z/VM 6.3: Changes in Memory Management

Session 14686

John Franciscovich
francisj@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

IBM*	System z10*	System z196
IBM Logo*	Tivoli*	System z114
DB2*	z10 BC	System zEC12
Dynamic Infrastructure*	z9*	System zBC12
GDPS*	z/OS*	
HiperSockets	z/VM*	
Parallel Sysplex*	z/VSE	
RACF*	zEnterprise*	
System z*		

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

OpenSolaris, Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
INFINIBAND, InfiniBand Trade Association and the INFINIBAND design marks are trademarks and/or service marks of the INFINIBAND Trade Association.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

All other products may be trademarks or registered trademarks of their respective companies.

Notes:
Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Notice Regarding Specialty Engines (e.g., zIIPs, zAAPs and IFLs):

Any information contained in this document regarding Specialty Engines ("SEs") and SE eligible workloads provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”).

No other workload processing is authorized for execution on an SE.

IBM offers SEs at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Agenda

- Objectives and strategies of the z/VM Large Memory enhancement
- Key features of the z/VM Large Memory enhancement
  - Algorithmic concepts: new, changed, or obsolete
  - Basic flows and data structures
  - Knobs you can twist or set
- Planning for z/VM Large Memory
  - Paging DASD calculations
  - Reminders about best practices with respect to paging I/O
- Workloads
- CP Monitor and z/VM Performance Toolkit
- Summary

Objectives and Strategies

- Objectives:
  - Support 1024 GB aka 1 TB of central memory in a partition
  - Support large guests in such a context
  - Retain ability to overcommit memory

- Strategies:
  - Repair or replace memory management algorithms that
    - do not scale well
    - are grossly unfair

- Specifically:
  - ***Page reorder*** is a real problem area. Get rid of it.
  - ***Demand scan*** has scaling problems and frame ordering problems. Repair them.
    - Introduce a new ***global aging list*** concept to add accuracy to frame reclaim decisions.
    - Improve ***fairness*** of frame steal when memory is constrained.
  - Improve effectiveness of keeping storage specified by **SET RESERVED** in memory.
  - Extend **SET RESERVED** to **DCSS**es such as MONDCSS.

New Algorithms and Behaviors

New Approach: Highlights

- Objective: keep the ***available lists*** populated just right
- New visit heuristic tries to improve occupancy fairness in the face of storage constraint
- In-use frames are tracked by a new hierarchical data structure:
  - Valid, often-touched frames are at the top
  - Demand scan pushes frames downward as they seem to increase in reclaim appeal
  - Best reclaim candidates are at the bottom
- DASD use for paging is changed to be more friendly to reclaim and to storage subsystems
  - Pages valid on DASD are not rewritten
  - Pages are rewritten to the same slots
  - Channel program can do fully discontiguous reads or writes
  - Pages can be pre-written to DASD

New Approach: Management of The Available Lists

Old way

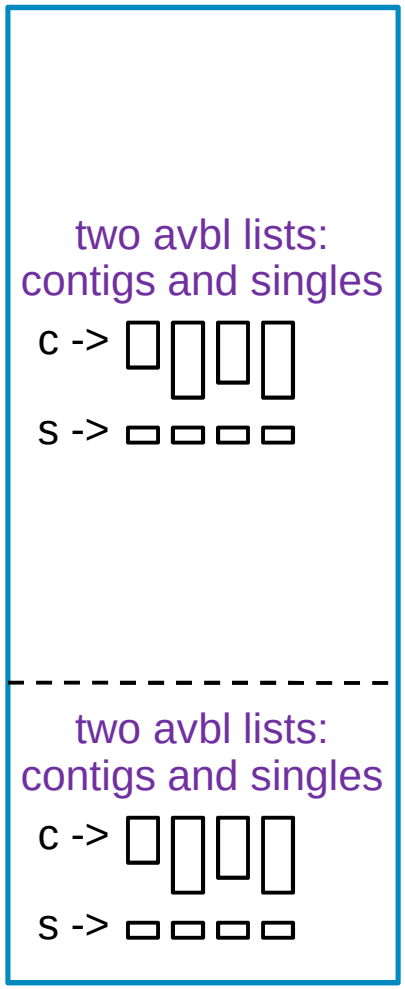
Each **list** had a low threshold and a high threshold

After every free storage request call, demand scan was kicked off if a **list** fell below its low threshold

The <2G lists were repopulated by demand scan

2 GB

<2G Use Policy:
Pre-6.2: *used <2G first*
In 6.2: *used <2G proportionally*
In 6.3: *uses <2G last*



New way

Each **kind of free storage request call** has a low and a high threshold:

- TYPE=ANY contigs
- TYPE=ANY singles
- TYPE=BELOW contigs
- TYPE=BELOW singles

Contiguous lists are protected from being completely raided by singles requests

After every request, the low threshold for **every type of request** is evaluated

If a TYPE=ANY low threshold is breached, demand scan is kicked off

If the <2G lists are empty, a frame table scan is kicked off

The Old Demand Scan Visit Policy

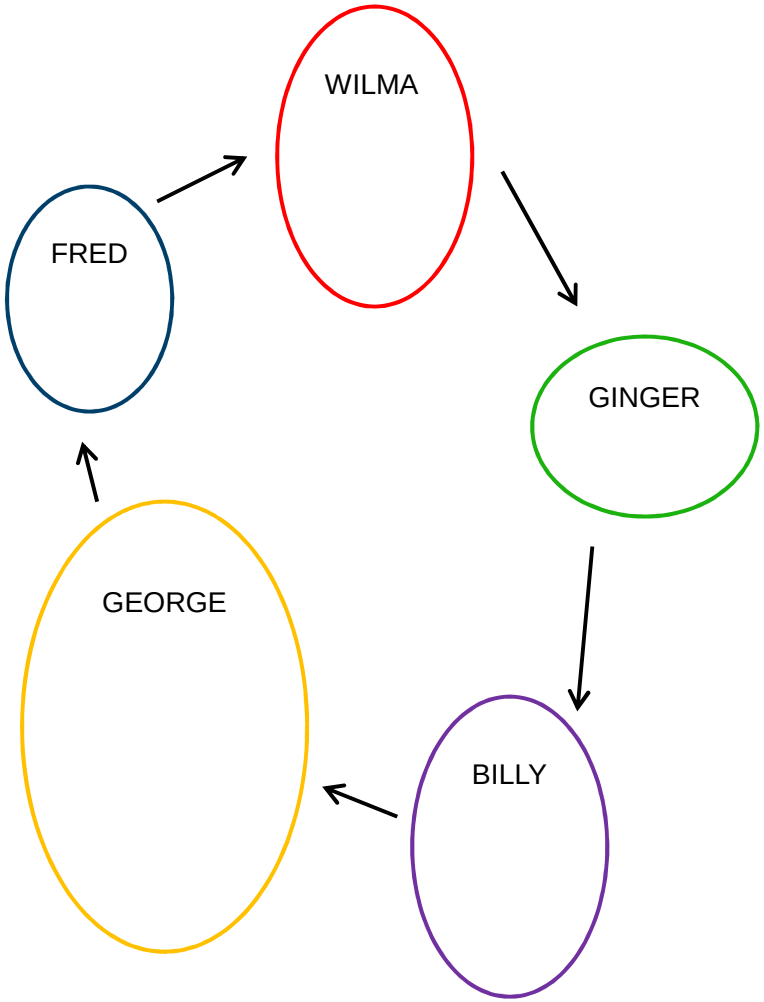
- Three-pass model:
  - **Pass 1:** tried to be friendly to dispatched users
    - Unreferenced shared-address-space pages
    - Long-term-dormant guests
    - Eligible-list guests
    - Dispatch-list guests' unreferenced pages down to Working Set Size
  - **Pass 2:** a little more aggressive… like pass 1 except:
    - Avoided shared address spaces
    - Would take from dispatch-list guests down to their SET RESERVED
  - **Pass 3:** emergency scan
    - Anything we can find

The Old Demand Scan Problems

- We found a number of problems over time, to various degrees, such as:
  - Pass 1 tended to be too soft.
  - Scheduler lists tended not to portray “active” in a way usable by storage management.
  - Stole a lot from the first few guests visited.
  - **SET RESERVED** was not being observed.

- It used the System z page reference bit R to track page changes
  - Required lots of RRBE instructions to keep track of recent reference habits
    - RRBE can be an expensive instruction
    - (Large resident frame list) + (long RRBE instruction) = problems in Reorder

New Approach: The New Demand Scan Visit Policy

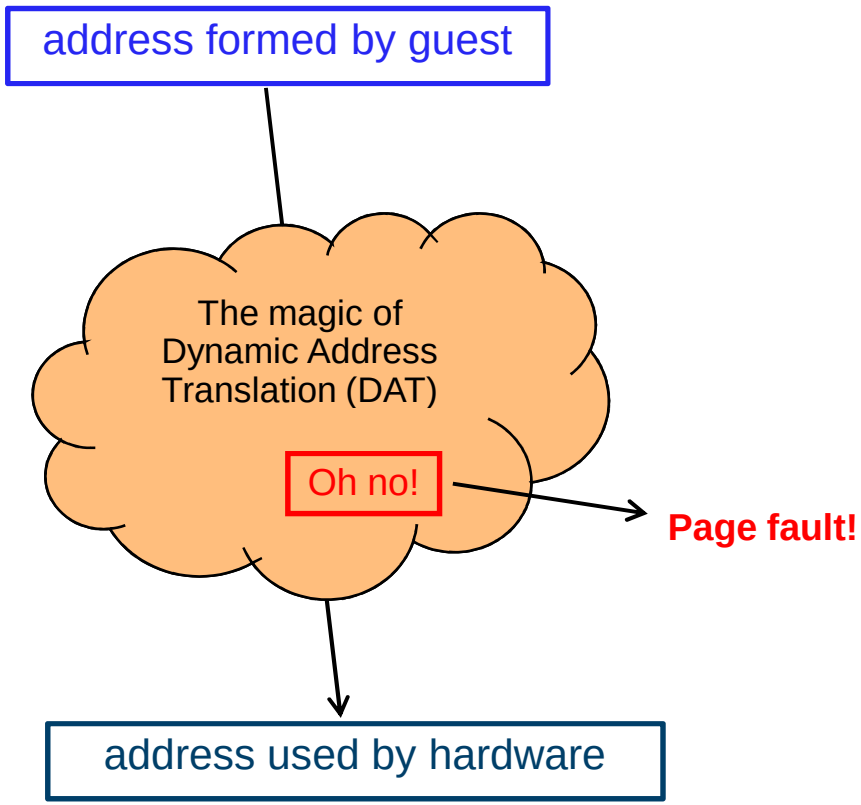


- **Used to:**
  - Visit according to scheduler lists
  - Take heavily at each visited guest
  - Start over at list tops every pass
  - Take from private VDISKs nearly last
  - A “take” was truly a ***reclaim*** of a frame
- **Now:**
  - Cyclically visits the logged-on guests
  - Keeps a visit cursor so it can resume
  - Takes a little and then moves to next
  - Takes from private VDISKs much earlier
  - A “take” is now just a push of in-use frames down toward ***eventual reclaim***
- **Effects**
  - Better equalizing in the face of storage constraint
  - Better equalizing on the notion of “hot” vs. “cold” pages

New Approach: Other New Things About Demand Scan

- Gives up control periodically
  - Lets other things happen
  - Avoids long-running “blackouts”
- Tries harder to be “fair” in the face of constraint.
- Aspects of “fairness”:
  - Use a guest’s size and estimation of its page touch rate to decide how much to take
  - Take from large guests who touch their pages less often before taking from small guests who touch their pages a lot
  - Treat identical guests identically
  - Don’t take from a guest’s working set if another guest is not stripped to its working set
  - During startup (when page touch rate data is available) take an amount of pages proportional to each guest’s size

New Approach: Trial Invalidation



- Page table entry (PTE) contains an "invalid" bit
- What if we:
  - Keep the PTE intact but set the "invalid" bit
  - Leave the frame contents intact
  - Wait for the guest to touch the page
- A touch will cause a page fault, but…
- On a fault, there is nothing really to do except:
  - Clear the "invalid" bit
  - Move the frame to the front of the frame list to show that it was recently referenced
- We call this **trial invalidation**.

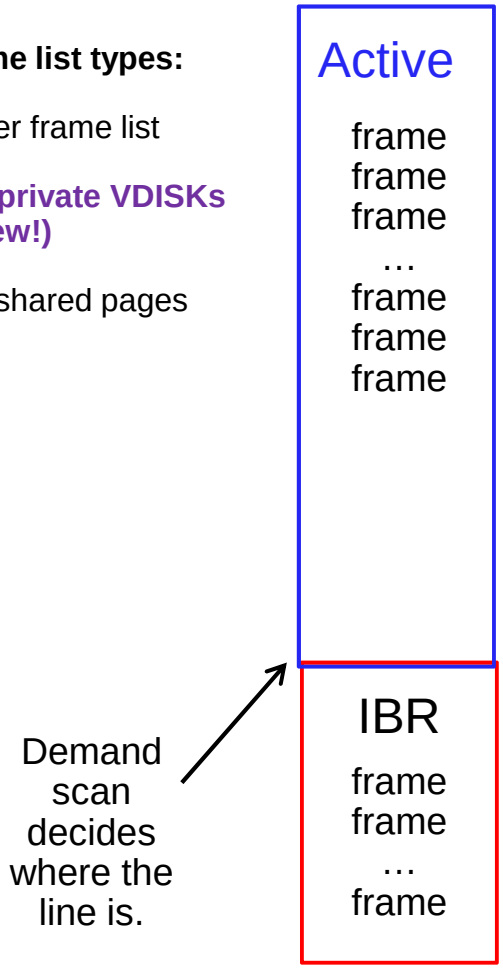
New Approach: Two-Section Frame-Owned Lists

Frame list types:

A user frame list

The private VDISKs (new!)

The shared pages



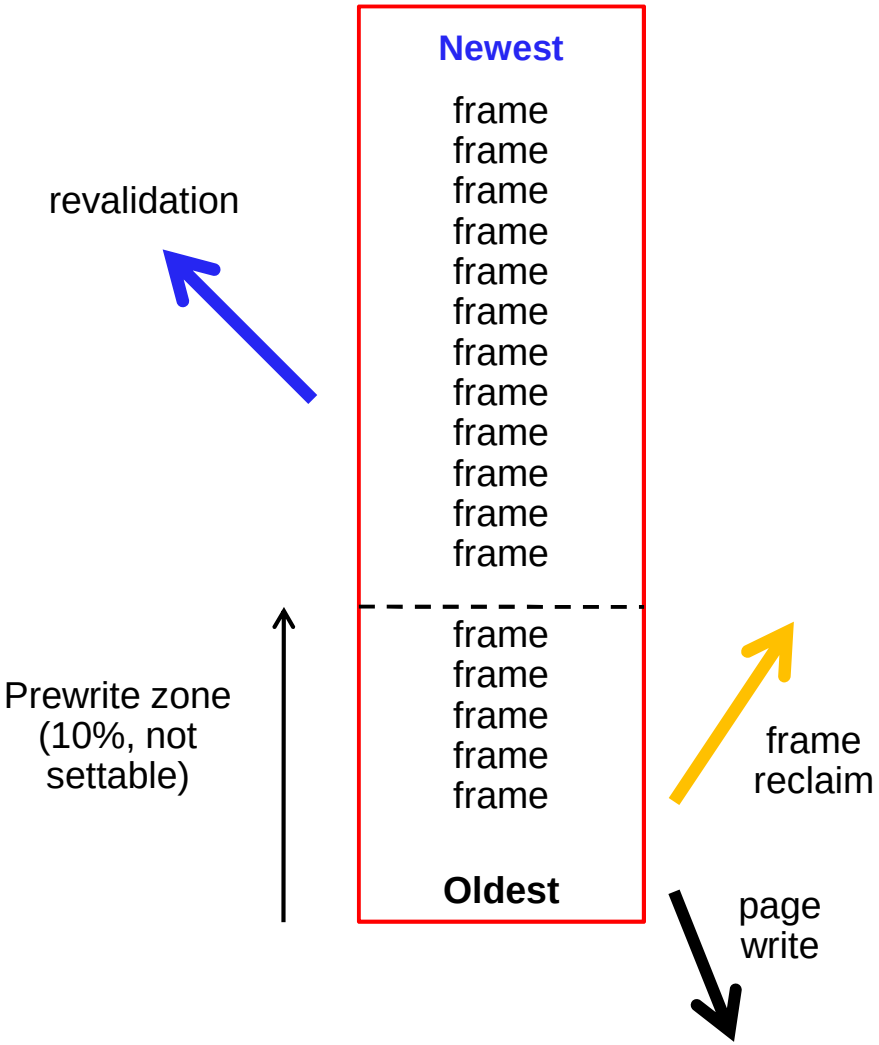
Active: frames that are in use
- Roughly in order by when they became valid.

No longer is a frame list ever searched, sorted, or reordered.

IBR: invalid but resident
- Marked invalid in page table entry
- If ever referenced, fault resolution moves to top of active
- This gives us a way to detect lack of reference

- "We try to keep [it] rather small."
- Influenced by revalidation rate.

New Approach: Global Aging List



- Size of global aging list can be specified…
  … but is best left to the system to manage

- All of the pages here are **IBR**

- Demand scan fills it from the top

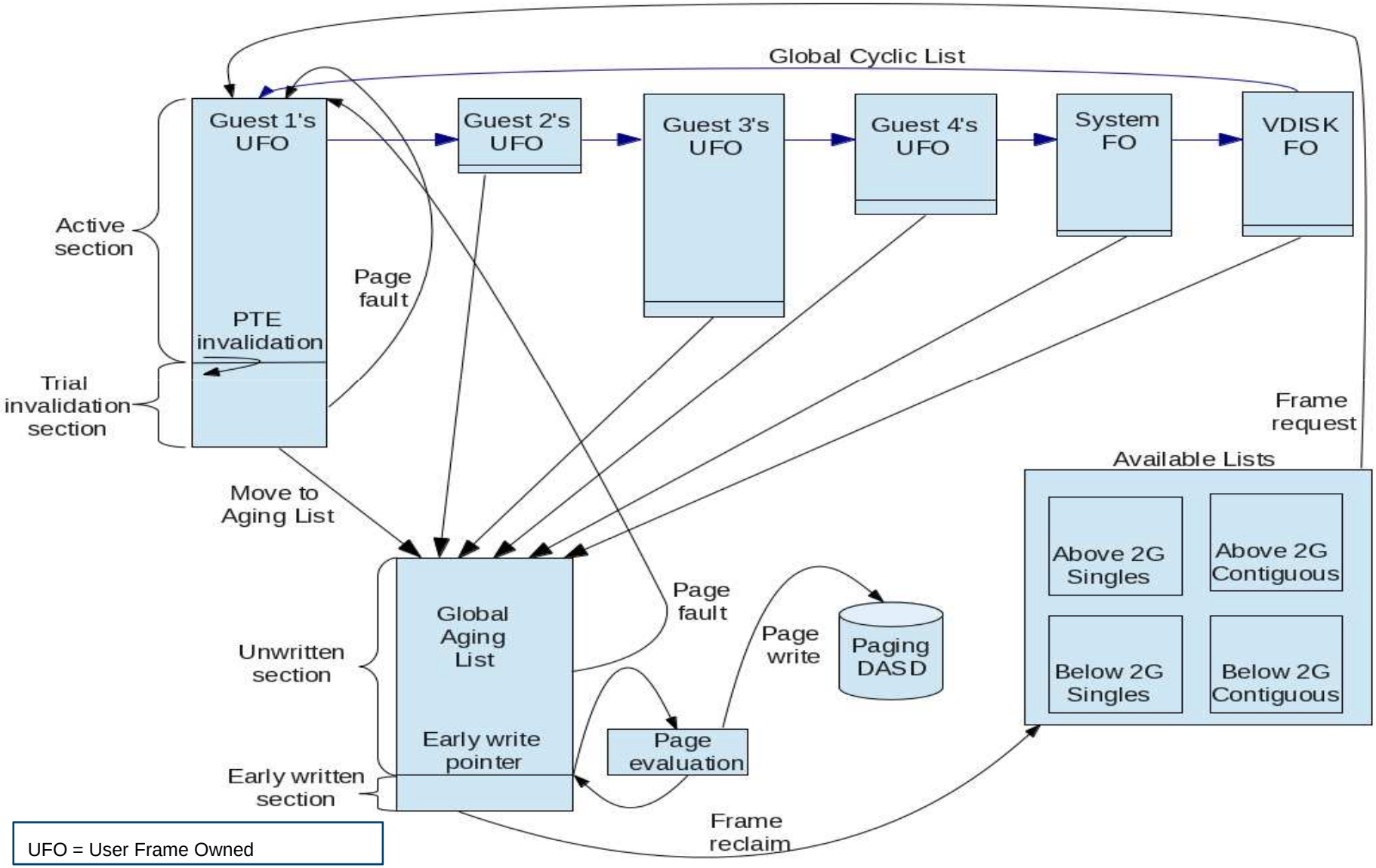
- Revalidated pages return to their owned lists

- We pre-write changed pages up from the bottom of the list.

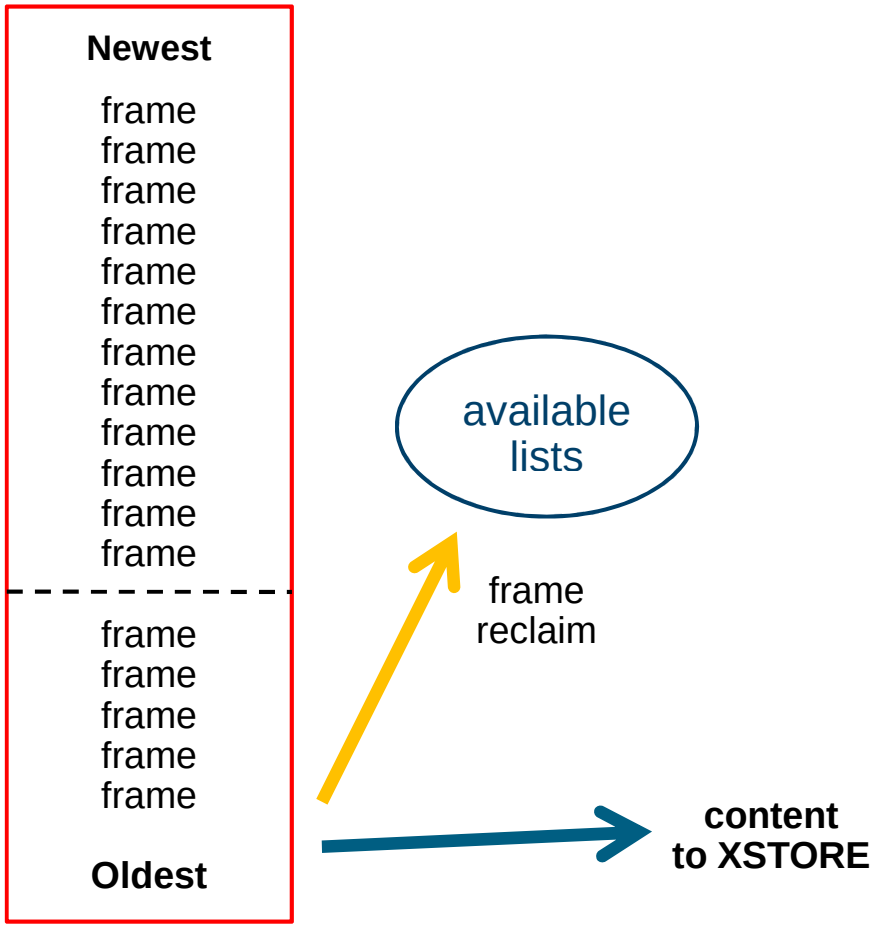
- The global aging list accomplishes the age-filtering process that XSTORE used to accomplish.

- We no longer suggest XSTORE for paging, but we will use it if it's there.

Memory Management Algorithm Visualization

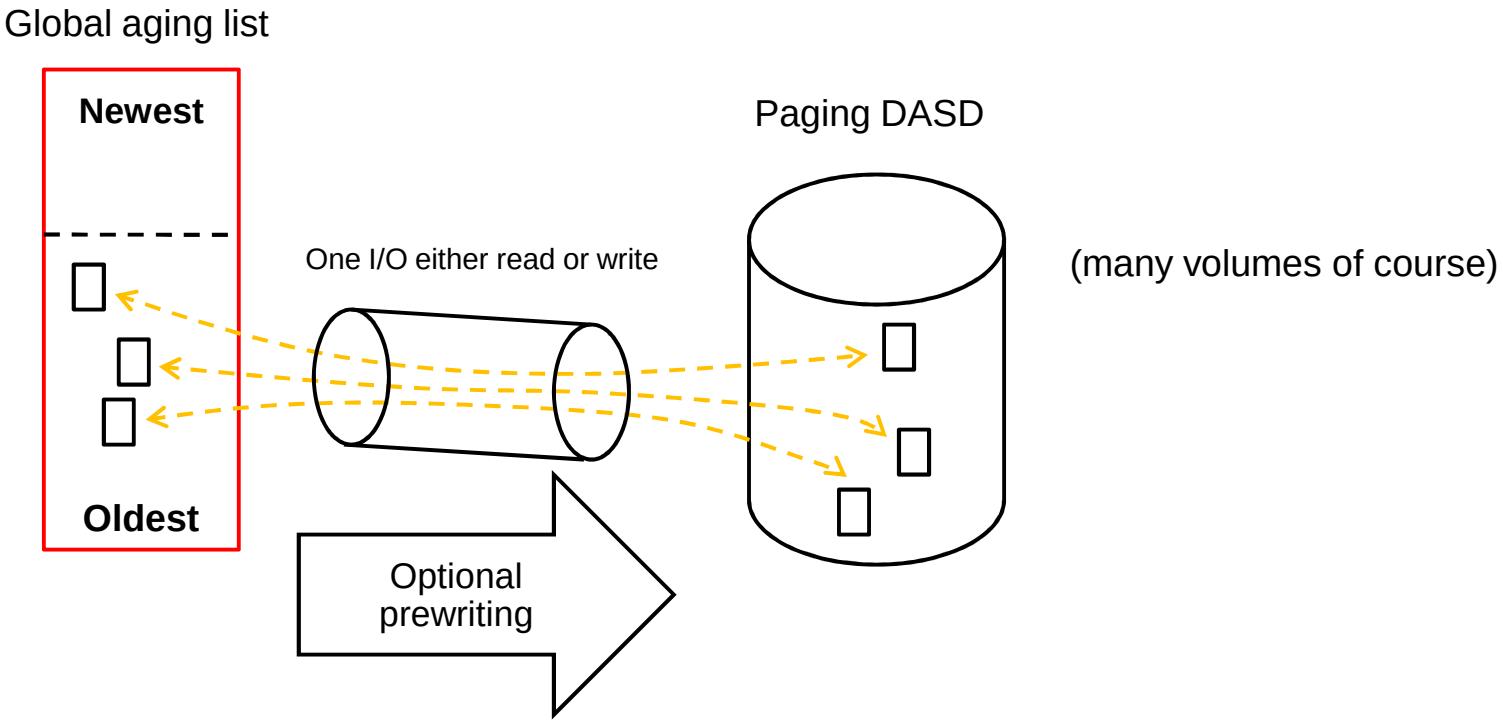


New Approach: What About XSTORE?



- We will use XSTORE if it is there.
- XSTORE is now the *second* line of defense.
- *When frame is reclaimed*, if XSTORE is present, we put a copy of the page there.
  - Even if the frame has already been prewritten
- On fault, if content is still in XSTORE, it comes back from there.
- If you decide to keep XSTORE, do NOT put MDC in XSTORE unless heavy CMS workload.

New Approach: How We Now Use Paging DASD



Highlights of new DASD techniques:

- A page almost always goes back to its same DASD slot.
  - *Exceptions: clogged or DRAINed volume*
- A page not changed since last read from DASD is almost never rewritten.
  - *Exceptions: DRAIN*
- The paging channel program can handle discontiguity on both ends, whether read or write.

New Approach: Large Real Implies Large Virtual, So…

- z/VM holds its DAT management structures in CP-owned pageable address spaces
- These *Page Table Resource Manager* address spaces are named PTRM0000, PTRM0001, …
- You will see them in the z/VM Performance Toolkit FCX134 DSPACESH report
- The number and size of these address spaces control how much logged-on guest real (aka virtual memory) the system can support

- In z/VM 6.2:
  - There were **16** of them: …, PTRM000F
  - Created as needed
  - With 16 of these, can address **8 TB** of virtual
- In z/VM 6.3:
  - There are now **128** of them: …, PTRM007F
  - All created at system initialization
  - With 128 of these, can address **64 TB** of virtual

New Behavior: CP SET RESERVED command

- We now do much better at honoring the setting
  - Revisit your uses to see whether you were trying to compensate

- Pages can be now be reserved for **NSS** and **DCSS** as well as virtual machines
  - Set ***after*** **CP SAVESYS** or **SAVESEG** of NSS or DCSS
  - Segment ***does not need to be loaded*** in order to **SET RESERVE** for it
  - A new instance of an NSS or DCSS ***does not*** inherit a pending-purge instance's RESERVED setting
  - Recommended for **MONDCSS**

- You can set a system-wide maximum (**SYSMAX**) on the number of reserved pages
- RESERVED settings ***do not*** survive IPL
  - Consider CP command in the CP directory (not for NSS or DCSS though)

Removed Behavior: Reorder

- We just don't do this anymore.
  - No longer a trade-off with larger virtual machines
- Commands remain for compatibility but have no impact
  - **CP SET REORDER** command gives RC=6005, "not supported".
  - **CP QUERY REORDER** command says it's OFF.
- You will no longer see reorder information in Monitor.
- Be aware of reorder settings when using LGR between z/VM 6.2 and z/VM 6.3

Changed Behavior: Eligible List

- One of the factors to the creation of an **eligible list** is the concept of “loading users”
  - Governed by **SET SRM LDUBUF**
  - A virtual machine is characterized as a “loading user” if its count of page faults in a dispatch slice exceeds a threshold
  - **SET SRM LDUBUF** attempts to keep the system from over-committing paging devices to the point of thrashing

- Changes in z/VM 6.3 paging algorithms can affect the number of virtual machines that are marked as “loading” users and therefore cause **eligible lists** to be formed where they had not formed prior to z/VM 6.3
  - Definition of page fault slightly different
  - Rate at which system can page fault has increased

- Recommend monitoring for eligible lists and adjusting the following as appropriate
  - **SET QUICKDSP**
  - **SET SRM LDUBUF**

- IBM is investigating improvements to avoid the unnecessary eligible list formation.

New or Changed Commands

Commands: Knobs You Can Twist

Concept	Knob	Comments
Size of the global aging list Whether early writes are allowed	Command: CP SET AGELIST … Config file: STORAGE AGELIST … Lookup: CP QUERY AGELIST	Sets the size of the global aging list, in terms of: - A fixed amount (e.g., GB) - A percent of DPA (preferred) The default is 2% of DPA. Seems OK. Sets whether early writes are allowed. (If storage-rich, say NO.)
Amount of storage reserved for a user or for a DCSS	Command: CP SET RESERVED … Config file: STORAGE RESERVED … Lookup: CP QUERY RESERVED …	You can set RESERVED for: - A user - An NSS or DCSS You can also set a SYSMAX on total RESERVED storage. Config file can only set SYSMAX.

Commands: Other Interesting “Queries”

Query or Lookup	Comments
CP INDICATE LOAD	The STEAL-nnn% field no longer appears in the output.
CP INDICATE NSS	Includes a new “instantiated” count. Number of pages that exist. Sum of locus counts might add to more than “instantiated”.
CP INDICATE USER	Includes a new “instantiated” count. Sum of locus counts might add to more than “instantiated”.
CP INDICATE SPACES	Includes a new “instantiated” count.

You Must Make a Plan

Planning for Large Memory

- **Normal best practices for migrating from an earlier release still apply.**
- **Change your paging XSTORE into central**
  - XSTORE provided an aging function. It helped catch reclaim selection "mistakes".
  - The new ***IBR*** concept and ***global aging list*** provide the same function but do so more efficiently in central.
- **Plan enough DASD paging space**
  - The system now prewrites pages to DASD.
  - See space calculation on a later slide
- **Plan a robust paging DASD configuration**
  - Use plenty of paging volumes
  - Make the volumes all the same size
  - Put only paging space on the volumes you use for paging
  - Spread the paging volumes through your logical control units
  - Avoid logical control units that you know are hot on application I/O
  - Use plenty of chpids
  - Do not use ESCON chpids
  - Do not mix ECKD paging and SCSI paging
  - Leave reserved slots in the CP-owned list

Planning for Large Memory

- Look at your **CP SET RESERVED** settings to make sure they're right.
  - Revisit scenarios where you looked at this capability and it wasn't effective

- Add **CP SET RESERVED** settings for DCSSes or NSSes if you like
  - MONDCSS is a good one to consider

- If you increase central, make sure you also increase dump space
  - More guidance available on www.vm.ibm.com/techinfo/
  - Download updated *"Allocating Space for CP Hard Abend Dumps"*

Planning DASD Paging Space

- Calculate sum of:
  - Logged-on virtual machines' primary address spaces, plus…
  - Any data spaces they create, plus…
  - Any VDISKs they use, plus…
  - Total number of shared NSS or DCSS pages, … and then …
  - Multiply this sum by 1.01 to allow for PGMBKs and friends

- Add to that sum:
  - Total number of CP directory pages (reported by DIRECTXA), plus…
  - Min (10% of central, 4 GB) to allow for system-owned virtual pages

- Then multiply by some safety factor (1.25?) to allow for growth or uncertainty

- Remember that your system will take a PGT004 if you run out of paging space

- Consider using something that alerts on page space, such as Operations Manager for z/VM

Planning to Learn About Your System's Performance

- While you are still on the earlier release, collect measurement data:
  - Know what your key success metrics are and what their success thresholds are
  - Transaction rates – *only you* know where these are on your workloads
  - MONWRITE files – some tips:
    - When: Daily peaks? Month-end processing? Quarter-end processing?

- Then go ahead and try z/VM 6.3

- When you start running on z/VM 6.3, collect the very same measurement data

- Compare z/VM 6.3 back to z/VM 6.2 to see what the effect is on your workload

Planning to Keep Your System Maintained

- Additional service has shipped, current install media includes second RSU (6302)

- Keep listening:
  - www.vm.ibm.com
  - The IBMVM listserver

- See also the PSP bucket for z/VM 6.3

Comments on Workloads

z/VM Large Memory: Amenable Workloads

- **Best benefit:** workloads highly affected by reorder or old demand scan
  - Large guests affected by reorder delays
  - Long demand scans looking for <2G frames

- **Less benefit:** workloads that were doing fine before
  - Storage-rich workloads
  - Running fine paging to only XSTORE
  - No problems with long demand scans
  - Small guests not affected by reorder

- Let's look at some examples

The “Sweet Spot” Workload

Our synthetic workload called *Sweet Spot* imitates behaviors we have seen in customer-supplied MONWRITE data.

	z/VM 6.2	z/VM 6.3	Delta	Pct. Delta
Cstore	256	384	128	
Xstore	128	**0**	-128	
External Throughput (ETR)	0.0746	0.0968	0.0222	29.8%
Internal Throughput (ITR)	77.77	105.60	27.83	35.8%
System Util/Proc	31.4	4.7	-26.7	-85.0%
T/V Ratio	1.51	1.08	-0.43	-28.5

By getting rid of both reorders and spin lock contention, we achieved huge drops in %CPU and T/V.

The “Sweet Spot” Workload

- Closer look at how the fairness and workloads may result in different results.
- Sweet Spot workload has four groups of virtual machines. Some benefit more than others.

	z/VM 6.2	z/VM 6.3	Delta	Pct. Delta
System External Throughput	0.0746	0.0968	0.0222	29.8%
User Group 1 ETR	0.0065	0.0128	0.0063	96.9%
User Group 2 ETR	0.0138	0.0236	0.0098	71.0%
User Group 3 ETR	0.0268	0.0264	-0.0004	-1.5%
User Group 4 ETR	0.0275	0.0341	0.0066	24.0%

Workload: The Apache Paging Workload

Our Linux-based workload called *Apache Paging* is built to page heavily to DASD almost no matter how much central or XSTORE we give it.

	z/VM 6.2	z/VM 6.3
Cstore (GB)	256	384
Xstore (GB)	128	0
External Throughput (ETR)	1.000	1.024
Internal Throughput (ITR)	1.000	1.017
Xstore paging / second	82489	0
DASD paging / second	33574	31376

This is an example of a workload where the limit comes from something large memory will not fix.

Large Memory Scaling Measurements

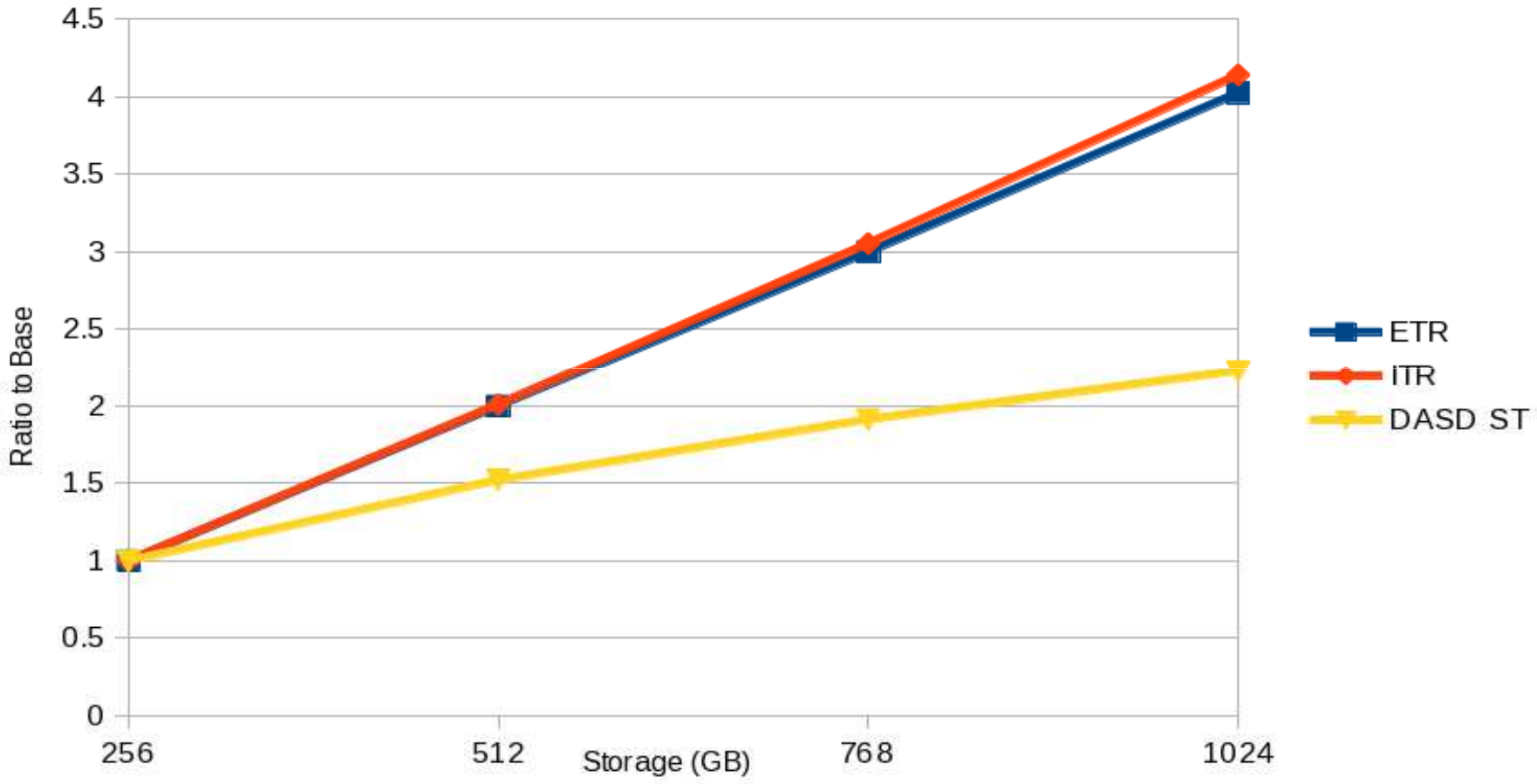
1. **VIRSTOR** – Test case system started with CMS boot strap with controls over memory reference patterns and processor usage.
   – Create workload similar to resource usage from customer Monwrite data

2. **Linux Apache Static Web serving**

- Measure and test levels of servers at peak usage for 256 GB in an overcommitted environment

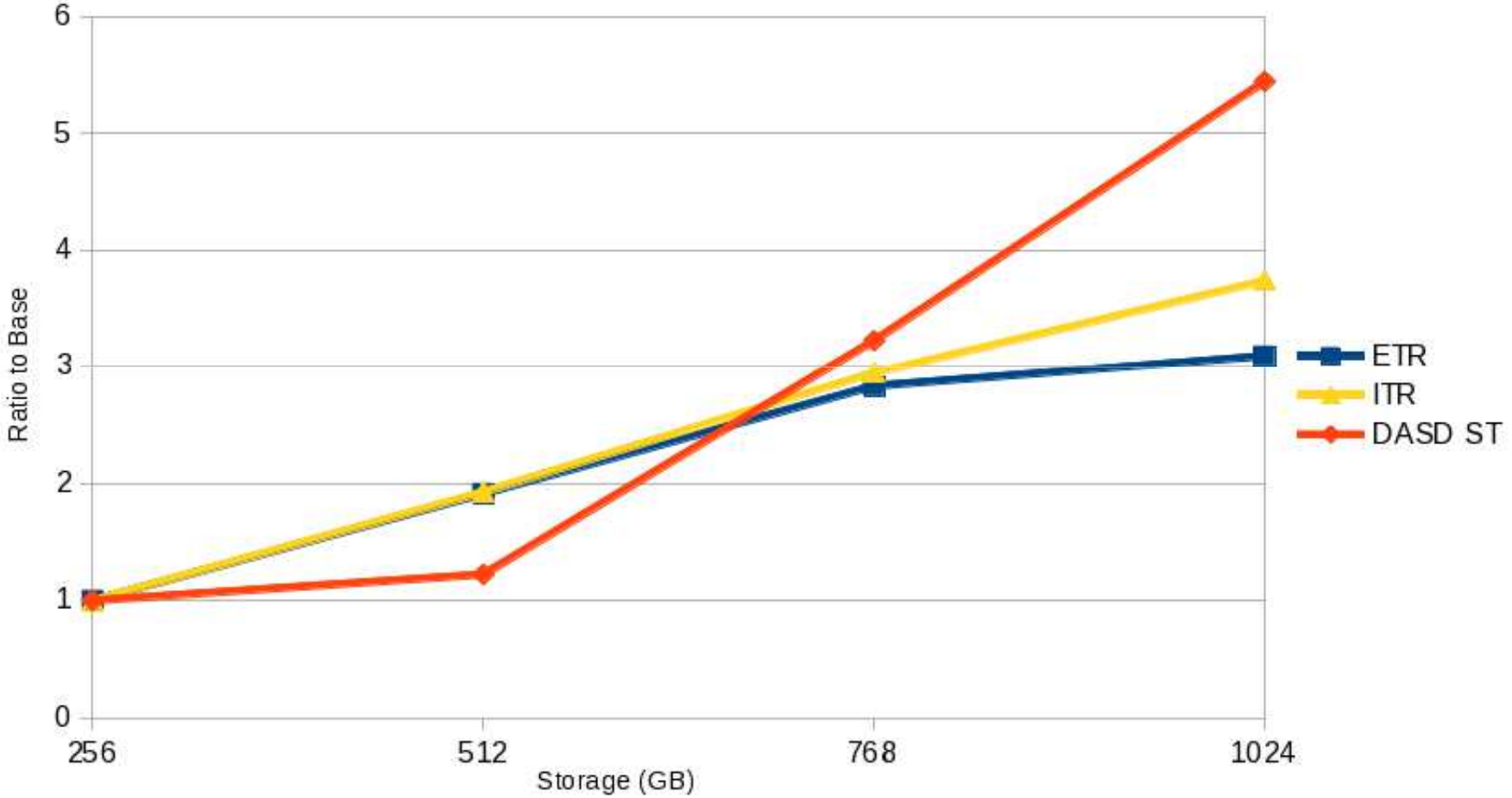
- Scale up from there to 1 TB
  – All resources scaled up, though note that while additional DASD space was provided, it was on the same storage server.

VIRSTOR Workload in Overcommitted Environment



ETR = External Throughput; ITR = Internal Throughput; DASD ST = DASD Service Time

Apache Workload in Overcommitted Environment



ETR = External Throughput; ITR = Internal Throughput; DASD ST = DASD Service Time

CP Monitor and z/VM Performance Toolkit

Session 14687: Understanding z/VM 6.3 through new Performance Toolkit Reports

Wednesday 4:30 PM - Grand Ballroom Salon G (Bill Bitner)

CP Monitor Records

No new Monitor records, only big changes….

Domain	Record	Name	Type	Title	Fields, N / D / C
D0	R3	MRSYTRSG	sample	Real Storage Data (Global)	D C
D0	R4	MRSYTRSP	sample	Real Storage Data (Per Processor)	D
D0	R6	MRSYTASG	sample	Auxiliary Storage (Global)	N C
D0	R7	MRSYTSHS	sample	Shared Storage Data	D
D0	R23	MRSYTLCK	sample	Formal Spin Lock Data	N C
D1	R7	MRMTRMEM	config	Memory Configuration Data	N
D1	R15	MRMTRUSR	config	Logged on User	C
D2	R4	MRSCLADL	event	Add User to Dispatch List	D C
D2	R5	MRSCLDDL	event	Drop User from Dispatch List	D C
D2	R6	MRSCLAEL	event	Add User to Eligible List	C
D2	R8	MRSCLSTP	event	System Timer Pop	D
D3	R1	MRSTORSG	sample	Real Storage Management (Global)	N D C
D3	R2	MRSTORSP	sample	Real Storage Activity (Per Processor)	D
D3	R3	MRSTOSHR	sample	Shared Storage Management	N C
D3	R14	MRSTOASI	sample	Address Space Information Record	N C
D3	R15	MRSTOSHL	event	NSS/DCSS/SSP Loaded into Storage	N
D3	R16	MRSTOSHD	event	NSS/DCSS/SSP Removed From Storage	N C
D4	R2	MRUSELOF	event	User Logoff Data	N D C
D4	R3	MRUSEACT	sample	User Activity Data	N D C
D4	R9	MRUSEATE	event	User Activity Data at Transaction End	D C

As usual, the Monitor records will be on www.vm.ibm.com at GA.

z/VM Performance Toolkit: Highlights

- Changed screens:
  - FCX102 SYSTEM, Some Internal System Counters
  - FCX103 STORAGE, General Storage Utilization
  - FCX133 NSS, NSS and DCSS Utilization and Paging Activity
  - FCX146 AUXLOG, Auxiliary Storage Utilization, by Time
  - FCX147 VDISKS, Virtual Disks in Storage
  - FCX265 LOCKLOG, Spin Lock Log, by Time
- Deleted screens:
  - FCX254 AVAILLOG, Available List Management, by Time
  - FCX259 DEMNDLOG, Demand Scan Details, by Time
- New screens:
  - FCX290 UPGACT, User Page Activity — *page state transition rates*
  - FCX291 UPGACTLG, User Page Activity (benchmarks a user)
  - FCX292 UPGUTL, User Page Utilization Data — *page residency counts*
  - FCX293 UPGUTLLG, User Page Utilization Data (benchmarks a user)
  - FCX294 AVLB2GLG, Available List Data Below 2G, by Time — *available list counts*
  - FCX295 AVLA2GLG, Available List Data Above 2G, by Time
  - FCX296 STEALLOG, Steal Statistics, by Time — *steal algorithm activity*
  - FCX297 AGELLOG, Age List Log, by Time — *global aging list activity*

z/VM Performance Toolkit: New Columns and Concepts

New Field	What this means
Inst	*Instantiations:* the rate at which valid memory is being created *Instantiated:* the amount of valid memory
Relse	*Releases:* the rate at which memory is being released
Inval	*Invalidations:* the rate at which demand scan is marking memory invalid as a way to determine whether it is being touched
Reval	*Revalidations*: the rate at which invalid pages are being made valid because somebody touched them
Ready	*Ready reclaims* or *ready steals:* the frame was found and selected for reclaim and had already been prewritten to auxiliary storage
Not Ready	*Notready reclaims* or *notready steals:* the frame was selected for reclaim but we had to wait for the auxiliary write (DASD) to finish before we could take it

z/VM Performance Toolkit: New Columns and Concepts

New Field	What this means
PNR	*Private, not referenced*: the page was read from aux as part of a block read, but it is still marked invalid because nobody has touched it yet
x<2G or *x*>2G	*Below 2 GB* or *Above 2 GB*: tells where the real backing frames are in real central
Sing	*Singles:* free frames surrounded by in-use frames (cannot coalesce)
Cont	*Contigs*: free frames in strings of two or more
Prot	*Protect threshold:* number of frames a singles-obtain must leave on a contigs-list

z/VM Performance Toolkit: New Report FCX292 UPGUTL

```
FCX292  Run 2013/04/10 07:38:36          UPGUTL                                                    Page  103
                                         User Page Utilization Data
From 2013/04/09 16:02:10                                                              SYSTEMID
To   2013/04/09 16:13:10                                                              CPU 2817-744    SN A6D85
For    660 Secs 00:11:00                 "This is a performance report for SYSTEM XYZ"            z/VM   V.6.3.0 SLU 0000
```

		Storage																
		Resident																
	Data							Invalid But Resident									Base	
	Spaces				Total		Locked		UFO		PNR		AgeList				Space	Nr of
Userid	Owned	WSS	Inst	Resvd	T_All	T>2G	L<2G	L>2G	U<2G	U>2G	P<2G	P>2G	A<2G	A>2G	XSTOR	AUX	Size	Users
>>Mean>>	.0	5284M	6765M	5611	5286M	27M	5259M	1010	232K	6565	2238K	59588	26M	53080	107M	.0	1815M	7108M	73
User Class Data:																			
CMS1_USE	.0	3320K	19M	.0	484K	.0	484K	.0	4096	.0	69632	.0	244K	.0	344K	.0	19M	2047M	1
LCC_CLIE	.0	364M	485M	.0	365M	11264	365M	.0	208K	.0	325K	.0	2686K	.0	8177K	.0	164M	1024M	8
LXA_SERV	.0	7974M	10G	.0	7978M	41M	7937M	.0	206K	9984	3327K	90624	39M	80725	161M	.0	2719M	10240M	48
User Data:																			
DISKACNT	.0	4976K	5156K	0	4K	0	4K	0	0	0	4K	0	0	0	0	0	5152K	32M	
DTCVSW1	.0	184K	11M	0	196K	8K	188K	8K	4K	0	4K	0	0	0	168K	0	11M	32M	
DTCVSW2	.0	180K	11M	0	184K	0	184K	0	4K	0	4K	0	0	0	164K	0	10M	32M	
EREP	.0	4912K	4944K	0	4K	0	4K	0	0	0	4K	0	0	0	0	0	4940K	32M	
FTPSERVE	.0	84K	5764K	0	88K	0	88K	0	4K	0	4K	0	0	0	76K	0	5760K	32M	
GCSXA	.0	204K	208K	0	8K	0	8K	0	4K	0	4K	0	0	0	0	0	200K	16M	
LCC00001	.0	364M	488M	0	365M	0	365M	0	204K	0	228K	0	2884K	0	8660K	0	192M	1024M	
LCC00002	.0	369M	492M	0	371M	20K	371M	0	204K	0	224K	0	2312K	0	7736K	0	159M	1024M	
LCC00003	.0	363M	484M	0	364M	0	364M	0	204K	0	252K	0	2852K	0	8372K	0	215M	1024M	
LCC00004	.0	363M	483M	0	363M	16K	363M	0	204K	0	228K	0	2724K	0	8512K	0	185M	1024M	

Look for the new concepts: Inst IBR UFO PNR AgeList

Amounts are in bytes, suffixed. Not page counts!

FCX113 UPAGE is still produced.

z/VM Performance Toolkit: New Report FCX292 UPGUTL

```
<------------------------- Resident ------------------------------>
                                    <------ Invalid But Resident ------>
<---- Total ----> <-Locked--> <-- UFO --> <-- PNR --> <-AgeList->
T_All  T<2G  T>2G  L<2G  L>2G  U<2G  U>2G  P<2G  P>2G  A<2G  A>2G XSTOR  AUX
 365M     0  365M     0  204K     0  228K     0 2884K     0 8660K     0 192M
```

- Look for the new concepts: Inst IBR UFO PNR AgeList
- Amounts are in bytes, suffixed. Not page counts!
- FCX113 UPAGE is still produced.

z/VM Performance Toolkit: New Report FCX290 UPGACT

```
FCX290   Run 2013/04/10 07:38:36         UPGACT                                                                  Page  102
                                         User Page Activity
From 2013/04/09 16:02:10                                                                                  SYSTEMID
To   2013/04/09 16:13:10                                                                                  CPU 2817-744    SN A6D85
For     660 Secs 00:11:00                "This is a performance report for SYSTEM XYZ"                    z/VM    V.6.3.0 SLU 0000
_____________________________________________________________________________________________________________________________________

 ______      .     .     .     .     .     .     .     .     .     .     .     .     .
                <------------------------------ Storage ------------------------------>
                                                    <------- Movement/s -------->
            Stl <--- Transition/s ----> <-Steal/s->                         <Migrate/s> Nr of
Userid       Wt  Inst Relse Inval Reval Ready NoRdy  PGIN PGOUT Reads Write MWrit  Xrel Users
>>Mean>>    1.0  143K  5142  849K  718K  999K    .0    .0    .0  958K  761K    .0    .0    73
User Class Data:
CMS1_USE    1.0 15515 15801  2377  1632  5145    .0    .0    .0    .0  1980    .0    .0     1
LCC_CLIE    1.0  658K 20875  488K  486K 60875    .0    .0    .0 54212 22869    .0    .0     8
LXA_SERV    1.0  108K  1095 1191K  994K 1506K    .0    .0    .0 1447K 1153K    .0    .0    48

User Data:
DISKACNT    1.0     0     0     0     0     0     0     0     0     0     0     0     0
DTCVSW1     1.0     0     0  3072  2855     0     0     0     0     0     0     0     0
DTCVSW2     1.0     0     0  3004  2780     0     0     0     0     0     0     0     0
EREP        1.0     0     0     0     0     0     0     0     0     0     0     0     0
FTPSERVE    1.0     0     0  1434  1434     0     0     0     0     0     0     0     0
GCSXA       1.0     0     0     0     0     0     0     0     0     0     0     0     0
LCC00001    1.0  601K 18686  501K  498K 65139     0     0     0 49866 23670     0     0
LCC00002    1.0  657K 24955  487K  486K 54725     0     0     0 44522 18991     0     0
LCC00003    1.0  565K 23012  485K  481K 64065     0     0     0 44783 19859     0     0
LCC00004    1.0  602K 24104  499K  495K 63178     0     0     0 48811 24588     0     0
LCC00005    1.0  717K 25675  500K  499K 65865     0     0     0 66002 28753     0     0
```

Look for the new concepts: Inst Relse Inval Reval Ready NoRdy

z/VM Performance Toolkit: New Report FCX290 UPGACT

```
FCX290   Run 2013/04/10 07:38:36          UPGACT                                                   Page  102
                                          User Page Activity
From 2013/04/09 16:02:10                                                                   SYSTEMID
To   2013/04/09 16:13:10                                                                   CPU 2817-744    SN A6D85
For     660 Secs 00:11:00                 "This is a performance report for SYSTEM XYZ"    z/VM    V.6.3.0 SLU 0000

                  <------------------------------- Storage ------------------------------->
                                                  <-------- Movement/s -------->
           Stl <--- Transition/s ----> <-Steal/s->                          <Migrate/s> Nr of
```

	Stl	<--- Transition/s		---->		<-Steal/s->	
Userid	Wt	Inst	Relse	Inval	Reval	Ready	NoRdy
>>Mean>>	1.0	143K	5142	849K	718K	999K	.0
User Class Data:							
CMS1_USE	1.0	15515	15801	2377	1632	5145	.0
LCC_CLIE	1.0	658K	20875	488K	486K	60875	.0
LXA_SERV	1.0	108K	1095	1191K	994K	1506K	.0

- Look for the new concepts: Inst Relse Inval Reval Ready NoRdy

z/VM Performance Toolkit: New Report FCX295 AVLA2GLG

```
FCX295  Run 2013/04/10 07:38:36          AVLA2GLG                                                    Page   25
                                         Available List Data Above 2G, by Time
From 2013/04/09 16:02:10                                                                  SYSTEMID
To   2013/04/09 16:13:10                                                                  CPU 2817-744   SN A6D85
For    660 Secs 00:11:00                 "This is a performance report for SYSTEM XYZ"    z/VM   V.6.3.0 SLU 0000
```

	<---------- Storage ---------->				<--Times-->		<-Frame Thresh-->		
Interval	<Available>		<Requests/s>		<Returns/s>		<-Empty/s->		Sing	<-Contigs->	
End Time	Sing	Cont	Sing	Cont	Sing	Cont	Sing	Cont	Low	Low	Prot
>>Mean>>	23M	267M	47M	59M	47M	51M	.0	.0	1310	15	15
16:02:40	0	938M	32M	126M	502K	30310	.0	.0	1332	15	15
16:03:10	152K	4556K	50M	89M	49M	59M	.0	.0	1168	15	15
16:03:40	400K	4824K	68M	82M	71M	79M	.0	.0	1321	15	15
16:04:10	0	5896K	49M	72M	52M	70M	.0	.0	2409	15	15
16:04:40	0	2124K	40M	60M	41M	59M	.0	.0	1308	15	15
16:05:10	876K	3488K	54M	52M	55M	51M	.0	.0	1118	15	15
16:05:40	0	3624K	53M	58M	54M	57M	.0	.0	1409	15	15
16:06:10	2016K	4464K	49M	57M	51M	56M	.0	.0	1273	15	15

Look for the new concepts: Singles Contigs Prot

Amounts are in bytes, suffixed. Not page counts!

FCX254 AVAILLOG is no longer produced.

z/VM Performance Toolkit: New Report FCX295 AVLA2GLG

```
FCX295  Run 2013/04/10 07:38:36          AVLA2GLG
                                         Available List Data Above 2G, by Time
From 2013/04/09 16:02:10
To   2013/04/09 16:13:10
For    660 Secs 00:11:00                 "This is a performance report for SYS
```

```
           <------------ Storage -------------> <--Times--> <-Frame Thresh-->
 Interval  <Available> <Requests/s> <Returns/s> <-Empty/s->  Sing <-Contigs->
 End Time   Sing  Cont  Sing   Cont  Sing  Cont  Sing  Cont   Low   Low  Prot
 >>Mean>>    23M  267M   47M    59M   47M   51M    .0    .0  1310    15    15
 16:02:40     0  938M   32M   126M  502K 30310    .0    .0  1332    15    15
 16:03:10   152K 4556K   50M    89M   49M   59M    .0    .0  1168    15    15
```

- Look for the new concepts: Singles Contigs Prot
- Amounts are in bytes, suffixed. Not page counts!
- FCX254 AVAILLOG is no longer produced.

z/VM Performance Toolkit: New Report FCX296 STEALLOG

```
FCX296  Run 2013/04/10 07:38:36          STEALLOG                                                    Page   62
                                         Frame Steal Statistics, by Time
From 2013/04/09 16:02:10                                                              SYSTEMID
To   2013/04/09 16:13:10                                                              CPU 2817-744   SN A6D85
For    660 Secs 00:11:00                 "This is a performance report for SYSTEM XYZ"  z/VM   V.6.3.0 SLU 0000
```

	Pct	<-------- Storage/s -------->							<-Completions/s->				<-Users/s->		<- Age List --->		
Interval	Time	Total	Write	<--User--->		<-Shared-->		<Pvt Vdisk>		AgeL	Need	Time	Sys	Travs			<-Stor	Skip/s-->	
End Time	Actv	Stoln	OnDmd	Inval	Reval	Inval	Reval	Inval	Reval	Reval	Met	Limit	Req	/s	Visit	Skip	Pin	Ser	Resv
>>Mean>>	2.6	71M	.0	61M	36M	16099	1589	.0	.0	15M	115.1	.0	.0	5.8	283.1	5.8	.0	.0	.0
16:02:40	.0	.0	.0	.0	123K	.0	136.5	.0	.0	4639K	.0	.0	.0	.0	.0	.0	.0	.0	.0
16:03:10	2.4	82M	.0	3085K	45M	69632	4506	.0	.0	5301K	111.5	.1	.0	2.3	25.4	2.2	.0	.0	.0
16:03:40	3.4	102M	.0	36M	70M	39595	.0	.0	.0	11M	236.1	.0	.0	6.7	203.2	6.7	.0	.0	.0
16:04:10	3.5	94M	.0	75M	37M	13926	1092	.0	.0	18M	124.4	.0	.0	7.5	363.9	7.5	.0	.0	.0
16:04:40	3.2	84M	.0	68M	37M	9148	1092	.0	.0	15M	39.2	.0	.0	5.7	303.4	5.7	.0	.0	.0
16:05:10	3.0	80M	.0	70M	36M	16521	2867	.0	.0	16M	122.1	.0	.0	6.9	345.1	6.9	.0	.0	.0
16:05:40	2.9	80M	.0	71M	41M	11332	1092	.0	.0	17M	135.5	.0	.0	7.0	340.7	6.9	.0	.0	.0
16:06:10	2.8	78M	.0	70M	40M	11742	1092	.0	.0	16M	131.7	.0	.0	6.7	330.8	6.7	.0	.0	.0
16:06:40	2.7	74M	.0	71M	35M	10240	2731	.0	.0	17M	134.8	.0	.0	6.8	341.8	6.8	.0	.0	.0

- Look for the new concepts: Stoln Inval Reval etc.
- Amounts are in bytes, suffixed. Not page counts!

z/VM Performance Toolkit: New Report FCX297 AGELLOG

```
FCX297  Run 2013/04/10 07:38:36          AGELLOG                                                    Page   63
                                         Age List Log, by Time
From 2013/04/09 16:02:10                                                          SYSTEMID
To   2013/04/09 16:13:10                                                              CPU 2817-744   SN
A6D85
For    660 Secs 00:11:00                 "This is a performance report for SYSTEM XYZ"   z/VM   V.6.3.0 SLU
0000
```

```
_
```

					<---------- Storage ---------->					<----- Revalidation ------>			
					<-- Steal Ready --->	<--- Not Ready ---->			%Of	<------- Storage/s ------->			
Interval	Size	S	E	<-List Size-->	<--RefOnly-->	<--Changed-->	<Evaluating->		Pages	<--RefOnly-->	<--Changed-->		
End Time	%DPA	Z	W	Target	Current	NoWrt	Write	Write	PndWrt	Refd	Change	Eval	NoWrt	Write	NoWrt	Write
>>Mean>>	2.0	V	Y	7800M	7793M	177M	2K	602M	5481M	951K	98K	10	8595K	.0	6154K	359300
16:02:40	2.0	V	Y	7800M	7653M	51816K	0	725M	6620M	0	0	10	657954	.0	3919K	79736
16:03:10	2.0	V	Y	7800M	7800M	27972K	0	747M	4243M	4812K	548K	10	1079K	.0	3697K	537395
16:03:40	2.0	V	Y	7800M	7800M	21472K	0	756M	2173M	2596K	0	10	7429K	.0	3532K	36045
16:04:10	2.0	V	Y	7800M	7799M	770M	36K	10452K	3069M	0	0	10	13340K	.0	4293K	427622
16:04:40	2.0	V	Y	7800M	7799M	660M	0	120M	4756M	0	0	10	11392K	.0	3982K	3140.3
16:05:10	2.0	V	Y	7800M	7800M	218M	0	559M	5175M	2900K	276K	10	10095K	.0	5406K	534528
16:05:40	2.0	V	Y	7800M	7799M	229M	0	551M	5398M	0	0	10	10542K	.0	6067K	428851
16:06:10	2.0	V	Y	7800M	7800M	205M	0	570M	5548M	3824K	368K	10	10395K	.0	5503K	326861
16:06:40	2.0	V	Y	7800M	7800M	157M	0	623M	5648M	760K	24K	10	9115K	.0	7718K	305562

- Look for the new concepts: Write PndWrt etc.
- Amounts are in bytes, suffixed. Not page counts!

Summary

z/VM Large Memory: Summary

- **Objective was to get rid of algorithmic constraints that stopped growth**
- **Things we got rid of:**
  - Reorder
  - Using the scheduler lists to visit users
  - Taking a large amount when we visit a user
  - Excessively favoring VDISKs as regards memory residency
  - Problems in evaluating depletion of available lists
  - Excessive or unnecessary rewriting of DASD
  - Dependency on long-running System z instructions
- **Things we added:**
  - Visiting all users round-robin
  - Taking only a little when we visit
  - Visiting VDISKs sooner
  - Detecting available list depletion a little more smartly
  - Scatter-to-scatter paging channel program
  - Using trial invalidation
- **Effect: workloads constrained on z/VM 6.2 should go better on z/VM 6.3**

References

- z/VM CP Planning and Administration
- z/VM CP Commands and Utilities
- z/VM Performance Report: www.vm.ibm.com/perf/

Thanks!

John Franciscovich
IBM
z/VM Design and Development
Endicott, NY

francisj@us.ibm.com

Session 14686