



Ninety Minutes of Tuning Recommendations for DB2 for z/OS Version 8



Akira Shibamiya
IBM Silicon Valley Laboratory

August 14, 2007
Session 1320

Notes

- **Abstract:** This presentation covers the performance considerations, such as a potential change in the use of CPU time, I/O time, and virtual storage as a DB2 for z/OS subsystem is migrated to V8 compatibility mode and then to new function mode. It also provides performance monitoring and tuning tips on CPU and virtual storage usage of applications migrating to V8.

The new zIIP off-load feature is also described.

- **Speaker:** Akira Shibamiya, IBM Silicon Valley Laboratory, shibamiy@us.ibm.com

Acknowledgment and Disclaimer

- Measurement data included in this presentation are obtained by the members of the DB2 performance department at the IBM Silicon Valley Laboratory.
- The materials in this presentation are subject to
 - Enhancements at some future date,
 - A new release of DB2, or
 - A Programming Temporary Fix
- The information contained in this presentation has not been submitted to any formal IBM review and is distributed on an “As Is” basis without any warranty either expressed or implied. The use of this information is a customer responsibility.

For applications not taking advantage of V8 performance features

- I/O time
 - No change for 4K page I/O
 - Significant sequential I/O time improvement possible for 8K, 16K, or 32K page because of bigger Vsam Control Interval size (NFM)
 - Up to 70% i/o data rate (MB/sec) improvement
 - Also Vsam i/o striping now supported
- Some CPU time increase is expected in order to support a dramatic improvement in user productivity, availability, scalability, portability, family consistency, ...
 - DBM1 virtual storage constraint relief with 64bit instructions
 - Long names, long index keys
 - Longer and more complex SQL statements
- Incremental performance improvement to offset the increase

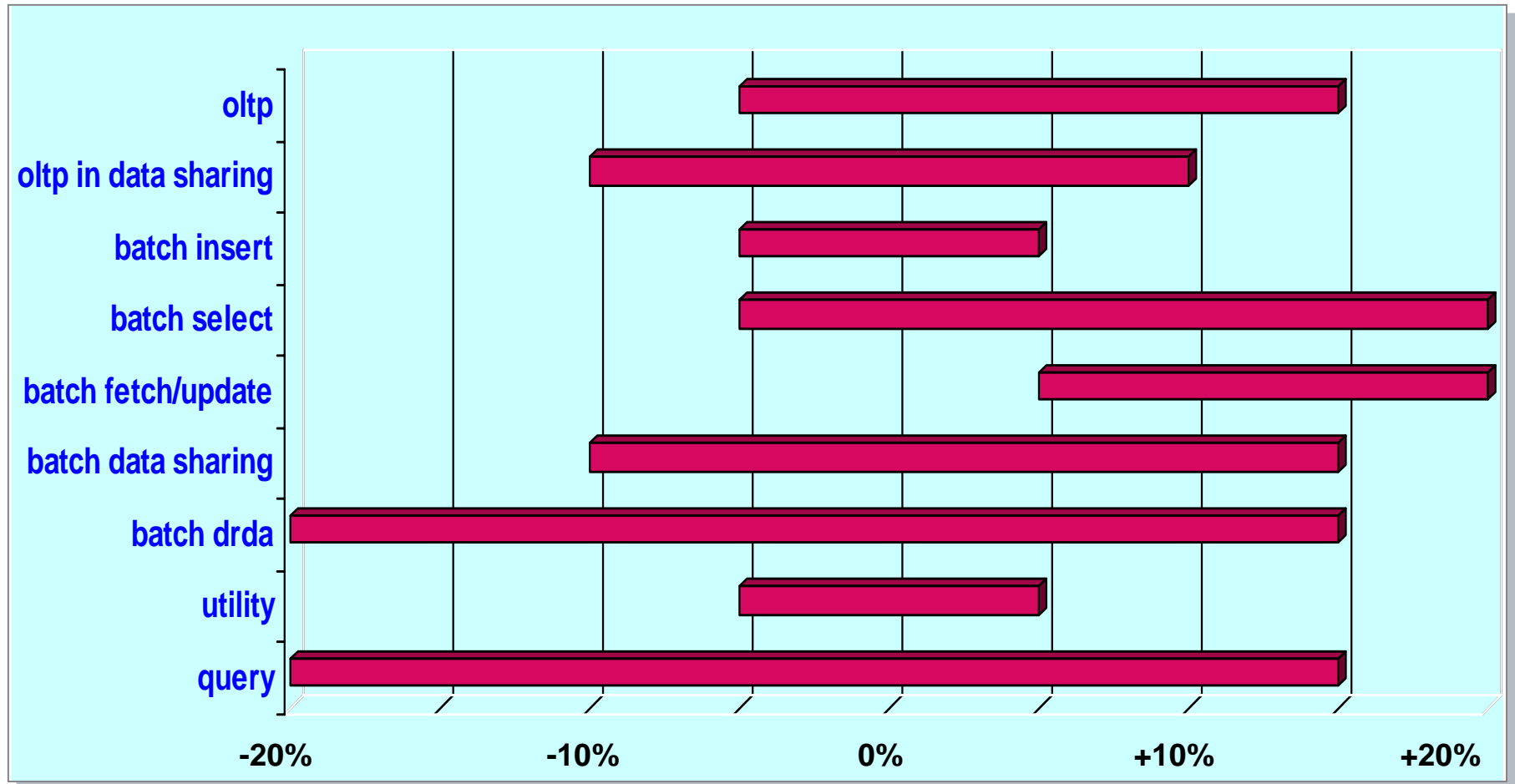
CPU change based on laboratory measurements

with no application nor aggressive configuration/environment change

- **'+' means cpu increase, '-' means reduction, compared to V7**
- **-5 to +15% online transaction**
- **-10 to +10% online transaction in data sharing (NFM)**
- **-5 to +20% batch**
 - **-5 to +5% insert**
 - **-5 to +20% select**
 - **+5 to 20% fetch, update**
- **-10 to +15% batch data sharing**
- **-20 to +15% batch DRDA**
- **-5 to +5% utility**
- **-20 to +15% query**



CPU change - continued



CM (Compatibility Mode) versus NFM (New Function Mode)

- ➔ Typically, no significant performance difference between CM and NFM except in data sharing for workloads with**
- No application change**
 - No aggressive configuration/environment change**

Example of What CM Supports

- **Most access path selection enhancements**
 - **Mismatched data type made indexable V8 PK12389
12/05**
- **DBM1 virtual storage constraint relief**
- **64bit IRLM**
- **Lock avoidance in Select Into with CurrentData
Yes and overflow rows**
- **180 CI limit removal in list prefetch and castout
I/O**
- **Long-term page fix option by buffer pool**

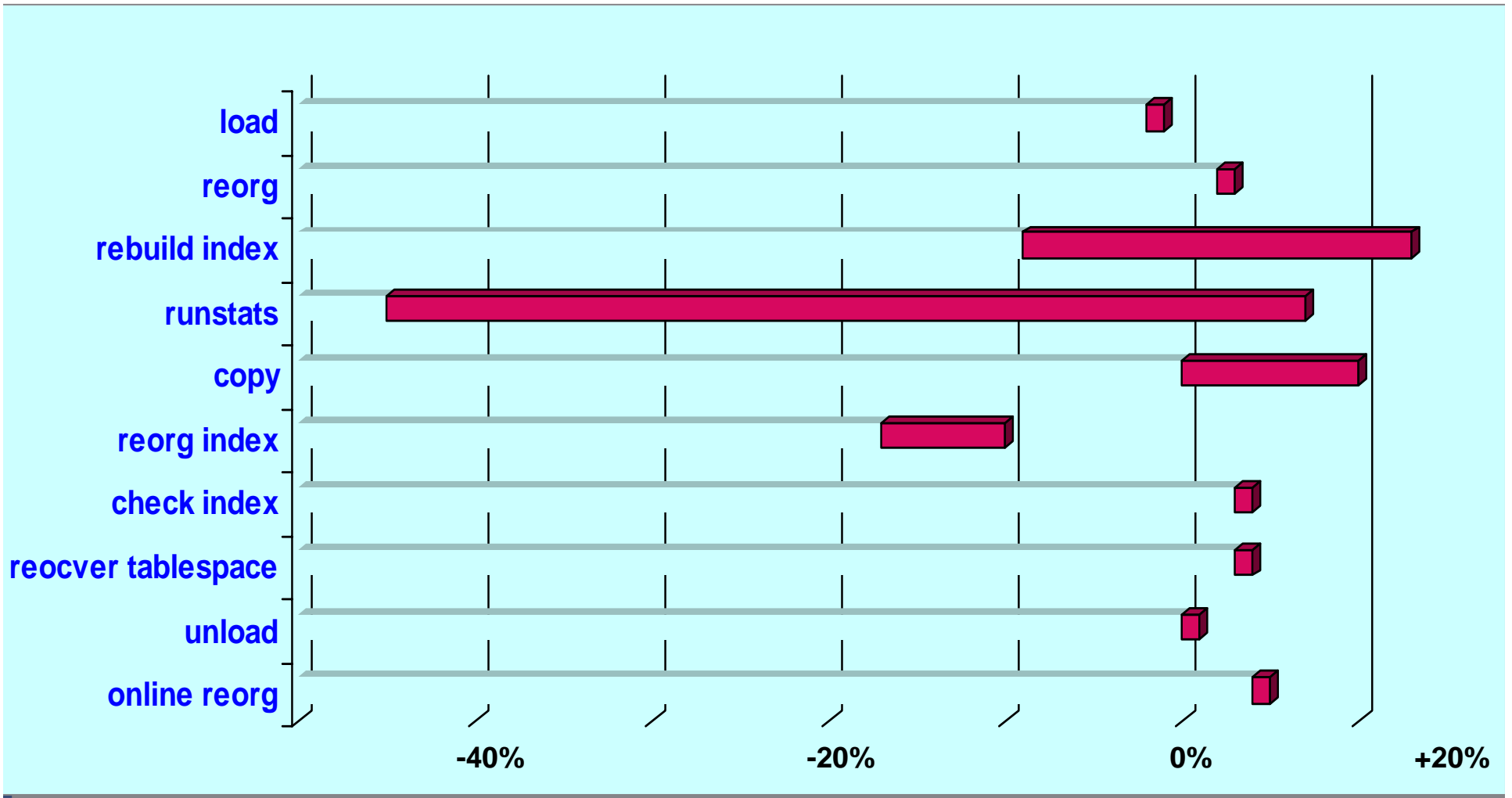
Example of What CM Supports - continued

- **SMF 89 performance enhancement (usage pricing)**
- **Data sharing immediate write default change**
- **CF batching**
- **Implicit multi-row operation in DRDA**
- **Backward index scan to avoid sort**
- **Most utility enhancements**
- **zIIP**

Example of What NFM, but not CM, Supports

- Improved package performance PK28637 9/06
- Pageset/partition lock contention reduction in data sharing with release commit bind option
- Bigger Vsam Control Interval size for bigger page
- Automatic multi-row fetch in
 - DSNTIAUL
 - DSNTPE4
 - QMF
- In-memory workfile in star join
- Work file materialization for rows in UDF
- New application functions such as MQT

Utility CPU Time Change



NOTES

- **Recent performance-related apars to help utilities**
 - **V8 PK18059 3/06 Reorg with Sortdata No option to avoid a large requirement for DASD space**
 - **V8 PK19373 3/06 Better estimate for DFSORT MAINSIZE instead of 2MB**
 - **V7 PK14477 4/06 Allow more parallel tasks in index build for Load and Reorg for large databases needing SORTNUM 255 which always forced 1 task.**
 - **PK36329 5/07 for Rebuild Index.**
 - **V8 PK25742 6/06 Mapping table index build performance improvement in Online Reorg**
 - **II14047 Informational apar on use of DFSORT by DB2 V8 utilities**
 - **V7 PK35390, V8 PK34441 2/07 Online Reorg performance/availability**

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**
 - **V8 PK31412 9/06 #Plan/package with disabled sproc**
 - **Reduce package overhead, especially when small number of short-running SQL calls/package**
 - **Take advantage of some ALTERed objects; for example**
 - **Matching index access or index-only after Alter Index Add Column (NFM)**
 - **Index-only after Alter padded to non padded index (NFM)**

NOTES

- ➔ **Automatic CPU performance enhancements or options introduced to compensate for increased CPU time to support new V8 functions are described next.**

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

- 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 = $[\text{pages read} + \text{pages written}] / [\text{number of buffers}]$

One example with 250,000 buffers total (1000MB)

| | #buffers | Pages read or written | Buffer i/o intensity |
|-----------------------------|----------|-----------------------|----------------------|
| BP0 catalog/directory | 5000 | 25000 | 5 #2 |
| BP1 workfile | 25000 | 50000 | 2 #3 |
| BP2 in-memory index or data | 75000 | 250 | 0.003 |
| BP3 other index | 132500 | 250000 | 1.9 #4 |
| BP4 other data | 12500 | 250000 | 20 #1 |

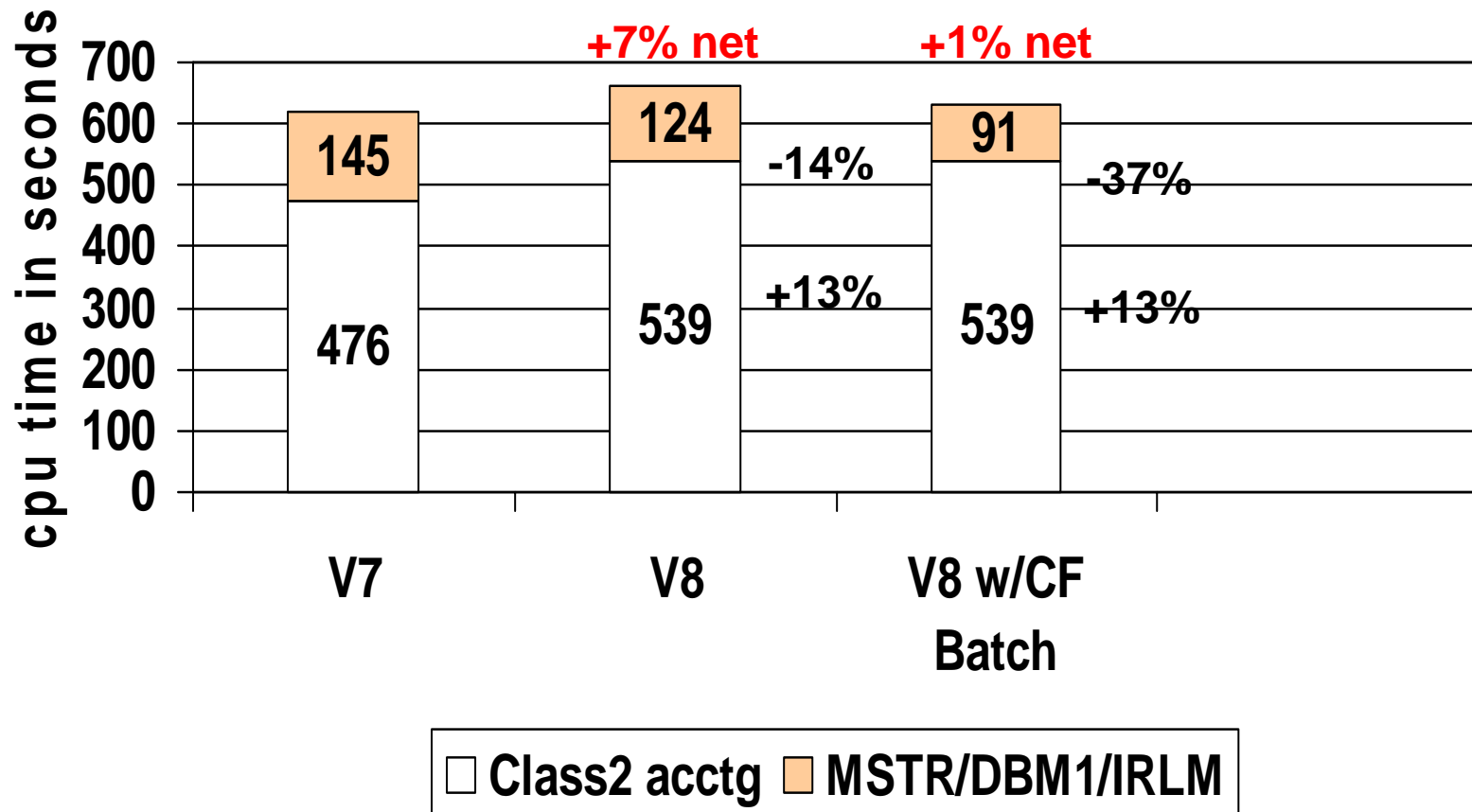
➡ BP4, 0, 1, 3 in that order. Don't bother with BP2.

CPU 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 for 2way data sharing IRWW**
- **CF Batching**

Batching of multiple CF write and castout read requests into one CF access with z/OS1.4 and CF level 12

Impact on Fetch/Update batch transaction



More CPU Tuning - continued

- **If DBM1 virtual storage constrained in V7, recheck various actions taken to reduce virtual storage usage at the cost of additional CPU time, such as**
 - **Reduced size of buffer pools and other pools**
 - **Bind option release commit and/or no thread reuse to reduce thread and EDM storage size**
 - **EDM best fit to reduce EDM pool size**
 - **MINSTOR and CONTSTOR to reduce thread storage size**
 - **Lower DSMAX to reduce storage for data set control blocks and compression dictionary**

NOTES

- **Watch out on CPU increase from**
 - **Non padded index (default with V8 install) with small varchar columns**
 - **Possibly except when index-only access**
 - **Data Partitioned Secondary Index in some cases**
 - **Many column processing**
 - ➔ **Consider Alter to less expensive column type**
 - **V5 Alter Varchar length, but Varchar no longer necessary to alter length**

More CPU Tuning - continued

- **Package-related performance**
 - **V8 PK26879 7/06 High LC32 with a large number of package search lists with wildcard (collid.*)**
 - **Avoid package-level accounting if possible to minimize CPU time and IFC data volume increase**
 - **V8 PK28561 9/06 to reduce this problem**
 - **Acctg class 7, 8, and 10 for detailed package acctg**
 - **V8 PK28637 9/06 (NFM) Package performance improvement including trigger when no hit in EDM pool**

More CPU Tuning - continued

- **If high unsuccessful autho check from cache, make sure at least the default autho cache size used**
 - **CACHEPAC=CACHERAC= 32KB V7, 100KB V8**
- **SMF 89 (usage pricing) data collection change**
 - **Beneficial when many concurrent short-running SQL calls**
- **Further lock avoidance**
 - **Lock in Select Into in CS and Currentdata Yes**
 - **Overflow lock for variable-length or compressed rows**

More CPU Tuning - continued

- **Minimize unnecessary trace**
 - Example from customer's DB2 statistics report in IFC records per commit V7->V8
 - (1) More detailed package accounting in V8
 - (2) Phantom or orphaned trace because monitoring (eg vendor tool) stopped but not DB2 trace. The same CPU overhead as real trace.
 - Display Trace to check

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

More CPU Tuning - continued

- **Remove 180 VSAM Control Interval limit in list prefetch and castout write i/o**
 - **Fewer SIO's, I/O interrupts**
 - **More pages read per list prefetch**
 - **More pages written per castout write I/O**
 - **2 to 3% overall CPU time reduction for IRWW**

- **Automatic multi-row operation in DRDA, DSNTEP4(NFM), DSNTIAUL(NFM), QMF(NFM)**
 - **Up to 50% cpu reduction compared to V7**
 - **PK19191 2/06 to correctly show number of rows sent from Server to Requestor**

Other Recent V8 Regression Apars


- **PK18275 2/06 Multi-column merge join replaced by single column merge join**
- **PK18162 4/06 Increased cpu time from CAF DSNTRACE**
- **PK22611 5/06 Fix extra data sharing locks in Update resulting from PK15056 1/06**
 - **Also avoid unnecessary sequential prefetch scheduling**
 - **0 seq pref count as long as no read I/O**
- **PK26692 9/06 Loop in DSNXECLF for dynamic SQL with many statement sections in a single DBRM resulting in increased short Prepare CPU time**
- **V7/V8 PK29963 10/06 +20% cpu from more Getpage in Sequence processing**
- **PK05360 11/06 No hybrid join with multi-row Fetch**
- **PK40010 3/07 EDM pool full when many repeated stored proc invocations without commit (NFM)**

NOTES

- **Less regression possible if**
 - **IRLM PC YES and LOCKPART YES in V7**
 - **Hiperpool/Dataspace buffer pool used in V7**
 - **SMF 89 (usage pricing) active**
 - **DRDA, DSNTPEP4, DSNTIAUL, QMF which can automatically exploit multi-row operation**
 - **Bind option release commit in data sharing**
 - **I/O-intensive workload**
 - **Long-term page fix and/or 180 CI limit removal help**

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

NOTES

- **Less DBM1 SRB time from**
 - **Long term page fix option especially in prefetch and write i/o**
 - **180 VSAM Control Interval limit removal in list prefetch and castout i/o**
 - **z/OS 1.4 Coupling Facility Level 12 batching of multiple CF write and castout read requests into 1 CF access**
 - **Bigger benefit for Insert/Update/Delete-intensive application**
- **Less MSTR SRB time from**
 - **Default immediate write change from NO(Phase2) to Phase1**
 - **MSTR SRB shifted to Acctg TCB time – no net change**
- **Less IRLM time from**
 - **Reduced global and false contention for pageset/partition locks when release commit in data sharing (NFM)**
 - **Less need for release deallocate bind option**

V8 Virtual Storage Usage in DBM1 Address Space

'typical' V7 below 2GB storage usage shown

EDM DBD pool 10 to 250MB

Buffer pool 0 to 1GB

[Dataspace lookaside buffer 0 to 100MB]

Buffer control blocks 1MB to 300MB

RID pool 4 to 80MB

Castout engine work area 0 to 80MB

Compression dictionary 0 to 500MB

[RDS OP pool 5 to 500MB]

Dyn Stmt Cache control blocks 0 to 200MB

BufMgr/DataMgr Trace Table 10 to 100MB

2GB

2GB

Other EDM pool 20 to 300MB

Local Dynamic Statement Cache 0 to 300MB

Thread and stack storage 50 to 800MB

Notes

- **DBM1 virtual storage constraint relief improves scalability of performance**
 - As the processor power continues to grow, linear scalability, or ability to exploit increasing processor power without encountering a bottleneck which prevents the full CPU usage, becomes more important.
 - Bigger buffer pool and cache to reduce i/o bottleneck and CPU overhead
 - ➡ Without 64bit support, it was difficult to exploit more than 20GB of real storage
 - Up to 32GB z800, 64GB z900, 256GB z990, 512GB z9
 - CPU-Storage trade-off
- **However, thread storage is still below 2GB in V8**
 - Hence, maximum number of threads supported, such as CTHREAD (2000) and MAXDBAT (1999), is not increased.

Estimation of V8 Below 2GB DBM1 Use, based on V7 Stats

- **Average estimates**
 - **Thread storage: +40 to 90% (40% for system, 40 to 90% for user thread)**
 - **Stack storage: +100%**
 - **Dynamic statement cache: +60%**
 - **EDM pool: roughly the same (more if <40% DBD, less if >40%)**
 - **Trace table: -50%**
 - **DSC control block: -70%**
 - **RID pool: -90%**
 - **Others: -100%**
- **Most customers get some to good relief but a small% of customers may get small% increase in DBM1 below 2GB use**

Notes

- For a fair comparison, use the same pool size, # of threads, etc. instead of the defaults which may have changed.
- Bigger thread/stack storage in V8 for
 - Long names, keys, statements, other new functions
 - A portion of RDS op pool for dynamic SQL
- ➡ How much more room in DBM1 address space below 2GB depends on % of storage used for threads, stacks, local DSC, and EDM pool for CT, PT, SKCT, SKPT versus others

| Customer 1 (Europe) | V7 measured | V8 estimated |
|-----------------------------------|--------------------|---------------------|
| Virtual Pool | 0 MB | 0 MB |
| Buffer control block | 78 | 0 |
| Dataspace lookaside buffer | 5 | 0 |
| Castout buffer | 38 | 0 |
| EDM pool | 118 | 118 |
| Compression dictionary | 340 | 0 |
| 1170 system agents | 73 | 103 |
| 25 user threads | 18 | 35 |
| RDS OP pool | 29 | 0 |
| RID pool | 92 | 10 |
| DSC control block | 53 | 16 |
| Trace table | 15 | 7 |
| Stack storage | 49 | 98 |
| TOTAL DBM1 below 2GB | 908MB | 387MB(-57%) |

Notes

- Negligible local dynamic statement cache
- No virtual buffer pool as dataspace buffer pool is used instead.
- Assumes
 - Same number of system agents in V8
 - 90% increase in user thread storage
- ➔ Good DBM1 virtual storage constraint relief for this customer
 - ➔ Even though dataspace buffer pool was used exclusively
 - ➔ Because of large compression dictionary and small thread/stack storage

| Customer 2 (US) | V7 measured | V8 estimated |
|-----------------------------------|--------------------|---------------------|
| Virtual Pool | 98 MB | 0 MB |
| Buffer control block | 13 | 0 |
| Dataspace lookaside buffer | 6 | 0 |
| Castout buffer | 17 | 0 |
| EDM pool | 71 | 71 |
| Compression dictionary | 51 | 0 |
| 837 system agents | 64 | 90 |
| 493 user threads | 291 | 553 |
| RDS OP pool | 420 | 0 |
| Dynamic Statement Cache | 41 | 66 |
| DSC control block | 42 | 13 |
| Trace table | 36 | 18 |
| Stack storage | 143 | 286 |
| TOTAL DBM1 below 2GB | 1293MB | 1097MB(-18%) |

Notes

- Negligible RID pool
- Both virtual buffer pool and dataspace buffer pool used here
- Assumes
 - Same number of system agents in V8
 - 90% increase in user thread storage
- ➔ DBM1 virtual storage constraint relief not as good as the Customer 1 because of large thread/stack storage

Virtual Storage-Related Tuning, if necessary

- Remember CPU and virtual storage trade-off
- Bind option release commit vs deallocate
- If necessary, reduce MAXKEEPD to reduce local DSC and rely more on global DSC which is above 2GB
- CONTSTOR and MINSTOR to reduce thread storage, especially for >1MB thread storage

Virtual Storage-Related Tuning - continued

- **Reduce Common area usage to enable additional private virtual storage availability below 2GB, eg DBM1, DDF, ...**
 - **If V7, use IRLM PC=YES to reduce Extended CSA**
 - **ECSA use by IMS**
 - **ELPA use by WebSphere**
 - **WebSphere V5 and V6 recommend load to private**
- **If V7, use Dataspace buffer pool**
- **Large buffer pool to help DBM1 virtual storage by possibly reducing number of threads**
 - **Database I/O wait time the majority of DB2 elapsed time in many customer workloads**

Recent Virtual Storage Reduction Apars

- **V8 PK21237 5/06 Reset stack for Castout and P-lock/Notify exit engines**
 - Also reduce the number of deferred write engines, castout engines, and GBP write engines from 600 to 300.
 - Up to 100MB reduction in system thread and stack storage possible in a data sharing environment
- **V8 PK25326 6/06 P-lock/Notify exit engine storage reduction via contraction before suspending**

NOTES

■ DBM1

- V8 PK21268 5/06 Current Path storage out of stack and above 2GB
- V8 PK21892 5/06 Stack storage for CICS threads
- V8 PK21861 6/06 Large cached SQL statement pools with infrequent commit and/or high concurrent full prepare

■ DDF

- V8 PK20157 3/06 Running out of DDF VS with LOBs
- V8 PK22442 5/06 and PK23743 8/06 DDF AS storage shortage while processing dist threads with hundreds of output columns causing large SQLDA

Real Storage (RS) Usage

- From V1 R1 in 1985 to the present, real storage usage growing at about 20 to 30% yearly to support performance scalability
 - More and bigger buffer pools, other pools, threads, ...
- V8 continues a similar trend
 - By removing bottlenecks which would have prevented the exploitation of bigger real storage
 - ➔ Rule-of-Thumb: If everything under user control is kept constant, 5 to 25% increase in overall real storage for active BP (buffer pool) between 1 and 10GB
 - Bigger %increase if <1GB BP, large IRLM lock table, and/or many threads
 - Less %increase if >10GB BP
 - Assuming VS/RS PTF maintenance kept current

Notes

- **Example of more real storage usage**
 - Higher default and maximum buffer pool size, RID pool size, sort pool size, EDM pool size, ...
 - Bigger and possibly more threads
 - Bigger modules, control blocks, internal working storage
 - More parallelism enabled
 - Parallel sort for multiple tables in composite
 - Parallel multi-column merge join
 - **Real storage leak**
 - PK19769 3/06, PK33273 1/07 Pool reset to unback pages with real storage
 - PK25427 8/31/06, OA15666 7/06 Correct accounting of real storage usage and z/OS to free real storage

Z9 Integrated Information Processor (zIIP)

- **ZIIP intended to reduce the total cost of ownership**
- **Prereuisites: DB2 for z/OS V8 CM, z/OS 1.6, z9 processor**
- **SYS1.PARMLIB(IEAOPTxx) PROJECTCPU=YES for projection without zIIP**
- **Off-loadable enclave SRBs in 3 areas**
 - **DRDA over TCP/IP**
 - **Parallel query**
 - **Load, Reorg, Rebuild Utility**

DRDA over TCP/IP PK18454 6/06

- **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**
 - **Runs in DBM1, not Workload Manager, address space under enclave SRB**

Tivoli Omegamon DB2PE Accounting Report with CLI SQL DRDA zIIP Redirect

CONNTYPE: DRDA

| <u>AVERAGE</u> | <u>APPL(CL.1)</u> | <u>DB2 (CL.2)</u> | |
|----------------|-------------------|-------------------|--|
| CP CPU TIME | 0.001197 | 0.000751 | Chargeable CPU time including |
| AGENT | 0.001197 | 0.000751 | <u>IIPCP CPU</u> but not <u>IIP CPU TIME</u> |
| PAR.TASKS | 0.000000 | 0.000000 | |
| IIPCP CPU | 0.000000 | N/A | zIIP-eligible work executed on CP |
| IIP CPU TIME | 0.001480 | 0.000911 | zIIP-eligible work executed on zIIP |

→ Redirect% = $1480 / (1197 + 1480) = 55\%$ for this workload

Tivoli Omegamon DB2PE Accounting Report with CLI SQL DRDA zIIP Redirect Estimate

CONNTYPE: DRDA

| <u>AVERAGE</u> | <u>APPL(CL.1)</u> | <u>DB2 (CL.2)</u> | |
|----------------|-------------------|-------------------|--|
| CP CPU TIME | 0.002754 | 0.001726 | Chargeable CPU time including |
| AGENT | 0.002754 | 0.001726 | <u>IIPCP CPU</u> but not <u>IIP CPU TIME</u> |
| PAR.TASKS | 0.000000 | 0.000000 | |
| IIPCP CPU | 0.001534 | N/A | zIIP-eligible work executed on CP |
| IIP CPU TIME | 0.000000 | 0.000000 | zIIP-eligible work executed on zIIP |

 **Estimated Redirect% = $1534/2754 = 55\%$ for this workload**

Parallel Query – PK27578 7/06

- **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 in V9**
 - **Optimized access path under parallelism separate from sequential access**
 - **Some examples of higher zIIP redirect percentage**
 - **37% V8 and 62% V9 offload in 100 star join queries**
 - **76% offload in V8 and V9 in 140 non star join parallel queries**
 - **48% V8 and 60% V9 offload in another 60 non star join parallel queries**

Tivoli Omegamon DB2PE Accounting Report with Local Parallel Query zIIP Redirect

PLANNAME: DSNTEP81

| <u>AVERAGE</u> | <u>APPL(CL.1)</u> | <u>DB2 (CL.2)</u> | |
|----------------|-------------------|-------------------|--|
| CP CPU TIME | 19.374 | 19.366 | Chargeable CPU time including |
| AGENT | 6.779 | 6.771 | <u>IIPCP CPU</u> but not <u>IIP CPU TIME</u> |
| PAR.TASKS | 12.594 | 12.594 | |
| IIPCP CPU | 2.814 | N/A | zIIP-eligible work executed on CP |
| IIP CPU TIME | 35.887 | 35.887 | zIIP-eligible work executed on Ziip |

➔ Total zIIP-eligible work % = $(2.814+35.887)/(19.374+35.887) = 70\%$

➔ Actual Redirect%= $35.887/(19.374+35.887)=65\%$ for this workload

Tivoli Omegamon DB2PE Accounting Report with Local Parallel Query zIIP Redirect Estimate

PLANNAME: DSNTEP81

| <u>AVERAGE</u> | <u>APPL(CL.1)</u> | <u>DB2 (CL.2)</u> | |
|----------------|-------------------|-------------------|--|
| CP CPU TIME | 54.690 | 54.682 | Chargeable CPU time including |
| AGENT | 6.775 | 6.767 | <u>IIPCP CPU</u> but not <u>IIP CPU TIME</u> |
| PAR.TASKS | 47.915 | 47.915 | |
| IIPCP CPU | 38.243 | N/A | zIIP-eligible work executed on CP |
| IIP CPU TIME | 0.000 | 0.000 | zIIP-eligible work executed on zIIP |

 **Estimated Redirect%= $38.243/54.690 = 70\%$ for this workload**

This page is intentionally left blank.

Load, Reorg, Rebuild Utility – PK19920 6/06, PK27712 8/06, PK30087 9/06

- **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**

NOTES

- **Variations in percentage redirect for various utilities are primarily determined by the percentage of CPU time consumed by build index processing, which is redirected, to the total CPU time for a given utility.**
 - **E.g. Partition Load/Reorg spends most of the CPU time in build index phase and consequently is in a position to gain the biggest redirect percentage, especially with more indexes.**
- **Less percentage redirect in Rebuild Index with more indexes because of added cost of sort. This also explains smaller percentage redirect than Reorg Index which does no sort.**

Tivoli Omegamon DB2PE Accounting Report with Rebuild Index Utility zIIP Redirect

PLANNAME: DSNUTIL

| <u>AVERAGE</u> | <u>CLASS1</u> |
|----------------|---------------|
| CP CPU TIME | 64.421 |
| AGENT | 0.017 |
| PAR.TASKS | 64.404 |
| IIPCP CPU | 6.158 |
| IIP CPU TIME | 16.236 |

Chargeable CPU time including
IIPCP CPU but not IIP CPU TIME

zIIP-eligible work executed on CP

zIIP-eligible work executed on zIIP

➔ Total zIIP-eligible work % = $(6.158+16.236)/(64.421+16.236) = 27\%$

➔ Actual Redirect%= $16.236/(64.421+16.236)= 20\%$ for this workload

NOTES

- **Biggest percentage redirect for**
 - **Load/Reorg partition with multiple indexes**
 - **Reorg Index**
- **Less for**
 - **Load/Reorg partition with one index or entire tablespace**
 - **Rebuild Index**
- **Typically less than 10% elapsed time and CPU time overhead for execution unit switch from TCB to enclave SRB during Index Build phase**
- **See Gopal Krishnan's zIIP presentation in Session 1782 of October 2007 IOD conference for more details**

Reference

- V8 manuals, especially Performance Monitoring and Tuning section of Administration Guide
- Redbooks at www.redbooks.ibm.com
 - DB2 UDB for z/OS V8 Technical Preview SG24-6871
 - DB2 UDB for z/OS V8 Everything you ever wanted to know... SG24-6079
 - DB2 UDB for z/OS V8 Performance Topics SG24-6465
 - A Deep Blue View of DB2 Performance: IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS SG24-7224
- More DB2 for z/OS information at www.ibm.com/software/db2zos
 - E-support (presentations and papers) at www.ibm.com/software/db2zos/support.html
- zIIP Reference Information: www.ibm.com/systems/z/ziip