



DB2 9 for z/OS Performance Update

Gareth Jones
DB2 for z/OS Development
IBM Silicon Valley Laboratory
jonesgth@uk.ibm.com





Important Disclaimer

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.

WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE.

IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.

NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:

- CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
- ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.



V7 to V8 CPU Time Change

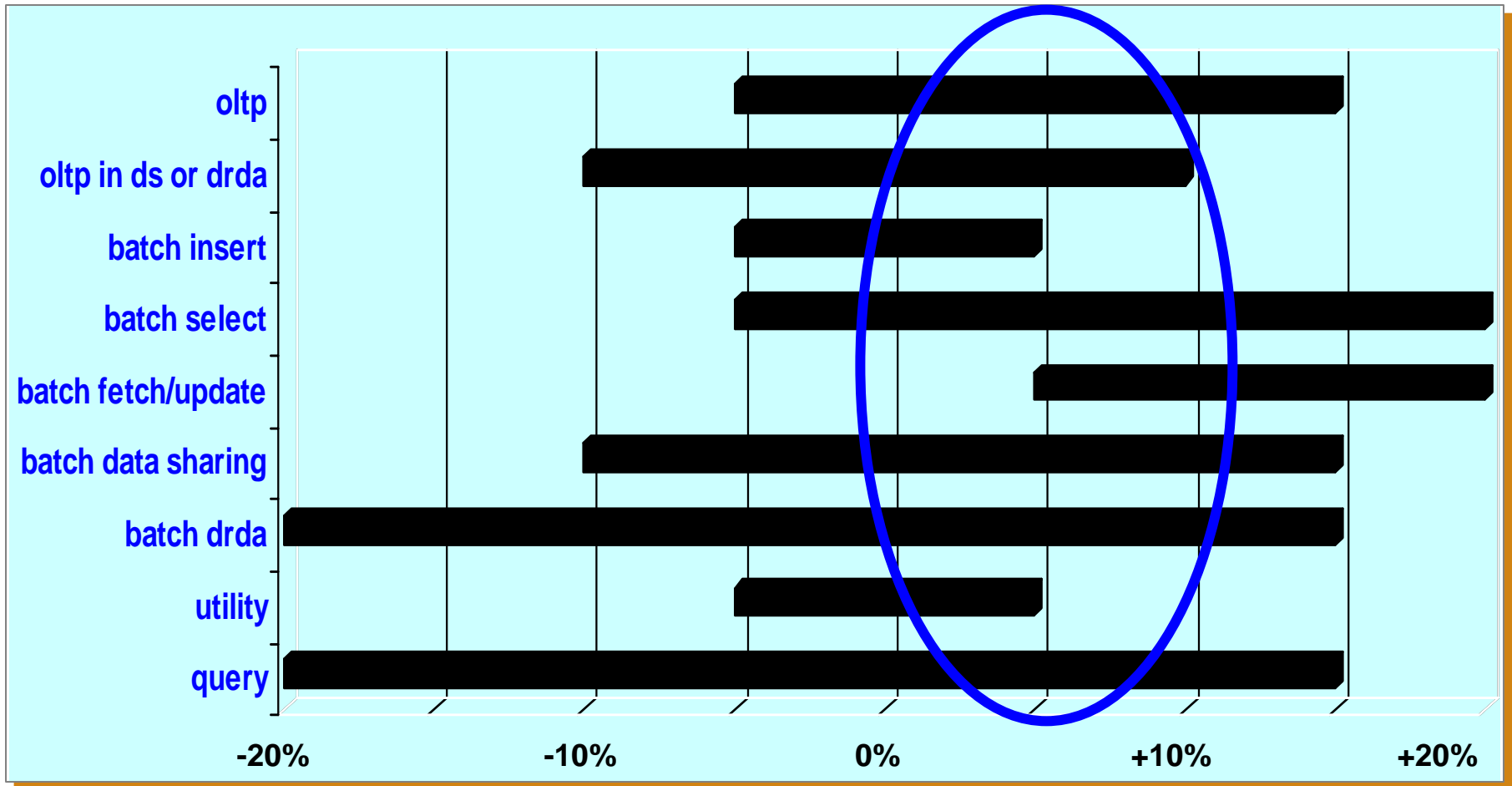
- **Some CPU time increase is inevitable to support a dramatic improvement in user productivity, availability, scalability, portability, family consistency, etc.**
 - Eg DBM1 virtual storage constraint relief, long names, long index keys, longer and more complex SQL statements
- **A heavy investment in minimizing such CPU time increase**
 - Incremental performance improvements to compensate for increased CPU time elsewhere



Improvement to offset CPU time increase to support new functions

- Better access path **(CM)**
- Lock avoidance in overflow rows and SELECT INTO with CURRENTDATA(YES) **(CM)**
- 180 CI limit removal in list prefetch and castout write I/O **(CM)**
- Long term page fix for buffer pools **(CM)**
- SMF89 usage pricing **(CM)**
- CF batching **(CM)**
- Automatic multi-row operation in DRDA **(CM)**
- Backward index scan to avoid sort **(CM)**
- zIIP offload **(CM)**
- Pageset/partition lock contention reduction in data sharing with bind option release resource at commit **(NFM)**
- Automatic multi-row Fetch in DSNTIAUL, DSNTPEP4, QMF **(NFM)**

V7 to V8 CPU Time Change typically observed





Tuning for CPU Usage in V8

- Rebind plans/packages
 - Better access path selection, especially beneficial for complex query
 - Enable SPROC (fast column processing) for 64bit
 - 0 to 10% CPU time impact
 - IFCID 224 trace record for plan/package with disabled sproc
 - Reduce package overhead, especially when small number of short-running SQL calls/package

CPU Time Tuning in Data Sharing

- Group-wide shutdown and restart to reduce global and false contentions for pageset/partition locks when release commit in data sharing (**NFM**)
 - 6% overall CPU reduction for 2way data sharing IRWW



Long-term page fix option for buffer pool (BP) with frequent I/O's (CM)

- DB2 BPs have always been strongly recommended to be backed up 100% by real storage
 - To avoid paging which occurs even if only one buffer short of real storage because of Least-Recently-Used buffer steal algorithm
 - Given 100% real storage, might as well page fix all buffers just once to avoid the cost of page fix and free for each and every I/O
- Up to 8% reduction in overall IRWW transaction cpu time
- New option: ALTER BPOOL(name) PGFIX(YES/NO)
- Recommended for BPs with high buffer i/o intensity = [pages read + pages written]/[number of buffers]
 - Pages read = total number of pages read in a given statistics interval including synchronous read and sequential, dynamic, and list prefetch = QBSTRIO + QBSTSP + QBSTDPP + QBSTLPP
 - Pages written = total number of pages written in a given statistics interval including synchronous write and deferred write = QBSTPWS

Caution on observed CPU time increase in V8

- In general, higher %CPU increase in acctg but
 - lower %CPU increase, or even reduction, in MSTR, DBM1, and IRLM address spaces possible
 - lower %CPU increase overall especially in data sharing or I/O-intensive application
- Example of IRWW



	Non data sharing	Data sharing
Acctg class2	+3%	+11%
MSTR/ DBM1/ IRLM	-15%	-52%
Total	-1%	-11%

- Also, usage pricing improvement in class1, but not class2, acctg



DB2 9 for z/OS Application Performance





V9 SELECT, OPEN/FETCH Performance

- Sort (CM)
- DPSI improvement (CM)
- Cross-query block optimization (CM)
- Pair-wise join in star schema (CM)
- Sparse index or in-memory data extension (CM)
- More parallelism (CM)
- Histogram statistics (NFM)
- Index on expression (NFM)
- Automatic reoptimization (NFM)
- Varchar performance improvement (NFM)
- Concurrency improvement (NFM)



Sort Performance Improvement

- Group By sort and Distinct sort (CM)
 - Group By sort improvement when no column function and no available index
 - Eg `SELECT a1 FROM A GROUP BY a1`
 - Distinct sort improvement when no index or only non unique index available
 - Eg `SELECT DISTINCT a1 FROM A`
 - Up to 2 times improvement



Sort Performance - continued

- Fetch First N Rows with Order By (CM)
 - Example: Get top 10 Americans in income tax paid
 - Avoid tournament tree sort for small N
 - Up to -50% cpu in one measurement test
 - Supported in Subselect also in V9 (NFM)
- In-memory work file for small sorts (CM)
 - 10 to 30% cpu reduction for short-running SQL calls with small sorts
 - Beneficial for online transaction with relatively short-running SQL calls in which the number of rows sorted can be small.



Sort Performance - continued

- Heavier use of 32K work file BP to help large work file record performance (CM)
 - V8 uses 4K BP if less than 4KB row
 - V9 uses 32K BP for more records to gain improved performance, especially I/O time
 - Less work file space and faster I/O, for example 15 2050byte records on one 32K page vs 8 records on 8 4K pages
 - Some measurement example
 - <10% difference for 50 and 85byte records as 4K BP continues to be used
 - 10-50% improvement for 95byte and bigger records because of 32K BP



Access Path Enhancements

- Histogram statistics over a range of column values (NFM)
 - For more precise filter factor estimates and better access path selection
 - Useful in range as well as equal predicates with high cardinality, eg NAME in contrast to STATE
 - Equal-depth (each interval with roughly same number of rows)
- Cross query block optimization in complex query (CM)
 - Optimization across, rather than within, query blocks
 - More predicate transitive closure across query blocks

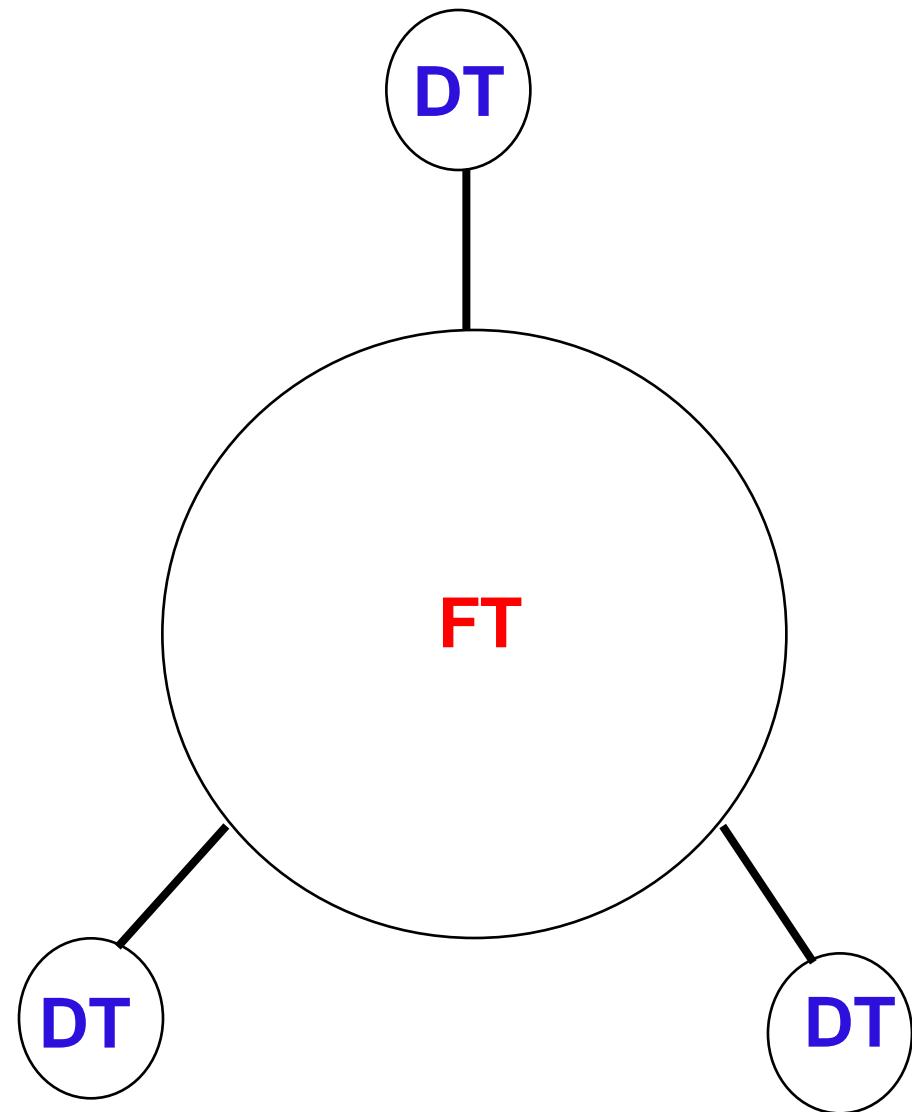


Access path - continued

- Index on expression (NFM)
 - Example: Create Index xxx on Employee(salary+bonus, bonus*100/salary)
 - Orders of magnitude improvement if a predicate using such an index
 - Extra cost in Load, Insert, Update on key value, Rebuild Index, Check Index, and Reorg tablespace but not Reorg index as expressions are evaluated in Insert or index rebuild
 - Not eligible for zIIP offload as index expression evaluation done at load or unload rather than build index phase

Access Path - continued

- Pair-wise join in star schema queries (CM)
 - Each dimension table joins with fact table separately
 - Using single column indexes, instead of difficult-to-tune multi-column index(es), in parallel
 - Each pair of candidate ridlists are AND'd together to produce a final result (Dynamic Index Anding)
 - Especially effective when indexes are not well designed and tuned for performance





Access Path - continued

- Pair-wise join in star join queries - continued
 - More parallelism and more zIIP offload
 - For 100 star join query example, 37% average offload in V8 and 62% in V9
 - These figures are very sensitive to queries themselves as well as hardware environment, eg zIIP utilization
 - Optimization at runtime for further performance improvement
 - Ridpool overflow to DASD via work file
 - Supported in star schema only for now
 - Some parallelism even when degree 1



Access Path - continued

- Optimizer cost model update (CM), eg
 - Cluster ratio
 - Parallel access path selection separate from sequential
 - List prefetch based on Cost Model, not cluster ratio
 - Comprehensive Runstats data recommended to keep any performance regression to a minimum
 - Statistics Advisor can help identify the useful statistics to collect
 - For a given query in V8 (CM)
 - For an entire workload in V9 (CM)



Access Path - continued

- REOPT AUTO (NFM)
 - REOPT None, Once, or Always in V8
 - Automatic REOPT based on parameter marker change
 - Only if dynamic statement caching on
- Generalized sparse index or in-memory data for all accesses, not just star join (CM)
- More parallelism leading to more zIIP offload (CM)
 - Especially in star schema queries
 - Access path selection for parallelism separate from sequential
- Plan Stability (PK52522, PK52523)



Data Partitioned Secondary Index (DPSI) Improvement (CM)

- Enhanced page range screening
 - To avoid having to visit all partitions when DPSI is used wherever possible
- More parallelism with DPSI
 - Parallel support for DPSI when DPSI is used as index access to data and ordering is expected
 - Example: `SELECT * FROM T1 ORDER BY C1` with DPSI on C1
- More index lookaside
- Index-only access with DPSI and ORDER BY
- Unique DPSI support when DPSI columns are superset of partitioning columns
 - E.g. partitioning key on c1.c2 and DPSI key on c1.c2.c3



Concurrency

- Default bind option change (NFM)
 - Isolation RR to CS and CURRENTDATA Yes to No
 - For reduced locking CPU and better concurrency

- Skip locked data option in SELECT (NFM)
 - This option skips locked row or page, while Uncommitted Read isolation mode does not
 - For CS or RS, not UR or RR
 - Conditional instead of unconditional lock request issued
 - Instant lock with immediate unlock for each row or page

Insert/Update/Delete Performance

- Scalability
 - Log latch contention
 - Index page P-lock contention
 - Index tree latch contention
- CPU time
 - Index lookaside
 - Table space append option
 - Index usage history
 - XML Insert CPU reduction – PK66218/PK75613





Log-related enhancements

- LC19 Log latch contention relief in data sharing (NFM)
 - Less DB2 spin for TOD clock to generate unique LRSN
 - No longer holds on to log latch while spinning
 - Also data sharing CPU time reduction especially with faster processor
- No Log table space option where appropriate (NFM)
 - No difference in accounting CPU
 - Significant accounting CPU time reduction possible if high log latch contention
 - If log I/O-bound, then good elapsed time reduction
- Archive log to use BSAM, enabling (NFM)
 - I/O striping
 - Compression if Extended Format data set



Index lookaside for additional indexes (CM)

- In V8, for clustering index only in Insert, none for Delete
- In V9, possible for more indexes in both Insert and Delete
- Big reduction in the number of index Getpages possible
 - In one benchmark of heavy insert into a large table with 3 indexes, all in ascending index key sequence,
 - $0+6+6=12$ index Getpages per average insert in V8
 - $0+1+1=2$ in V9



Randomized index key to avoid hot spots (NFM)

- Can be beneficial for data sharing because of index page P-lock contention
- CREATE/ALTER INDEX ... column-name RANDOM, instead of ASC or DESC
- Trade-off between contention relief and additional Getpage, read/write I/O, and lock request
 - Better for indexes resident in buffer pool

Tablespace append option in Insert (CM)

- CREATE/ALTER TABLE ... APPEND YES
- To reduce longer chain of spacemap page search as tablespace keeps getting bigger



Bigger preformatting quantity and trigger ahead (CM)

- From 2 (V8) to 16 (V9) cylinders if >16cyl allocation
- 27% faster Insert with ESS 800 in one measurement
- 47% faster with DS8300 turbo
- Wait for asynchronous preformat shows up in x'09' Lock wait



Index page split reduction

- Bigger index page (NFM)
 - 4K, 8K, 16K, or 32K page
 - Up to 8 times less index split
 - Good for heavy inserts to reduce index splits
 - Especially recommended if high LC6 contention in data sharing
 - 2 forced log writes per split in data sharing
 - Or high LC254 contention in non data sharing shown in IFCID57
- Asymmetric index page split depending on an insert pattern when inserting in the middle of key range (NFM)
 - Instead of 50-50 split
 - Up to 50% reduction in index split
 - 20% class 2 cpu and 31% elapsed time improvement in one data sharing measurement
 - 10% cpu and 18% elapsed time improvement in one non data sharing measurement



Real Time Stats

SYSINDEXSPACESTATS.LASTUSED (NFM)

- Indicates when index used in SELECT/FETCH, searched UPDATE/DELETE, and Referential Integrity check, but not INSERT, LOAD, etc.
- RTS optional in V7/V8, always in V9 (catalog)
 - In-memory statistics transferred to RTS tables every 30 minutes by default
- Useful in getting rid of unnecessary indexes
- PK44579 8/07 to support the use of index in Referential Integrity, Rid list processing, set functions, and XML values index



Very Heavy Insert Performance Measurement Example

- Up to -18% cpu in non data sharing

- Up to -65% cpu in data sharing
 - -3x Getpage and index page P-lock
 - -25x index page P-lock suspension/negotiation



Topics common to most SQL calls

- Native SQL procedure
- LOB
- VARCHAR performance improvement
- Declared global temp table



NOTES

- V9 ALTER TABLE Column Implicitly Hidden
 - Can reduce the column processing cost in SELECT *
 - Columns defined implicitly hidden can be explicitly referenced in Select list



Native SQL Procedure (NFM)

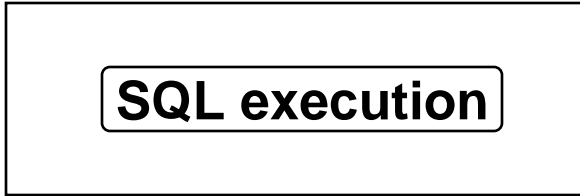
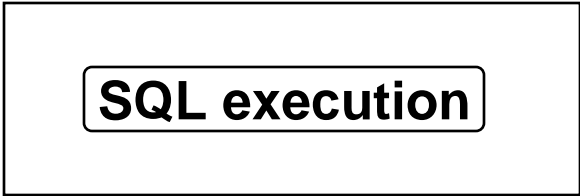
- Avoid the stored proc invocation overhead and roundtrip between Work Load Manager and DBM1 address spaces for each SQL call
 - 0 to 40% ITR improvement compared to external SQL procedure observed
 - No difference if long-running SQL
- Some SELECT replaced by SET statements
- zIIP-eligible if DRDA as it runs in DBM1, not WLM, address space under DDF enclave SRB
- V9 PK45265 8/07 (also V8 PK28046 9/06) to reduce LC24, EDM pool full, and unavailable storage when same external stored procedure or native SQL procedure is called multiple times without an intervening commit or close result sets

External Stored Procedure

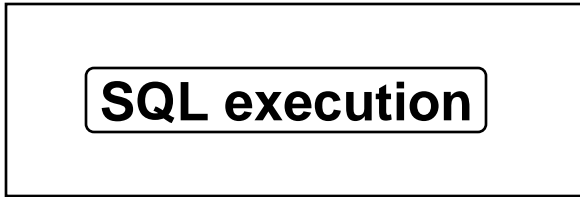
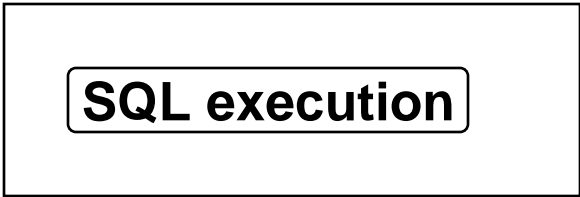
Native SQL procedure

stored proc call

WLM<->DBM1



WLM<->DBM1



Not zIIP-eligible except stored proc call,
result set, and commit processing

zIIP-eligible



LOB Enhancements

- Reorg LOB to reclaim space (CM)
 - In V8, LOB Reorg did not reclaim free space, leading often to a bigger table space as a result of Reorg.
 - In V9, free space is reclaimed. A general recommendation is to Reorg when the free space is bigger than the used space; i.e. `SYSTABLESPACESTATS.SPACE > 2 * DATASIZE / 1024` in Real Time Statistics.
 - Another LOB Reorg indicator is `REORGDISORGLOB / TOTALROWS > 50%` in Real Time Statistics to keep pages of a given LOB together for an efficient read/write performance.
- LOB read/write I/O performance improvement (CM)
 - From doubling of prefetch and deferred write quantity
 - From 8 times increase in preformat quantity
 - Described in System Performance section

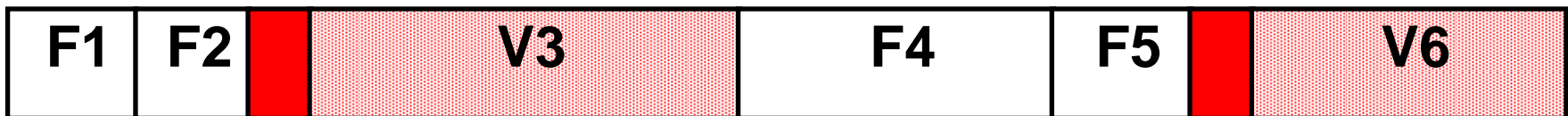


LOB Enhancements - continued

- LOB lock avoidance (NFM)
 - Up to 100% reduction in Lock and Unlock requests in Fetch
 - One measurement with SAP optimized LOB streaming
 - -67% IRLM requests
 - -26% class2 accounting CPU time
 - -14% elapsed time
- LOB/XML flow optimization by size (NFM)
 - V9 DRDA LOB handling instead of SAP optimized LOB handling
 - Additional 11% elapsed time and 2% CPU time improvement
- LOB insert with -14% CPU time (CM)
 - Despite +50% IRLM requests to avoid lock escalation

VARCHAR Performance Improvement (NFM)

- Remember the tuning recommendation for rows with many columns with any varchar present?



- V9 DB2 internally executes this recommendation and more
- V8 and prior: any column after the first varchar requires offset calculation
- V9: all columns directly accessible
- PK78958 disables conversion from BRF to RRF for COMPRESS YES table spaces/partitions
- Open APAR PK78959 allows conversion from/to BRF/RRF via REORG



DGTT – Declared Global Temp Table

- Workfile and temp database are now stored in workfile database (CM)
- PK43765 6/07 to reduce LC24 DGTT prefetch latch contention
 - Prefetch quantity increased from 8 to segsize with 16 default and 64 maximum
- 30 to 60% faster for SELECT COUNT
 - Bigger prefetch quantity and 32K workfile
- 5 to 15% faster and less CPU for INSERT
 - Bigger preformat quantity and asynchronous preformat in V9 but not V8

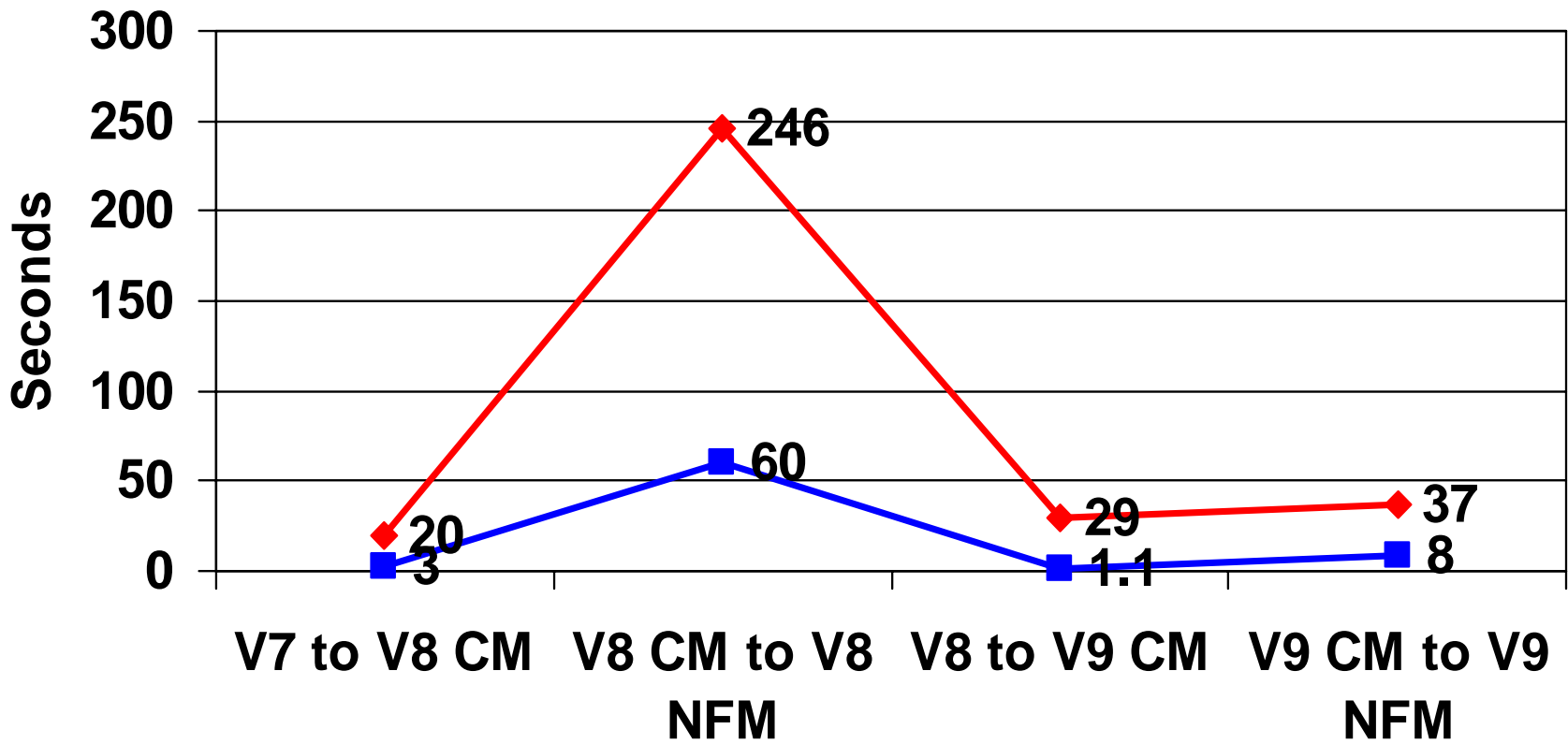


DB2 9 for z/OS System Performance



V7-V8-V9 Catalog Migration – 700MB catalog

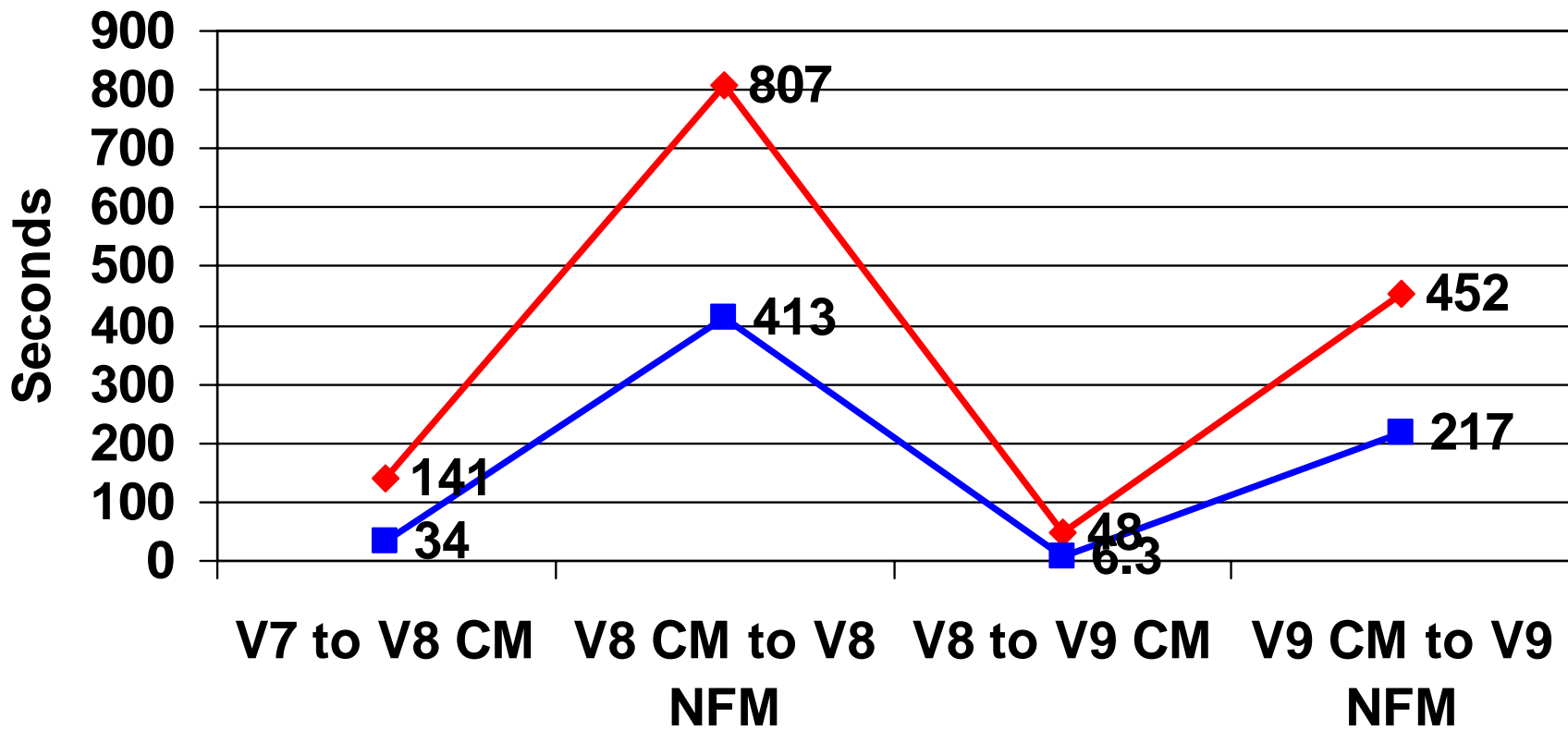
◆ Elapsed Time ■ CPU Time



V7-V8-V9 Catalog Migration

– 15GB catalog

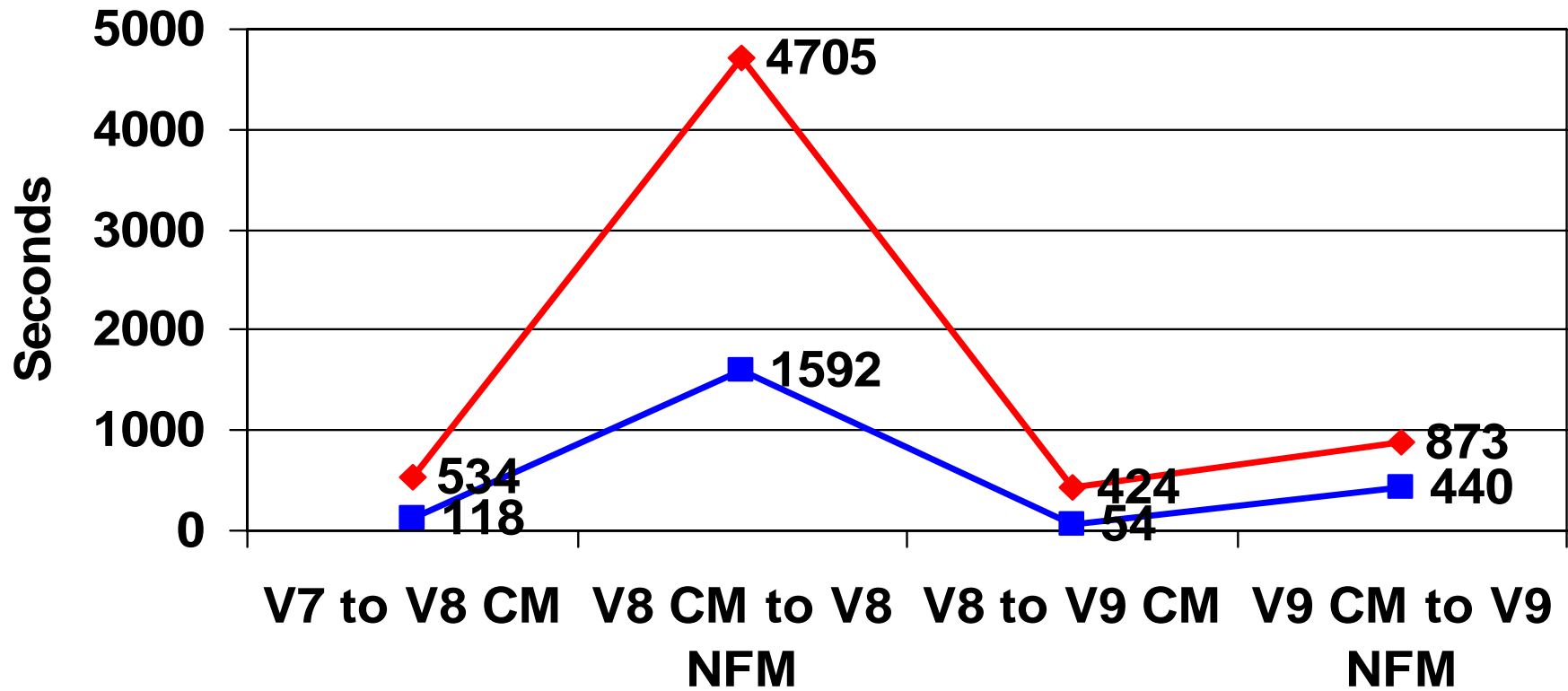
—◆— Elapsed Time —■— CPU Time



V7-V8-V9 Catalog Migration

– 28GB catalog

—◆— Elapsed Time —■— CPU Time



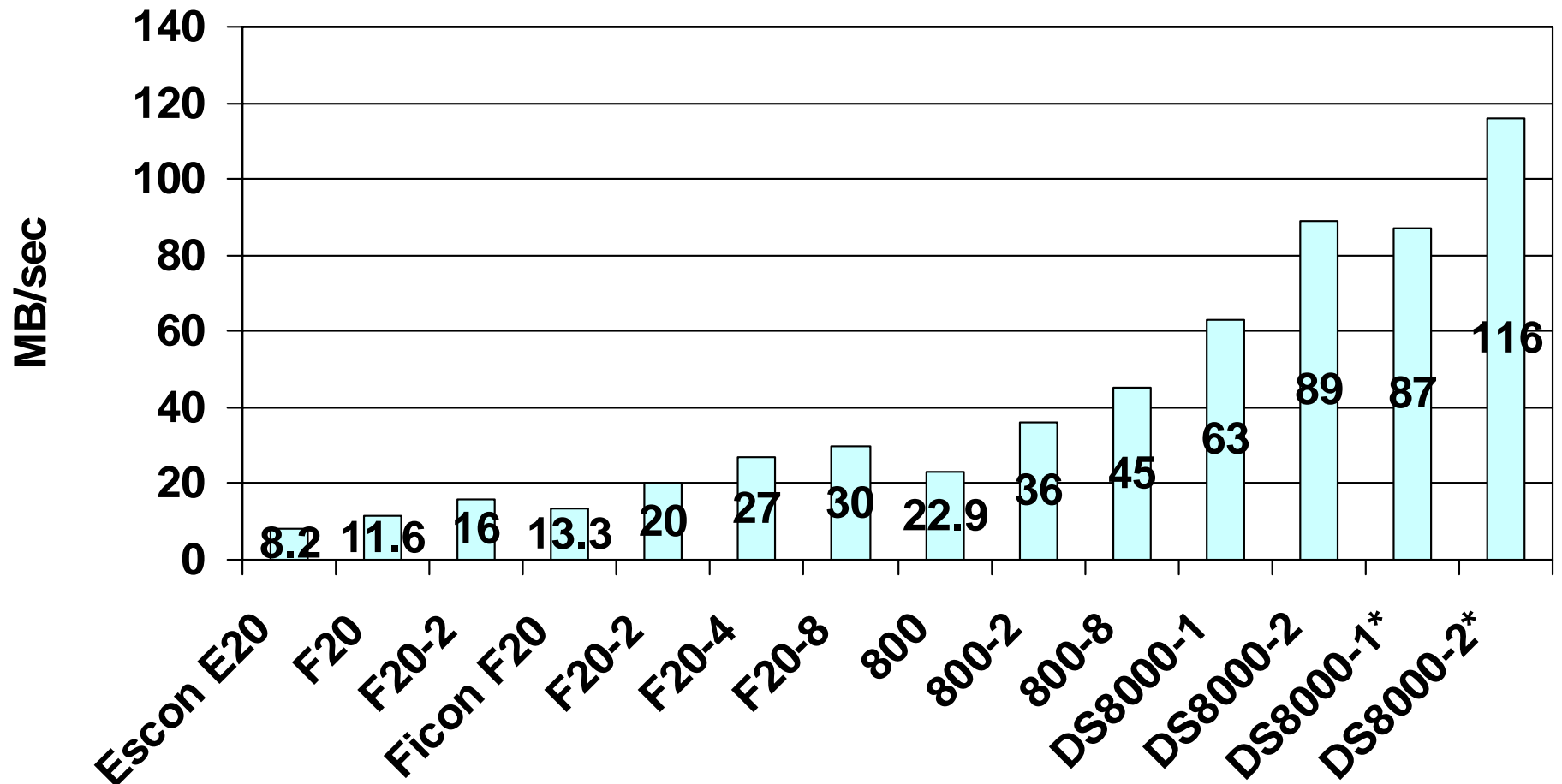


Synergy with new I/O hardware

- DS8000 with Ficon Express and MIDAW (Modified Indirect Data Address Word)
 - MIDAW requires z9 (2094) and z/OS1.6 OA10984 8/05, 13324/13384 9/05
 - Sequential read throughput from cache
 - 40MB/sec on ESS 800
 - 69MB/sec with DS8000
 - 109MB/sec with DS8000 and MIDAW
 - 138MB/sec with 2 stripes
 - Bigger read, write, preformat quantity in DB2 9
 - 183MB/sec in sequential read with 2 stripes
 - Similarly for write
 - Performance gap between EF(Extended Format) and nonEF datasets practically gone

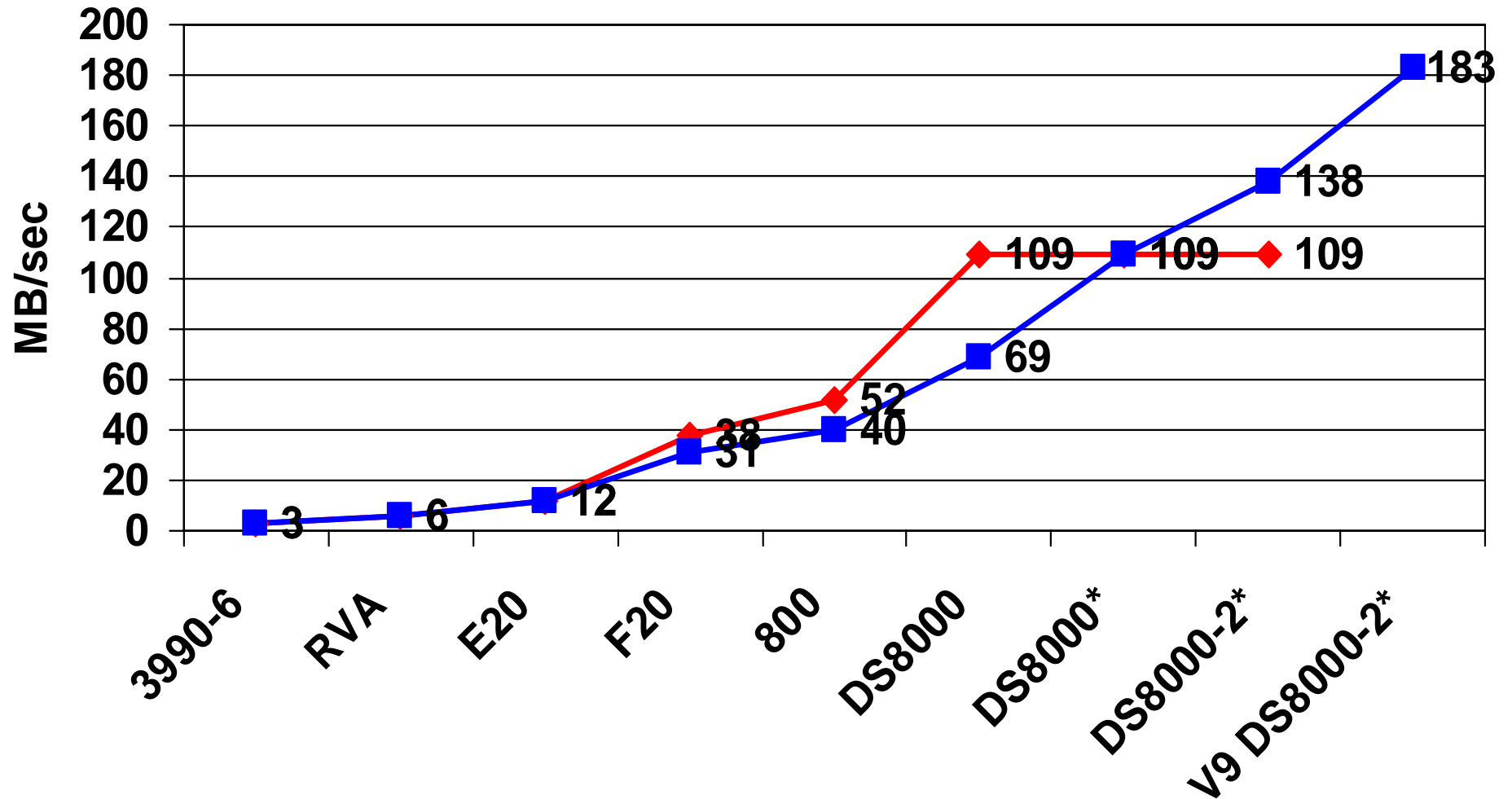
Maximum observed rate of active log write

- First 3 use Escon channel, the rest is Ficon.
- -N indicates N i/o stripes; * MIDAW



MB/sec in sequential prefetch from cache (*MIDAW)

◆ non EF ■ EF





Buffer Manager Performance

- Efficient large BP (>5GB) support
 - Fix RSM UIC update for LRU in z/OS1.8
 - More evenly balanced buffer hash chain
 - Also in V8 PK29626 8/06

- Bigger prefetch and deferred write quantity for bigger buffer pool (CM)
 - Max of 128 V8 ->256KB V9 in SQL tablespace scan
 - 256 V8 ->512KB V9 in utility
 - +36% MB/sec in non striped prefetch
 - +47% in 2-striped prefetch -> more effective striping



Another I/O Time Improvement

- Replace all sequential prefetch, except in tablespace scan, with dynamic prefetch in SQL calls (CM)
 - Dynamic prefetch is more intelligent and robust
 - A big reduction in synchronous read I/O's possible in some cases



Synergy with New CPU Hardware

- Data compression
 - Z900 (2064-1) up to 5 times faster than G6 turbo (9672), instead of normal 1.15 to 1.3 times, in compression and decompression
 - Z990(2084) 1.4 times additional speed up compared to z900 turbo in decompression
 - Z990 1.5 times faster than z900 turbo on average
 - But decompression is $1.5 \times 1.4 = 2.1x$ faster

- Faster Unicode conversion with z900, and more with z990



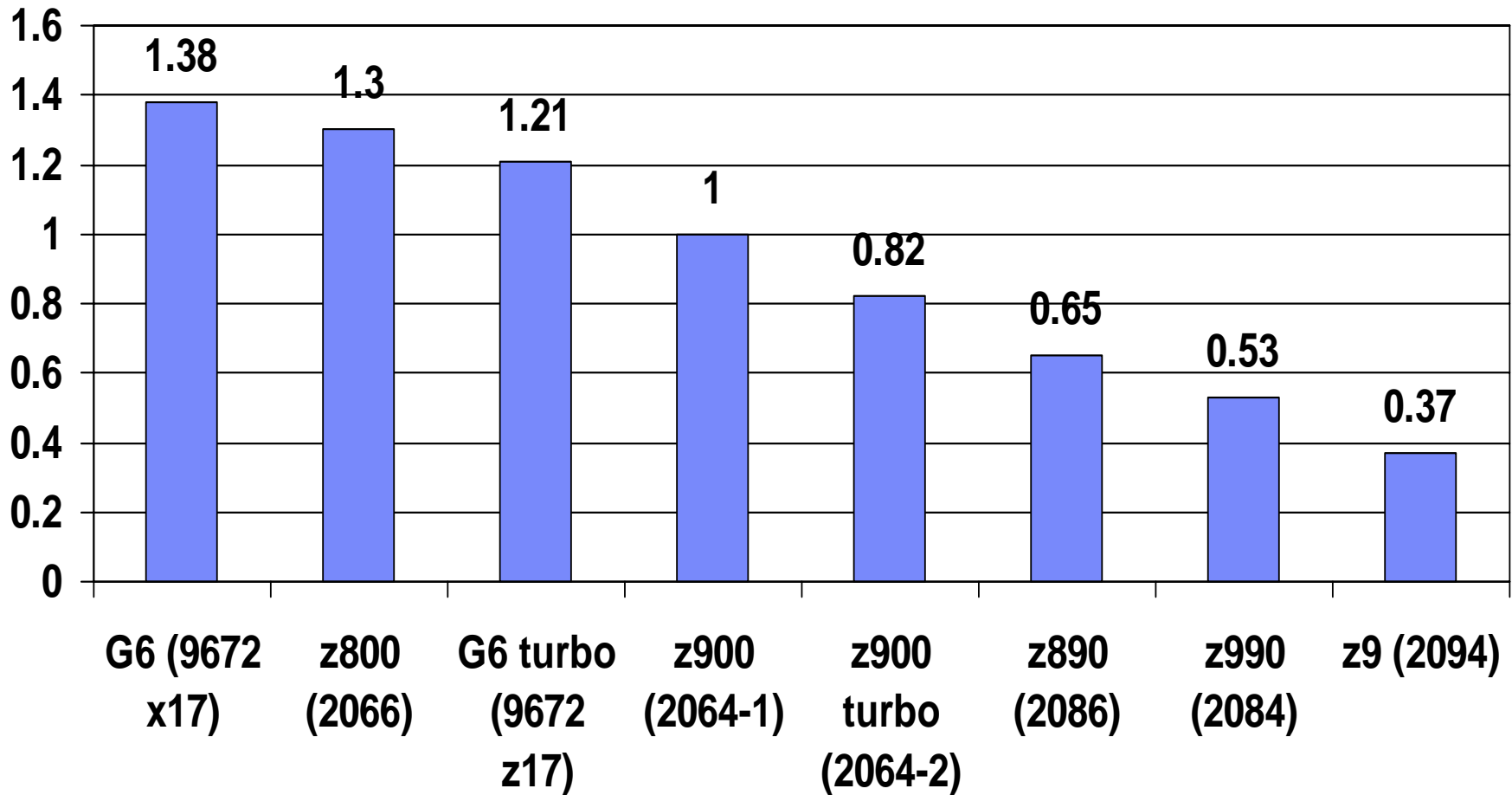
Synergy with new CPU hardware - continued

- z990 (2084)
 - More than 2 times faster row-level encryption
 - V9 long displacement instruction hardware support, simulated by microcode on z900
 - Most impact on input/output column processing
 - V9 cpu vs V8 on z990 or later: -5 to -15% if column-intensive application
 - V9 cpu vs V8 on z900: +5 to 15%, more if many columns

- z9 (2094)
 - MIDAW to improve I/O performance
 - zIIP offload to reduce total cost of ownership

- z10 (2097)
 - DECFLOAT facility

CPU Time Multiplier for various processor models



z9/z10 Integrated Information Processor (zIIP)

- ZIIP intended to reduce the total cost of ownership
- Prerequisites: Current releases of and z/OS, z9 processor or later
- `SYS1.PARMLIB(IEAOPTxx) PROJECTCPU=YES` for projection without zIIP
- Off-loadable enclave SRBs in 4 areas
 - DRDA over TCP/IP
 - Load, Reorg, Rebuild Utility
 - Parallel query
 - XML Parsing

DRDA over TCP/IP

- External stored procedure, user defined function, and SNA are not zIIP-eligible
 - However, stored procedure call, result set, and commit processing that run under enclave SRB are eligible for zIIP redirect
- V9 native SQL procedure is off-loadable under DRDA (NFM)
 - Runs in DBM1, not Workload Manager, address space under enclave SRB

Parallel Query

- For relatively long-running queries, not short
 - E.g. seconds rather than milliseconds of z9 CPU time
 - A portion of the child task processing redirected after certain CPU usage threshold is reached for each parallel group
- More query parallelism leading to more zIIP offload in V9 (CM)
 - Optimized access path under parallelism separate from sequential access
 - #CPs and #zIIPs both considered in parallel degree



XML Offload to zAAP/zIIP

- zIIP (DRDA SRB) - z/OS 1.10 XML System Services enables all XML parsing in enclave SRB mode to be eligible for zIIP.
 - Retrofitted on z/OS V1.8 and V1.9 via APAR OA23828.
- zAAP (System z Application Assist Processor) offload of XML System Services in TCB mode
 - Prerequisites: DB2 9 **NFM**, z9, z/OS 1.10, 1.9 or 1.8 (with PTF OA20308)
 - XML zAAP whitepaper on <http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101088>
- XML Insert/Update/Load offload percentage depends on document size, complexity, number of indexes, etc.

Load, Reorg, Rebuild Utility

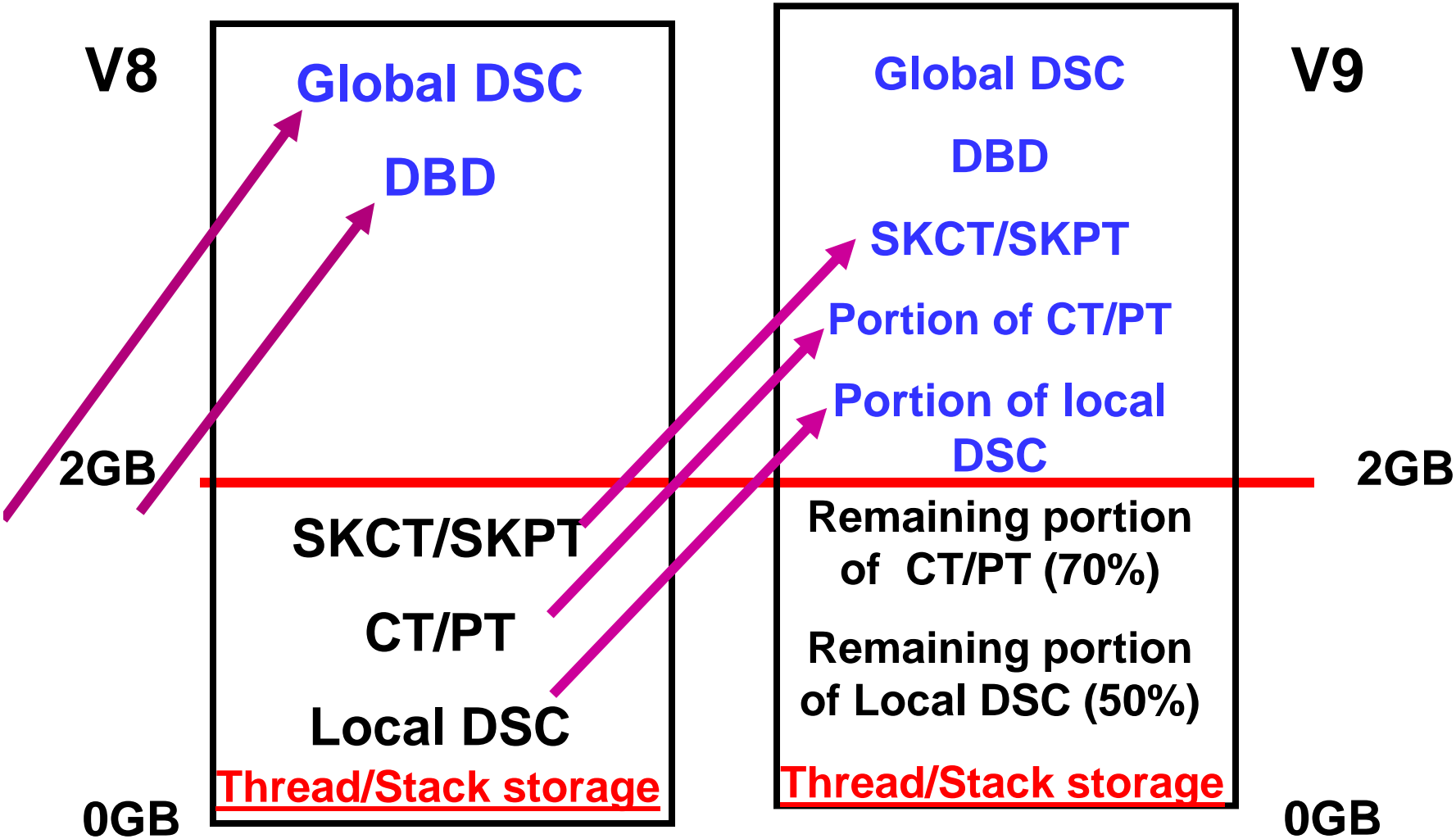
- Example of effective offloaded CPU time with 4 CPs and 2 zIIPs
 - 5 to 20% Rebuild Index
 - 10 to 20% Load/Reorg partition with one index or entire tablespace
 - 40% Rebuild Index logical partition of NPI
 - 40 to 50% Reorg Index
 - 30 to 60% Load/Reorg partition with more than one index
- Higher percentage redirect as the ratio of #zIIPs to #CPs goes up
- For more details on zIIP off-load, please refer to Gopal Krishnan's "Leveraging zIIP with DB2 for z/OS" session 1342 in August 2008 Share conference:
http://ew.share.org/proceedingmod/abstract.cfm?abstract_id=18554



DBM1 Virtual Storage below 2GB (CM)

- 3 major storage areas still below 2GB in V8
 - EDM pool containing SKCT, SKPT, CT, and PT
 - Local dynamic statement cache
 - Thread and stack storage
- V9 EDM pool
 - SKCT/SKPT moved above 2GB
 - A portion of CT/PT moved above 2GB
 - Needs V9 rebind
 - Average estimated reduction of 60% but there is a wide fluctuation from 20 to 90%
 - Rough estimation of V9 EDM pool below 2GB = $2 \times 70\% \times [\text{V8 CT/PT pages used}]$
- V9 Local dynamic statement cache
 - 60% reduction in one measurement, 40% in another
 - Rough estimation of V9 local dynamic statement cache = 50% of V8 local dynamic statement cache
- V9 User thread storage, System thread storage, Stack storage
 - Current expectation of less than 10% difference overall
- Potential reduction in DBM1 virtual storage below 2GB of 0 to 300MB depending on how much thread/stack storage

Summary of DBM1 Virtual Storage in V8 and V9





Virtual and Real Storage - continued

- Real storage – If everything under user control such as buffer/storage pool size, #concurrent threads, etc. is kept constant,
 - 5 to 25% increase in overall real storage from V7 to V8, primarily depending on active buffer pool size
 - Less than 10% from V8 to V9

- DDF virtual storage below 2GB
 - 15 to 40% reduction in V9
 - via shared storage between DDF and DBM1 above 2GB (CM)

One measurement example

Virtual storage below 2GB from DB2 Stats	V8	V9
DBM1 below 2GB used	1091MB	819MB
Local dynamic statement cache	466	172
Thread/stack storage	500	528
EDM	110	110
DBM1 real storage	1935	2203 +14%

Real storage frames from RMF	V8	V9	%delta
DBM1	497K	564K	+14%
DDF	171K	115K	-33%
MSTR	17K	16K	
IRLM	4K	4K	
TOTAL	690K	699K	+1%



Utility CPU time reduction (CM) – primarily from index processing

- 5 to 20% in Recover index, Rebuild Index, Reorg Tablespace/Partition
- 5 to 30% in Load
- 20 to 60% in Check Index
- 35% in Load Partition
- 30 to 50% in Runstats Index
- 40 to 50% in Reorg Index
- Up to 70% in Load Replace Partition with NPIs and dummy input



Reorg Utility Performance (CM)

- Parallel unload/reload by partition
 - 10 to 40% faster in one measurement

- Eliminate Build2 phase in Online Reorg Partition with NPI (Non Partitioning Index) for better availability
 - But higher CPU time and elapsed time when few out of many partitions, especially with more NPIs, are Reorg'd as entire NPIs copied to shadow dataset
 - Additional temporary DASD space needed
 - NPIs are automatically Reorg'd also



Miscellaneous Utility Enhancement (CM)

- Tablespace Image Copy with Checkpage option always
 - Added overhead for Checkpage practically eliminated
 - LRU (Least Recently Used)->MRU (Most Recently Used) buffer steal
 - Protects contents of buffer pool
 - 15% less CPU than V8 with Checkpage option, same as V8 without Checkpage option in one measurement

- Check Index parallelism for sharelevel reference
 - Up to -30% elapsed time with less than +5% cpu



Other Online Utility Improvement (CM) NOTES

- Online Check Index V8 PQ96956 10/05

- Online Rebuild Index
 - Read/Write instead of Read-only access allowed
 - For unique index, update of non indexed columns and all deletes, but not insert, allowed

- Online Check LOB

- Online Check Data



Index Compression (NFM)

Difference between data and index compression

	Data	Index
Level of compression	Row	Page (1)
CPU overhead	In Acctg	In Acctg and/or DBM1 SRB
Comp in DASD	Yes	Yes
Comp in BP and Log	Yes	No
Comp Dictionary	Yes	No (2)
'Typical' Comp Ratio CR	10 to 90%	25 to 75% (3)



Index Compression - continued

- DSN1COMP utility to simulate compression ratio without real index compression
- Work area for compressed index I/O is long term page fixed.
 - So a long term page fix is recommended for
 - Non compressed index buffer pool if significant DASD read/write and/or GBP read/write
 - Compressed index buffer pool also if significant GBP read/write



Dataset Open/Close Performance

- Buffer Manager Open/Close service tasks increased from 20 to 40 to speed up massive dataset open/close (CM)
 - Also only one close at a time while up to 20 concurrent open prior to V8 PK28008 8/06 and PK33496 1/07 which avoids the problem of deferred close not being able to keep up with dataset open resulting in the number of open data sets much greater than DSMAX.
 - Also V8 PK42106 5/07 to limit the number of deferred closes scheduled which causes performance degradation.
 - Up to 40 concurrent dataset open or close in V9
- ACCESS DATABASE(...) SPACENAM(...)
 - MODE(OPEN) to physically open dataset on the local member (CM)
 - MODE(NGBPDEP) to convert to non GBP dependency (CM)

Instrumentation CPU Time Reduction, Enhancements

- Minimize phantom or orphaned trace records (CM)
 - Example from customer’s DB2 V8 statistics report in IFC records per commit
 - (1) **Phantom or orphaned trace** because monitoring (eg vendor tool) stopped but not DB2 trace. The same CPU overhead as real trace.
 - V9 tries to eliminate orphaned trace records
- Capture missing wait time in class3 accounting to reduce NOT ACCOUNTED time (CM)
 - Active log read
 - TCP/IP to transmit the LOB
- Package-level trace filter in Trace Command

IFC DEST	Written	Others (1)
SMF	2	0
OP5	0	4
OP6	0	4
OP7	0	4
OP8	2	0
Others	0	0



General Transaction CPU Usage Trend

- 5 to 10% increase on average from V7 to V8
- No change on average from V8 to V9
 - 0 to 5% improvement for column-intensive transaction, especially with VARCHAR (NFM)

CPU reduction also from

- Long displacement instructions (CM)
- DDF/DBM1 shared storage (CM)
- Partition declaim DSNB1DCM (CM)
- Index access (CM)
- JDBC/CLI commit DSNXECW (CM)



Reference

- Redbooks at www.redbooks.ibm.com
 - DB2 9 for z/OS Performance Topics SG24-7473
 - Updated with new features added via maintenance stream, provides an APAR list including performance sensitive ones.
 - DB2 9 for z/OS Technical Overview SG24-7330

- DB2 for z/OS home page at www.ibm.com/software/db2zos
 - E-support (presentations and papers) at www.ibm.com/software/db2zos/support.html



Old slides (APAR information)



IBM

ON DEMAND BUSINESS

© 2009 IBM Corporation



Additional V9 Performance-Sensitive Apars

- PK41878 Slow query on a partitioned table space that uses Data Partitioned Secondary Index
- ➔ PK42008 Excessive locking on each and every partition even with lock avoidance
- PK41323 Improve performance of implicit Create/Drop of table space
- PK42801 Unnecessary storage request in Insert
- PK42005 Unnecessary calls to DBET checking in read-only Select and Fetch
- PK46972 System hang or deadlock when degree any with DPSI
- ➔ PK41380 Remove serviceability code from DSNIOST2 mainline and put it in error routine
- ➔ PK54988 Reduce excessive CPU time in DSNB1CPS, especially with many partitions



Additional V9 Performance-Sensitive Apars

- PK41899, PK45916 SORTNUM elimination
- PK44026 Enable dynamic prefetch for
 - UNLOAD phase of REORG INDEX
 - UNLOAD and BUILD phases of REORG TABLESPACE PART
 - BUILD phase of LOAD REPLACE PART
 - RUNSTATS phase of RUNSTATS INDEX
- PK42005 7/07 Unnecessary calls to DBET checking in read-only Select and Fetch
- PK46972 8/07 System hang or deadlock when degree any with DPSI
- ➔ PK41380 9/07 Remove serviceability code from DSNIOST2 mainline and put it in error routine
- ➔ PK54988 12/07 Reduce excessive CPU time in DSNB1CPS, especially with many partitions