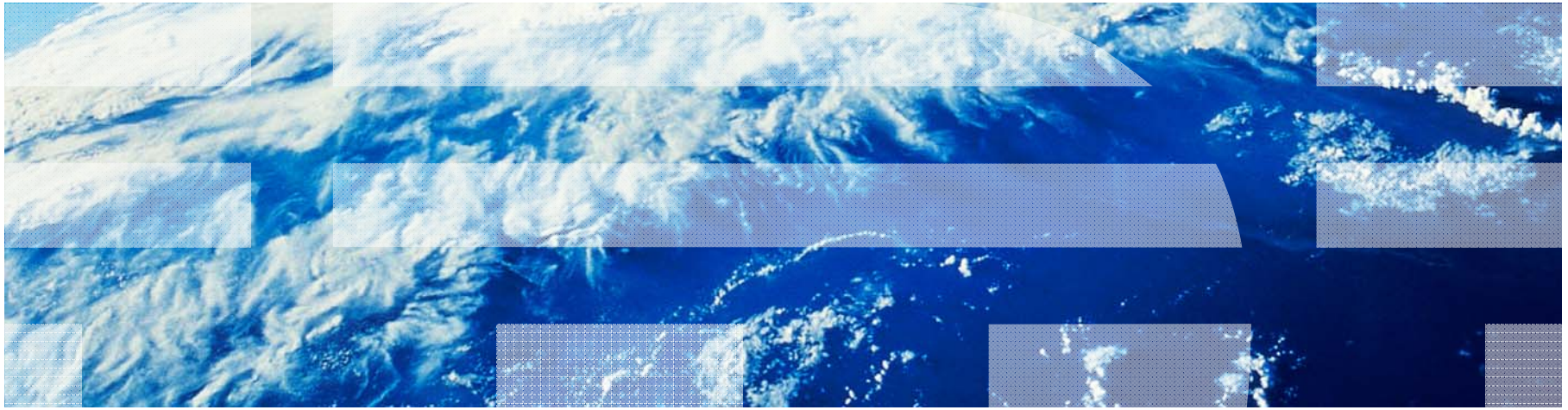
PE11(AIX)

AIX 5L/6 Performance Tuning Part II: Tactics for Tuning Indicated Performance Issues

Earl Jew
IBM Field Technical Sales Specialist for Power Systems and Storage
IBM Regional Designated Specialist - Power/AIX Performance and Tuning
400 North Brand Blvd., Suite 700 c/o IBM, Glendale, CA, USA 91203
earlj@us.ibm.com (310)251-2907



IBM Power Systems Technical University 2010: Las Vegas – Earl Jew

- A Comparative Overview of 2010 IBM Storage Technology Offerings for Power P/AIX
  - *SO03(Storage) Tuesday 13:00-14:15 Palma A&B*
  - *SO03(Storage) Wednesday 9:00-10:15 Rio Pavilion 5*

- Part I: Tactics for Monitoring Indications of Performance Issues
  - *PE10(AIX) Monday 16:15-17:30 Conga*
  - *PE10(AIX) Thursday 9:00-10:15 Amazon I*

- **Part II: Tactics for Tuning Indicated Performance Issues**
  - *PE11(AIX) Wednesday 13:00-14:15 Rio Pavilion 5*
  - *PE11(AIX) Thursday 16:15-17:30 Amazon I*

- Part III: Real-world Case-studies illustrating the Recognition and Remedy of Performance Issues
  - *PE12(AIX) Wednesday 16:15-17:30 Rio Pavilion 5*
  - *PE12(AIX) Friday 9:00-10:15 Amazon J*

RawIO vs LVM/JFS2:
Which provides the highest performance/throughput? RawIO
Which is more convenient to create and manage? LVM/JFS2

Most agree: An RDBMS-on-RawIO is fast but very inconvenient to create&manage.

Most agree: An RDBMS-on-LVM/JFS2 is slower but far easier to create&manage.

If an RDBMS must run as fast as the technology can manage, then use RawIO.

Otherwise, our challenge is making an RDBMS-on-LVM/JFS2 run as fast as it can.

This presentation is about optimizing performance of an RDBMS-on-LVM/JFS2, with consideration for RawIO.

Strategic Thoughts, Concepts, Considerations, and Tactics

- Devise tactics to relieve exhaustions by exploiting surplus resources
  - Determine points of exhaustion, limitation, and over-commitment
  - Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

- Recognize-and-Remedy the “bottlenecks” in AIX VMM resources
  - Study the mechanics of AIX Virtual Memory Management (VMM)
  - Practice monitoring the behaviors of the AIX VMM mechanisms
  - Understand the influence of *vmo/ioo/no* tuning parameters on AIX VMM behaviors

- Match/place RDBMS “tablespaces” with the best mount-options or Go-Raw
  - Exercise&experiment with the various JFS2 mount-options as well as Going Raw
  - Devise ways to characterize I/O patterns in routinely-active RDBMS “tablespaces”

Strategic Thoughts: Monitoring AIX 5L/6.1 LPARs

- Many AIX performance-degrading scenarios can be readily characterized by monitoring AIX dynamically (real-time) as well as cumulatively (ie. vmstat –sv).
- By understanding and interpreting the output of mundane AIX commands better&deeper, areas of resource exhaustion, limitation and over-commitment, as well as, resource under-utilization, surplus and over-allocation, can be distinguished.

- This presentation focuses on the tactical -- meaning your daily “keyboard awareness”.
- This will explain the numbers presented by AIX commands (*vmstat, mpstat, iostat, ps*, etc.) and formulate the severity of performance issues, if any.

- Most cumulative indicators are counts-per-scale over days-uptime.
- Many dynamic indicators are comparing ranges&ratios of system resources.
- **Scaled-definitions** define **blue/surplus, green/normal, yellow/warning, red/serious** and Flashing-Red-with-Sirens/**critical** status-conditions.

Strategic Concepts: Monitoring AIX 5L/6.1 LPARs

- Can the capabilities&capacities (C&C) of the infrastructure manage the workload?
- Is there an appropriate balance of hardware capabilities&capacities for the workload?
- To answer, AIX must be monitored relative to its hardware resources&infrastructure.

- Note the size, scale, technology and implementation of the given LPAR
- Note the LPAR's ratio-of-resources, ie. CPU-to-RAM-to-SAN I/O for the workload
- Note the same of other same-frame "sibling" LPARs, if any

- Review the historical/accumulated count-of-events over days-uptime
- Determine points of exhaustion, limitation, and over-commitment
- Determine surplus resources: CPUcycles, RAM, SAN I/O throughput, etc.
- Review exhaustions and surpluses in-light of workload expectations

- Monitor dynamic AIX behaviors to characterize the given workload
- For example: Is this a Think-Think or a Move-the-Data workload?

Strategic Considerations: Monitoring AIX 5L/6.1 LPARs

- Monitor dynamic AIX behaviors using a 1 or 2 second sampling interval (vs >30secs)
- Verify a stressful workload exists: "We can't tune what is not being taxed"
- Discontinue active efforts when done: "If/when it runs fast enough, we're tuned"
- Build with track-able discrete structures: "We can't tune what can't be tracked"
- Monitor spikes,peaks,bursts and burns: "We tune the intensities, not the sleepy-times"
- Establish dynamic baselines by monitoring real-time AIX behaviors by ranges&ratios
- Watch AIX behaviors with the goal of characterizing the workload **(`vmstat -Iwt 2`)**

```
System Configuration: lcpu=48 mem=174080MB

   kthr           memory                          page                    faults           cpu       time
---------- -------------------- ---------------------------------- ----------------- ----------- --------
  r   b   p        avm        fre    fi    fo    pi    po    fr     sr    in     sy    cs us sy id wa hr mi se
 23  12   0   16896791    6008393 10421  7588     0     0  6336   6693  5488 2166351 648185 34 27 27 12 13:23:04
  0   0   0   16918278    5986559  9317  2992     0     0  2207   2361  5551 1992963 630548 32 27 27 14 13:23:05
 24  16   0   16908381    5996756  5554  2118     0     0  5702   6059  5366 2176668 645810 32 27 28 13 13:23:06
 26  17   0   16902538    6002492  6608  2412     0     0  6606   7143  6174 2275536 680525 33 25 27 15 13:23:07
 19  21   0   16897984    6007108  5991  1907     0     0  5948   6331  5623 3373138 700564 32 26 28 14 13:23:08
 18  18   0   16897189    6007718  5659  1793     0     0  5432   5713  5285 2053519 665453 28 24 33 15 13:23:09
 21  13   0   16898035    6006971  6530  1552     0     0  6600   6946  5853 1918286 714013 28 24 34 14 13:23:10
 23  17   0   16896745    6008235  5987  2032     0     0  5937   6284  5531 2390617 774752 31 26 30 14 13:23:11
 29  18   0   16898852    6006076  7040  3969     0     0  6868   7550  6050 2663508 811273 32 28 26 14 13:23:12
 21  20   0   16900383    6004434  8590  7483     0     0  8284   8852  8094 2583303 805351 34 29 24 14 13:23:13
 36  15   0   16896715    6010736 11738 16017     0     0  7487   8181  8730 2502543 790705 35 29 26 10 13:23:14
 24  14   0   16900246    6004726 11607 10387     0     0  5444   5983  6291 2971775 820218 35 28 26 11 13:23:15
 27  11   0   16898966    6006775 12471 13520     0     0  6065   6665  6831 1947291 863025 35 29 26 10 13:23:16
```

Strategic Tactics: Monitoring/Tactically-Tuning AIX 5L/6.1 LPARs

- Monitor AIX behaviors with the goal of characterizing the workload
- Use the workload characterization to guide AIX 5L/6.1 tactical-tuning efforts

Tuning Strategy example 1

- Determine points of exhaustion, limitation, and over-commitment
- Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.
- Devise tactics to relieve exhaustions by exploiting surplus resources

Tuning Strategy example 2

- Study the mechanics of AIX Virtual Memory Management (VMM)
- Understand the influence of *vmo/ioo/no* tuning parameters on AIX VMM dynamic behaviors
- Practice monitoring the behaviors of the AIX VMM mechanisms
- Recognize-and-Remedy the “bottlenecks” in AIX VMM resources

Tuning Strategy example 3

- Exercise&experiment with the various JFS2 mount-options as well as Going Raw
- Devise ways to characterize I/O patterns in routinely-active RDBMS “tablespaces”
- Match/place RDBMS “tablespaces” with the best JFS2 mount-options including Going Raw

- Strategic Thoughts, Concepts, Considerations, and Tactics

- **Devise tactics to relieve exhaustions by exploiting surplus resources**
  - Determine points of exhaustion, limitation, and over-commitment
  - Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

- Recognize-and-Remedy the “bottlenecks” in AIX VMM resources
  - Study the mechanics of AIX Virtual Memory Management (VMM)
  - Practice monitoring the behaviors of the AIX VMM mechanisms
  - Understand the influence of *vmo/ioo/no* tuning parameters on AIX VMM behaviors

- Match/place RDBMS “tablespaces” with the best mount-options or Go-Raw
  - Exercise&experiment with the various JFS2 mount-options as well as Going Raw
  - Devise ways to characterize I/O patterns in routinely-active RDBMS “tablespaces”

Virtually without exception, change these **AIX 5.3** default values

- Set `vmo:lru_file_repage=0`; default=1 # **Mandatory critical change**
  - This change directs `lrud` to steal only JFS/JFS2 file-buffer pages unless/until `numperm/numclient` is less-than/equal-to `vmo:minperm%`, at which point `lrud` begins stealing both JFS/JFS2 file-buffer pages and computational memory pages.
  - Essentially stealing computational memory invokes pagingspace-pageouts.
  - I have found this change already made by most AIX 5.3 customers.

- Set `vmo:page_steal_method=1`; default=0 # helpful, not critical
  - This change switches the `lrud` page-stealing algorithm from a physical memory address page-scanning method (=0) to a List-based page-scanning method (=1).

- Set `ioo:sync_release_ilock=1`; default=0 # helpful, not critical
  - Default value =0 means that the i-node lock is held while all dirty pages of a file are flushed; thus, I/O to a file is blocked when the **syncd** daemon is running. Setting =1 will cause a **sync**() to flush all I/O to a file without holding the i-node lock, and then use the i-node lock to do the commit.

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ uptime; vmstat -s
02:28PM   up 270 days,  20:29,  19 users,  load average: 27.21, 37.41, 42.25
        842229409191 total address trans. faults
        246764231347 page ins
        203496170416 page outs
             5727761 paging space page ins
            15309228 paging space page outs
                   0 total reclaims
        183629569151 zero filled pages faults
          1647822525 executable filled pages faults
        915567964999 pages examined by clock
              362090 revolutions of the clock hand
        358382319357 pages freed by the clock
          2516004617 backtracks
           130450223 free frame waits
                   0 extend XPT waits
         24009520583 pending I/O waits
        450099399145 start I/Os
         40534913894 iodones
        188600873632 cpu context switches
        107406884121 device interrupts
         19811930837 software interrupts
         37095518459 decrementer interrupts
           103705513 mpc-sent interrupts
           103671523 mpc-received interrupts
          6976371410 phantom interrupts
                   0 traps
      1076715870409 syscalls
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ uptime; vmstat -v
02:28PM   up 270 days,  20:29,  19 users,  load average: 27.21, 37.41, 42.25
             8388608 memory pages
             7961714 lruable pages
               19421 free pages
                   3 memory pools
             2180469 pinned pages
                80.0 maxpin percentage
                10.0 minperm percentage
                90.0 maxperm percentage
                37.9 numperm percentage
             3019754 file pages
                 0.0 compressed percentage
                   0 compressed pages
                37.9 numclient percentage
                90.0 maxclient percentage
             3019754 client pages
                   0 remote pageouts scheduled
            12018771 pending disk I/Os blocked with no pbuf
             9551839 paging space I/Os blocked with no psbuf
                2484 filesystem I/Os blocked with no fsbuf
                   0 client filesystem I/Os blocked with no fsbuf
            55225325 external pager filesystem I/Os blocked with no fsbuf
                   0 Virtualized Partition Memory Page Faults
                0.00 Time resolving virtualized partition memory page faults
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
# lvmo -a -v rootvg
vgname = rootvg
pv_pbuf_count = 512
total_vg_pbufs = 1024          # total_vg_pbufs/pv_pbuf_count = 1024/512 = 2 LUNs comprise rootvg:
max_vg_pbuf_count = 16384
pervg_blocked_io_count = 90543
pv_min_pbuf = 512
global_blocked_io_count = 12018771
…
…
# lvmo -a -v apvg15
vgname = apvg15
pv_pbuf_count = 512
total_vg_pbufs = 15872         # total_vg_pbufs/pv_pbuf_count = 15872/512 = 31 LUNs comprise apvg15:
max_vg_pbuf_count = 524288
pervg_blocked_io_count = 517938
pv_min_pbuf = 512
global_blocked_io_count = 12018771

# lvmo -a -v pgvg01
vgname = pgvg01
pv_pbuf_count = 512
total_vg_pbufs = 1024          # total_vg_pbufs/pv_pbuf_count = 1024/512 = 2 LUNs comprise pgvg01:
max_vg_pbuf_count = 16384
pervg_blocked_io_count = 8612687
pv_min_pbuf = 512
global_blocked_io_count = 12018771
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ vmstat -Iwt 2

System configuration: lcpu=18 mem=32768MB ent=6.00

   kthr           memory                          page                    faults                cpu             time
----------- --------------------- ------------------------------------ ------------------ ----------------------- --------
  r   b   p       avm        fre    fi    fo    pi    po    fr     sr    in     sy    cs us sy id wa    pc    ec hr mi se
 25   3   0   5403734      18626 17042  6736     0     0  6497  24593  4103  37276  7087 85 15  0  0  6.80 113.3 14:28:16
 22   4   0   5403132      18759 19591  8210     0     0 27150 107775  3790  39022  6245 85 15  0  0  6.84 114.0 14:28:18
 28   2   0   5407123      18292 17030  8713     0     0 27473 103637  3617  39771  6359 84 16  0  0  6.81 113.5 14:28:20
 30   3   0   5418622      18786 12707 11550     0     0 29271 108882  4158  71405  6649 86 14  0  0  6.81 113.5 14:28:22
 19   3   0   5414411      18794 13177  8559     0     0 19272  64745  4339  36455  7449 88 12  0  0  6.84 114.0 14:28:24
 28   4   0   5416616      18525 13023 10181     0     0 24070  85363  4435  33957  7941 86 13  0  0  6.78 113.0 14:28:26
 25   3   0   5472944      18249  9108  5960     0     0 42642 156140  3258  60123  6341 85 15  0  0  6.58 109.7 14:28:28
108   1   0   5619126      18633  2114  1807     0     0 76629 383833  2928  82847  6936 85 15  0  0  6.56 109.4 14:28:30
 73   5   0   5545685      75755  9729  9528     0     0 10840  41984  5499  88876 21629 87 12  0  0  6.68 111.3 14:28:32
 24   8   0   5436253     132608 12152 14570     0     0     0      0  4278  53647  9979 86 14  0  0  6.43 107.1 14:28:34
 79   3   0   5547792      19025 10822  2538     0     0 11493  43139  3543  82226  7429 88 12  0  0  6.48 108.1 14:28:36
208   1   0   5673214      18408  2754  2388     0     0 66976 365501  2443  69464  5481 86 14  0  0  6.53 108.8 14:28:38
164   1   0   5701925      31006  4768  6468     0     0 31420 152572  3426  75063 13962 88 12  0  0  6.94 115.7 14:28:40
 25   6   0   5613435      63434 10954 17812     0     0     0      0  3595  56182 15972 86 14  0  0  6.63 110.5 14:28:42
 18   5   0   5481337     145518 14634 10809     0     0     0      0  4488  40217 16754 87 13  0  0  6.70 111.6 14:28:44
 21   5   0   5408368     152678 12081 21144     0     0     0      0  4545  51876  8548 86 14  0  0  6.43 107.1 14:28:46
 39   1   0   5486801      38491  7932 10307     0     0     0      0  2976  86471  6668 90 10  0  0  6.88 114.6 14:28:48
181   1   0   5605899      18456  2696  3451     0     0 55412 203724  2056  68444  4977 87 13  0  0  6.67 111.2 14:28:50
203   0   0   5670273      20271  3484  2410     0     0 37625 183709  2076  88672  8015 90 10  0  0  6.50 108.3 14:28:52
119   1   0   5631053      54296  6963  7904     0     0 11675  40951  3123  35073 13698 90 10  0  0  6.39 106.5 14:28:54
   kthr           memory                          page                    faults                cpu             time
----------- --------------------- ------------------------------------ ------------------ ----------------------- --------
  r   b   p       avm        fre    fi    fo    pi    po    fr     sr    in     sy    cs us sy id wa    pc    ec hr mi se
 43   1   0   5405007     235360 11588 11259     0     0     0      0  4718 110398 16216 87 13  0  0  6.36 106.1 14:28:56
 26   1   0   5401711     190079 13925 10499     0     0     0      0  4365 471926  7553 91  9  0  0  6.77 112.9 14:28:58
 37   2   0   5416248     112153 12428 20164     0     0     0      0  4720  33316  7453 90 10  0  0  6.92 115.3 14:29:00
 26   3   0   5402388      78319 14677  9921     0     0     0      0  5639  54747  8715 92  8  0  0  7.49 124.8 14:29:02
 23   3   0   5401223      18984 14682 16610     0     0   849   2843  5124  44305  7869 92  8  0  0  7.54 125.7 14:29:04
 28   3   0   5395071      18536 13429 22088     0     0 31416  97936  4757  32289  7480 88 12  0  0  7.57 126.1 14:29:06
 22   1   0   5394300      18406 13274 13593     0     0 26274  89779  4941  32586  7512 91  9  0  0  7.53 125.4 14:29:08
 23   1   0   5393822      18538 12715 10443     0     0 22100  56253  5807  30463  8032 90 10  0  0  7.48 124.6 14:29:10
 22   4   0   5394369      18345 11872 14275     0     0 25415  76097  4171  32998  6334 88 12  0  0  7.25 120.8 14:29:12
 21   4   0   5395734      18556 11271 16273     0     0 27796  89028  3468  70431  6281 86 14  0  0  6.73 112.2 14:29:14
 21   3   0   5399263      18982 12180  8788     0     0 22973  67836  3980  29017  7071 89 11  0  0  6.79 113.1 14:29:16
 23   2   0   5403524      18923 12234 12432     0     0 26389  72108  4131  40505  7164 90 10  0  0  6.94 115.7 14:29:18
 22   4   0   5401685      19327 11426 13672     0     0 24234  75055  3461  39043  6602 89 11  0  0  6.74 112.3 14:29:20
 30   7   0   5399467      19172 13270 20191     0     0 32198  99442  3794  31399  6558 85 15  0  0  6.75 112.4 14:29:22
 30   2   0   5408432      19178 15532 10837     0     0 30625  91603  3656  59516  7554 86 14  0  0  6.81 113.5 14:29:24
```

`vmstat -s` # Writes to standard output the contents of the sum structure, which contains an absolute count of paging events since system initialization.

address translation faults
Incremented for each occurrence of an address translation page fault. I/O may or may not be required to resolve the page fault. Storage protection page faults (lock misses) are not included in this count.

page ins
Incremented for each page read in by the virtual memory manager. The count is incremented for page ins from page space and file space. Along with the page out statistic, this represents the total amount of real I/O initiated by the virtual memory manager.

page outs
Incremented for each page written out by the virtual memory manager. The count is incremented for page outs to page space and for page outs to file space. Along with the page in statistic, this represents the total amount of real I/O initiated by the virtual memory manager.

paging space page ins
Incremented for VMM initiated page ins from paging space only.

paging space page outs
Incremented for VMM initiated page outs to paging space only.

...

pages examined by the clock
VMM uses a clock-algorithm to implement a pseudo least recently used (lru) page replacement scheme. Pages are *aged* by being examined by the clock. This count is incremented for each page examined by the clock.

revolutions of the clock hand
Incremented for each VMM clock revolution (that is, after each complete scan of memory).

pages freed by the clock
Incremented for each page the clock algorithm selects to free from real memory.

[Continued]

`vmstat -s` # [continued] Writes to standard output the contents of the sum structure, which contains an absolute count of paging events since system initialization.

backtracks
Incremented for each page fault that occurs while resolving a previous page fault. (The new page fault must be resolved first and then initial page faults can be *backtracked*.)

free frame waits
Incremented each time a process requests a page frame, the free list is empty, and the process is forced to wait while the free list is replenished.

extend XPT waits
Incremented each time a process is waited by VMM due to a commit in progress for the segment being accessed.

pending I/O waits
Incremented each time a process is waited by VMM for a page-in I/O to complete.

start I/Os
Incremented for each read or write I/O request initiated by VMM.

iodones
Incremented at the completion of each VMM I/O request.

CPU context switches
Incremented for each processor context switch (dispatch of a new process).

device interrupts
Incremented on each hardware interrupt.

software interrupts
Incremented on each software interrupt. A software interrupt is a machine instruction similar to a hardware interrupt that saves some state and branches to a service routine. System calls are implemented with software interrupt instructions that branch to the system call handler routine.

decrementer interrupts
Incremented on each decrementer interrupt.

…

`vmstat -v` # Writes to standard output various statistics maintained by the Virtual Memory Manager. The **-v** flag can only be used with the **-s** flag.

memory pages
Size of real memory in number of 4 KB pages.

lruable pages
Number of 4 KB pages considered for replacement. This number excludes the pages used for VMM internal pages, and the pages used for the pinned part of the kernel text.

free pages
Number of free 4 KB pages.

memory pools
Tuning parameter (managed using **vmo**) specifying the number of memory pools.

pinned pages
Number of pinned 4 KB pages.

maxpin percentage
Tuning parameter (managed using **vmo**) specifying the percentage of real memory which can be pinned.

minperm percentage
Tuning parameter (managed using **vmo**) in percentage of real memory. This specifies the point below which file pages are protected from the re-page algorithm.

maxperm percentage
Tuning parameter (managed using **vmo**) in percentage of real memory. This specifies the point above which the page stealing algorithm steals only file pages.

numperm percentage
Percentage of memory currently used by the file cache.

[Continued]

`vmstat -v` # [Continued] Writes to standard output various statistics maintained by the Virtual Memory Manager. The **-v** flag can only be used with the **-s** flag.

file pages
Number of 4 KB pages currently used by the file cache.

…

numclient percentage
Percentage of memory occupied by client pages.

maxclient percentage
Tuning parameter (managed using vmo) specifying the maximum percentage of memory which can be used for client pages.

client pages
Number of client pages.

…

pending disk I/Os blocked with no pbuf
Number of pending disk I/O requests blocked because no pbuf was available. Pbufs are pinned memory buffers used to hold I/O requests at the logical volume manager layer.

paging space I/Os blocked with no psbuf
Number of paging space I/O requests blocked because no psbuf was available. Psbufs are pinned memory buffers used to hold I/O requests at the virtual memory manager

filesystem I/Os blocked with no fsbuf
Number of filesystem I/O requests blocked because no fsbuf was available. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer.

client filesystem I/Os blocked with no fsbuf
Number of client filesystem I/O requests blocked because no fsbuf was available. NFS (Network File System) and VxFS (Veritas) are client filesystems. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer.

external pager filesystem I/Os blocked with no fsbuf
Number of external pager client filesystem I/O requests blocked because no fsbuf was available. JFS2 is an external pager client filesystem. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer.

…

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ uptime; vmstat -s
 12:46AM   up 139 days,   1:29,  0 users,  load average: 9.24, 4.21, 2.99
         36674080366 total address trans. faults
        303594828999 page ins                      # filesystem reads from disk; vmstat:page:fi
         65127100071 page outs                     # filesystem writes to disk; vmstat:page:fo
                  17 paging space page ins         # vmstat:page:pi
                 166 paging space page outs        # vmstat:page:po
                   0 total reclaims
         10153151099 zero filled pages faults
              379929 executable filled pages faults
        790677067990 pages examined by clock       # vmstat:page:sr
              102342 revolutions of the clock hand
        323578511315 pages freed by the clock      # vmstat:page:fr
           216779474 backtracks
           173781776 free frame waits
                   0 extend XPT waits
         13118848968 pending I/O waits
        369118024444 start I/Os
         21394237531 iodones
        115626032109 cpu context switches          # vmstat:faults:cs
         25244855068 device interrupts             # fc/ent/scsi interrupts; vmstat:faults:in
          3124067547 software interrupts
         14571190906 decrementer interrupts        # lcpu decrementer “clock” interrupts
            56397341 mpc-sent interrupts
            56396919 mpc-receive interrupts
            32316580 phantom interrupts
                   0 traps
        739431511068 syscalls
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ uptime; vmstat -v
 12:46AM   up 139 days,   1:29,  0 users,  load average: 9.24, 4.21, 2.99
           16318464 memory pages
           15690385 lruable pages
               5271 free pages
                  2 memory pools
            3290976 pinned pages
               80.0 maxpin percentage
                5.0 minperm percentage
               80.0 maxperm percentage
               73.3 numperm percentage
           11508824 file pages
                0.0 compressed percentage
                  0 compressed pages
               73.3 numclient percentage
               80.0 maxclient percentage
           11508824 client pages
                  0 remote pageouts scheduled
           14108165 pending disk I/Os blocked with no pbuf
                  0 paging space I/Os blocked with no psbuf
               2228 filesystem I/Os blocked with no fsbuf
                383 client filesystem I/Os blocked with no fsbuf
          572404510 external pager filesystem I/Os blocked with no fsbuf
                  0 Virtualized Partition Memory Page Faults
               0.00 Time resolving virtualized partition memory page faults
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
# lvmo -a -v rootvg
vgname = rootvg
pv_pbuf_count = 512
total_vg_pbufs = 1024      # total_vg_pbufs/pv_pbuf_count = 1024/512 = 2 LUNs comprise rootvg:
max_vg_pbuf_count = 16384
pervg_blocked_io_count = 384
pv_min_pbuf = 512
global_blocked_io_count = 14108165
…
…
# lvmo -a -v bkup3vg
vgname = bkup3vg
pv_pbuf_count = 512
total_vg_pbufs = 3072      # total_vg_pbufs/pv_pbuf_count = 3072/512 = 6 LUNs comprise bkup3vg:
max_vg_pbuf_count = 65536
pervg_blocked_io_count = 14104581
pv_min_pbuf = 512
global_blocked_io_count = 14108165

# lvmo -a -v tspdsvg
vgname = tspdsvg
pv_pbuf_count = 512
total_vg_pbufs = 10240     # total_vg_pbufs/pv_pbuf_count = 10240/512 = 20 LUNs comprise tspdsvg:
max_vg_pbuf_count = 65536
pervg_blocked_io_count = 0
pv_min_pbuf = 512
global_blocked_io_count = 14108165
```

Determine points of exhaustion, limitation, and over-commitment
Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

```
$ vmstat -Iwt 2

System configuration: lcpu=16 mem=63744MB

   kthr           memory                          page                      faults           cpu       time
----------- --------------------- ------------------------------------ ------------------ ----------- --------
  r   b   p        avm        fre    fi    fo    pi    po    fr     sr    in     sy    cs us sy id wa hr mi se
  8  12   0    4774322       4812 126320     8     0     0 126599 383255  5929  52461  4988 59 24  2 15 00:46:30
 11  10   0    4774086       5029 137106    10     0     0 136814 462957  5207  55496  4544 63 24  2 11 00:46:32
 13   7   0    4774166       4980 141378     4     0     0 141081 384636  5467  59121  4446 65 25  1  9 00:46:34
 17   5   0    4774086       5170 138429    10     0     0 138939 310915  5709  56538  4554 65 24  1 10 00:46:36
  8  13   0    4774086       5037 141245    15     0     0 141284 353826  5107  58776  4336 65 25  1 10 00:46:38
 13   7   0    4774086       5371 136991     0     0     0 136566 323578  5948  57384  4880 63 25  2 11 00:46:40
  8  13   0    4774088       5237 139668    14     0     0 139323 332047  5540  57614  4799 64 25  2  9 00:46:42
 17   6   0    4774362       5409 142134    11     0     0 142301 345935  4359  58811  3844 66 24  1  9 00:46:44
 11   9   0    4774085       4937 136534     7     0     0 135960 305148  5774  56377  4621 63 24  2 11 00:46:46
 15   6   0    4774085       5048 134094     7     0     0 133845 325715  5679  53900  4847 62 24  2 12 00:46:48
 12   8   0    4774085       5198 128991    19     0     0 128666 322754  5021  52163  4501 60 24  2 14 00:46:50
 11   9   0    4774085       4914 131840     0     0     0 130851 297965  5540  52615  4413 61 24  2 12 00:46:52
 11   9   0    4774091       5113 140019     4     0     0 139699 344134  4849  56378  4002 65 25  1  9 00:46:54
 11  12   0    4774091       5321 145256    42     0     0 145967 356880  4478  60992  3468 67 27  1  6 00:46:56
 13   8   0    4774174       4787 143054   100     0     0 142896 336635  4557  58782  3416 65 28  1  6 00:46:58
 13   9   0    4774247       4677 146945    15     0     0 146980 393169  5064  62780  4441 67 27  1  6 00:47:00
 17   6   0    4774495       4771 147824    73     0     0 147967 362560  4985  61154  4347 68 26  1  6 00:47:02
 12  11   0    4774603       5127 146230    41     0     0 146557 373037  4341  56926  3951 68 26  1  6 00:47:04
  8  15   0    4774171       5147 146123    36     0     0 145475 440309  4758  58606  4141 68 25  1  6 00:47:06
 13   9   0    4774171       5340 144846    60     0     0 144601 380263  4730  54216  4240 66 26  1  7 00:47:08
   kthr           memory                          page                      faults           cpu       time
----------- --------------------- ------------------------------------ ------------------ ----------- --------
  r   b   p        avm        fre    fi    fo    pi    po    fr     sr    in     sy    cs us sy id wa hr mi se
 11  10   0    4774171       4967 142111    26     0     0 141856 443839  5623  54031  5273 65 26  1  8 00:47:10
 12   8   0    4774171       5128 142800    28     0     0 143306 347534  5975  54293  5422 65 26  1  8 00:47:12
 11  10   0    4774171       5250 138248     8     0     0 137183 426141  5799  51232  5433 63 26  1 10 00:47:14
 12   9   0    4774157       5187 136132    23     0     0 136488 345613  6058  55570  5321 63 26  2 10 00:47:16
 12   9   0    4774156       4784 136341     6     0     0 136261 372360  6108  50819  5054 64 25  2 10 00:47:18
 13   8   0    4774160       5083 146439     2     0     0 146938 397102  5471  57358  4675 67 25  1  7 00:47:20
  9  12   0    4774160       5121 143074    34     0     0 142776 393705  5757  53468  4913 65 25  1  8 00:47:22
 14   7   0    4774159       5157 148219     2     0     0 147934 406539  5706  54665  4960 66 26  2  6 00:47:24
 10  10   0    4774159       5037 145049    15     0     0 143521 457916  6290  53916  5606 65 27  1  7 00:47:26
 11  10   0    4774159       5086 143519    16     0     0 144003 360368  6233  54209  5682 65 25  1  9 00:47:28
 12   9   0    4774159       3197 145260     3     0     0 143900 377970  5511  54682  5032 65 25  2  8 00:47:30
 10   9   0    4462574     140352 144664    12     0     0 57948 169669  5418  56517  4794 62 25  2 11 00:47:32
  8  12   0    4462814       4954 144484    64     0     0 76281 221144  6125  71414  6600 61 25  2 11 00:47:34
 13  10   0    4462931       5268 146550    86     0     0 147300 335611  6258  71903  6651 63 29  2  7 00:47:36
 10  11   0    4462896       5053 142026    65     0     0 141733 421671  5819  74578  6511 61 29  2  8 00:47:38
 11  10   0    4462261       5118 136256    48     0     0 135386 334709  5080  75260  6171 59 28  3 10 00:47:40
```

12018771 pending disk I/Os blocked with no pbuf # 270days
14108165 pending disk I/Os blocked with no pbuf # 139days

There are three tactics for relieving **pbuf** exhaustion.

- Add more **LUN**s to the LVM:VG that needs them per **lvmo -a -v $VG** output
  - Each LUN adds its allocation of **pv_min_pbuf=512**(default) **pbuf**'s to the LVM:VG pool

- Use **AIX:lvmo** to manually add more **pbuf**'s to the LVM:VG that needs them
  - **AIX:lvmo** requires root-user privilege to execute

- Set **AIX:ioo:pv_min_pbuf=2048**; this will increase **all** **pbuf** allocations by 4X
  - Default value: **AIX:ioo:pv_min_pbuf=512**

15309228 paging space page outs # 270days
9551839 paging space I/Os blocked with no psbuf # 270days

- **psbuf** exhaustion is best-resolved by precluding pagingspace-pageout events
- Allocation of **psbuf**s is static in pinned memory and cannot be increased
- **psbuf** exhaustion is a relative-measure of pagingspace-pageout intensity
  - i.e. too many pagingspace-pageouts occurring in a short span-of-time
  - Confirm by observing ratio of:
    paging space page outs : paging space I/Os blocked with no psbuf
- If sudden computational-memory overgrowth is for-purpose/by-design and/or cannot be avoided, then create a highly write-expedient AIX:pagingspace
  - Create a dedicated LVM:VG called pagingspacevg
  - Map multiple LUNs of equal size and RAID characteristics to the LVM:VG
  - Map each LUN (hdisk) directly to one LVM:LV (that is, no PP-striping, etc.)
  - A convenient total size for AIX:pagingspace is a multiple of **lruable pages**
  - Ensure a generous allocation of **pbuf**s to LVM:VG:pagingspacevg (see above)
  - The goal of this tactic is **WYSIWYG**: What you see is what you get
- AIXperftuning is easier when you build monitor-able/track-able “firm” structures

2228 filesystem I/Os blocked with no fsbuf
383 client filesystem I/Os blocked with no fsbuf

- **filesystem I/Os blocked with no fsbuf** **# mostly JFS**
  - If many, increase **ioo:numfsbufs to** 512,1024 or 2048 per severity of blocked I/Os
  - Default value of **ioo:numfsbufs=192**
  - JFS fsbufs are per-filesystem static-allocations in pinned memory
  - Must re-mount (**umount; mount**) filesystems for effect

- **client filesystem I/Os blocked with no fsbuf** **# NFS/Veritas**
  - If substantial blocks using **vxfs**, and blocks are not due to NFS I/O, then either:
    - research how to tune Veritas-on-AIX; ***sorry, I rarely encounter Veritas/vxfs on AIX***
    - re-examine "Why Veritas/vxfs?" and/or re-implement using JFS2
  - If substantial blocks and not using **vxfs**, then verify which NFS version is in-use
    - it may be: NFS V2 (old), NFS V3 (likely) or NFS V4 (new)
    - increase **nfso:nfs_v3_pdts, nfs_v3_vm_bufs** or the like per version in-use
    - client fsbufs are per-filesystem static-allocations in pinned memory
    - Must re-mount (**umount; mount**) filesystems for effect

55225325 external pager filesystem I/Os blocked with no fsbuf
572404510 external pager filesystem I/Os blocked with no fsbuf

- **external pager filesystem I/Os blocked with no fsbuf # JFS2**
  - If substantial blocked I/Os, increase **ioo:j2_nBufferPerPagerDevice**
  - Increase to 768,1024 or 2048 of static fsbuf allocations per severity of blocked I/Os
  - Default value of **ioo:j2_nBufferPerPagerDevice=512**
  - JFS2 fsbufs are per-filesystem static-allocations in pinned memory
  - Must re-mount (**umount; mount**) filesystems for effect
  - Increase **ioo:j2_dynamicBufferPreallocation** (see next below)

- **external pager filesystem I/Os blocked with no fsbuf # JFS2**
  - If substantial blocked I/Os, also increase **ioo:j2_dynamicBufferPreallocation**
  - Increase to 256 for greater dynamic fsbuf allocations when static fsbufs are exhausted
  - Default value of **ioo:j2_dynamicBufferPreallocation=16**
  - When no longer needed, JFS2 dynamic fsbufs are returned/released to freemem

- Strategic Thoughts, Concepts, Considerations, and Tactics

- Devise tactics to relieve exhaustions by exploiting surplus resources
  - Determine points of exhaustion, limitation, and over-commitment
  - Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

- **Recognize-and-Remedy the "bottlenecks" in AIX VMM resources**
  - Study the mechanics of AIX Virtual Memory Management (VMM)
  - Practice monitoring the behaviors of the AIX VMM mechanisms
  - Understand the influence of **`vmo/ioo/no`** tuning parameters on AIX VMM behaviors

- Match/place RDBMS "tablespaces" with the best mount-options or Go-Raw
  - Exercise&experiment with the various JFS2 mount-options as well as Going Raw
  - Devise ways to characterize I/O patterns in routinely-active RDBMS "tablespaces"

Study the mechanics of AIX Virtual Memory Management (VMM)
Practice monitoring the behaviors of the AIX VMM mechanisms

- **pages examined by the clock** (and thus not freed)
  VMM uses a clock-algorithm to implement a pseudo least recently used (lru) page replacement scheme. Pages are *aged* by being examined by the clock. This count is incremented for each page examined by the clock. The dynamic-counterpart to this is **`vmstat:page:sr`**
- **revolutions of the clock hand**
  Incremented for each VMM clock revolution (that is, after each complete scan of memory).
- **pages freed by the clock** (and thus not examined)
  Incremented for each page the clock algorithm selects to free from real memory. The dynamic-counterpart to this is **`vmstat:page:fr`**
- **free frame waits**
  Incremented each time a process requests a page frame, the free list is empty, and the process is forced to wait while the free list is replenished.

```
System configuration: lcpu=18 mem=32768MB ent=6.00
   kthr           memory                          page                    faults                cpu             time
---------- --------------------- ------------------------------------ ----------------- ---------------------- --------
  r    b    p        avm        fre    fi    fo    pi    po    fr      sr    in     sy    cs us sy id wa    pc     ec hr mi se
 25    3    0    5403734      18626 17042  6736     0     0  6497   24593  4103  37276  7087 85 15  0  0  6.80 113.3 14:28:16
 22    4    0    5403132      18759 19591  8210     0     0 27150 107775  3790  39022  6245 85 15  0  0  6.84 114.0 14:28:18
 28    2    0    5407123      18292 17030  8713     0     0 27473 103637  3617  39771  6359 84 16  0  0  6.81 113.5 14:28:20
 30    3    0    5418622      18786 12707 11550     0     0 29271 108882  4158  71405  6649 86 14  0  0  6.81 113.5 14:28:22
 19    3    0    5414411      18794 13177  8559     0     0 19272  64745  4339  36455  7449 88 12  0  0  6.84 114.0 14:28:24
 28    4    0    5416616      18525 13023 10181     0     0 24070  85363  4435  33957  7941 86 13  0  0  6.78 113.0 14:28:26
 25    3    0    5472944      18249  9108  5960     0     0 42642 156140  3258  60123  6341 85 15  0  0  6.58 109.7 14:28:28
108    1    0    5619126      18633  2114  1807     0     0 76629 383833  2928  82847  6936 85 15  0  0  6.56 109.4 14:28:30
```

130450223 free frame waits # 270days
173781776 free frame waits # 139days

- Check if all-the-following is true:
  - substantial **free frame waits** over days-uptime
  - persistent bursts&burns of vmstat **fr:sr** "scanning&freeing" activity in **vmstat -Iwt 2**
  - **[vmstat:page:fi** + **vmstat:page:fo]** is often greater than **vmstat:memory:fre**
  - **vmo -L** shows **vmo:minfree=960**(default) and **vmo:maxfree=1088**(default)

- If all-the-above is true, then increasing both **vmo:minfree** and **vmo:maxfree** will reduce the incidence of **free frame waits**, but not eliminate them.

```
# vmo -L
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT           TYPE
------------------------------------------------------------------------------
maxfree                   1088   1088   1088   8      200K   4KB pages        D
------------------------------------------------------------------------------
minfree                   960    960    960    8      200K   4KB pages        D
------------------------------------------------------------------------------
```

130450223 free frame waits # 270days
173781776 free frame waits # 139days

Increasing both **vmo:minfree** and **vmo:maxfree** reduces **free frame waits**.

Both **vmo:minfree** and **vmo:maxfree** are **counts of 4k mempages per memory pool.**

To illustrate: Assume 3 mempools and **vmo:minfree=960** and **vmo:maxfree=1088**
- **sys_minfree** = 3 mempools * **minfree** = 3 * 960 = 2880 (default minfree value)
- **sys_maxfree** = 3 mempools * **maxfree** = 3 * 1088 = 3264 (default maxfree value)

Thus when **vmstat:memory:fre** is less_than **sys_minfree=2880**, **lrud** begins to **fr:sr** (or scan&steal) pages until **vmstat:memory:fre** is greater_than **sys_maxfree=3264.**

- LPARs with more installed gbRAM tend to have more mempools, but not always.
- LPARs with more installed gbRAM should target higher **sys_minfree** and **sys_maxfree**.

130450223 free frame waits # 270days
173781776 free frame waits # 139days

Both **vmo:minfree** and **vmo:maxfree** are allocations of 4k mempages **per memory pool.**

Lets assume 4 mempools and **vmo:minfree=(4*2048)** and **vmo:maxfree=(5*2048)**

sys_minfree = 4 mempools * **minfree** = 4 * (4*2048) = 32768

sys_maxfree = 4 mempools * **maxfree** = 4 * (5*2048) = 40960

- For your LPAR, find the count of memory pools in the **vmstat -v** command output.
- LPARs have different counts of memory pools and amounts of installed gbRAM
  - LPARs w/4-12gbRAM should target sys_minfree=12288 & sys_maxfree=16384
  - LPARs w/12-24gbRAM should target sys_minfree=16384 & sys_maxfree=20480
  - LPARs w/24-36gbRAM should target sys_minfree=30720 & sys_maxfree=36864
  - LPARs w/36-72gbRAM should target sys_minfree=40960 & sys_maxfree=49152
  - LPARs w/72-128gbRAM should target sys_minfree=51200 & sys_maxfree=61440
- Note: I recommend setting **ioo:j2_maxPageReadAhead=2048**, hence 2048 above.

Understand the influence of **vmo/ioo/no** tuning parameters on AIX Behaviors

- **15309228 paging space page outs** # 270days
  166 paging space page outs # 139days
- No matter the days-uptime, if there is greater than 5-digits of **paging space page outs**, the root-cause warrants your attention -- to the exponential-degree beyond 5-digits.
- If **AIX 5.3**, then ensure **vmo:lru_file_repage=0** (default=1). If =1, then enjoy your bonus for miraculously improving system performance&throughput; **change this to =0**.

- Execute **vmstat -v** and compare the following values/settings:
  - **minperm** should be 10, 5 or 3; default=20
  - **maxperm** should be 80 or higher; default=80 or 90
  - **maxclient** should be 80 or higher; default=80 or 90
  - **numperm** real-time percent of non-computational memory (includes client below)
  - **numclient** real-time percent of JFS2/NFS/vxfs filesystem buffer-cache
- **paging space page outs** are triggered when **numperm** or **numclient** is less-than-or-equal-to **minperm**. Typically **numperm** and **numclient** is greater than **minperm**, and as such, no **paging space page outs** can be triggered.

[Continued]

```
15309228 paging space page outs # 270days
     166 paging space page outs # 139days
```

[Continued]

- **paging space page outs** are triggered when **numperm** or **numclient** is less-than-or-equal-to **minperm**. Typically **numperm** and **numclient** is greater than **minperm**, and as such, no **paging space page outs** can be triggered.

- Routinely execute **vmstat –Iwt 2** to monitor [**vmstat:memory:avm * 4096**] relative to the amount of installed gbRAM. If/as this approaches-or-exceeds the installed gbRAM, **paging space page outs** are triggered, likely causing mysteriously erratic and unfounded Stop&Go system performance. This is notably ugly, but it not AIX's fault…

- This is not an error-condition; rather, it is normal but horrendously-poor performance. Nothing of this will be noted in **errprt**. **Note: Do not confuse with a "system hang".**

- When **paging space page outs** are triggered, something has grown too big, or too many, or both. The cause may be an unexpected dramatic-increase in RDBMS user-connections, i.e. when typically only ~300 user-sessions grows to over ~2600 user-sessions.

[Continued]

```
15309228 paging space page outs # 270days
     166 paging space page outs # 139days
```

[Continued]

- When **`paging space page outs`** are triggered, something has grown too big, or too many, or both. The cause is often an unexpected dramatic-increase in RDBMS user-connections, i.e. when typically only ~300 user-sessions grows to over ~2600 user-sessions.
- **Exception:** The JFS/JFS2 Twist-Up, i.e. numperm>minperm but numclient<minperm, thus triggering unfounded **`paging space page outs`**.

- Otherwise investigate the cause(s) of computational memory overgrowth, and reduce its demand and/or add more installed gbRAM to manage -- in-order to preclude the trauma of **`paging space page outs`**.

- In some legitimate cases, this overgrowth is for-purpose/by-design and/or cannot be avoided. See above to create a highly write-expedient AIX:pagingspace.

- Strategic Thoughts, Concepts, Considerations, and Tactics

- Devise tactics to relieve exhaustions by exploiting surplus resources
  - Determine points of exhaustion, limitation, and over-commitment
  - Determine surplus resources: CPUcycles, RAM, SAN I/O thruput, etc.

- Recognize-and-Remedy the “bottlenecks” in AIX VMM resources
  - Study the mechanics of AIX Virtual Memory Management (VMM)
  - Practice monitoring the behaviors of the AIX VMM mechanisms
  - Understand the influence of `vmo/ioo/no` tuning parameters on AIX VMM behaviors

- **Match/place RDBMS “tablespaces” with the best mount-options or Go-Raw**
  - **Exercise&experiment with the various JFS2 mount-options as well as Going Raw**
  - **Devise ways to characterize I/O patterns in routinely-active RDBMS “tablespaces”**

Hot Tips when using default-mode LVM/JFS2 "rw" filesystems

- Implement dedicated JFS2-logfiles; try not to share logfiles between filesystems
  - Otherwise, if not dedicated, then for RAID-5 LUNs, using INLINE logfiles are acceptable

- Do not create "Large-Chunk" files across too few inodes
  - i.e. 35gb filesystems full-of-content housed in only 5 inodes (thus only 5 big files)
  - More inodes controlling smaller files lowers the incidence of inode-lock contention

- Do not create NFS-exported filesystems housing too many inodes
  - Creating up to 5-digits of inodes per NFS fs is acceptable/well-tolerated
  - Creating 6-digits of inodes on an NFS fs adds notable lookuppn/sec syscall traffic
  - Creating 7-digits of inodes on an NFS fs adds risk of intolerable latency&overhead
  - Don't even think about 8-digits of inodes on an NFS-exported filesystem. Oops…

- Consider adopting **`mount -o noatime`** as part of a standard universal JFS2 mount policy

- Monitor **`vmstat -v`** for **`pbuf`** and **`fsbuf`** exhaustions (see above for details&remedy)
  - increase **`ioo:j2_dynamicBufferPreallocation=256`** (default=16; see above)

Consider universally adopting `mount -o noatime`

New mount option – `noatime`

Ingo Molnar (Linux kernel developer) said:

– "It's also perhaps the most stupid Unix design idea of all times. Unix is really nice and well done, but think about this a bit: 'For every file that is read from the disk, lets do a ... write to the disk! And, for every file that is already cached and which we read from the cache ... do a write to the disk!'"

If you have a lot of file activity, you have to update a lot of timestamps

– File timestamps
- File creation (ctime)
- File last modified time (mtime)
- File last access time (atime)

– New mount option **`noatime`** disables ***last access time*** updates for JFS2

– File systems with heavy inode access activity due to file opens can have

APARs

– IZ11282 AIX 5.3

– IZ13085 AIX 6.1

JFS2 inode-locking with default-mount filesystems

Each file has a data structure associated with it, called an *inode*. When a file is accessed for reading, the contents of the inode do not change, whereas writes to a file do change the contents of the inode (and the contents of the file).

JFS2 uses a read-shared, write-exclusive inode lock which allows multiple readers to access the file simultaneously, but requires that the lock be held in exclusive mode when a write access is made.

The inode lock imposes write serialization at the file level. Serializing write accesses ensures that data inconsistencies due to overlapping writes do not occur. Serializing reads with respect to writes ensures that the application does not read stale data.

[Continued]

JFS2 inode-locking with default-mount filesystems [Continued]

JFS2 uses a read-shared, write-exclusive inode lock which allows multiple readers to access the file simultaneously, but requires that the lock be held in exclusive mode when a write access is made.

Given this, imagine the difference in concurrent read-write access to a single 35gb file (and its single inode) versus thirty-five 1gb files (and thirty-five inodes).

Multiple read-accesses to the 35gb file and the thirty-five 1gb files is unhindered. But introduce one write to the 35gb file, and the inode locks-out all other reads&writes until released. Contrast this to one write to one of thirty-five 1gb files: We still continue with read-write access to thirty-four 1gb files.

Do not create "Large-Chunk" files across too few inodes. Having more inodes controlling smaller files lowers the incidence of inode-lock contention.

Devise ways to characterize I/O patterns in routinely-active RDBMS "tablespaces"; use `filemon` for detailed logical volume statistics

`filemon` detailed logical volume statistics:

- VOLUME -- Name of the logical volume.
- Description -- Describes contents of the logical volume.
- Reads -- Number of read requests made against the volume.
- read sizes (blks) -- Read transfer-size statistics (avg/min/max/sdev), in 512-byte blocks.
- read times (msec) -- Read response-time statistics (avg/min/max/sdev), in milliseconds.
- read sequences -- Number of read sequences. A sequence is a string of 512-byte blocks that are read consecutively. It indicates the amount of sequential access.
- read seq. lengths -- Statistics describing the lengths of the read sequences, in blocks.
- Writes -- Number of write requests made against the volume.
- write sizes (blks) -- Write transfer-size statistics.
- write times (msec) -- Write-response time statistics.
- write sequences -- Number of write sequences. A sequence is a string of 512-byte blocks that are written consecutively.
- write seq. lengths -- Statistics describing the lengths of the write sequences, in blocks.

Exercise&experiment with JFS2 default mount and Raw I/O

By default, file pages can be cached in real memory for file systems. The caching can be disabled using direct I/O or concurrent I/O mount options; also, the Release-Behind mount options can be used to quickly discard file pages from memory after they have been copied to the application's I/O buffers if the read-ahead and write-behind benefits of cached file systems are needed.

JFS2 default mount -- AIX uses file caching as the default method of file access. However, file caching consumes more CPU and significant system memory because of data duplication. The file buffer cache can improve I/O performance for workloads with a high cache-hit ratio. And file system readahead can help database applications that do a lot of table scans for tables that are much larger than the database buffer cache.

Raw I/O -- Database applications traditionally use raw logical volumes instead of the file system for performance reasons. Writes to a raw device bypass the caching, logging, and inode locks that are associated with the file system; data gets transferred directly from the application buffer cache to the disk. If an application is update-intensive with small I/O requests, then a raw device setup for database data and logging can help performance and reduce the usage of memory resources.

Exercise&experiment with the Direct I/O and Concurrent I/O

By default, file pages can be cached in real memory for file systems. The caching can be disabled using direct I/O or concurrent I/O mount options; also, the Release-Behind mount options can be used to quickly discard file pages from memory after they have been copied to the application's I/O buffers if the read-ahead and write-behind benefits of cached file systems are needed.

- **Direct I/O** – DIO is similar to rawIO except it is supported under a file system. DIO bypasses the file system buffer cache, which reduces CPU overhead and makes more memory available to others (that is, to the database instance). DIO has similar performance benefit as rawIO but is easier to maintain for the purposes of system administration. DIO is provided for applications that need to bypass the buffering of memory within the file system cache. For instance, some technical workloads never reuse data because of the sequential nature of their data access. This lack of data reuse results in a poor buffer cache hit rate, which means that these workloads are good candidates for DIO.

- **Concurrent I/O** -- CIO supports concurrent file access to files. In addition to bypassing the file cache, it also bypasses the inode lock that allows multiple threads to perform reads and writes simultaneously on a shared file. CIO is designed for relational database applications, most of which will operate under CIO without any modification. Applications that do not enforce serialization for access to shared files should not use CIO. Applications that issue a large amount of reads usually will not benefit from CIO either.

Exercise&experiment with JFS2 Release-behind mechanisms

Release-behind-read and release-behind-write allow the file system to release the file pages from file system buffer cache as soon as an application has read or written the file pages. This feature helps the performance when an application performs a great deal of sequential reads or writes. Most often, these file pages will not be reassessed after they are accessed.

Without this option, the memory will still be occupied with no benefit of reuse, which causes paging eventually after a long run. When writing a large file without using release-behind, writes will go very fast as long as pages are available on the free list. When the number of pages drops to **minfree**, VMM uses its LRU algorithm to find candidate pages for eviction.

This feature can be configured on a file system basis. When using the **mount** command, enable release-behind by specifying one of the three flags below:

- The release-behind sequential read flag (rbr)
- The release-behind sequential write flag (rbw)
- The release-behind sequential read and write flag (rbrw)

A trade-off of using the release-behind mechanism is that the application can experience an increase in CPU utilization for the same read or write throughput rate (as compared to not using release-behind). This is because of the work required to free the pages, which is normally handled at a later time by the LRU daemon. Also, note that all file page accesses result in disk I/O because file data is not cached by VMM. However, applications (especially long-running applications) with the release-behind mechanism applied are still supposed to perform more optimally and with more stability.

Match/place RDBMS “tablespaces” with the best mount-options or Go-Raw

- Review component technology of the infrastructure; ensure proper tuning-by-hardware
- Review implemented AIX constructs, i.e. “firm” near-static structures and settings
- Create a complete technology-stack map of I/O
  - i.e. RAIDset/tech->LUN-> [LVMvg:lv::JFS2mtpt w/options]->*logical_content*
- Review historical/accumulated AIX events, usages, pendings, counts, blocks, etc.
- Monitor dynamic AIX command behaviors, i.e. ps, vmstat, mpstat, iostat, topas, etc.
- Devise tactics to relieve exhaustions by exploiting surplus resources
- Recognize-and-Remedy the “bottlenecks” in AIX VMM resources
- Implement dedicated JFS2-logfiles; try not to share logfiles between filesystems
- Do not create “Large-Chunk” files across too few inodes
- Adopt **`mount -o noatime`** as part of a standard universal JFS2 mount policy
- Use **`filemon`** to characterize I/O patterns in routinely-active RDBMS “tablespaces”
- Exercise&experiment with all JFS2 mount options as well as Raw I/O
- Match&place individual RDBMS “tablespaces” with the best mount-options or Go-Raw

Trademarks

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

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*, AS/400®, e business(logo)®, DBE, ESCO, eServer, FICON, IBM®, IBM (logo)®, iSeries®, MVS, OS/390®, pSeries®, RS/6000®, S/30, VM/ESA®, VSE/ESA, WebSphere®, xSeries®, z/OS®, zSeries®, z/VM®, System i, System i5, System p, System p5, System x, System z, System z9®, BladeCenter®

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
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.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
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.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* 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.

Disclaimers

No part of this document may be reproduced or transmitted in any form without written permission from IBM Corporation.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This information could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The performance data contained herein was obtained in a controlled, isolated environment. Actual results that may be obtained in other operating environments may vary significantly. While IBM has reviewed each item for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customer experiences described herein are based upon information and opinions provided by the customer. The same results may not be obtained by every user.

Reference in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead. It is the user's responsibility to evaluate and verify the operation on any non-IBM product, program or service.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR INFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted according to the terms and conditions of the agreements (e.g. IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. IBM is not responsible for the performance or interoperability of any non-IBM products discussed herein.

Disclaimers Continued

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of 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.

The providing of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

IBM customers are responsible for ensuring their own compliance with legal requirements. It is the customer's sole responsibility to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws.

IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law.

The information contained in this documentation is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information provided, it is provided "as is" without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this documentation or any other documentation. Nothing contained in this documentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

PE11(AIX)

AIX 5L/6 Performance Tuning Part II: Tactics for Tuning Indicated Performance Issues

Thank You

Earl Jew
IBM Field Technical Sales Specialist for Power Systems and Storage
IBM Regional Designated Specialist - Power/AIX Performance & Tuning
400 North Brand Blvd., Suite 700 c/o IBM, Glendale, CA, USA 91203
earlj@us.ibm.com (310)251-2907