Dynamic Platform Optimizer

DPO

Advanced Technology Support, Europe.

Gareth Coates
gaz@uk.ibm.com
@power_gaz

Agenda

- My headline comments to the developers

- Motivation

- What is DPO
  - What does it do and how does it do it?

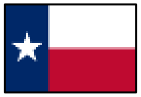
- Our testing

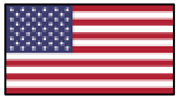
- Our recommendations

POWER7
BUILT ON
Power™
Advanced Technical Support (Europe)

My headline comments to the developers

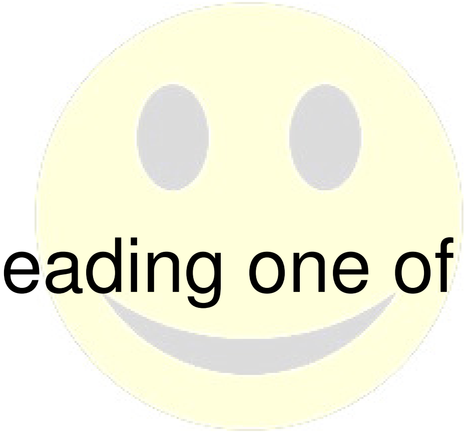
- EMEA ATS received a Power 760
  - on loan from Austin, Texas in November 2012.





- The product was not announced till 5th February 2013
- I was really keen to test DPO
- I gave feedback to the developers and the following slide shows my headline comments.
- The presentation I sent, (pre-announce) was of course, IBM Confidential, but this one is not.
- It was internal IBM communication, so I could be blunt!

Observations



- Working on DPO has been like reading one of those books that you can’t put down!

- Shuffling the VMs (LPARs) by hand, checking the affinity and then watching the optimiser fix it all; has been great.
  - I have certainly consolidated my understanding of POWER7 and POWER7+ LPAR placement and affinity implications

Now released so no longer confidential!

- As the system used for testing is not yet announced, this document is IBM Confidential.
  - And that’s the only reason that I haven’t been tweeting hard about this technology too – **it is really great!**

@power_gaz

Motivation

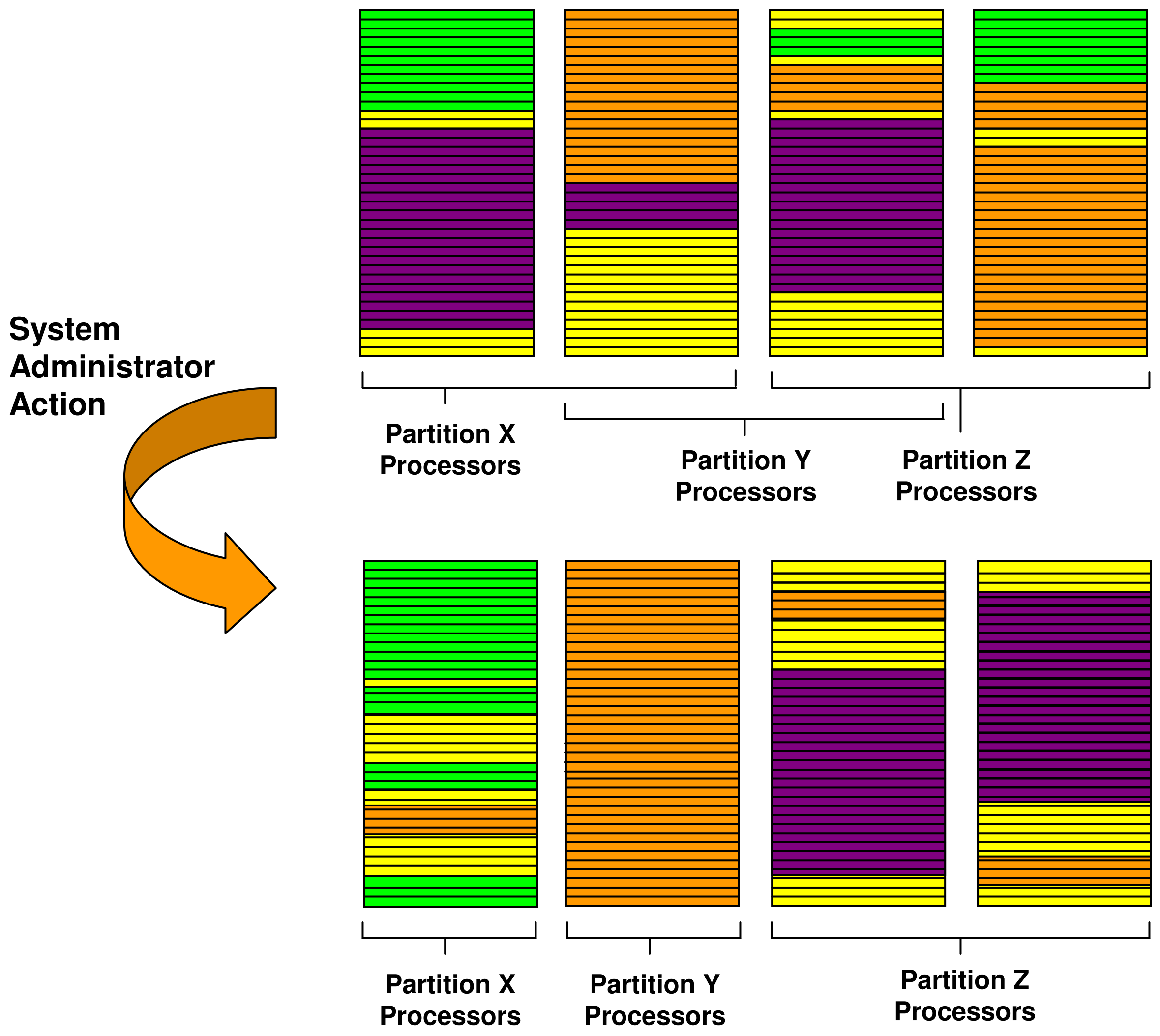
- Partition placement can become sub-optimal
  - Dynamic creation and deletion of partitions
  - DLPAR operations
  - Partition Mobility
  - Hibernation

- Platform will provide a mechanism to optimize partition placement dynamically

Think of it as 52 card pickup - and sort

- Benefits include
  - Improved performance in a cloud environment
  - Dynamically adjust topology after mobility
  - Simple to use and predicted “score”

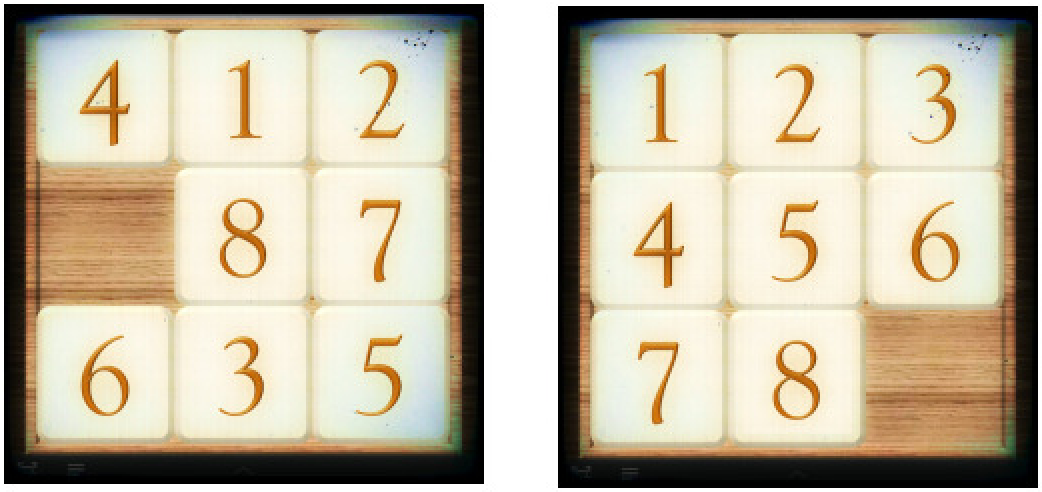
Dynamic Platform Optimizer



Legend

- 🟩 Partition X Memory
- 🟧 Partition Y Memory
- 🟪 Partition Z Memory
- 🟨 Free LMBs

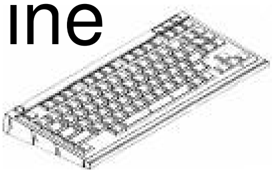
Solution Similar to Solving a Tile Puzzle



- More free blocks make it easier (and quicker)
- More non-relocatable blocks make it tougher
- At least one free memory block required
  - – Hypervisor will use unlicensed memory for the purpose of relocation

Additional Details

- Optimizer is launched via HMC command-line interface



- Requested/protected partition lists
  - Sets of partitions can be prioritized or protected (untouched) by the DPO operation
- Impacted partitions notified at the end of operations
  - Partitions will re-fetch affinity properties in response to notification
- Notion of current and potential “affinity score”
  - Enables system administrator to make decisions about value of running optimizer

Additional Details

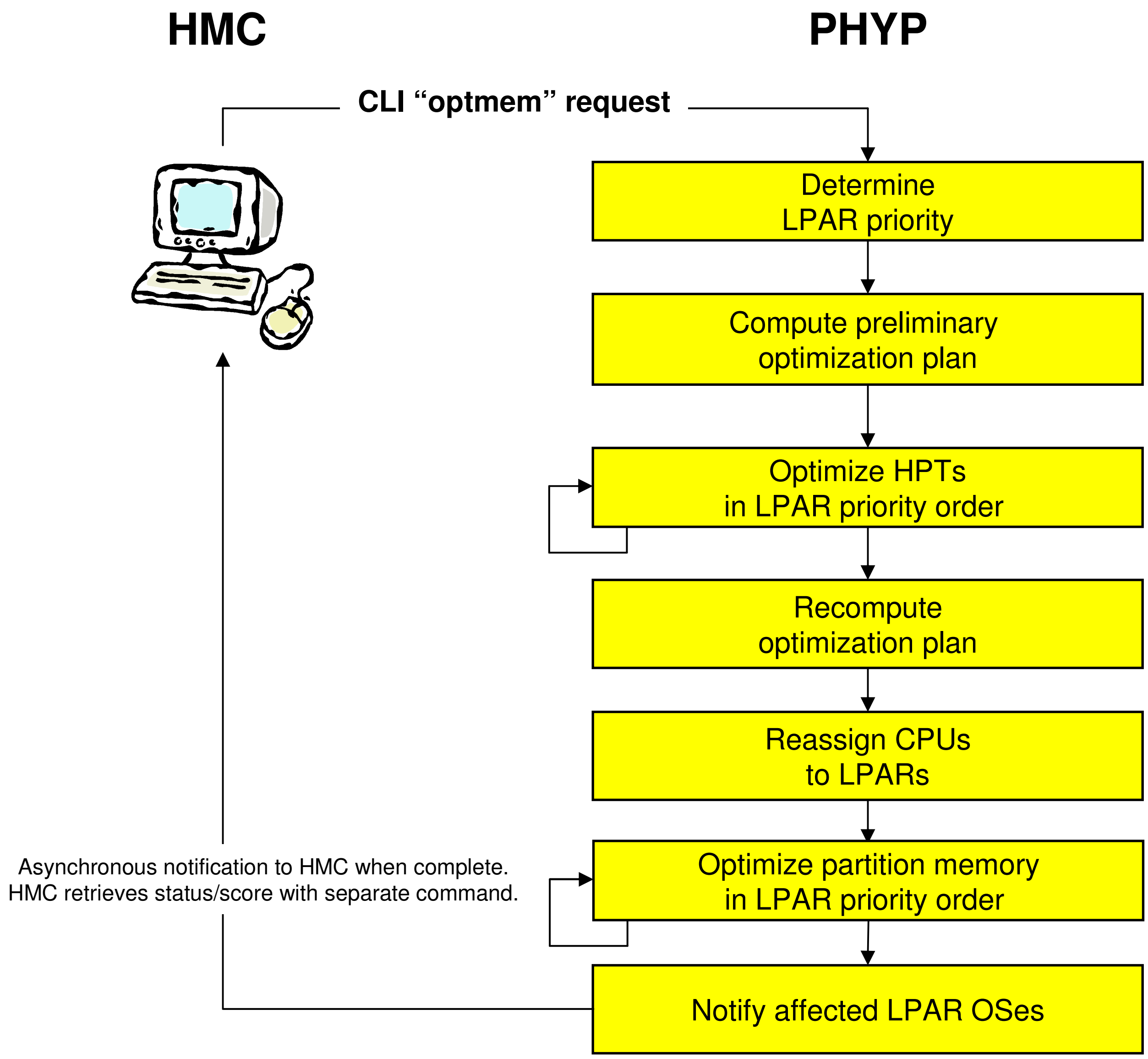
- Hypervisor utilises underlying technology developed for CHARM to relocate memory and virtual CPUs
  - Relocation transparent to partitions

- Enterprise models support CUoD
  - PoD – Processor on Demand
  - MoD – Memory on Demand

POWER7
BUILT ON
Power™
Advanced Technical Support (Europe)

Power Systems Capacity on Demand: http://www-03.ibm.com/systems/power/hardware/cod/index.html

Implementation Details – Work Flow



HMC

PHYP

CLI “optmem” request

Determine LPAR priority

Compute preliminary optimization plan

Optimize HPTs in LPAR priority order

Recompute optimization plan

Reassign CPUs to LPARs

Optimize partition memory in LPAR priority order

Notify affected LPAR OSes

Asynchronous notification to HMC when complete.
HMC retrieves status/score with separate command.

Same high-level flow for optimization and score prediction. No LMBs or CPUs actually moved for prediction. Predicted score based on predicted virtual memory/CPU layout.

HPT objects require contiguous LMBs. Some HPT objects may not be moved to desired location due to fragmentation caused by garded memory, TCE tables, etc.

Based on new memory layout.

One LMB at a time. Atomic units (relocation granules) are 512K.

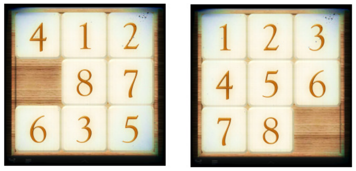
LMB = logical Memory Block
HPT = Hardware Page Table

Implementation Details -- Misc.

- Optimization Priority Order
  - Primarily based on user-defined affinity group ID (255-1).
  - Otherwise, based on CPU/memory resources (more = higher priority)

- CPU Reassignment
  - Fast operation – not dependent on optimization priority

- Partition LMB moves
  - Planned in partition priority order
  - Higher priority partitions finish earlier
  - Can be long-running operation



- CPU Cycles
  - LMB relocation performed in multiple threads.
  - PHYP dispatcher “steals” cycles periodically to perform work in the background.
  - Cycles not stolen from protected partitions.

- Partition OS Reaffinitization
  - AIX: 6.1 TL8+, AIX 7.1 TL2+
  - IBM i: 7.1
  - Linux: Some reaffinitization in RHEL7/SLES12. Fully implemented in follow-on release.

HMC CLI: Starting/Stopping a DPO Operation

```
# optmem –m managed_system –t affinity –o start
        [--id requested_partition_list]
        [--xid protect_partition_list]
```

- Partition lists are comma-separated and can include ranges.
  eg: --xid 5,10,16-20
- Requested partitions:LPARs that should be prioritized (default = all)
- Protected partitions: LPARs that should not be touched (default = none)

```
# optmem –m managed_system –t affinity –o stop
```

HMC CLI: DPO Status

```
# lsmemopt -m managed_system

in_progress=0,status=Finished,type=affinity,opt_id=1,pro
  gress=100,
  requested_lpar_ids=none,protected_lpar_ids=none,
  “impacted_lpar_ids=106,110”
```

- Unique optimization identifier
- Estimated progress %
- LPARs that were impacted by the optimization
  (i.e. had CPUs, memory, or their hardware page table moved)

HMC CLI: Current and Predicted Affinity Scores

```
# lsmemopt -m managed_system -o currscore

# lsmemopt -m managed_system -o calcscore
      [--id request_partition_list]
      [--xid protect_partition_list]
```

- Currscore computes the current system-wide affinity score (0-100)

- Calcscore computes the predicted system-wide score that would probably be the result of running a DPO operation (includes optional parms just like the actual DPO operation)

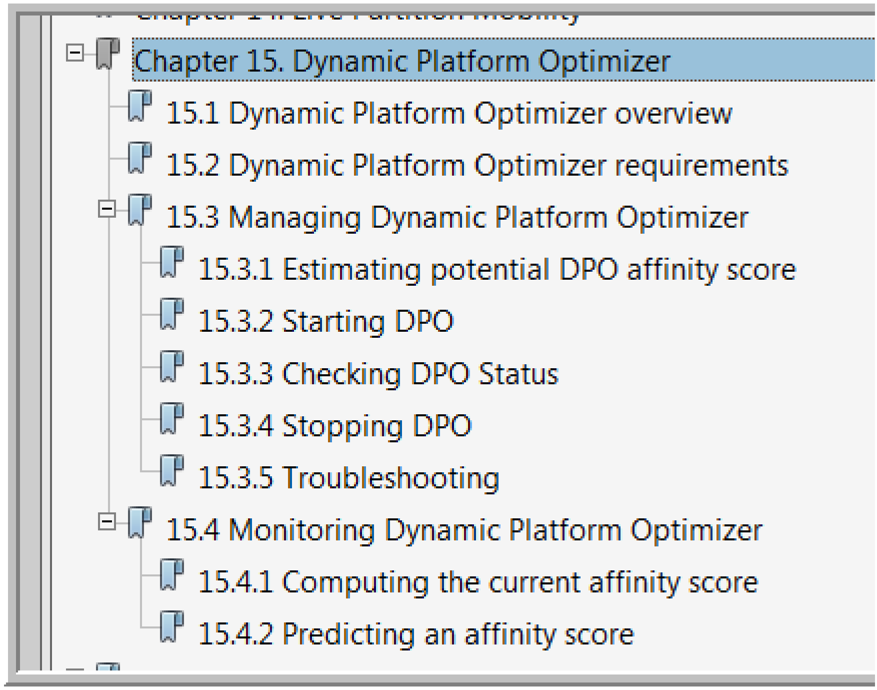
HMC CLI: Example

```
 # lssyscfg -r sys -F name
zg23ae
zg24he

# lsmemopt -m zg24he -o currscore
curr_sys_score=84

# lsmemopt -m zg24he -o calcscore
curr_sys_score=84,predicted_sys_score=86,"requested_lpar_ids=1,2,17,105,106,107,
  108,109,110,111",protected_lpar_ids=none

# optmem -m zg24he -t affinity -o start

# lsmemopt -m zg24he
in_progress=0,status=Finished,type=affinity,opt_id=2,progress=0,requested_lpar_i
  ds=none,protected_lpar_ids=none,”impacted_lpar_ids=106,110”

# lsmemopt -m zg24he -o currscore
curr_sys_score=86
```

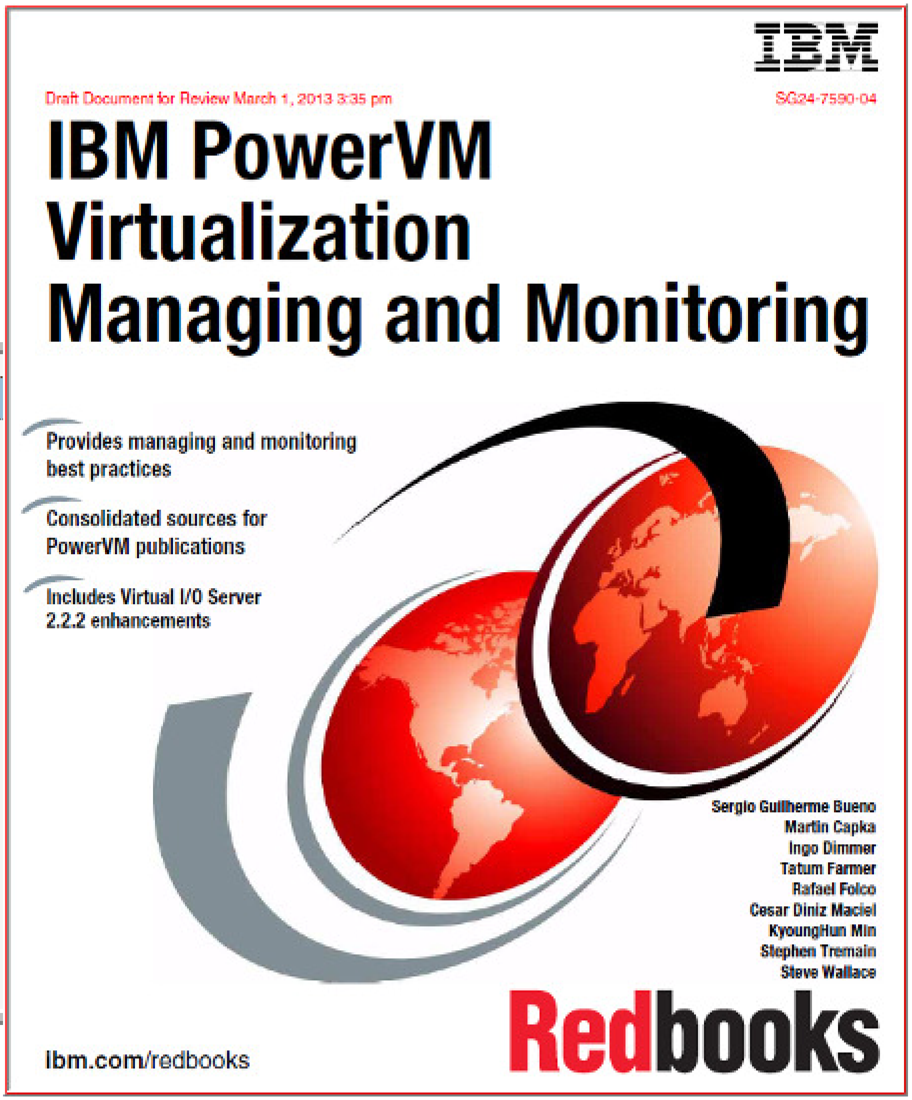
More Information

- http://www.redbooks.ibm.com/redpieces/abstracts/sg247590.html

- RedBook
- March 2013 update



Chapter 14. Live Partition Mobility
- Chapter 15. Dynamic Platform Optimizer
  - 15.1 Dynamic Platform Optimizer overview
  - 15.2 Dynamic Platform Optimizer requirements
  - 15.3 Managing Dynamic Platform Optimizer
    - 15.3.1 Estimating potential DPO affinity score
    - 15.3.2 Starting DPO
    - 15.3.3 Checking DPO Status
    - 15.3.4 Stopping DPO
    - 15.3.5 Troubleshooting
  - 15.4 Monitoring Dynamic Platform Optimizer
    - 15.4.1 Computing the current affinity score
    - 15.4.2 Predicting an affinity score



Draft Document for Review March 1, 2013 3:35 pm

SG24-7590-04

IBM PowerVM Virtualization Managing and Monitoring

Provides managing and monitoring best practices

Consolidated sources for PowerVM publications

Includes Virtual I/O Server 2.2.2 enhancements

Sergio Guilherme Bueno
Martin Capka
Ingo Dimmer
Tatum Farmer
Rafael Folco
Cesar Diniz Maciel
KyoungHun Min
Stephen Tremain
Steve Wallace

ibm.com/redbooks

Redbooks

Our Tests

- Type_model
  - 8408-E8D
- 24 POWER7+ cores at 3.136 GHz
- 128GB RAM
- Current hypervisor dispatch wheel time : 10 mS
- LMB= 128MB
- Enable Static Power Saver mode
- Idle Power Saver Enable – defaults
- Tuning Parameters (ASMI and AIX) – defaults
- AME was not enabled in any LPARs

You can change this now
10mS or 50 mS

Keep it the same on all your servers – if not, LPM is not available

LPARs

1	full_system	aixlinux
21	claret-vios1	vioserver
22	claret-vios2	vioserver
31	claret1	aixlinux
32	claret2	aixlinux
33	claret3	aixlinux
34	claret4	aixlinux
35	claret5	aixlinux
36	claret6	aixlinux
37	claret7	aixlinux
38	claret8	aixlinux
39	claret9	aixlinux
40	claret10	aixlinux
41	claret11	aixlinux
99	claret-bigdummy	aixlinux

LPAR usage for this testing

- claret-vios1 and claretvios2
  - Fully redundant no changes to config
  - CE=2.0 VP=2 weight=200

VIOS

- claret1 through claret10
  - Various changes to config during testing
  - Not all are used in all tests
  - Always uncapped with weight=128

AIX LPARs

- claret-bigdummy
  - Desires all CPU and all RAM
  - Started to SMS to grab all free resources, then reset

To “reset” placement

- full_system
  - Full system partition – not used in this testing
- claret11
  - Not used in this testing

NOT USED

Software versions

- HMC V7R770

- VIOS
  - NIM installed VIOS 2222
  - ioslevel reports: 2.2.2.1

- AIX
  - A mix of
    - 7100-00-07-1228
    - 7100-01-05-1228

Tools used

New URLs

- nmon
  - Native version in AIX
  - Analyser version 34a from https://www.ibm.com/developerworks/wikis/download/attachments/53871868/nmon_analyser.zip?version=18
- nstress
  - A suite of performance utilities written by Nigel Griffiths
  - https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/Power%20Systems/page/nstress
  - ncpu hammers cpus
  - nmem hits memory (and uses cpu to do so)
- Parallel worms
  - Draws squiggles on the screen and gives an update/s output

General methodology

- We created a number of LPAR and System profiles for easy control of multiple LPARs

- We created a set of scripts to carry out various DLPAR operations
  - Designed specifically to mess up the affinity
  - We avoided “round numbers” like 4 and 16 preferring eg: 11

- We captured resource configurations throughout

- We start some load in the LPARs

- Then we run the optimiser

- Capturing stats throughout

- We looked at resource configurations to see the effect

Load generator and monitor

- nmon data gathering is started

- nmem -m 254 -s 300 -z 20
  - This mallocs 254MB RAM
  - then touches the memory pages at random

- One (or 3) instance is run for each VP configured

- Shortly afterwards the optimiser is started

Pass 17 – Simulate 10 LPARs growing over time

- We ran literally dozens of tests
- This is one example:
- All LPARs except VIOS were shut down
- claret-bigdummy was started and stopped
- `optmem` was run and a dump was taken

The dump RSCDUMP.109D58R.02000006.20130201001519

```
|-----------|----------------------|--------------|-------|---------------|---------------|-------|
|  Domain   |    Procs       Units |    Memory    |       |  Proc Units   |    Memory     | Ratio |
| SEC | PRI | Total | Free | Free  | Total | Free |  LP   |  Tgt  | Aloc  |  Tgt  | Aloc  |       |
|-----|-----|-------|------|-------|-------|------|-------|-------|-------|-------|-------|-------|
|  0  |     | 1200  |   0  |    0  |  512  |   1  |       |       |       |       |       |    0  |
|     |  0  |  600  |   0  |    0  |  256  |   0  |       |       |       |       |       |    0  |
|     |     |       |      |       |       |      |   99  |  600  |  600  |  248  |  248  |       |
|     |  1  |  600  |   0  |    0  |  256  |   1  |       |       |       |       |       |    0  |
|     |     |       |      |       |       |      |   21  |  200  |  200  |   64  |   64  |       |
|     |     |       |      |       |       |      |   22  |  200  |  200  |   64  |   64  |       |
|     |     |       |      |       |       |      |   99  |  200  |  200  |  117  |  117  |       |
|-----|-----|-------|------|-------|-------|------|-------|-------|-------|-------|-------|-------|
|  1  |     | 1200  |   0  |    0  |  512  |   0  |       |       |       |       |       |    0  |
|     |  2  |  600  |   0  |    0  |  256  |   0  |       |       |       |       |       |    0  |
|     |     |       |      |       |       |      |   99  |  600  |  600  |  249  |  249  |       |
|     |  3  |  600  |   0  |    0  |  256  |   0  |       |       |       |       |       |    0  |
|     |     |       |      |       |       |      |   99  |  600  |  600  |  245  |  245  |       |
|-----|-----|-------|------|-------|-------|------|-------|-------|-------|-------|-------|-------|
```

- The two VIOS are nicely positioned
  - with their entitlement of 2.0 and their 8GB RAM
  - 64*128MB (LMB) = 8192MB
  - In one domain

- We don't care that LPAR id 99 is fragmented

> Don't ask me how to create these dumps, I am not allowed to tell you – remember, I was Beta testing and using engineering tools.

Start the LPARs

- We started claret1 through claret10 (System profile)
  - Each had CE=1.0 VP=1 1GB

- We ran

  -
    ```
    lsmemopt -m $CLARET -o calcscore -F "curr_sys_score,predicted_sys_score"
    ```
  - 99,100

- We took a dump

The dump RSCDUMP.109D58R.03000006.20130201004119

```
|-----------|-----------------------|---------------|------|---------------|---------------|-------|
|   Domain  |     Procs       Units |     Memory    |      |   Proc Units  |     Memory    | Ratio |
| SEC | PRI | Total | Free  | Free  | Total |  Free |  LP  |  Tgt  |  Aloc |  Tgt  |  Aloc |       |
|-----|-----|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|
|   0 |     |  1200 |   600 |     0 |   512 |   342 |      |       |       |       |       |  1335 |
|     |   0 |   600 |   600 |     0 |   256 |   248 |      |       |       |       |       |   967 |
|     |   1 |   600 |     0 |     0 |   256 |    94 |      |       |       |       |       |     0 |
|     |     |       |       |       |       |       |   21 |   200 |   200 |    64 |    64 |       |
|     |     |       |       |       |       |       |   22 |   200 |   200 |    64 |    64 |       |
|     |     |       |       |       |       |       |   39 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   40 |   100 |   100 |     8 |     8 |       |
|-----|-----|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|
|   1 |     |  1200 |   400 |     0 |   512 |   405 |      |       |       |       |       |  2371 |
|     |   2 |   600 |   100 |     0 |   256 |   196 |      |       |       |       |       |  4593 |
|     |     |       |       |       |       |       |   31 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   34 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   36 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   37 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   38 |   100 |   100 |     8 |     8 |       |
|     |   3 |   600 |   300 |     0 |   256 |   209 |      |       |       |       |       |  2449 |
|     |     |       |       |       |       |       |   32 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   33 |   100 |   100 |     8 |     8 |       |
|     |     |       |       |       |       |       |   35 |   100 |   100 |     8 |     8 |       |
|-----|-----|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|
```

- All the LPARs have good affinity

Script

- We ran "dpo_test17“
  - Gathering stats throughout …
  - It adds 1152MB to each LPAR in turn
  - Then goes round the loop 7 more times
  - That gave us: 71,96
  - Then it adds 0.5 to CE and 1 to VP of each LPAR
  - Then adds the same amount again
  - We ended up with
  - CE=2 VP=3 10GB per LPAR
  - No available CPUs, 3.25 GB available RAM
  - Score: 71,99
  - Then it takes a dump

RAM but no CPU

CPUs in multiple domains

The dump RSCDUMP.109D58R.0E000006.20130201011708

```
|-----------|--------------------------|-----------------|------|-----------------|-----------------|--------|
|  Domain   |   Procs           Units  |     Memory      |      |   Proc Units    |     Memory      | Ratio  |
| SEC | PRI |  Total |  Free  |  Free  |  Total |  Free  |  LP  |   Tgt  |  Aloc  |   Tgt  |  Aloc  |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
|  0  |     |  1200  |   600  |     0  |   512  |    27  |      |        |        |        |        |   632  |
|     |  0  |   600  |   600  |     0  |   256  |    27  |      |        |        |        |        |   632  |
|     |     |        |        |        |        |        |  31  |        |        |    80  |    18  |        |
|     |     |        |        |        |        |        |  32  |        |        |    80  |    18  |        |
|     |     |        |        |        |        |        |  33  |        |        |    80  |    18  |        |
|     |     |        |        |        |        |        |  34  |        |        |    80  |    27  |        |
|     |     |        |        |        |        |        |  35  |        |        |    80  |    18  |        |
|     |     |        |        |        |        |        |  36  |        |        |    80  |    27  |        |
|     |     |        |        |        |        |        |  37  |        |        |    72  |    18  |        |
|     |     |        |        |        |        |        |  38  |        |        |    72  |    27  |        |
|     |     |        |        |        |        |        |  39  |        |        |    23  |    23  |        |
|     |     |        |        |        |        |        |  40  |        |        |    27  |    27  |        |
|     |  1  |   600  |     0  |     0  |   256  |     0  |      |        |        |        |        |     0  |
|     |     |        |        |        |        |        |  21  |   200  |   200  |    64  |    64  |        |
|     |     |        |        |        |        |        |  22  |   200  |   200  |    64  |    64  |        |
|     |     |        |        |        |        |        |  39  |   100  |   100  |    57  |    57  |        |
|     |     |        |        |        |        |        |  40  |   100  |   100  |    53  |    53  |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
|  1  |     |  1200  |   400  |     0  |   512  |     0  |      |        |        |        |        |     0  |
|     |  2  |   600  |   100  |     0  |   256  |     0  |      |        |        |        |        |     0  |
|     |     |        |        |        |        |        |  31  |   100  |   100  |        |    53  |        |
|     |     |        |        |        |        |        |  34  |   100  |   100  |        |    44  |        |
|     |     |        |        |        |        |        |  36  |   100  |   100  |        |    44  |        |
|     |     |        |        |        |        |        |  37  |   100  |   100  |     8  |    44  |        |
|     |     |        |        |        |        |        |  38  |   100  |   100  |     8  |    51  |        |
|     |  3  |   600  |   300  |     0  |   256  |     0  |      |        |        |        |        |     0  |
|     |     |        |        |        |        |        |  31  |        |        |        |     9  |        |
|     |     |        |        |        |        |        |  32  |   100  |   100  |        |    62  |        |
|     |     |        |        |        |        |        |  33  |   100  |   100  |        |    62  |        |
|     |     |        |        |        |        |        |  34  |        |        |        |     9  |        |
|     |     |        |        |        |        |        |  35  |   100  |   100  |        |    62  |        |
|     |     |        |        |        |        |        |  36  |        |        |        |     9  |        |
|     |     |        |        |        |        |        |  37  |        |        |        |    18  |        |
|     |     |        |        |        |        |        |  38  |        |        |        |     2  |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
```

optimiser

- We ran dpo_nmon17
  - Which starts nmon on each client
  - Sleeps for a minute
  - Starts an nmem thread per VP
  - Sleeps for a minute
  - Then optimises

Result

EXACTLY WHAT WAS PREDICTED

- Final score: 99,99
- It took less than 8 minutes and 16 seconds
- Let’s see what it did
  - The dump is on the next foil

Much better !!!

Dump RSCDUMP.109D58R.0C000007.20130201031958

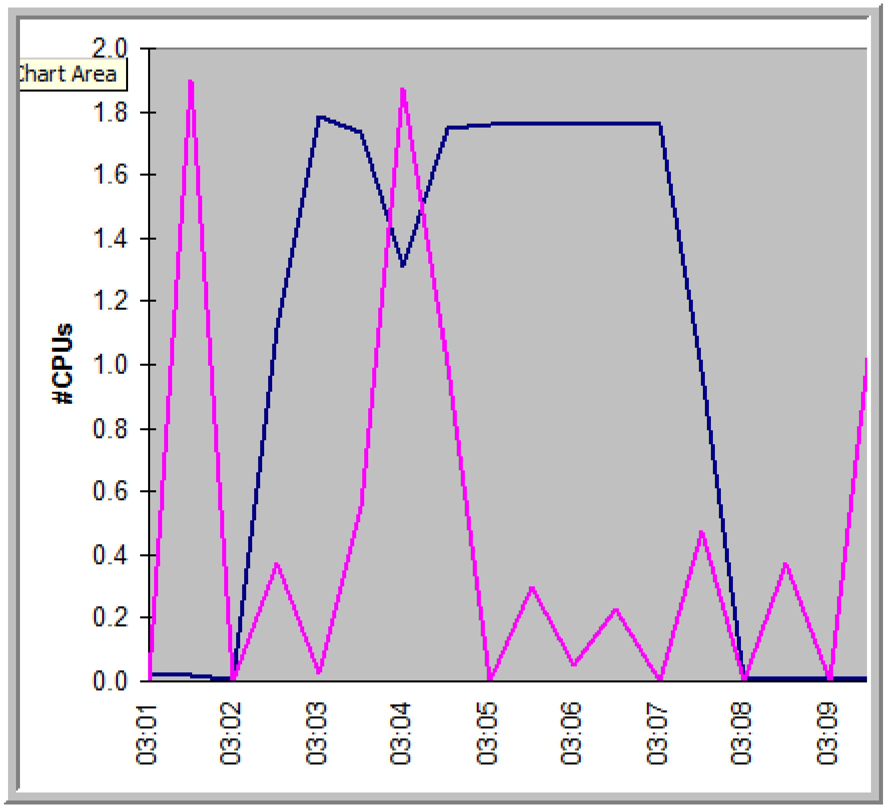
```
|-----------|--------------------------|-----------------|------|-----------------|-----------------|--------|
|  Domain   |   Procs          Units   |     Memory      |      |   Proc Units    |     Memory      | Ratio  |
| SEC | PRI |  Total |  Free  |  Free  |  Total |  Free  |  LP  |   Tgt  |  Aloc  |   Tgt  |  Aloc  |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
|  0  |     |   1200 |      0 |      0 |    512 |     26 |      |        |        |        |        |      0 |
|     |   0 |    600 |      0 |      0 |    256 |      0 |      |        |        |        |        |      0 |
|     |     |        |        |        |        |        |   37 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   38 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   39 |    200 |    200 |     76 |     76 |        |
|     |   1 |    600 |      0 |      0 |    256 |     26 |      |        |        |        |        |      0 |
|     |     |        |        |        |        |        |   21 |    200 |    200 |     64 |     64 |        |
|     |     |        |        |        |        |        |   22 |    200 |    200 |     64 |     64 |        |
|     |     |        |        |        |        |        |   36 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   39 |        |        |      4 |      4 |        |
|     |     |        |        |        |        |        |   40 |        |        |      4 |      4 |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
|  1  |     |   1200 |      0 |      0 |    512 |      1 |      |        |        |        |        |      0 |
|     |   2 |    600 |      0 |      0 |    256 |      0 |      |        |        |        |        |      0 |
|     |     |        |        |        |        |        |   31 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   34 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   40 |    200 |    200 |     76 |     76 |        |
|     |   3 |    600 |      0 |      0 |    256 |      1 |      |        |        |        |        |      0 |
|     |     |        |        |        |        |        |   32 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   33 |    200 |    200 |     80 |     80 |        |
|     |     |        |        |        |        |        |   35 |    200 |    200 |     80 |     80 |        |
|-----|-----|--------|--------|--------|--------|--------|------|--------|--------|--------|--------|--------|
```

- Most LPARs are fine, sitting in a single Domain

- LPARid 39 is in two parts
  - In the same Secondary but a different Primary Domain

- LPARid 40 is in two parts
  - In different Secondary Domains

The nmon output

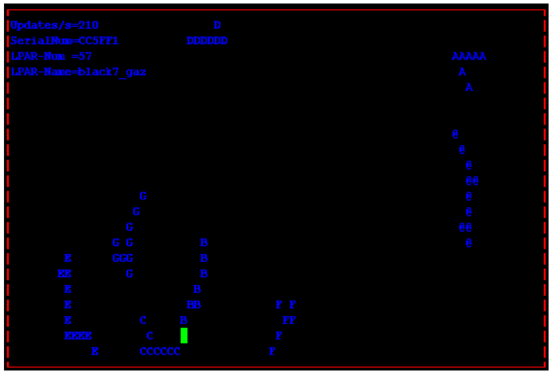


claret1

- All the LPARs showed similar results
  - There was a short dip in cpu activity
  - About 1 minute
  - The dips were offset slightly in each LPAR
  - This suggests that PHYP is using cpu cycles so they are not available to the shared pool
  - Which is what we expect
  - This is a good result as we have much better affinity and had only a short and small reduction in performance getting there

Performance impact

- Parallel worms was very useful
  - It clearly showed a drop in activity as `nmem` kicked in
  - And then a further short dip as the optimiser ran
  - A couple of times, it seemed to “block” momentarily, but the responsiveness was still there over a period of say a quarter of a second
  - In other words you wouldn’t be picking up the phones!



```
Updates/s=210                 D
SerialNum=CC5FF1          DDDDDD
LPAR-Num =57                                                     AAAAA
LPAR-Name=black7_gaz                                              A
                                                                   A


                                                                 @
                                                                  @
                                                                   @
                                                                   @@
                   G                                               @
                  G                                                @
                 G                                                @@
               G G          B                                      @
        E      GGG          B
       EE        G          B
        E                  B
        E                 BB           F F
        E          C     B              FF
        EEEE        C    █             F
            E      CCCCCC             F
```

Worst case

- Worst score seen during our investigations:
  - curr_sys_score=37
- OK, we were
  - stopping/starting LPARs
  - DLPARing
- As we were carrying out various test on DPO and other features of the equipment
- Such a low score is unlikely in most circumstances
- The optimizer took it to 95 !

Observations

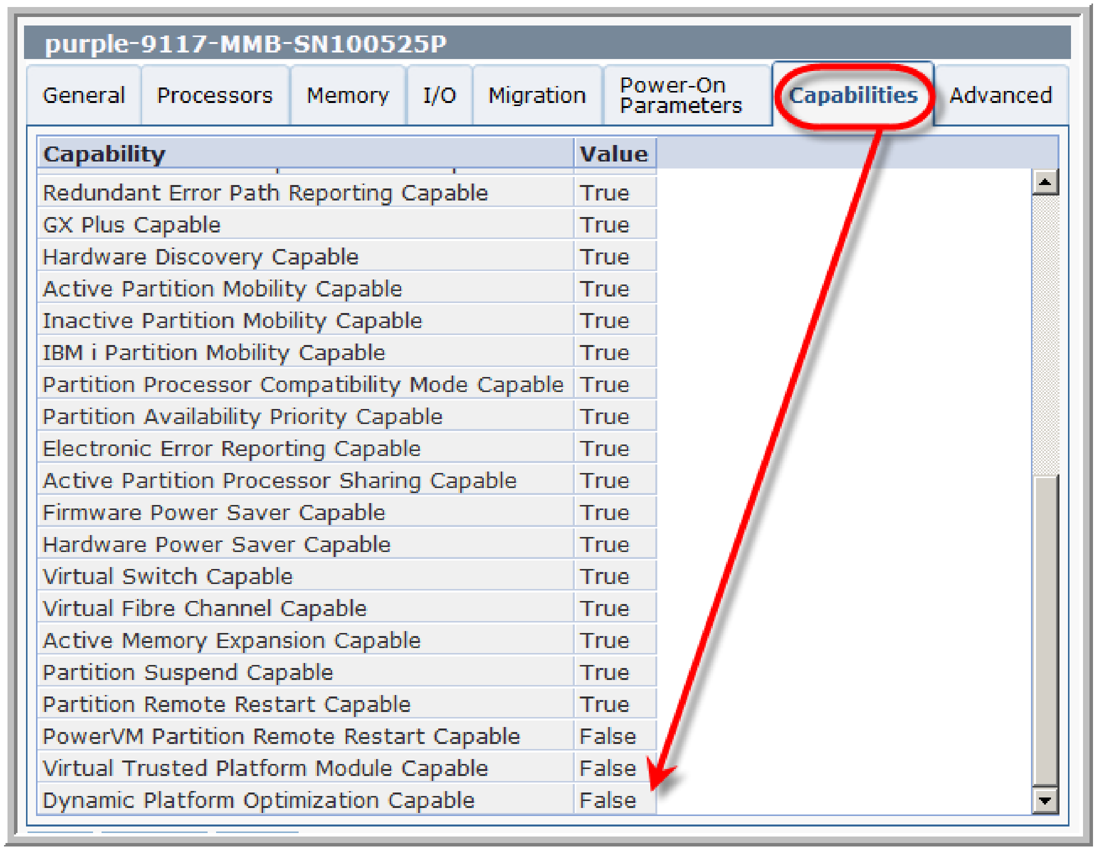
- We noticed that there seemed to be a lag between changes happening on the system and the HMC being aware
  - This is to be expected
  - The HMC needs to refresh from the FSP

Observations

- Try running DPO on a 770 (B)

```
hscroot@hmc11:~> lsmemopt -m $PURPLE
HSCL02D0 This operation is not allowed because the managed system does not support
  Dynamic Platform Optimization.
hscroot@hmc11:~>
```

- Good: correct, and a sensible error message☺



Capability	Value
Redundant Error Path Reporting Capable	True
GX Plus Capable	True
Hardware Discovery Capable	True
Active Partition Mobility Capable	True
Inactive Partition Mobility Capable	True
IBM i Partition Mobility Capable	True
Partition Processor Compatibility Mode Capable	True
Partition Availability Priority Capable	True
Electronic Error Reporting Capable	True
Active Partition Processor Sharing Capable	True
Firmware Power Saver Capable	True
Hardware Power Saver Capable	True
Virtual Switch Capable	True
Virtual Fibre Channel Capable	True
Active Memory Expansion Capable	True
Partition Suspend Capable	True
Partition Remote Restart Capable	True
PowerVM Partition Remote Restart Capable	False
Virtual Trusted Platform Module Capable	False
Dynamic Platform Optimization Capable	False

Let’s enable DPO on a system

```
hscroot@blackhmc:~> lssyscfg -r sys -F name
black-9119-FHB-02C5FF1

hscroot@blackhmc:~> BLACK=$(lssyscfg -r sys -F name)

hscroot@blackhmc:~> lssyscfg -m $BLACK -r sys -F dynamic_platform_optimization_capable
0
```

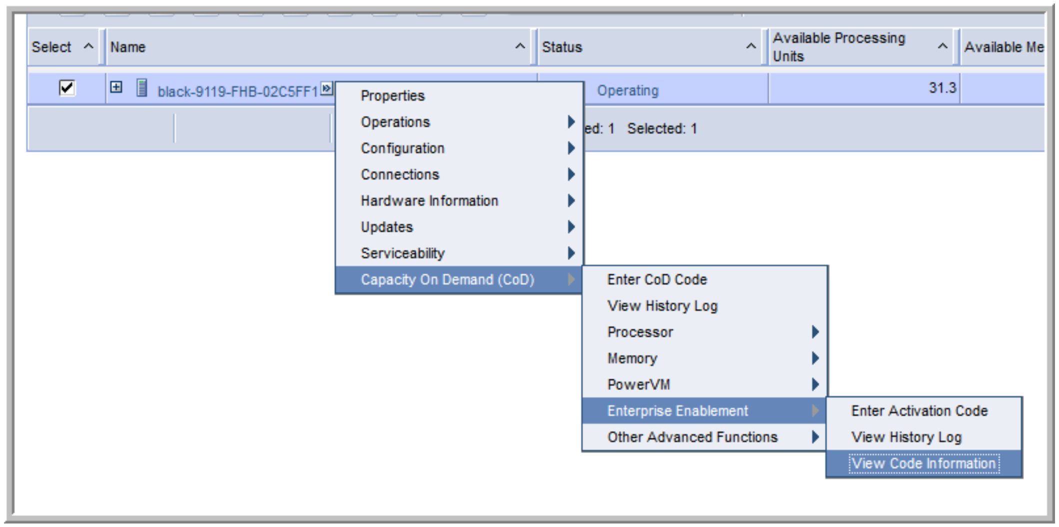
OK as there is only one system. Otherwise try adding a grep or cut’n’paste

- Need to order VET code
  - and you need this information to do so

VET

Virtualization Enablement Technology

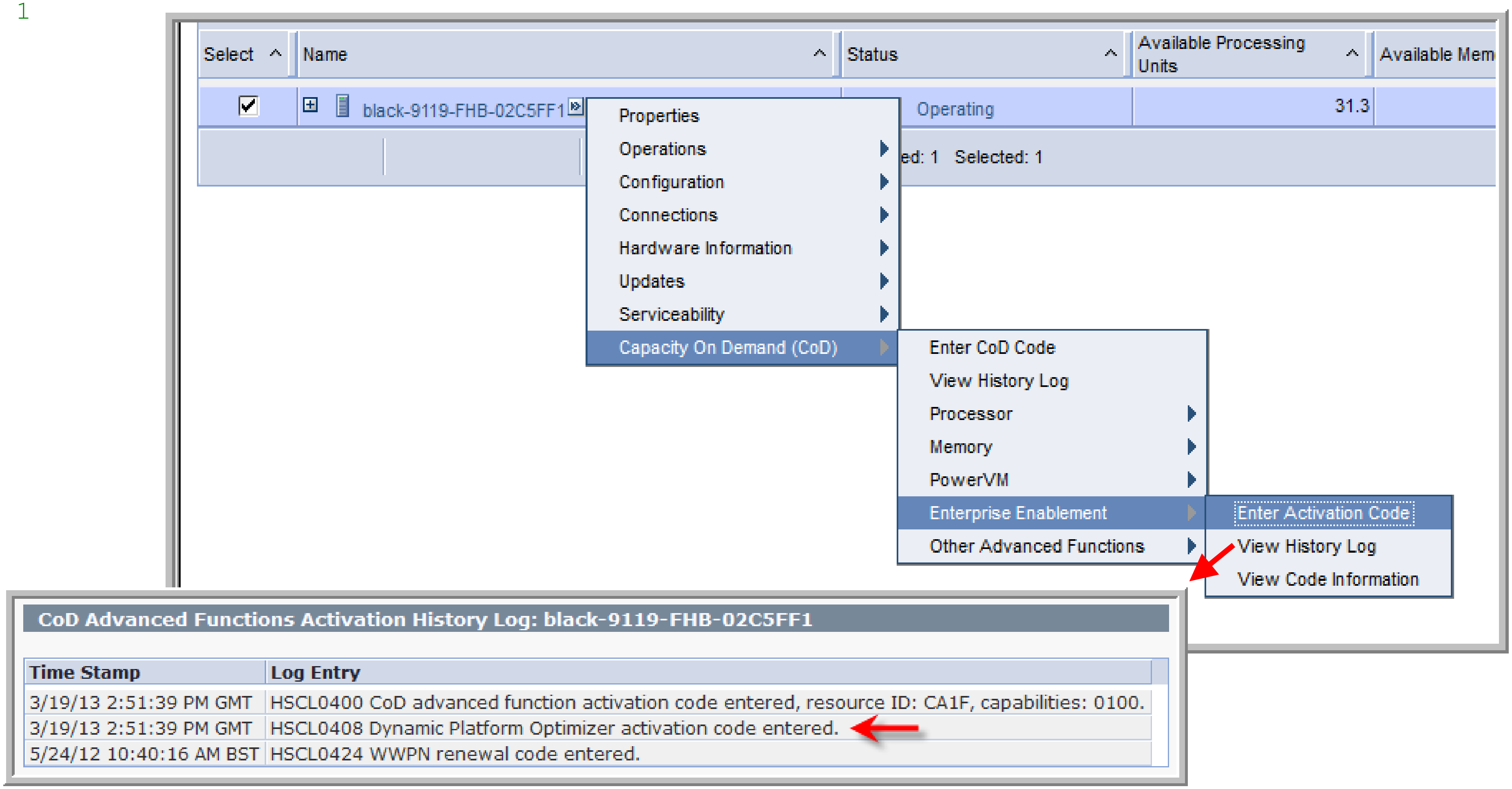
```
hscroot@blackhmc:~> lsvet -t code -m $BLACK | sed -e s/,/\\n/g
sys_type=9119
sys_serial_num=02-C5FF1
anchor_card_ccin=52C4
anchor_card_serial_num=0U-E05E005
anchor_card_unique_id=0208070737773E9F
resource_id=CA1F
activated_resources=0000
sequence_num=0040
entry_check=46
hscroot@blackhmc:~>
```



- You can check codes here
  - http://www-912.ibm.com/pod/pod

Enter the VET code

```
hscroot@blackhmc:~> chvet -m $BLACK -o e -k 3DB5F4F383E1A9A6CA1F00000100004188

hscroot@blackhmc:~> lssyscfg -m $BLACK -r sys -F dynamic_platform_optimization_capable
1
```



Select	Name	Status	Available Processing Units	Available Mem
☑	black-9119-FHB-02C5FF1	Operating	31.3	

ed: 1   Selected: 1

CoD Advanced Functions Activation History Log: black-9119-FHB-02C5FF1

Time Stamp	Log Entry
3/19/13 2:51:39 PM GMT	HSCL0400 CoD advanced function activation code entered, resource ID: CA1F, capabilities: 0100.
3/19/13 2:51:39 PM GMT	HSCL0408 Dynamic Platform Optimizer activation code entered.
5/24/12 10:40:16 AM BST	HSCL0424 WWPN renewal code entered.

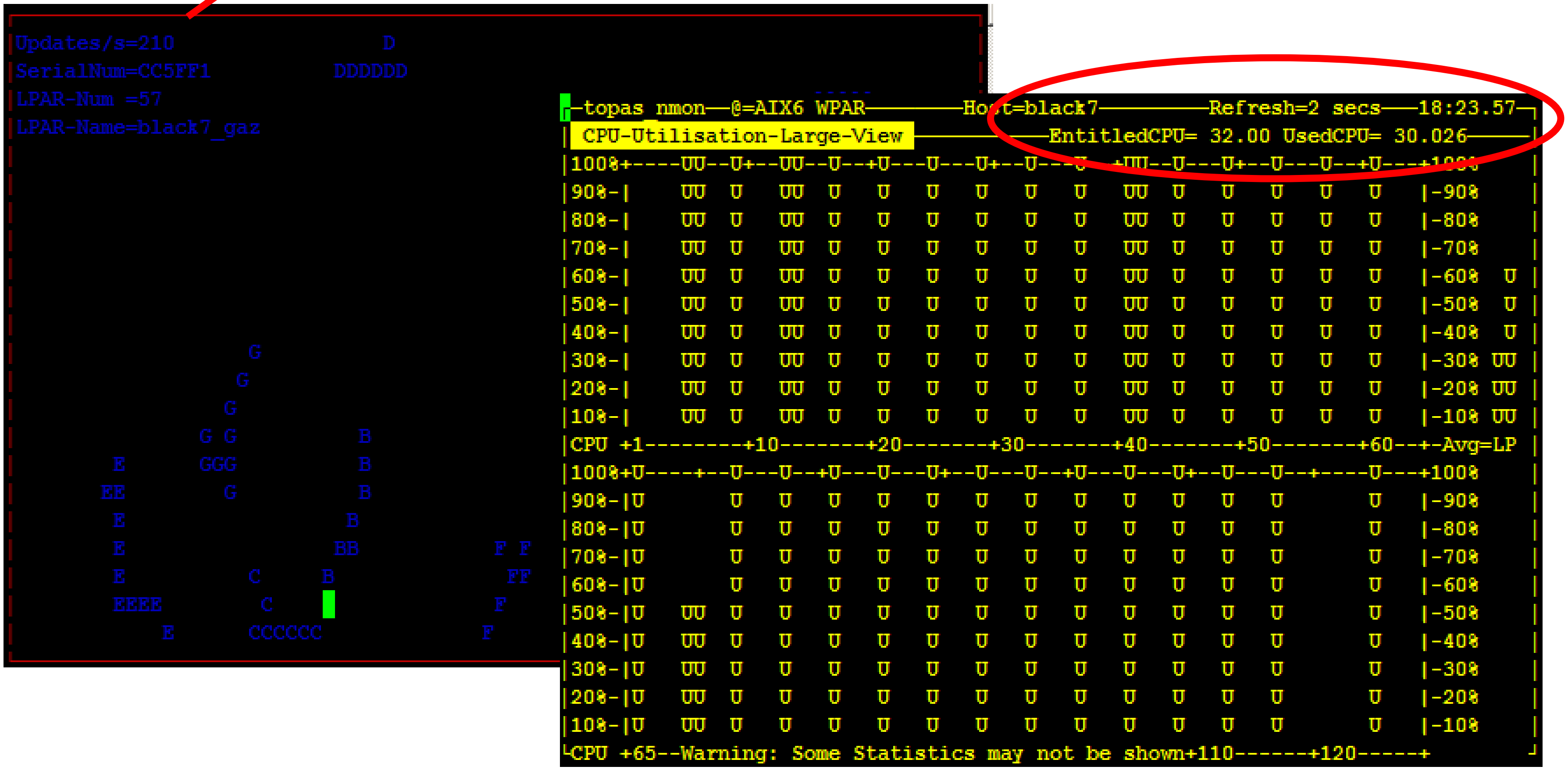
Using it on a Power 795

```
hscroot@blackhmc:~> lsmemopt -m $BLACK -o calcscore
curr_sys_score=60,predicted_sys_score=100,"requested_lpar_nam
  es=jpgtemp,02-C5FF1,02-C5FF1-
  16min,blackvios1,blackvios2,blackhpcvios1,black1_guests,bla
  ck3_ralf,black4_AndyT,black5_watts1t,black6_AIX61TL6,black7
  _gaz,black8_clive,black9-
  andyt,blackhpc1","requested_lpar_ids=1,5,6,31-33,51,53-
  59,61",protected_lpar_ids=none
hscroot@blackhmc:~>
```

```
black7:/# lssrad -av
REF1    SRAD          MEM        CPU
0
           0    12909.88       0-127
           1    17181.00
1
           2    57500.75
           3    61252.75
black7:/#
```

black7:/# ./paraworms 8 32

About 200-230 updates/sec



```
Updates/s=210                 D
SerialNum=CC5FF1          DDDDDD
LPAR-Num =57
LPAR-Name=black7_gaz







                   G
                  G
                 G
               G G          B
        E      GGG          B
       EE        G          B
        E                  B
        E                 BB           F F
        E          C     B              FF
        EEEE        C                   F
            E      CCCCCC             F
```

```
┌─topas nmon──@=AIX6 WPAR──────────Host=black7─────────Refresh=2 secs───18:23.57─┐
│ CPU-Utilisation-Large-View ───────────EntitledCPU= 32.00 UsedCPU= 30.026──────│
|100%+----UU--U+--UU--U--+U---U---U+--U---U--+UU--U---U+--U---U--+U----+100%    |
|90%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-90%    |
|80%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-80%    |
|70%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-70%    |
|60%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-60%  U |
|50%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-50%  U |
|40%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-40%  U |
|30%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-30% UU |
|20%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-20% UU |
|10%-|    UU  U   UU  U   U   U   U   U   U   UU  U   U   U   U   U   |-10% UU |
|CPU +1--------+10-------+20-------+30-------+40-------+50-------+60--+-Avg=LP |
|100%+U----+--U---U--+U---U---U+--U---U--+U---U---U+--U---U--+----U----+100%    |
|90%-|U       U   U   U   U   U   U   U   U   U   U   U   U       U   |-90%    |
|80%-|U       U   U   U   U   U   U   U   U   U   U   U   U       U   |-80%    |
|70%-|U       U   U   U   U   U   U   U   U   U   U   U   U       U   |-70%    |
|60%-|U       U   U   U   U   U   U   U   U   U   U   U   U       U   |-60%    |
|50%-|U   UU  U   U   U   U   U   U   U   U   U   U   U   U       U   |-50%    |
|40%-|U   UU  U   U   U   U   U   U   U   U   U   U   U   U       U   |-40%    |
|30%-|U   UU  U   U   U   U   U   U   U   U   U   U   U   U       U   |-30%    |
|20%-|U   UU  U   U   U   U   U   U   U   U   U   U   U   U       U   |-20%    |
|10%-|U   UU  U   U   U   U   U   U   U   U   U   U   U   U       U   |-10%    |
└CPU +65--Warning: Some Statistics may not be shown+110-------+120-----+       ┘
```

```
hscroot@blackhmc:~> optmem -m $BLACK -o start -t affinity
```

```
black7:/# lssrad -av
REF1   SRAD        MEM      CPU
0
          0   12909.88      56-59 72-75 88-91 104-107 120-123
          1   17181.00      24-27 36-39 48-51 64-67 80-83 96-99 112-115
1
          2   57500.75      0-11 16-19 28-31 40-43 52-55 68-71 84-87 100-103 116-119
          3   61252.75      12-15 20-23 32-35 44-47 60-63 76-79 92-95 108-111 124-127
black7:/#
```

```
hscroot@blackhmc:~> optmem -m $BLACK -o start -t affinity
```

```
black7:/# lssrad -av
REF1   SRAD        MEM      CPU
0
          0   12909.88      56-59 72-75 88-91 104-107 120-123
          1   17181.00      24-27 36-39 48-51 64-67 80-83 96-99 112-115
1
          2   57500.75      0-11 16-19 28-31 40-43 52-55 68-71 84-87 100-103 116-119
          3   61252.75      12-15 20-23 32-35 44-47 60-63 76-79 92-95 108-111 124-127
black7:/#
```

The correct syntax for the `lssrad` command is:

```
man lssrad
```

Scheduler Resource Allocation Domain

Observations

- We do NOT recommend running a cron job, automatically to optimise memory on all systems

- We DO recommend running a nightly cron job, automatically to gather the current and predicted scores from all servers and then mail the sysadmin

- To get the best performance it is recommended to populate all available memory slots

Demonstration

```
lssrad -av

/paraworms 8 32

BLACK=$(lssyscfg -r sys -F name)

lsmemopt -m $BLACK

lsmemopt -m $BLACK -o currscore

lsmemopt -m $BLACK -o calcscore

optmem -m $BLACK -o start -t affinity
while :
do
   date
   lsmemopt -m $BLACK -F in_progress,progress
   sleep 10
   echo
done

lssrad -av
```

Summary

- We ran loads of different tests and the score achieved was always as good as, or better than predicted.

- It requires a little analysis to be sure where the memory actually is.
  - Not that it was ever something for the uninitiated.

- The tool to look at resource locations is `lssrad`

- We think DPO is good and that you, our customers will like it.

- There is a small, noticeable, but certainly tolerable dip while the optimisation runs, but it is worth it!

Thanks !!!

W5: Pillars of Star Formation © NASA