0.00
                                   STDDEV             0                              HST       0.0   EXP BLK  0.39
                                                                                     APPL%     3.5   EXP SHR  0.00
                                                                                     EX VEL%  31.1
```

  - IMS Dispatcher statistics
    - Reported by IMSPA, Omegamon, IMF, etc.

```
DISPATCHER STATISTICS
NAME #TCBS              TSK CREATE    CREATE/SEC            ITASK DISP      DISP/SEC            ITSK SUSP      SUSP/SEC
          TASK REALTIME PER TCB REAL/#TCB/SEC   IMS BUSY TIME PER TCB BUSY/#TCB/SEC      CPU TIME PER TCB  CPU/#TCB/SEC
LOG      1                3,662        31.29                10,181        87.01                7,098        60.66
                      116,364,187    994,565.70               226,631     1,937.01              180,526     1,542.95
CTL      1               14,473        123.70                23,982        204.97               19,284        164.82
                      116,384,222    994,736.94             2,427,571     20,748.47            1,674,493    14,311.90
```

* note that the values shown in these examples are not high

Some things to check

- Hardware config changes
  - Increase in number of engines
    - More dependents processing more work
    - More demand on CTL region
  - Software changes
    - Network into IMS
    - Exit routines
    - PARDLI option?
  - FP page fix options
    - Most common issue
    - Page fix/free can be very costly with large pools
      - DBFP setting
        - use 1 or 2-3600 if many start/stop regions (BMP's?)
    - DFSFIXxx
      - Blocks=FP or EPST to avoid I/O fix/free

How many tran classes should I use?

- Easier to use and manage just a few
  - Can start more regions with less consideration
  - Sysdef simplified
  - Possibly run with fewer regions
  - Performance can be more variable
- More classes make tuning easier
  - More than 255 possible in V9
    - Don't get too carried away however
  - More difficult to manage
  - Makes PWFI more effective
  - Fewer schedules
- Many other factors
  - #PST's
  - PSB, CSAPSB, PSBW, EPCB, DMBW
  - just to name a few

Class scheduling considerations

- With fewer classes
  - Make use of MAXRGN, PARLIM, PROCLIM
    - Region thrashing may occur otherwise
  - Might run fine until:
    - higher activity or system delay causes queuing
  - Scheduling might get more frequent with higher loads
    - Just the opposite of what is desired
  - Scheduling pools may be affected to a larger degree
- With more classes
  - Less prone to fluctuations in load
  - PWFI more effective
    - Fewer possible tran codes per region
  - More control over resource usage
    - But more user control required
- Bottom line - minimize scheduling

Do I need more log buffers?

- Check the logger statistics

LOGGER STATISTICS	#	#/SEC
LOG BLOCK SIZE	26,624	
NUMBER OF BUFFERS	10	
WADS TRACK GROUPS	4	
NUMBER OF RECORDS	54,447	465.35
CHECK WRITE REQUESTS	5,357	45.78
WAIT WRITE REQUESTS	497	4.24
WAIT 4 BUFF CKPT	**0**	**.00**
WAIT 4 BUFF NON-CKPT	**0**	**.00**
AWE SUBMITTED ON WRT	1,632	13.94
WADS EXCPVRS	3,034	25.93
2K SEGMENT WRITES	7,735	66.11
OLDS WRITES	400	3.41
OLDS READS	0	.00
INTERNAL CHKW REQ	13	.11
ACCUM WAIT TIME	30,782	263.09

- Check the RMF DASD report

STORAGE GROUP	DEV NUM	DEVICE TYPE	VOLUME SERIAL	LCU	ACTIVITY RATE	RESP TIME	IOSQ TIME	DPB DLY	CUB DLY	DB DLY	PEND TIME
	3203	3390	IMS001	0078	65.259	5	0	0.0	0.0	0.0	0.2

Log buffer considerations

- No waits means there are enough
- A few waits are generally tolerable
  - A few during checkpoint may be almost unavoidable
    - How much checkpoint data is written
  - Large numbers during non-checkpoint mean investigate
- Are you getting good DASD response times
  - Check both primary and secondary
    - IMS writes concurrently but waits for both
- What is the log blocksize
  - 26K generally gives you the best buffer utilization
  - 24K (or some multiple of 4K) required for 64 bit support
  - Smaller blocks may require more buffers
- If you increase # of buffers
  - Check WADs track groups
    - Don't necessarily need 1 per buffer
  - More virtual/real storage used

Do I need to run with all traces active?

- Traces being active may significantly aid in problem resolution
  - Sometimes a little overhead is worth the cost
- Most traces are very little (relatively) overhead
  - When 'ON' in storage versus 'OUT' to data set
  - 'OUT' tends to be more of a volume issue than CPU
  - Make sure to allocate DFSTRAxx data sets
- Some traces can be noticeable overhead however
  - FAST
  - FPTT
  - LATC
  - IRLM
- Some traces only for special purposes
  - Can cause significant volume to log
  - Use only when necessary
  - Traces like
    - LINE, LINK, NODE, PSB, PROGRAM, TRANSACTION

What about the DFSSPM member?

- First check the statistics
  - Each DFSSPM pool will have statistics like this

```
FIXED STORAGE POOL:  EMHB
CURRENT POOL SIZE :      429152
MAX POOL SIZE     :      462168
# BYTE IN OVERSIZE:           0
OVERALL POOL SIZE :      462168
SUBPOOL NUMBER    :         231
ABOVE/BELOW 16M   :       ABOVE
BUFFERSET                   01          02          03          04          05          06          07          08
OVERSIZE
BUFFER SIZE (BYTES)         264         520        1032        2056        4104        8200       16392       32776
PRIMARY BLK BUFFERS          64          64          32          32          16           8           4           4
INITIAL (Y/N)                 N           N           N           N           N           N           N           N
SECONDARY BLK BUFFS          32          32          16          16           8           4           2           2
MAX BLK SINCE INIT            0           0           0           0          13           0           0           0
MAX BLK SINCE CHKPT           0           0           0           0          13           0           0           0
0
MIN BLK SINCE CHKPT           0           0           0           0          12           0           0           0
0
AVERAGE BLKS ALLOC            0           0           0           0          12           0           0           0
MAX BUF SINCE INIT            0           0           0           0         105           0           0           0
0
MAX BUF SINCE CHKPT           0           0           0           0         104           0           0           0
MIN BUF SINCE CHKPT           0           0           0           0         103           0           0           0
AVERAGE BUFFS ALLOC           0           0           0           0         103           0           0           0
TOTAL GET REQUESTS            0           0           0           0         128           0           0           0
GET REQ PER SECOND          .00         .00         .00         .00        1.09         .00         .00         .00
.00
PGLOAD REQUIRED               0           0           0           0           0           0           0           0
0
PGLOAD-IWAIT REQD             0           0           0           0           0           0           0           0
0
BLK ALLOCATE REQD             0           0           0           0           0           0           0           0
0
BLKS RELEASED                 0           0           0           0           1           0           0           0
0
CURRENT BLOCK COUNT           0           0           0           0        1645           0           0           0
0
CURRENT BUFFER COUNT          0           0           0           0       13200           0           0           0
WASTED STORAGE                0           0           0           0      210176           0           0           0
```

What about the DFSSPM member?

- Normally no overrides are necessary
  - Fixed storage manager is pretty efficient
  - Most buffer sizes are not gotten unless needed
  - Minimizes wasted space
- Check for 'OVERSIZE' usage
  - These requests will be separate getmain/freemain
  - If small numbers then don't worry
  - If significant then consider adding another size to the pool
    - Getmains and Freemains in large numbers are bad
- Compare 'average req size' to buffer size
  - Again, only if large numbers of requests
  - For larger buffer sizes this can reduce storage usage considerably if only 'missing' by a small number

What is the cost of using shared queues?

- Understand the benefits first
  - Not going into that part here
- If anyone quotes you a percentage - it is wrong
  - One number does not fit all
    - For some the % overhead will be small
    - For others it may be huge
- Many variables
  - z/OS CPU speed compared to CF
  - Types of CF links
  - MVS logger configuration
  - Application (dependent region / DLI) cpu consumption
    - Relative to TM
  - IFP's and/or MPP's
    - Relative use of each

Shared Queues estimation

- Best case will be FP tran handled locally
  - Tran processed by IMS on which it arrived
  - Controlled by
    - Available regions
    - Local only, local first, global only options
  - Overhead only incurred when necessary
- Worst case will be Full Function processed globaly
  - Up to 12 CF accesses per tran
  - Includes queue structures plus MVS logger
- Can make rough estimate by
  - Find some other LIST structure already in use
    - Get response time from RMF
  - Multiply that number by 1.25 for system overhead
  - Multiply that result by 12
  - Mulitply that result by trans per second to get approximate additional cpu consumption

Can latch contention be a problem?

- Well of course it can
- Normally the highest latch contention if for the logger
  - Not usually an issue - just a way of life
    - Check buffers, blocksize, dasd response
- Monitor others for significant changes
- Sample latch statistics

```
LATCH STATISTICS: NUMBER IN INTERVAL / PER SECOND
        EXCL GRANT  SHR GRANT EXCL IWAIT  SHR IWAIT EXCL OWAIT  SHR OWAIT  EXCL BUSY   SHR BUSY  EX+SH WAIT TIME
DISP             0          0          0          0          0          0          0          0                0
                .00        .00        .00        .00        .00        .00        .00        .00
CTXT             0          0          0          0          0          0          0          0                0
                .00        .00        .00        .00        .00        .00        .00        .00
```

- Watch scheduling latches
  - TCTB, APSB, PDRB, PSBP, DMBP, PSBB, DMBB, PDRP
  - If conention then cure is to reduce scheduling
  - DMBB contention may compression/expansion routines
    - Not all compression is good but that's a DB issue
  - ACTL is IMS monitor
    - Check blocksize/bufno of monitor

How Can I reduce checkpoint impact?

- Tune the logger
  - All the things we already talked about like blksize, bufno, etc.
- Look for queues of messages not being processed
  - Qpurge must write 'aged' messages to DASD
- Clean up the gen's
  - Remove unused terminals, programs, trans.
- Use ETO if not all sessions are in use concurrently
- Shared queues
  - Spread the terminal blocks across multiple systems
  - Won't help if you clone the same statically genned terminals
- Stagger checkpoints with multiple shared systems
- Watch out for interaction of time driven checkpoints with CPLOG value

Summary

- Tried to hit some recurring questions
- Lots of others not addressed
- Can't possibly cover them all
- Systems getting more and more complex
- Monitor regularly
- Don't be afraid to ask questions

RETURN TO INDEX