MQ V8 for z/OS Bufferpools

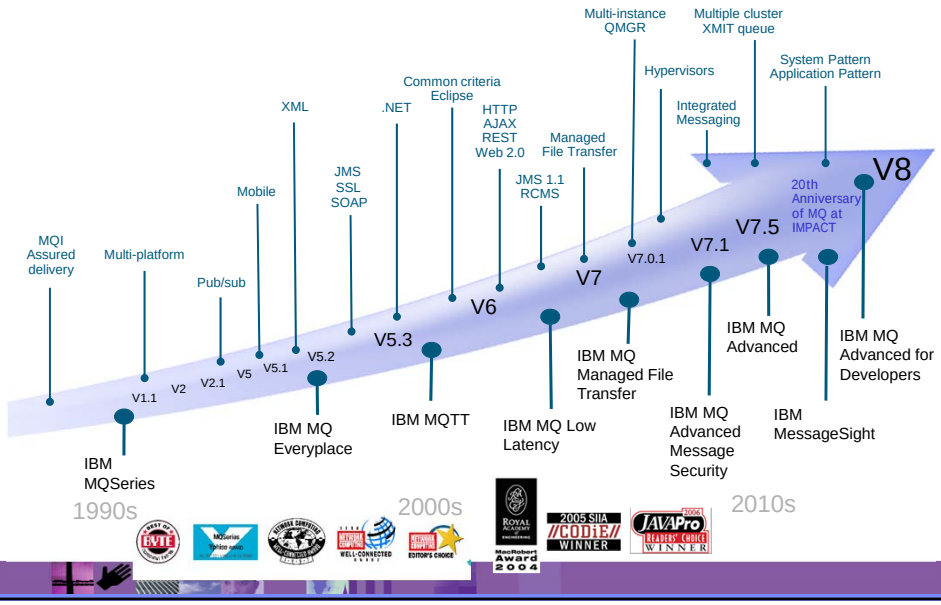
WebSphere software

Lyn Elkins – elkinsc@us.ibm.com

Mitch Johnson – mitchj@us.ibm.com

Wildfire
Technical Hands-On Workshops
IBM zGrowth Team

Timeline



Agenda

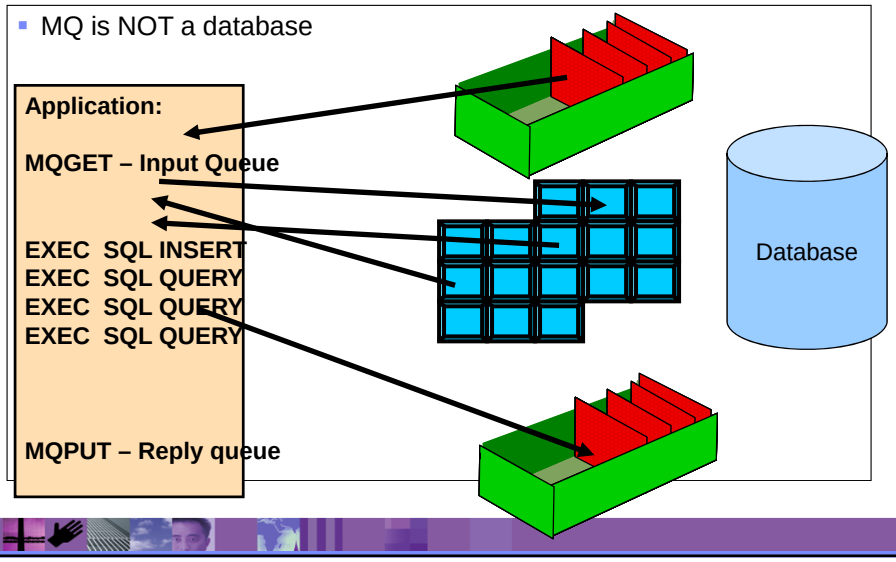
- Warnings and considerations
- How messages are stored in MQ
- Above the bar (64-bit) bufferpools
- Advantages and maybe not so much
- Summary
-

Warnings and considerations

- Having more bufferpools and pages is not the ultimate solution
  - If your performance problems are log I/O related this will not help you
  - Allocating more, and larger, bufferpools may just expose other issues
- Using above the bar bufferpools may require the purchase of additional memory

One of these things is not like the other

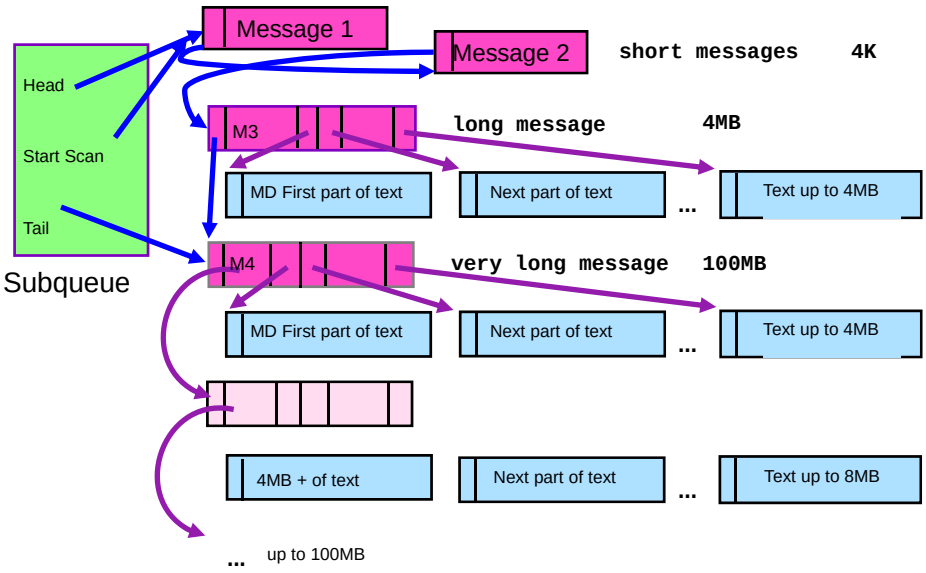
- MQ is NOT a database



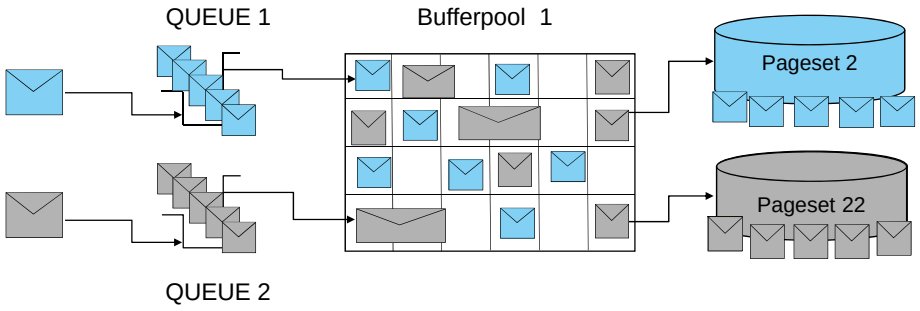
How private queue messages are stored in MQ

- Shared queue message storage is different
- This section will discuss how messages are stored internally for private queues
  - This does not change no matter where the bufferpool is allocated

Private Queue Message Storage



Internally the messages are mapped something like this



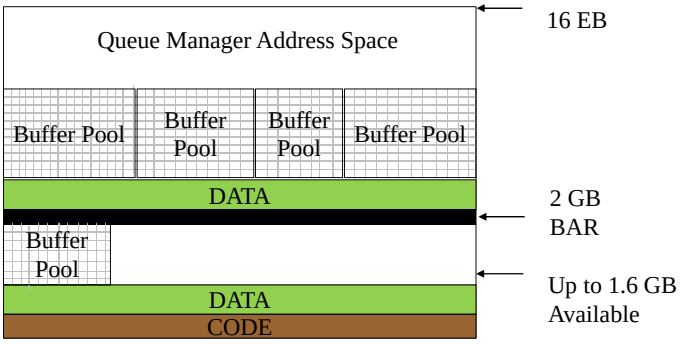
MQ for z/OS: 64-bit bufferpools

- 64-bit buffer pools in MQ for z/OS
  - Allows large numbers of messages to be cached before writing to pagesets
  - Allows MQ to exploit the vast amount of storage on today's machines
- Improves performance of putting/getting messages by minimizing disk I/O
  - What's the BEST I/O?
- Minimizes administrative overhead of managing buffer pools
  - You can now have bufferpools 0-99
- Buffer pool LOCATION attribute says where it is located relative to the bar
  - BELOW: The default. Buffer pool is located below the bar in 31 bit storage
  - ABOVE: Buffer pool is located above the bar in 64 bit storage
  - This can be altered dynamically
- Storage can be pagefixed based on pageclass attribute

IBM MQ for z/OS Wildfire Workshop

MQ for z/OS: 64-bit bufferpools

- Buffer pools above the bar can (theoretically) use up to 16 EB storage
- Increased maximum size of pool to 999,999,999 buffers
  - Was 500,000
- Allows up to 100 buffer pools
  - Was 16
  - Can have a 1-1 page set – buffer pool mapping
- The storage available above the bar
  - Set by MEMLIMIT or systems settings



Definition Changes

- To implement expanded bufferpools
  - New attributes added to the BUFFPOOL commands
    - LOCATION
      - BELOW – default, the pool is taken from the 2G address space
      - ABOVE – the pool is allocated above the 2G bar
    - PAGECLAS
      - 4KB – each 4K page is pageable by the operating system
        - This is the only option for pools defined in BELOW the bar storage
      - 4KBFIXED – each 4K page is fixed in memory
    - REPLACE/NOREPLACE
      - Should this definition override the what is held in the log of the queue manager?

```
DEFINE BUFFPOOL( 0 ) BUFFERS( 50000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
DEFINE BUFFPOOL( 1 ) BUFFERS( 20000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
DEFINE BUFFPOOL( 2 ) BUFFERS( 50000 ) LOCATION( ABOVE ) +
       PAGECLAS( 4KB )   REPLACE
DEFINE BUFFPOOL( 3 ) BUFFERS( 20000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
DEFINE BUFFPOOL( 10 ) BUFFERS( 1000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
DEFINE BUFFPOOL( 11 ) BUFFERS( 1000 ) LOCATION( ABOVE ) +
       PAGECLAS( 4KB) REPLACE
DEFINE BUFFPOOL( 12 ) BUFFERS( 1000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
DEFINE BUFFPOOL( 13 ) BUFFERS( 1000 ) LOCATION( ABOVE ) +
       PAGECLAS( 4KB )   REPLACE
DEFINE BUFFPOOL( 14 ) BUFFERS( 1000 ) LOCATION( BELOW ) +
       PAGECLAS( 4KB ) NOREPLACE
```


MEMLIMIT

- V7.1 MEMLIMIT

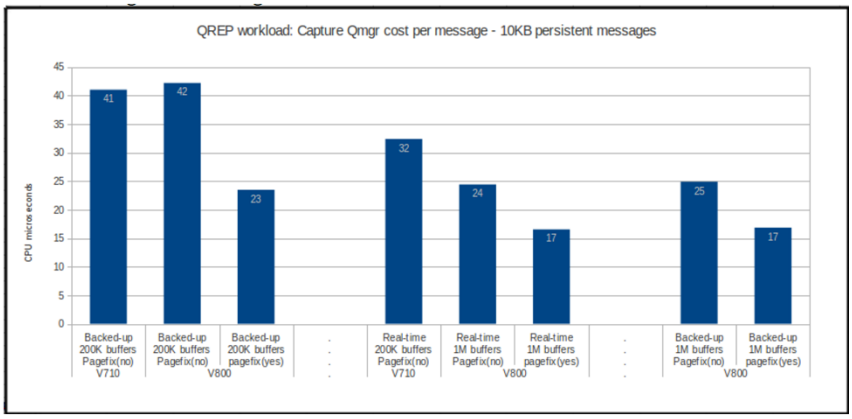
```
//QML1MSTR PROC
//PROCSTEP EXEC PGM=CSQYASCP,REGION=0M MEMLIMIT=2G
//*
```

  - Allocates 64-bit storage for MQ's use
    - Indexes
    - Security cache
    - SMDS buffers
- V8.0 MEMLIMIT

```
//QML1MSTR PROC
//PROCSTEP EXEC PGM=CSQYASCP,REGION=0M MEMLIMIT=7G
//*
```

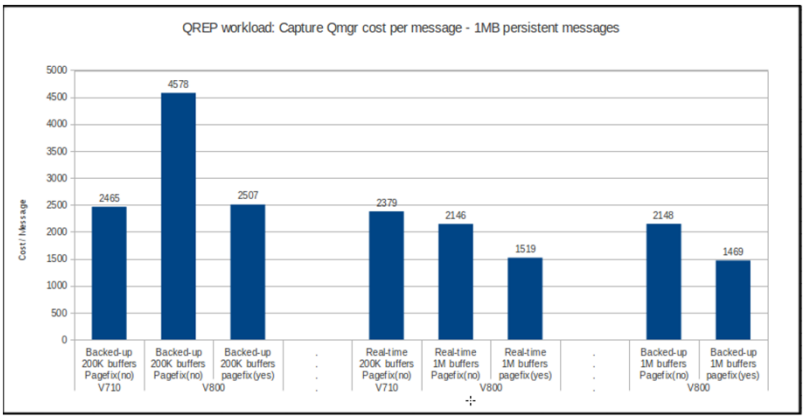
  - Same storage areas and the above the bar bufferpools (sized for example in MQ V8 Redbook)

QREP and similar workload performance



IBM MQ for z/OS Wildfire Workshop

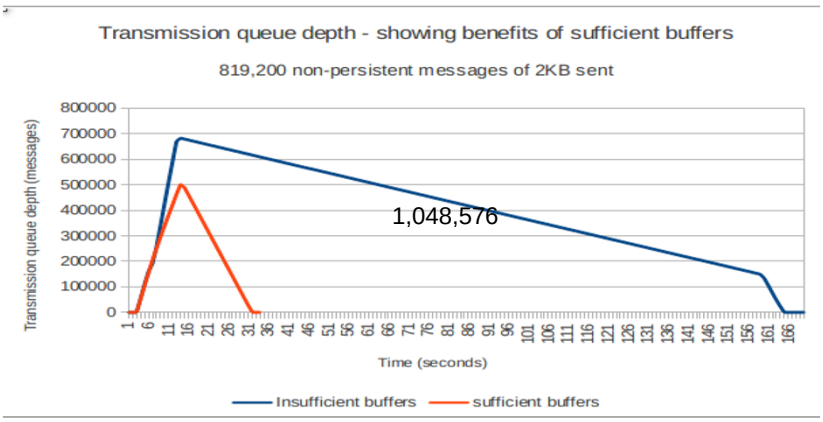
QREP and similar workload performance



The benefit of sufficient buffers

- While QREP was one of the focus areas when testing above the bar bufferpools, any application that overfills a bufferpool will benefit.
- Another example illustrated in the upcoming MP1J, MQ for z/OS performance, is the effect of having sufficient buffers on a normal transmission queue.
- In this example a transmission queue has filled due to an unavailable receiver queue. Please note the following:
  - The below the bar bufferpool had 200,000 pages – which is quite large for a buffer pools
  - The above the bar bufferpools had a **1,048,576 page (**4GB) buffer pool for the transmission queue.
  - Each test put the same number of messages, batching them at 200 messages per commit.
  - The large bufferpool peaked at 500,000 messages and did not have to write any to the pageset during processing.

Comparing time to drain a transmission queue



Calculating Memory requirements

- As of MQ V7.1 the queue manager set the MEMLIMIT to 2G
  - This storage is used by MQ for things like the INDEX storage, security cache, etc.
  - If bufferpools are above the bar are being allocated, more storage may be needed.
  - To calculate the additional storage requirements:
    - Multiply the number of pages by 4096
    - Calculate the number 200-byte control blocks required for all messages (all bufferpools)
    - Tally the storage requirements and add that to the 2G MEMLIMIT

64 Bit Buffer Pools: Performance Summary

Single Requester per Queue:

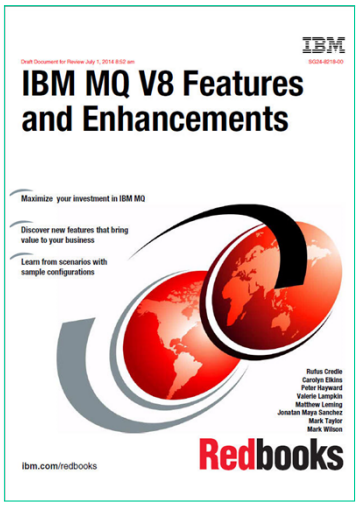
Test	Transaction Rate (per second)	Transaction Cost (cpu microseconds)	LPAR %Busy	Channel Path %Busy
31-bit	232762	35.92	54%	56%
64-bit	235217	37.48	57%	57.4%
64-bit (enough buffers)	324213	38.12	83%	0.07%
64-bit (4GB per buffer pool)	341412	38.23	83%	0.08%

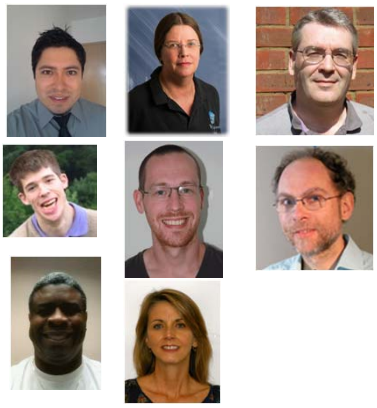
2 Requesters per Queue:

Test	Transaction Rate (per second)	Transaction Cost (cpu microseconds)	LPAR %Busy	Channel Path %Busy
31-bit	149140	42.3	42%	75.4%
64-bit	145623	44.84	43.5%	75.9%
64-bit (enough buffers)	384062	40.65	99.59%	0.08%
64-bit (4GB per buffer pool)	370546	52.15	99.69%	0.07%

•16 CP LPAR
•Each transaction puts and gets a random message from a pre loaded queue. Second test requires a doubling in buffer pool size

For more info … Already available





https://www.redbooks.ibm.com/Redbooks.nsf/RedpieceAbstracts/sg248218.html

Questions???