VM/ESA Data in Memory Techniques

SHARE 89 - Summer Meeting
Session 9223

Bill Bitner
IBM Corp
1701 North St.
Endicott, NY 13760
(607) 752-6022
bitner@vnet.ibm.com
USIB1E29 at IBMMAIL

Disclaimer

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used instead.

Any performance data contained in this document was determined in a controlled environment and, therefore, the results which may be obtained in other operating environments may vary significantly.

Users of this document should verify the applicable data for their specific environments.

It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country or not yet announced by IBM. Such references or information should not be construed to mean that IBM intends to announce such IBM products, programming, or services.

Should the speaker start getting too silly, IBM will deny any knowledge of his association with the corporation.

Trademarks

The following are trademarks of the IBM Corporation:

- IBM
- VM/ESA

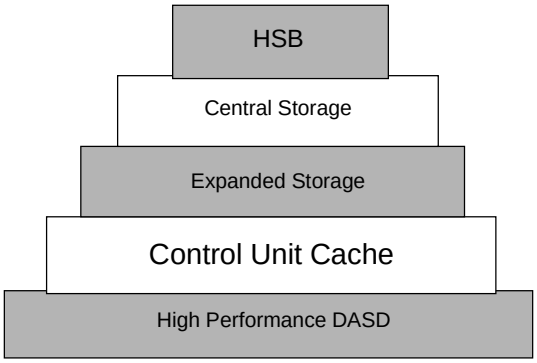
Introduction

- How many places in VM/ESA do we use the term "Caching"?
- What is the best method to speed up spool I/O?
- Should I use virtual disk in storage or minidisk cache to speed things up?
- Topics Covered:
  - data attributes
  - saved segments
  - minidisk cache
  - VM data spaces
  - virtual disk in storage
  - Control Unit Cache

We talk about data in memory or caching or buffering all the time with VM, but what all does it really involve. This presentation will attempt to describe the major DIM (data in memory) techniques available in VM/ESA and the strengths (or weaknesses) of each. Much of the material in this presentation was borrowed from other sources. In particular, I found a paper written by Kris Buelens and Guy De Ceulaer, both of IBM, to be very helpful. This paper is available on the VM home page. In addition, the redbook "VM/ESA Storage Management with Tuning Guidelines" has a lot of interesting reading.

Data in Memory

- Various techniques where data is kept in memory instead of going to the next level in the data access hierarchy to retrieve the data.
- Usually involves a trade-off of resources in higher level to avoid access of lower level.



Data in memory (DIM) concepts have been around for a long time. Any DIM presentation would be incomplete without the little pyramid diagram that shows the access being faster the higher up on the pyramid we go. DIM attempts to improve performance by avoiding the slower access of data on a lower level. However, this will often require greater use of the resources at a higher level. The top of this diagram shows high speed buffer (HSB) which is sometimes referred to as processor cache. It holds data and instructions that the processing unit can execute or process. Other than the jump between central and expanded storage, the delta in performance between levels is rather significant.

Data Attributes

Before asking which DIM technique to use, think about the attributes of the data.

- Does data need to be executed? (code)
- Read and write characteristics
  - Ratio read to write
  - Reference patterns
- CP or Guest or Preferred Guest I/O?
- Access density
- Temporary or permanent data?
- Is data shared between virtual machines? systems?

We often rush into asking whether one certain DIM method is better than another. This, in my opinion, is a mistake. You know most performance questions are answered with "it depends", so first, ask "on what does it depend?". The attributes listed here are some of the answers. Does the data need to be executed? VM data spaces are great, but you can not execute instructions directly from a VM data space. Do we mostly read this data or is there a lot of update activity as well? MDC can be great for read data, but does not benefit write I/O. The I/O interface used to get the data can be important as well, especially in a guest environment where you might be trading off processor resources (I/O Assist) for quicker access. For CP I/O, many of the features managed by CP (e.g. MDC or virtual disk in storage) cannot be used. Access density will play into how much storage up the upper level will be needed. If we can treat data as volatile, that allows more options for performance, such as virtual disk in storage or VM data spaces. Whether data is shared between systems or users, and the degree of sharing can be important as to which DIM method we use.

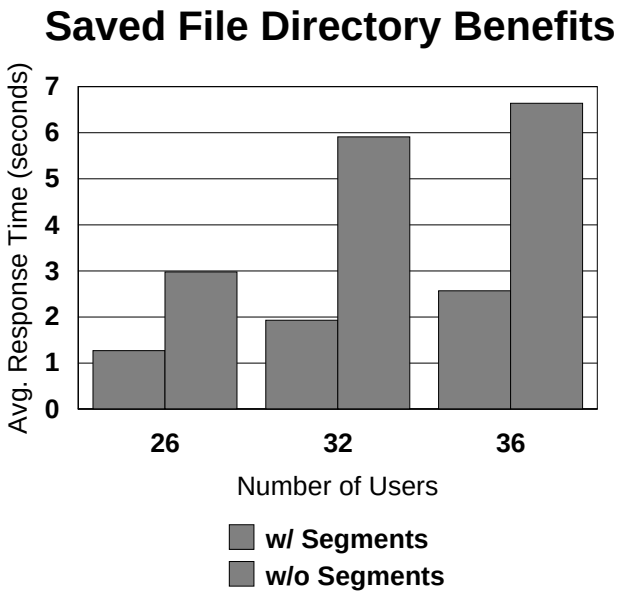
Saved Segments

- A range of pages of virtual storage you can define to hold data or reentrant code.
- Benefits:
  - Use virtual storage outside virtual machine size
  - Decreases I/O required to load the data
  - Can minimize real storage and I/O requirements by sharing the segment
- Uses:
  - FSTs saved via SAVEFD
  - Code in storage (CMSINST, Program Products)
- Benefit increases with degree of sharing

Saved segments are one of the more powerful tools. A saved segment is a range of pages of virtual storage you can define to hold data (which includes re-entrant code). This data can be shared between virtual machines.
We most often think about saved segments in terms of code for program products, but they can also be used for data. The benefits that can be gained from saved segments include using virtual storage outside the defined virtual machine size; descreasing I/O required to load code or data into storage; and when the segment is shared, can minimize storage and I/O resources. The benefit from this last area increases as the degree of sharing increases. Using SAVEFD to save the FSTs in a segment is highly recommeded.

Example of FSTs in Saved Segment

- Measurement results with CMS 5 on 4341-11 with 4MB central storage. 5000 files on tools disk.
- FST is 64 bytes per file. Savings: 64*5000*users



The chart above shows an example. It is admittedly an old example, but the dramatic results still hold true. These measurements were made on a 4341-11 with 4MB of central storage running VM/SP Release 5. The tools disk held 5000 files. There were only a few users on the system, by today's standards. It shows that the savings or improvement increases as the number of users (degree of sharing) increases. Now, while this was a very small machine and had slow I/O and paging, it also had much fewer users than VM/ESA systems of today.
An FST (file status table) entry is 64 bytes (for minidisk files). There is one FST for each file on the accessed minidisk. So the product of 64 times the number of files times gives us how much storage is required for the control blocks of an accessed minidisk. Without SAVEFD, this is the storage per user. So to compute total savings we would want to multiple times the number of users.

VM Data Spaces

- VM/ESA 1.1.1 - extended ESA/370 data spaces to DAT-off virtual machines
- Requires ESA-capable processor with ESA/390 VM data space support.
- Virtual machine mode XC
- Advantages:
  - Reduces storage requirements when shared
  - Reduces need to transfer data (IUCV, VMCF)
  - Increases virtual storage addressability to allow more virtual DIM
- Related features and other cool stuff
  - Mapped minidisks
  - Asynchronous page fault processing
  - Page reference services
  - Copy to Primary diagnose x'248'

There are two types of data spaces in the ESA world. A data space is like an address space, except that it contains data only and one cannot execute instructions directly from a data space. ESA/370 data spaces are sharable only by operating systems running in paging mode. They can exist on any ESA-capable processor. ESA/390 VM data spaces are a VM/ESA exclusive extension to the architecture which provides data spaces for DAT-off virtual machines such as CMS. This feature requires a processor with this feature (3090 and older processors do not provide this support). XC is the machine mode which provides VM data spaces. There are three major advantages to VM data spaces: they can reduce storage requirements when the data spaces are shared amongst users; they can reduce the need to transfer data by some communication interface; and they increase the virtual storage addressability to allow more buffering or caching in the virtual storage. There are a number of related features. Mapped minidisks allow one to map a minidisk to virtual storage in data spaces and reference the data by simply referencing the data space. Asynchronous page fault processing allows a virtual machine to handshake with CP so that a page fault does not serialize the entire virtual machine (not strictly a VM data space feature). An application can give hints to CP about paging by using Page Reference Services. While XC mode is required to directly access a VM data space, diagnose x'248' can be used to copy data from an address space to the primary space.

VM Data Spaces in Use

- SQL/DS and DSS feature
  - Mapped and unmapped DBspaces
  - Page Reference Services
  - Asynchronous Page Fault processing
  - Strength lies in added virtual storage and using CP paging I/O
- SFS Dircontrol Directories
  - FSTs and part of ADT in data space
  - File content mapped into data space
  - Similar performance to read-mostly minidisk
- VS FORTRAN V2R5
  - Uses extended addressing for large arrays
- OV/VM Calendar Feature
  - Internal data spaces and through SFS

Here are some examples of how and where VM data spaces are used today. SQL/DS has a DSS feature which is a very complete and extensive exploitation of VM data spaces. DSS uses mapped minidisks extensively as well as data spaces for work areas. Features such as page reference services and asynchronous page fault processing are also used. Parameter settings allow for control over just how much the data spaces are exploited. SQL uses the strength of CP paging I/O and added virtual storage addressing for its performance gains. The data spaces are not shared with the end users.

SFS uses of VM data spaces for dircontrol directories differ from SQL/DS DSS. SFS relies on the strength of sharing data spaces for its benefits. By keeping FSTs, part of the ADT, and the actual file data in VM data spaces, SFS minimizes communication between users and the server and shares data by keeping only one copy for the entire system in storage. Thus you get shared FSTs without crowding the virtual storage below the 16 meg line.

In addition, products such as VS FORTRAN and the OV/VM Calendar feature, uses VM data spaces. Non-IBM products also exploit VM data spaces.

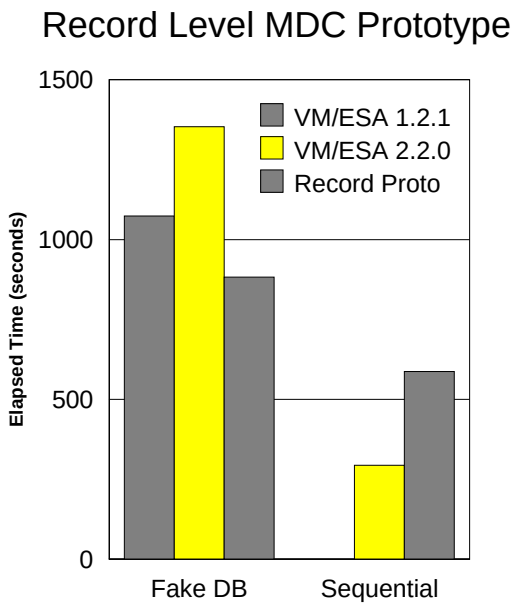
Minidisk Cache

- Enhanced in VM/ESA 1.2.2
  - Real or expanded storage
  - CMS and non-CMS minidisks and I/O
- Provides a read cache for minidisk I/O.
- No support for caching shared DASD volumes.
- Can cache minidisks shared between users
- System-wide cache
- NOMDCFS turns off fair share limit

Minidisk cache was enhanced in VM/ESA 1.2.2 to allow for caching in real storage and for non-CMS I/O and minidisks. MDC provides a good performing read cache. Some limitations are no specific support for minidisks shared between VM systems and trade off for preferred guest I/O. MDC is implemented as a system wide cache and allows for the minidisks to be shared between users. The fair share limits protect anyone user from using more than their fair share. This can be disabled for a particular user by using NOMDCFS directory option.

Minidisk Cache - Record Level

- MDC with track caching can result in worse performance for large database jobs.
- Prototype measured which provides record level MDC for CMS data.
- Development APAR VM61045 for 2.1.0 and 2.2.0.
- SET MDC MDISK option in APAR (and QUERY MDC)
- Directory options in future



When VM/ESA 1.2.2 enhanced minidisk cache, we knew there was potential for extreme workloads to run worse under the new MDC. Since then a number of customers have run into problems when running large databases that are built off of CMS flat files. "Large" is an important term. In these case, large means files that are 100s of cylinders in size.

Currently customers have found relief by increasing processor and or CU storage sizes, tuning the system, or tuning the application. We know this is not acceptable as a long term solution.

Some prototype measurements are shown of an enhancement that became APAR VM61045. (on RSU 9703 for 2.1.0 and 9706 for 2.2.0). This code adds a flavor of record-level MDC back into the system. It applies only to CMS data accessed through certain CMS I/O interfaces.

The measurements made included an application that modeled a customers database access pattern. You can see that not only is performance improved over the track MDC, but it is also better than the original record level MDC seen in the VM/ESA 1.2.1 measurement. A second workload was run that accessed a file sequentially. The results show that the track level MDC remains the better performer in this case.

Virtual Disk in Storage

- Introduced in VM/ESA 1.2.1
- Simulated FBA DASD
- Backed by an ESA/370 data space (system utility space)
- Appears like a minidisk on a 9336 device
- Volatile
- Fast read and write I/O

Virtual disk in storage was added in VM/ESA 1.2.1. It provides a volatile FBA minidisk. Volatile in that when the last link to the minidisk is dropped, the disk and its contents are lost. It is implemented as a system utility space (i.e. ESA/370 data space). It does not require XC mode. Virtual disk in storage provides very fast read and write I/O, however this is at a trade off in storage requirements.

Virtual Disk Trade Offs

- May make I/O constrained work CPU or storage constrained
- Think of it as additional working set
  - less paging area for others
  - sytem space has better priority in steal processing
  - minimize pages referenced (blocks touched)
- Keep eye on storage requirements of key servers

One needs to respect the tradeoffs being made for virtual disk in storage. Respect keeps you out of trouble. It is often helpful to think of a virtual disk in storage as just another working set or address space on the system. For best performance, you want to minimize the pages referenced in the space. For virtual disk in storage, this translates to minimizing the virtual disk blocks referenced. The total size of the virtual disk controls the upper bound. Note that unlike other PGMBKs (page management blocks), the ones for the virtual disk space are not pageable.

As a system space, it is given better treatment. Regular user virtual storage pages will be stolen before a page that represents the virtual dasd blocks. The storage associated with a virtual disk in storage is not considered part of the user working set, but part of the system storage.

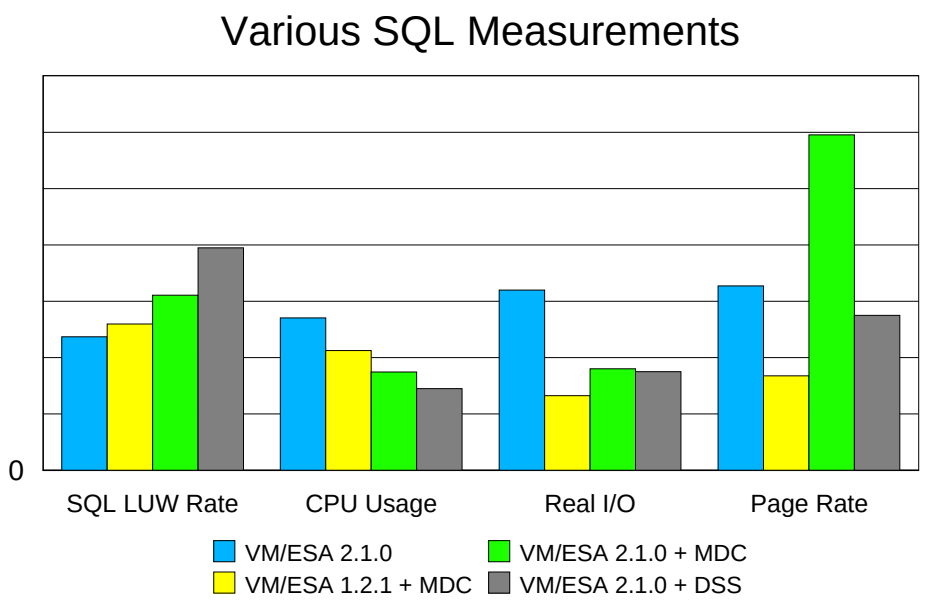
Also, keep an eye on the storage requirements of key servers on the system. You may never have needed to reserve pages prior to virtual disk in storage, but now you might need to. However, do not think that virtual disk in storage is a bad idea just because you need to reserve pages. It is just another trade off.

Control Unit Cache

- Lower on the hierarchy
- Caches all I/O including CP and preferred guest I/O
- Cache between shared volumes
- Write cache on some models
- Some is transparent to you (no stats and/or no control)

Control unit cache is lower on the hierarchy (well as low as I'm going in this presentation). The major strength here is that CU cache can help CP and preferred guest I/O, not just virtual minidisk I/O. Some caching control units support write cache and in many of these cases this is non-volatile. Control unit cache can also help when the data is on volumes shared between systems. Over the years, higher RAID levels have been implemented to provide cost-effective RAS solutions. Cache is often required in these environments to get acceptable performance. In some cases, the control unit provides a cache, but there is no information from VM provided on cache statistics or there is no control from the VM system to enable or disable the cache.

Example Comparison



This chart shows a collection of measurements from a customer running SQL/DS. The results show various tradeoffs and challenges when running different configurations. The four environments measured were: a plain SQL/DS system on VM/ESA 2.1.0, an SQL/DS system on VM/ESA 1.2.1, SQL/DS with MDC on VM/ESA 2.1.0, and SQL/DS plus Data Space Support on VM/ESA 2.1.0. Remember that VM/ESA 1.2.1 still used a record level cache for MDC. The first set of bars shows the rate of start LUWs for SQL. This is a measure of work. The most dramatic change here is with the DSS feature.The other three measures (CPU Usage, Real I/O, and Page Rate) are all normalized to the workload. While VM/ESA 2.1.0 with MDC was lower in CPU usage and did well with in the Real I/O area, the increase in paging caused severe response time problems.

Summary

DIM Method	CP ?	Write ?	Read ?	Code ?	NonVol ?
Saved Segments	No	No	Yes	Yes	No/Yes
VM Data Space	No	Yes	Yes	No	Yes/No
Virtual Disk	No	Yes	Yes	n/a	No
MDC	No	No	Yes	n/a	Yes
CU Cache	Yes	Yes	Yes	n/a	Mostly Yes

- There are various data in memory techniques in VM to choose from.
- Choose one based on the data attributes and the resources you have to trade for better performance.
- There are also non-IBM alternatives to the above and additional/complementary products that can help.

This presentation made an attempt to discuss DIM concepts and how they apply to VM. The table above tries to summarize different data attributes and which DIM methods match those attributes. Again, I would recommend looking at the attributes to help you understand which methods you should apply. In addition, let me note that for special requirements there are non-IBM products that are alternatives and extensions to the features discussed here.

References

1. VM/ESA 2.2.0 Performance SC24-5782.
2. "VM/ESA Data IN Memory Techniques" page, adapted from a paper by Kris Buelens and Guy De Ceulaer -- IBM Belgium. See http://www.vm.ibm.com/perf/tips/dim.html
3. VM/ESA 2.2.0 Planning and Administration SC24-5750.
4. VM/ESA 2.2.0 CP Programming Services SC24-5760.
5. ITSC Redbook "VM/ESA Storage Management with Tuning Guidelines", GG24-3934-01.
6. "Minidisk Cache Guidelines" page. See http://www.vm.ibm.com/perf/tips/prgmdcar.html