

IBM POWER Virtualization Best Practices Guide

Version 4.0

March 2020

CONTENTS

1	INTRODUCTION	7
2	VIRTUAL PROCESSORS	7
2.1	Sizing/configuring virtual processors.....	9
2.2	Entitlement vs. Virtual processors.....	10
2.3	Matching entitlement of an LPAR close to its average utilization for better performance.....	10
2.4	When to add additional virtual processors.....	11
2.5	How to estimate the number of virtual processors per uncapped shared LPAR	11
3	AIX VIRTUAL PROCESSOR MANAGEMENT - PROCESSOR FOLDING	11
3.1	VPM folding example	12
3.2	Tuning Virtual Processor Management Folding	13
3.3	POWER7/POWER7+/POWER8/POWER9 Folding	14
3.4	Relationship between VPM Folding and PHYP dispatching.....	14
4	AIX PERFORMANCE MODE TUNING	16
4.1	Processor Bindings in Shared LPAR.....	18
5	LPAR PAGE TABLE SIZE CONSIDERATIONS	18
6	ASSIGNMENT OF RESOURCES BY THE POWERVM HYPERVISOR	19
6.1	PowerVM Resource assignment ordering.....	19
6.2	Overview of PowerVM Hypervisor Resource Assignment.....	19
6.3	How to determine if an LPAR is contained within a chip or drawer/Dual Chip Module (DCM)	20
6.3.1	Displaying Resource Assignments for AIX	21
6.3.2	Displaying Resource Assignments for Linux	21
6.3.3	Displaying Resource Assignments for IBM i	22
6.4	Optimizing Resource allocation for affinity	23
6.5	Optimizing Resource Assignment – Dynamic Platform Optimizer.....	24
6.6	Partition Mobility (LPM) and DPO considerations for Linux	26
6.7	Affinity Groups	27
6.8	Lpar_placement=2.....	27
6.9	Lpar_placement considerations for failover/Disaster recovery	28
6.10	Server Evacuation Using Partition Mobility	29
6.11	PowerVM Resource Consumption for Capacity Planning Considerations	29
6.12	Licensing resources (COD)	29
7	ENERGY SCALE™	30
8	PROCESSOR COMPATIBILITY MODE	31
9	CONCLUSION.....	31
10	REFERENCES.....	31

Table of Figures

FIGURE 1 - HMC PROCESSOR CONFIGURATION MENU	8
FIGURE 2 - VPM FOLDING EXAMPLE.....	12
FIGURE 3 – VIRTUAL PROCESSOR UNFOLDING EXAMPLE	13
FIGURE 4 – PARTITION HOME CORES.....	14
FIGURE 5 – SHARED PARTITION PLACEMENT WITH VPM FOLDING ENABLED	15
FIGURE 6 - SHARED POOL CORE PLACEMENT WITH VPM FOLDING DISABLED	15
FIGURE 7 – PERFORMANCE MODES	17
FIGURE 8 – APPLICATION THREADS EXAMPLE	18

Disclaimer – IBM Virtualization Best Practices Guide

Copyright © 2020 by International Business Machines Corporation.

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 may include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or programs(s) at any time without notice. References 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.

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 NON-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.

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.

Statements regarding IBM's future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only.

The provision 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

U.S.A.

Acknowledgements

We would like to thank the many people who made invaluable contributions to this document. Contributions included authoring, insights, ideas, reviews, critiques and reference documents.

Our special thanks to key contributors from IBM Power Systems Performance:

Abraham Alvarez - Power Systems Performance

Clark Anderson - Power Systems Performance

Qunying Gao - Power Systems Performance

Dirk Michel – Power Systems Performance

Rick Peterson – Power Systems Performance

Our special thanks to key contributors from IBM Development:

Pete Heyrman – PowerVM Development

Glen Nelson – IBM i Development

Dan Toft – IBM i Development

Our special thanks to key contributors from IBM i Global Support Center:

Kevin Chidester -IBM i Performance

Our special thanks to key contributors from IBM Lab Services Power Systems Delivery Practice:

Eric Barsness –IBM i Performance

Tom Edgerton –IBM i Performance

Preface

This document is intended to address IBM POWER processor virtualization best practices to attain best logical partition (LPAR) performance. This document by no means covers all the PowerVM™ best practices so this guide should be used in conjunction with other PowerVM documents.

1 INTRODUCTION

This document covers best practices for virtualized environments running on POWER systems. This document can be used to achieve optimized performance for workloads running in a virtualized shared processor logical partition (SPLPAR) environment.

2 VIRTUAL PROCESSORS

A virtual processor is a unit of virtual processor resource that is allocated to a partition or virtual machine. PowerVM Hypervisor (PHYP) can map a whole physical processor core or can time slice a physical processor core.

PHYP time slices shared processor partitions (also known as “micro-partitions”) on the physical CPU’s by dispatching and un-dispatching the various virtual processors for the partitions running in the shared pool. The minimum processing capacity per processor is 1/20 of a physical processor core, with a further granularity of 1/100, and the PHYP uses a 10 millisecond (ms) time slicing dispatch window for scheduling all shared processor partitions' virtual processor queues to the PHYP physical processor core queues.

General	Processors	Memory	I/O	Virtual Adapters	Power Controlling	Settings	Tagged I/O	OptiConnect
---------	-------------------	--------	-----	------------------	-------------------	----------	------------	-------------

Detailed below are the current processing settings for this partition profile.

Processing mode

Dedicated
 Shared

Processing units

Total managed system processing units : 20.00

Minimum shared processing units :

Desired shared processing units :

Maximum shared processing units :

Shared processor pool: ▼

Virtual processors

Minimum processing units required for each virtual processor : 0.05

Minimum virtual processors :

Desired virtual processors :

Maximum virtual processors :

Sharing mode

Uncapped Weight : ▲▼

Processor compatibility mode: ▼

Figure 1 - HMC processor configuration menu

If a partition has multiple virtual processors, they may or may not be scheduled to run simultaneously on the physical processors.

Partition entitlement is the guaranteed resource available to a partition. A partition that is defined as capped, can only consume the processors units explicitly assigned as its entitled capacity. An uncapped partition can consume more than its entitlement but is limited by several factors:

- Uncapped partitions can exceed their entitlement if:
 - There is unused capacity in the shared pool
 - There are dedicated partitions configured to share their physical processors while active or inactive
 - unassigned physical processors are available
 - COD utility processors have been configured

- If the partition is assigned to a virtual shared processor pool, the capacity for all the partitions in the virtual shared processor pool may be limited
- The number of virtual processors in an uncapped partition throttles on how much CPU it can consume. For example:
 - An uncapped partition with 1 virtual CPU can only consume 1 physical processor of CPU resource under any circumstances
 - An uncapped partition with 4 virtual CPUs can only consume 4 physical processors of CPU
- Virtual processors can be added or removed from a partition using HMC actions. (Virtual processors can be added up to the Maximum virtual processors of an LPAR and virtual processors can be removed up to the Minimum virtual processors of an LPAR)

2.1 SIZING/CONFIGURING VIRTUAL PROCESSORS

The number of virtual processors in each LPAR in the system should not “exceed” the number of cores available in the system (CEC/framework) or if the partition is defined to run in specific virtual shared processor pool, the number of virtual processors should not exceed the maximum defined for the specific virtual shared processor pool. Having more virtual processors configured than can be running at a single point in time does not provide any additional performance benefit. Instead context switching overhead of excessive virtual processors can reduce performance.

*For example: A shared pool is configured with **32 physical cores**. Eight SPLPARs are configured in this case each LPAR can have up to **32 virtual processors**. This allows a LPAR that has 32 virtual processors to get 32 CPUs if all the other LPARs are not using their entitlement. **Setting > 32 virtual processors is not necessary as there are only 32 CPUs in the pool**. If there are sustained periods of time where there is sufficient demand for all the shared processing resources in the system or a virtual shared processor pool, it is prudent to configure the number of virtual processors to match the capacity of the system or virtual shared processor pool.*

A single virtual processor can consume a whole physical core under two conditions:

1. SPLPAR is given an entitlement of 1.0 or more processor
2. This is an uncapped partition and there is idle capacity in the system.

Therefore, there is no need to configure more than one virtual processor to get one physical core.

For example: A shared pool is configured with 16 physical cores. Four SPLPARs are configured each with entitlement 4.0 cores. We need to consider the workload’s sustained peak demand capacity when configuring virtual processors. If two of the four SPLPARs would peak to use 16 cores (max available in the pool), then those two SPLPARs would need 16 virtual CPUs. The other two peaks only up to 8 cores, those two would be configured with 8 virtual CPUs

2.2 ENTITLEMENT VS. VIRTUAL PROCESSORS

Entitlement is the capacity that a SPLPAR is guaranteed to get as its share from the shared pool. Uncapped mode allows a partition to receive excess cycles when there are free (unused) cycles in the system.

Entitlement also determines the number of SPLPARs that can be configured for a shared processor pool. That is, the sum of the entitlement of all the SPLPARs cannot exceed the number of physical cores configured in a shared pool.

For example: Shared pool has 8 cores, 16 SPLPARs are created each with 0.1 core entitlement and 1 virtual CPU. We configured the partitions with 0.1 core entitlement since these partitions are not running that frequently. In this example, the sum of the entitlement of all the 16 SPLPARs comes to 1.6 cores. The rest of 6.4 cores and any unused cycles from the 1.6 entitlement can be dispatched as uncapped cycles.

At the same time keeping entitlement low when there is capacity in the shared pool is not always a good practice. Unless the partitions are frequently idle or there is a plan to add more partitions, the best practice is that the sum of the entitlement of all the SPLPARs configured should be close to the capacity in the shared pool. Entitlement cycles are guaranteed, so while a partition is using its entitlement cycles, the partition is not pre-empted, whereas a partition can be preempted when it is dispatched to use excess cycles. Following this practice allows the hypervisor to optimize the affinity of the partition's memory and processors and it also reduces unnecessary preemptions of the virtual processors.

2.3 MATCHING ENTITLEMENT OF AN LPAR CLOSE TO ITS AVERAGE UTILIZATION FOR BETTER PERFORMANCE

The aggregate entitlement (min/desired processor) capacity of all LPARs in a system is a factor in the number of LPARs that can be allocated. The minimum entitlement is what is needed to boot the LPARs, however the desired is what an LPAR will get if there are enough resources available in the system. The best practice for LPAR entitlement would be to match the entitlement capacity to average utilization and let the peak be addressed by additional uncapped capacity.

The rule of thumb would be setting entitlement close to average utilization for each of the LPARs in a system, however this rule can be relaxed when there are cases where a LPAR must be given higher priority compared to other LPARs in a system. For example, if the production and non-production workloads are consolidated on the same system, production LPARs would be preferred to have higher priority over non-production LPARs. In which case, in addition to setting higher weights for uncapped capacity, the entitlement of the production LPARs can be raised while reducing the entitlement of non-production LPARs. This allows these important production LPARs to have better partition resource assignments (affinity) and these LPARs will have additional entitled capacity so not to rely solely on uncapped processing. At the same time if production SPLPAR is not using their entitled capacity, then that capacity can be used by non-

production SPLPAR and the non-production SPLPAR will be pre-empted if production SPLPAR needs its capacity.

2.4 WHEN TO ADD ADDITIONAL VIRTUAL PROCESSORS

When there is sustained need for a shared LPAR to use additional resources in the system in uncapped mode and there is available capacity, increasing virtual processors are recommended. Keep in mind that this may negatively affect the performance of other partitions on the system if resources become constrained.

2.5 HOW TO ESTIMATE THE NUMBER OF VIRTUAL PROCESSORS PER UNCAPPED SHARED LPAR

The first step is to monitor the utilization of each partition and for any partition where the average utilization is ~100% one virtual processor can be added. That is, use the capacity of the already configured virtual processors before adding more. Additional virtual processors are going to run concurrently if there are enough free processors available in the shared pool.

If the peak utilization is well below 50% mark, then there is no need for additional virtual processors. In this case, look at the ratio of virtual processors to configured entitlement and if the ratio is > 1 , then consider reducing the ratio.

For AIX, if there are too many virtual processors configured, the operating system can “fold” those processors so that the workload would run on fewer virtual processors to optimize virtual processor performance.

For example, if an SPLPAR is given a CPU entitlement of 2.0 cores and 4 virtual processors in an uncapped mode then the hypervisor could dispatch the virtual processors to 4 physical cores concurrently if there are free cores available in the system. The SPLPARs leverages unused cores and the applications can scale up to 4 cores. However, if the system does not have free cores then the hypervisor would have to dispatch 4 virtual processors on 2 cores, so the concurrency is limited to 2 cores. In this situation, each virtual processor is dispatched for reduced time slice as 2 cores are shared across 4 virtual processors. This situation could impact performance; therefore AIX operating system processor folding support may be able to reduce to number of virtual processors being dispatch such that only 2 or 3 virtual processors are dispatched across the 2 physical processors.

3 AIX VIRTUAL PROCESSOR MANAGEMENT - PROCESSOR FOLDING

For AIX partitions, the operating system dynamically manages the number of active virtual processors through VPM (Virtual Processor Management), also known as processor folding. VPM folding will be enabled for dedicated processor partitions if static power saving has been enabled for the system.

VPM determines the CPU utilization of the partition once per second and calculates the number of virtual processors needed based on the aggregate CPU usage plus an additional 20% for headroom. A virtual processor will be disabled (folded) if the aggregate CPU usage is below 80% of the enabled virtual processors. If available, an additional virtual CPU will be enabled (unfolded) if the CPU usage is beyond 80% of the enabled virtual processors.

3.1 VPM FOLDING EXAMPLE

The example below illustrates the unfolding of virtual processors. In this example the partition has eight virtual processors. The partition initially is idle and therefore has only one virtual processor unfolded.

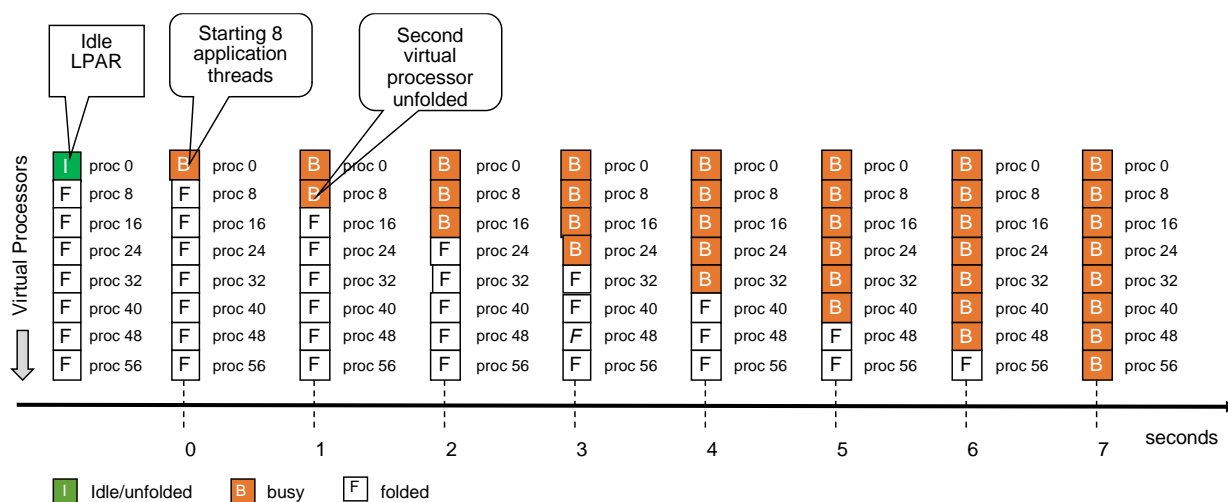


Figure 2 - VPM Folding Example

- At 0 seconds we start a workload that consists of eight compute intensive application threads. At this time only one virtual processor is unfolded. Therefore, all application threads are running on one processor. In raw performance throughput mode, the application threads will mainly run on the primary SMT thread since overall partition utilization is too low to place them on the secondary or tertiary SMT threads. In scaled throughput mode the application threads will be placed on primary and secondary or on all SMT threads depending on which mode has been set.
- At 1 second an additional virtual processor gets unfolded since the aggregate CPU utilization of the enabled virtual processors exceeded the folding threshold. The same happens every second until all eight virtual processors are unfolded.

Note: The number of unfolded virtual processors will be lower when running scaled throughput mode since the eight application threads would be placed on four or two virtual processors.

3.2 TUNING VIRTUAL PROCESSOR MANAGEMENT FOLDING

For most workloads, unfolding one virtual processor per second is sufficient. However, some workloads are very response time sensitive and require available CPU resources immediately to achieve good performance. Disabling VPM folding would be one option but that would cause all virtual processors to be active all the time, whether they are needed or not. For most cases, a better way is to unfold more virtual processors in addition to the number determined by VPM. This can be done through the schedo tunable `vpm_xvcpus`.

Taking the same example as in section 1.2.7 and setting `vpm_xvcpus` to 2 will unfold virtual processors as illustrated below:

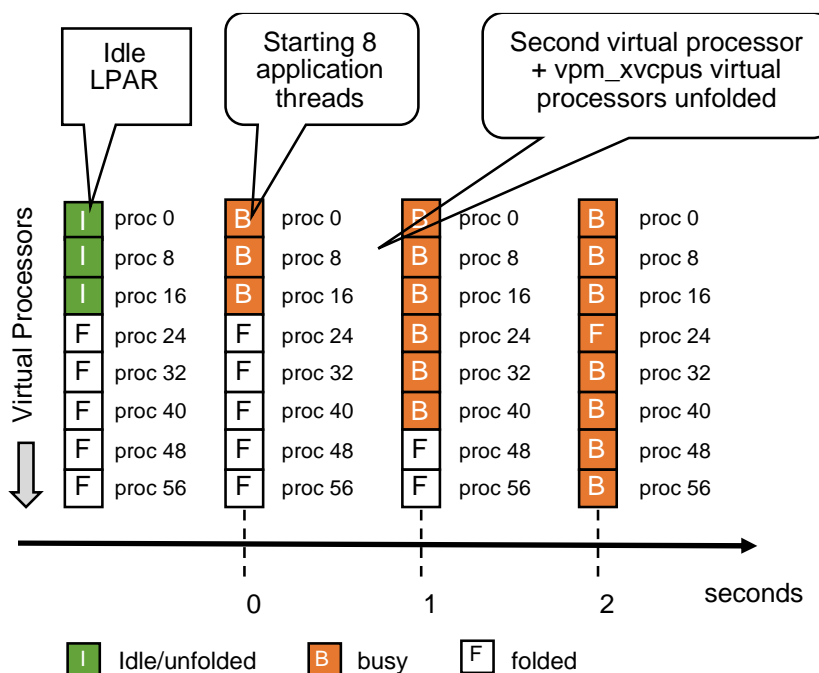


Figure 3 – Virtual Processor Unfolding Example

Like in the previous example, the partition has eight virtual processors. Three of the eight virtual processors are unfolded due to setting `vpm_xvcpus=2`.

- At 0 seconds we start a workload that consists of eight compute intensive application threads. The eight application threads are now spread across the three unfolded processors.
- At 1 second VPM determined that one additional virtual processor is needed. Three were enabled before, so the new calculated value is four, plus the additional specified by `vpm_xvcpus`.
- At 2 seconds, all virtual processors are unfolded.

The example above uses a `vpm_xvcpus` value of 2 to demonstrate how quickly virtual processors are unfolded through `vpm_xvcpus` tuning. A value `vpm_xvcpus` value of 1 usually is sufficient for response time sensitive workloads.

Note: Tuning Virtual Processor Management folding too aggressively can have a negative impact on PHYP's ability to maintain good virtual processor dispatch affinity. Please see section 3.4 which describes the relationship between VPM folding and PHYP dispatching.

3.3 POWER7/POWER7+/POWER8/POWER9 FOLDING

VPM folding supports multiple SRADs (Scheduler Resource Allocation Domain) on POWER7®, POWER7+™, POWER8®, and POWER9™ shared processor partitions. VPM folds and unfolds virtual processors per SRAD. Virtual processors fold and unfold more quickly on shared processor partitions that have more than one SRAD.

3.4 RELATIONSHIP BETWEEN VPM FOLDING AND PHYP DISPATCHING

VPM folding ensures that the number of active virtual processors is kept to what is really needed by the workload, plus a 20% additional headroom. This has a direct impact on PHYP's ability to maintain good dispatching affinity in the case appropriate partition sizing was done based on the Virtualization Best Practices.

The following example assumes a shared processor partition with 10 virtual processors and an entitlement of 6.0. The partition is placed in a shared processor pool that spawns across 24 cores and three 8-core chips.

The figure below illustrates the shared processor pool with the placement of the home cores (entitled cores) within the pool. Please note that the partition placement of the illustration below represents a simplified example to explain the relationship between VPM folding and PHYP dispatching. Actual partition placement or placement rules are not covered in this section.

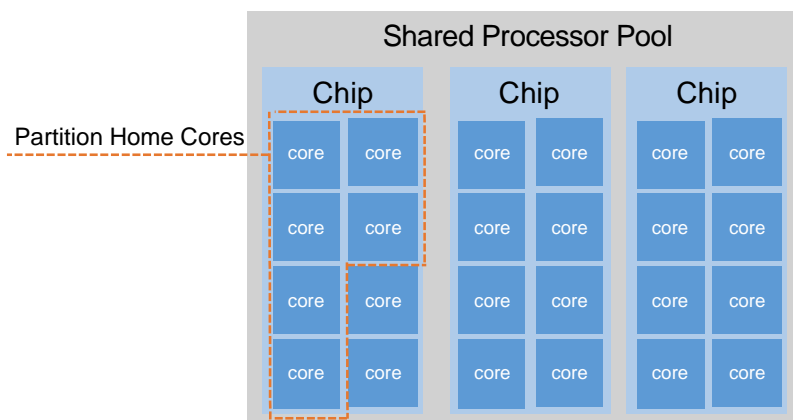


Figure 4 – Partition Home Cores

Let's assume that the workload put onto the partition consumes four physical core (PC 4.0) and that the shared processor pool is otherwise idle. With processor folding enabled, five virtual processors will be unfolded and PHYP can dispatch the virtual processor onto the home cores of the partition.

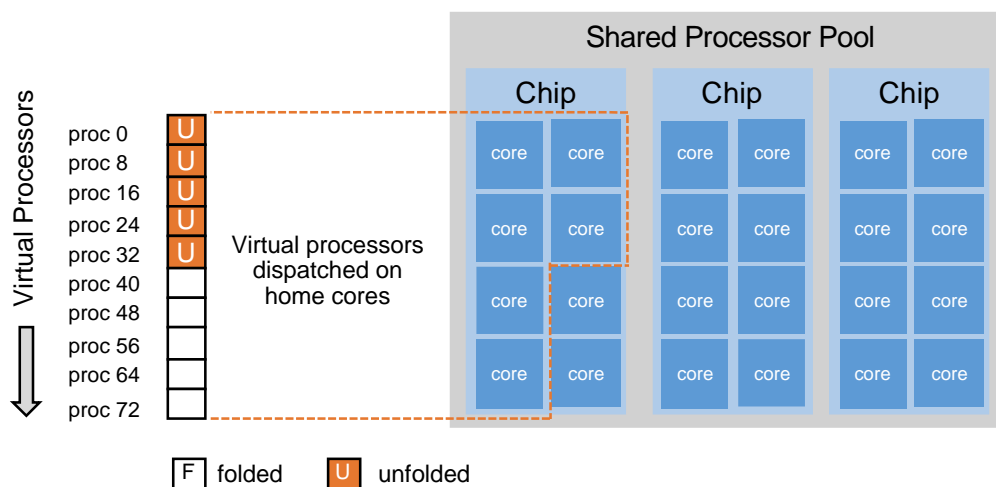


Figure 5 – Shared Partition Placement with VPM Folding Enabled

The figure below illustrates what will happen running the same workload with VPM folding disabled. With VPM folding disabled, all 10 virtual processors of the partition are unfolded. It is not possible to pack all 10 virtual processors onto the five home cores if they are active concurrently. Therefore, some virtual processors get dispatched on other cores of the shared pool.

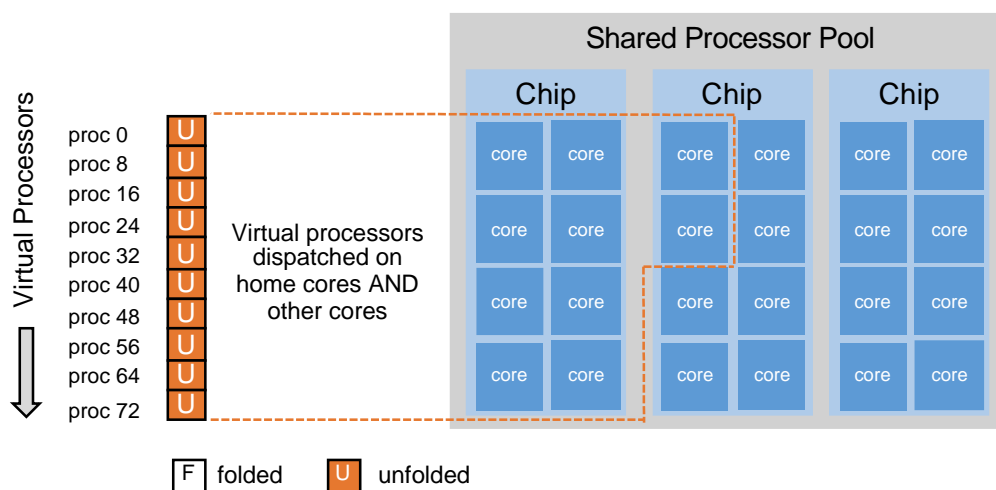


Figure 6 - Shared Pool Core Placement with VPM Folding Disabled

Ideally, the virtual processors that cannot be dispatched on their home cores get dispatched on a core of a chip that is in the same node. However, other chip(s) in the same node might be busy at the time, so a virtual processor gets dispatched in another node.

For the example above, there is no performance benefit of disabling folding. More virtual processors than needed are unfolded; physical dispatch affinity suffers and typically the physical CPU consumption increases.

4 AIX PERFORMANCE MODE TUNING

AIX provides a dispatching feature through the schedo tuneable `vpm_throughput_mode`, which allows greater control over workload dispatching. There are 5 options, 0,1,2,4 and 8 that can be set dynamically. Mode0 and 1 cause the AIX partition to run in raw throughput mode, and modes 2 through 8 switch the partition into scaled throughput mode.

The default behavior is raw throughput mode, same as legacy versions of AIX. In raw throughput mode (`vpm_throughput_mode=0`), the workload is spread across primary SMT threads.

Enhanced raw throughput mode (`vpm_throughput_mode=1`), behaves similar to mode0 by utilizing primary SMT threads; however, it attempts to lower CPU consumption by slightly increasing the unfold threshold policy. It typically results in a minor reduction in CPU utilization.

The new behavior choices are for scaled throughput mode SMT2 (`vpm_throughput_mode=2`), scaled throughput mode SMT4 (`vpm_throughput_mode=4`) and scaled throughput mode SMT8 (`vpm_throughput_mode=8`). These options allow for the workload to be spread across 2 through 8 SMT threads, accordingly. The throughput modes determine the desired level of SMT exploitation on each virtual processor core before unfolding another core. A higher value will result in fewer cores being unfolded for a given workload.

IBM Power Virtualization Best Practices Guide

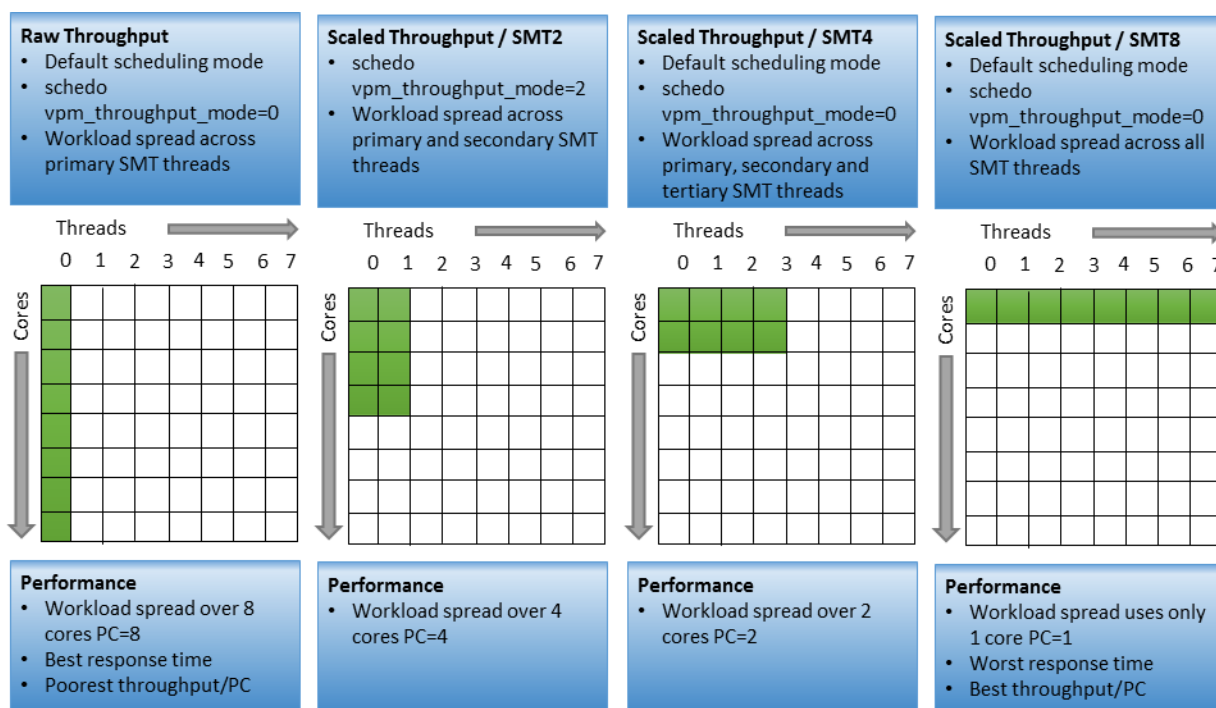


Figure 7 – Performance Modes

The AIX scheduler is optimized to provide best raw application throughput on POWER7 – POWER9 processor-based servers. Modifying the AIX dispatching behavior to run in scaled throughput mode will have a performance impact that varies based on the application.

There is a correlation between the aggressiveness of the scaled throughput mode and the potential impact on performance; a higher value increases the probability of lowering application throughput and increasing application response times.

The scaled throughput modes, however, can have a positive impact on the overall system performance by allowing those partitions using the feature to utilize less CPU (unfolding fewer virtual processors). This reduces the overall demand on the shared processing pool in an over provisioned virtualized environment.

The tuneable is dynamic and does not require a reboot; this simplifies the ability to revert the setting if it does not result in the desired behavior. It is recommended that any experimentation of the tuneable be done on non-critical partitions such as development or test before deploying such changes to production environments. For example, a critical database SPLPAR would benefit most from running in the default raw throughput mode, utilizing more cores even in a highly contended situation to achieve the best performance; however, the development and test SPLPARs can sacrifice by running in a scaled throughput mode, and thus utilizing fewer virtual processors by leveraging more SMT threads per core. However, when the LPAR utilization reaches maximum level, AIX dispatcher would use all the SMT8 threads of all the virtual processors irrespective of the mode settings.

The example below illustrates the behavior within a raw throughput mode environment. Notice how the application threads are dispatched on primary processor threads, and the secondary and tertiary processor threads are utilized only when the application thread count exceeds 8 and 16 accordingly.

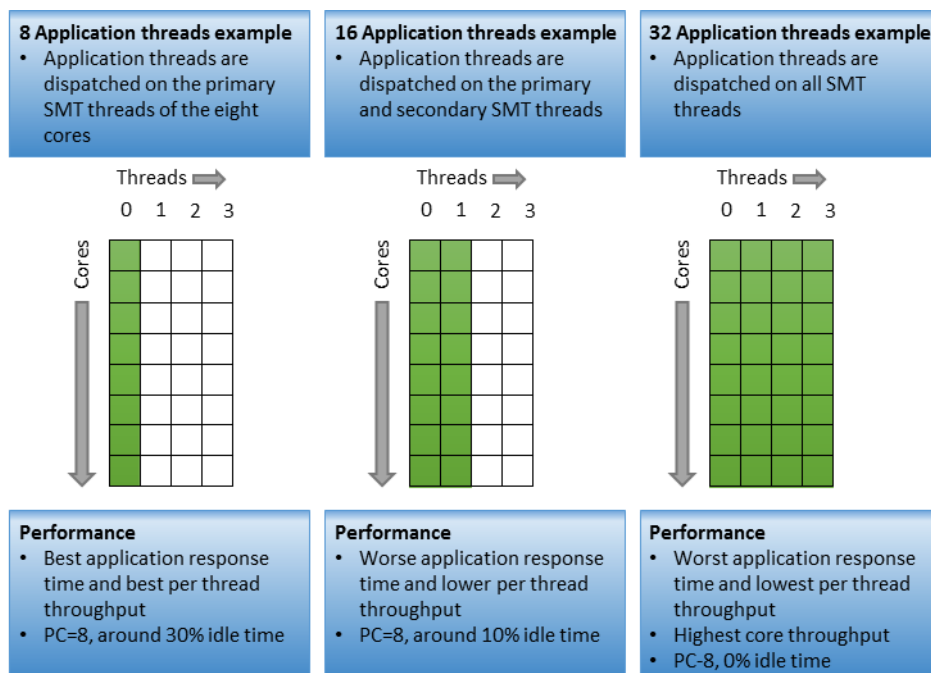


Figure 8 – Application Threads Example

4.1 PROCESSOR BINDINGS IN SHARED LPAR

Under AIX, binding processors is available to application running in a shared LPAR. An application process can be bound to a virtual processor in a shared LPAR. In a shared LPAR a virtual processor is dispatched by PowerVM hypervisor. In firmware level 730 and later, the PowerVM hypervisor maintains three levels of affinity for dispatching, core, chip and node level. By maintaining affinity at hypervisor level as well as in AIX, applications may achieve higher level affinity through processor bindings.

5 LPAR PAGE TABLE SIZE CONSIDERATIONS

The hardware page table of an LPAR is sized based on the maximum memory size of an LPAR not what is currently assigned (desired) to the LPAR. There are some performance considerations if the maximum size is set significantly higher than the desired memory.

1. Larger page table tends to help performance of the workload as the hardware page table can hold more pages. This will reduce translation page faults. Therefore, if there is enough memory in the system and would like to improve translation page faults, set your max memory to a higher value than the LPAR desired memory.
2. On the downside, more memory is used for hardware page table; this not only wastes memory but also makes the table become sparse which results in a couple of things:
 - a. dense page table tends to help better cache affinity due to reloads
 - b. lesser memory consumed by hypervisor for hardware page table more memory is made available to the applications
 - c. lesser page walk time as page tables are small

6 ASSIGNMENT OF RESOURCES BY THE POWERVM HYPERVISOR

6.1 POWERVM RESOURCE ASSIGNMENT ORDERING

When assigning resources to partition either at server boot or using the default options on the Dynamic Platform Optimizer, the hypervisor assigns resources one partition at a time until all partitions have been assigned resources. This implies that the partitions that are among the first to receive resources have a better chance of acquiring the optimum resources as compared to the last partitions to receive resources. The following is the order the hypervisor uses when determining the priority for resource assignment:

1. Partitions that have explicit affinity hints like affinity groups
2. Partitions with dedicated processor partitions
3. Higher uncapped weight for shared processor partitions
4. Partition size (the more processors and memory a partition has, the higher the priority)

For example, all the dedicated processor partitions are assigned resource before any shared processor partitions. Within the shared processor partitions, the ones with the highest uncapped weight setting will be assigned resources before the ones with the lower uncapped weights.

6.2 OVERVIEW OF POWERVM HYPERVISOR RESOURCE ASSIGNMENT

The hypervisor tries to optimize the assignment of resources for a partition based on the number of physical cores (dedicated processor partitions), entitled capacity (shared processor partitions), desired amount of memory and physical I/O devices assigned to the partition. The basic rules are:

1. Try to contain the processors and memory to a single processor chip that has the most I/O assigned to the partition.

2. If the partition cannot be contained to a single processor chip, try and assign the partition to a single DCM/Drawer that has the most I/O assigned to the partition.
3. If the partition cannot be contained to a single DCM/Drawer, contain the partition to the least number of DCMs/Drawers.

Notes:

1. The amount of memory the hypervisor attempts to assign not only includes the user defined memory but also additional memory for the Hardware Page Table for the partition and other structures that are used by the hypervisor to manage the partition.
2. The amount of memory available in a processor chip, DCM or Drawer is lower than the physical capacity. The hypervisor has data structures that are used to manage the system like I/O tables that reduce the memory available for partitions. These I/O structures are allocated in this way to optimize the affinity of I/O operation issued to devices attached to the processor chip.

6.3 HOW TO DETERMINE IF AN LPAR IS CONTAINED WITHIN A CHIP OR DRAWER/DUAL CHIP MODULE (DCM)

Each of the Operating Systems supported on PowerVM based servers provide a method of reporting physical processors and memory resources assigned to an individual partition. This section will explain a couple common characteristics of the information reported followed by instructions by OS type that can be used to display the resource assignment information.

The OS output for displaying resources assignments have the following common characteristics:

1. The total amount of memory reported is likely less than the physical memory assigned to the partition because some of the memory is allocated very early in the AIX boot process and therefore not available for general usage.
2. If the partition is configured with dedicated processors, there is a static 1-to-1 binding between logical processors and the physical processor hardware. A given logical processor will run on the same physical hardware each time the PowerVM hypervisor dispatches the virtual processor. There are exceptions like automatic processor recovery which would happen in the very rare case of a failure of a physical processor.
3. For partitions configured with shared processors, the hypervisor maintains a “home” dispatching location for each virtual processor. This “home” is what the hypervisor has determined is the ideal hardware resource that should be used when dispatching the virtual processor. In most cases this would be a location where the memory for the partition also resides. Shared processors allow partitions to be overcommitted for better utilization of the hardware resources available on the server so the OS output may report more logical processors in each domain than physical possible. For example, if you have a server with 8 cores per processor chip, a partition with 8.0 desired entitlement, 10 VPs and memory, the hypervisor is likely to assign resources for the partition from a single chip. If you examine the OS output with respect to processors, you would see all the

logical processors are contained to the same single domain as memory. Of course, since there are 10 VPs but only 8 cores per physical processor chip, not all 10 VPs could be running simultaneously on the “home” chip. Again, the hypervisor picked the same “home” location for all 10 VPs because whenever one of these VPs runs the hypervisor want to dispatch the partition as close to the memory as possible. If all the cores are already busy when the hypervisor attempts to dispatch a VP, the hypervisor will then attempt to dispatch the VP on the same DCM/drawer as the “home”.

6.3.1 Displaying Resource Assignments for AIX

From an AIX LPAR, the `lssrad` command can be used to display the number of domains a LPAR is using. The `lssrad` syntax is:

```
lssrad -av
```

If the cores and memory are located in multiple processor chips in a single drawer or DCM the output would look similar to

REF1	SRAD	MEM	CPU
0	0	31806.31	0-31
	1	31553.75	32-63

REF1 is the second level domain (Drawer or DCM). SRAD refers to processor chips. The “`lssrad`” results for REF1 and SRAD are logical values and cannot be used to determine actual physical chip/drawer/DCM information. Note that CPUs refer to logical CPUs so if the partition is running in SMT8 mode, there are 8 logical CPUs for each virtual processor configured on the management console.

6.3.2 Displaying Resource Assignments for Linux

From a Linux partition the `numactl` command can be used to display the resource assignment.

The `numactl` syntax is: `numactl -H`

If the processors and some memory are allocated from one chip and there is memory without any processors on another chip, the output would look like the following:

```
available: 2 nodes (0-1)
node 0 cpus:
node 0 size: 287744 MB
node 0 free: 281418 MB
node 1 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 22 23
node 1 size: 480256 MB
```

IBM Power Virtualization Best Practices Guide

```
node 1 free: 476087 MB
```

```
node distances:
```

```
node    0    1
      0:  10  40
      1:  40  10
```

A node from Linux corresponds to an individual processor chip so in this example the partition is using resources from 2 processor chips. On the first chip is 287,744 MB of memory and on the second chip is 480,256 MB of memory and 24 logical processors. Since this partition is running in SMT8 mode, this corresponds to 3 virtual processors with 8 hardware threads per virtual processor.

6.3.3 Displaying Resource Assignments for IBM i

IBM i has support in the Start SST (STRSST command) and the Dedicated Service Tools (DST) interface to display the resource assignment for a partition. The following steps can be used to display this information:

1. Use the DST function or the *STRSST* command to start the service tool function
2. Select the *Start a service tool*
3. Select the *Display/Alter/Dump*
4. Select the *Display/Alter storage or Dump to printer*
5. Select *Licensed Internal Code (LIC) data*
6. Select *Advanced analysis* (Note—you may need to roll to see additional options)
7. Enter *1* in the option column and enter *rmnodeinfo* in the command column
8. Enter *-N* in the Options prompt

The following is an example of the output from the *rmnodeinfo* command

```
Running macro: RMNODEINFO -N
rmnodeinfo macro start at 02/18/2020 15:22:16
Node statistics for 2 nodes across 2 node groups
Node #          |          0 |          1 |
=====
Hardware node ID |          1 |          2 |
Node group ID   |          0 |          1 |
Hardware group ID |          0 |          2 |
```

IBM Power Virtualization Best Practices Guide

```
# of Logical procs |          16 |          16 |
# of procs folded  |           8 |          16 |
# main store pages | 000748DB | 0007B8A8 |
```

This is an example of a partition that is spread across 2 processor chips (Node # 0 and 1). The *# of Logical procs* field indicates there are 16 logical processors on each of the two processor chips. The *# main store pages* field reports in hexadecimal the number of 4K pages assigned to each of the two processor chips.

6.4 OPTIMIZING RESOURCE ALLOCATION FOR AFFINITY

PowerVM when all the resources are free (an initial machine state or reboot of the CEC) will allocate memory and cores as optimally as possible. At server boot time, PowerVM is aware of all the LPAR configurations so, assignment of processors and memory are irrespective of the order of activation of the LPARs.

However, after the initial configuration, the setup may not stay static. There will be numerous operations that can change the resource assignments:

1. Reconfiguration of existing LPARs with updated profiles
2. Reactivating existing LPARs and replacing them with new LPARs
3. Adding and removing resources to LPAR dynamically (DLPAR operations)

Note: If the resource assignment of the partition is optimal, you should activate the partition from the management console using the “Current Configuration” option instead of activating with a partition profile. This will ensure that the PowerVM hypervisor does not re-allocate the resources which could affect the resources currently assigned to the partition.

If the current resource assignment is causing a performance issue, Dynamic Platform Optimizer is the most efficient method of improving the resource assignment. More information on the optimizer is available later in this section.

If you wish to try to improve the resource assignment of a single partition and do not want to use the dynamic platform optimizer, power off the partition, use the `chhwres` command to remove all the processor and memory resources from the partition and then activate the partition with desired profile. The improvement (if any) in the resource assignments will depend on a variety of factors such as: which processors are free, which memory is free, what licensed resources are available, configuration of other partitions and so on.

Fragmentation due to frequent movement of memory or processors between partitions can be avoided through proper planning ahead of the time. DLPAR actions can be done in a controlled way so that the performance impact of resource addition/deletion will be minimal. Planning for growth would help to alleviate the fragmentation caused by DLPAR operations.

6.5 OPTIMIZING RESOURCE ASSIGNMENT – DYNAMIC PLATFORM OPTIMIZER

A feature is available called the Dynamic Platform Optimizer. This optimizer automates the previously described manual steps to improve resource assignments. The following functions are available on the HMC command line:

“`lsmemopt -m <system_name> -o currscore`” will report the current affinity score for the server. The score is a number in the range of 0-100 with 0 being poor affinity and 100 being perfect affinity. Note that the affinity score is based on hardware characteristics and partition configurations so a score of 100 may not be achievable.

“`lsmemopt -m <system_name> -r sys -o calcscore [-p partition_names | --id partition_ids] [-x partition-names | --xid partition-ids]`” will report the potential score that could be achieved by optimizing the system with the Dynamic Platform Optimizer. Again, a calculated score of 100 may not be possible. The scoring is meant to provide a gauge to determine if running the optimizer is likely to provide improved performance. For example, if the current score is 85 and the calculated score is 90, running the optimizer may not have a noticeable impact on overall systems performance. The amount of gain from doing an optimization is dependent on the applications running within the various partitions and the partition resource assignments. If the -p, --id, --x and/or --xid parameters are specified with calcscore, this will approximate the score on a partition basis that could be achieved if the optmem command was run with the same parameters.

Also note that this is a system-wide score that reflects all the resources assigned to all the partitions. The resource assignments of individual partitions contribute to the system wide score relative to the amount of resources assigned to the partition (large processor/memory partition contributes more to the system wide score than small partition). Making any configuration change (even activating a partition that wasn't assigned resources previously) can change the overall score.

“`lsmemopt -m <system_name> -r lpar -o currscore | calcscore [-p partition_names | --id partition_ids] [-x partition-names | --xid partition-ids]` will report the current or potential score on a partition basis. The scoring is based on the same 0-100 scale as the system wide scoring. If the -p, --id, --x and/or --xid parameters are specified with calcscore, this will approximate the score on a partition basis that could be achieved if the optmem command was run with the same parameters.

“`lsmemopt -m <system_name>`” displays the status of the optimization as it progresses.

“`optmem -m <system_name> -o start -t affinity`” will start the optimization for all partitions on the entire server. The time the optimization takes is dependent upon: the current resource assignments, the overall amount of processor and memory that need to be moved, the amount of CPU cycles available to run this optimization and so on. See additional parameters in the following paragraphs.

“`optmem -m <system_name> -o stop`” will end an optimization before it has completed all the movement of processor and memory. This can result in affinity being poor for some partitions that were not completed.

Additional parameters are available on these commands that are described in the help text on the HMC command line interface (CLI) commands. One option on the `optmem` command is the exclude parameter (`-x` or `-xid`) which is a list of partition that should not be optimized (left as is with regards to memory and processors). The include option (`-p` or `-id`) is not a list of partitions to optimize as it may appear. Instead it is a list of partitions that are optimized first, followed by any unlisted partitions and ignoring any explicitly excluded partitions. For example, if you have partition ids 1-5 and issue “`optmem -m myserver -o start -t affinity -xid 4 -id 2`”, the optimizer would first optimize partition 2, then from most important to least important partitions 1, 3 and 5. Since partition 4 was excluded, its memory and processors remain intact.

Some functions such as dynamic lpar and partition mobility cannot run concurrently with the optimizer. If one of these functions is attempted, you will receive an error message on the management console indicating the request did not complete.

Running the optimizer requires some unlicensed memory installed or available licensed memory. As more free memory (or unlicensed memory) is available, the optimization will complete faster. Also, the busier CPUs are on the system, the longer the optimization will take to complete as the background optimizer tries to minimize its effect on running LPARs. LPARs that are powered off can be optimized very quickly as the contents of their memory does not need to be maintained. The Dynamic Platform Optimizer will spend most of the time copying memory from the existing affinity domain to the new domain. When performing the optimization, the performance of the partition will be degraded and overall there will be a higher demand placed on the processors and memory bandwidth as data is being copied between domains. Because of this, it might be good to schedule DPO to be run at periods of lower activity on the server.

When the optimizer completes the optimization, the optimizer can notify the operating systems in the partitions that physical memory and processors configuration have been changed. Partitions running on AIX 7.2, AIX 7.1 TL2 (or later), AIX 6.1 TL8 (or later), VIOS 2.2.2.0 (or later) and IBM I 7.3, 7.2 and 7.1 PTF with MF56058 have support for this notification. For older operating system versions that do not support the notification, the dispatching, memory

management and tools that display the affinity can be incorrect. Also, for these partitions, even though the partition has better resource assignments the performance may be adversely affected because the operating system is making decisions on stale information. For these older versions, a reboot of the partition will refresh the affinity. Another option would be to use the exclude option on the `optmem` command to not change the affinity of partitions with older operating system levels. Rebooting partitions is usually less disruptive than the alternative of rebooting the entire server.

6.6 PARTITION MOBILITY (LPM) AND DPO CONSIDERATIONS FOR LINUX

The base Linux operating system evolved from a non-virtualized environment where there was a single VM per server. Because of this initial design, there are currently some design assumption that impact some of the features available on PowerVM with respect to affinity. When a partition migrates from one server to another or if DPO is executed, the underlying resources assigned to the partition can change. For example, the partition may have had processors and memory from 3 nodes and after LPM or DPO the processor and memory are only assigned to 2 nodes. Linux design assumptions do not allow the new topology information to be presented to Linux, so the partition runs with stale information with respect to the resources assigned to the partition.

The performance effect of this stale information could result in some cases sub-optimal performance since Linux is making decisions based on outdated information. There are many cases though where there will be little to no noticeable performance effects. Situations that should be fine would be if the partition is placed on the same or fewer nodes when performing an LPM or DPO operation. For example, if the hypervisor was able to contain the partition to a single processor chip or single drawer, the operation is performed and the resulting resource assignment is similar, there should be little to no performance change. If a partition was nicely contained to a single chip or drawer but due to the lack of free resources the resource assignment after LPM or DPO caused the partition to receive resources from other nodes, the performance could be affected.

The following are some suggestions to help understand information about the resource assignments and ways to mitigate the effects:

1. Even though the Linux `numactl -H` output does not reflect the current resource assignment, the HMC `lsmemopt` command does reflect the current resource assignment. If the score is high (score of 100 is perfect), this indicates the partition has good assignment of resources and performance should not be affected because of the stale information. If the score is low, performance could be affected.
2. When performing LPM, you may want to migrate the partitions with the most memory and processors or the most performance sensitive partitions before other partitions. If a server has lots of free resources, the PowerVM hypervisor can optimize the assignment of resources. As resources start becoming consumed, this tends to “fragment” the available resources which could lead to less than ideal resource assignments.
3. When performing DPO, you may want to prioritize the placement order using the `-id` or `-x` option. These options indicate to the PowerVM hypervisor that the listed partitions should receive resource assignments before other partitions.
4. The more installed resources (processors and memory) the more likely the PowerVM hypervisor can optimize the resource assignments. The resources do not have to be licensed; they just need to be installed. The hypervisor can optimize even if the resources are “dark” resources.

5. There may be a performance improvement even with less than ideal resources that could be obtained if Linux is able to work with accurate resource assignment information. A partition reboot after LPM or DPO would correct the stale resource assignment information allowing the operating system to optimize for the actual resource configuration.

6.7 AFFINITY GROUPS

The following HMC CLI command adds or removes a partition from an affinity group:

```
chsyscfg -r prof -m <system_name> -i name=<profile_name>  
lpar_name=<partition_name>,affinity_group_id=<group_id>
```

where `group_id` is a number between 1 and 255 (255 groups can be defined - `affinity_group_id=none` removes a partition from the group).

Affinity groups can be used to override the default partition ordering when assigning resources at server boot or for DPO. The technique would be to assign a resource group number to individual partitions. The higher the number the more important the partition and therefore the more likely the partition will have optimal resource assignment. For example, the most important partition would be assigned to group 255, second most important to group 254 and so on. When using this technique, it is important to assign only one partition to a resource group.

6.8 LPAR_PLACEMENT=2

The management console supports a profile attribute named `lpar_placement` that provides additional control over partition resource assignment. The value of '2' indicates that the partition memory and processors should be packed into the minimum number of domains. In most cases this is the default behavior of the hypervisor, but certain configurations may spread across multiple chips/drawers. For example, if the configuration is such that the memory would fit into a single chip/drawer/book but the processors don't fit into a single chip/drawer/book, the default behavior of the hypervisor is to spread both the memory and the cores across multiple chip/drawer/book. In the previous example, if `lpar_placement=2` is specified in the partition profile, the memory would be contained in a single chip/drawer/book but the processors would be spread into multiple chips/drawers/books. This configuration may provide better performance in a situation with shared processors where most of the time the full entitlement and virtual processors are not active. In this case the hypervisor would dispatch the virtual processor in the single chip/drawer/book where the memory resides and only if the CPU exceeds what is available in the domain would the virtual processors need to span domains. The `lpar_placement=2` option is available on all server models and applies to both shared and dedicated processor partitions. For AMS partitions, the attribute will pack the processors but

since the memory is supplied by the AMS pool, the `lpar_placement` attribute does not pack the AMS pool memory. Packing should not be used indiscriminately; the hypervisor may not be able to satisfy all `lpar_placement=2` requests. Also, for some workloads, `lpar_placement=2` can lead to degraded performance.

6.9 LPAR_PLACEMENT CONSIDERATIONS FOR FAILOVER/DISASTER RECOVERY

Many customers use a pair or cluster of systems for recovery purposes where in the case of failure of partition or server the work is offloaded to another server. The placement of these partitions upon physical boundaries (drawers/books) is something to consider when optimizing the performance in preparation for a failover situation.

The entitlement of the partitions needs to accurately reflect the CPU consumption under normal load (i.e. if the partition is consuming 4.5 processor units on average, partition should be configured with at least 5 VPs and 4.5 processor units and not 5 VPs with 0.5 processor units. Since the hypervisor uses the entitlement to determine how much CPU a partition will consume, the resource assignments of partitions will not be optimal if the entitlement is undersized. After ensuring the entitlement is correct, the next step is to place partitions into hardware domains with spare capacity such that when a failover occurs, there is unused capacity available in the domain.

As an example, let's assume two 2-drawer model with 16 cores per drawer. On server ONE is Partition A1 which is configured for 16 virtual processors (VPs) and 8.0 cores and Partition B1 also is configured for 16 VPs and 8.0 cores. Server TWO is similar configuration with partitions A2 and B2. Without any other directive, the hypervisor may place partitions A1 and B1 into the same drawer since the resources can be contained within a drawer. When running normally (non-failover), each server is only licensed for 16 of the 32 installed cores. In the event of a failure of server, the transactions that were being processed by A1 will failover to A2, same for B1 failing over the B2. Also, for failover, all cores in the server are activated such that the workload can be contained in a single server. What would be ideal for placement in the failover scenario would be that partition A2 is contained in a drawer with 16 CPUs and similarly B2 is contained in a different drawer.

If all cores were licensed, one way to achieve this placement would be to configure A2 and B2 with 16.0 entitlement even though only 8.0 entitlement is required in the normal situation. Since each drawer has 16 cores, the hypervisor would be forced to place these partitions on different drawers. Another way to achieve the desired placement is through the configuration of memory. For example, if each drawer has 256GB of memory, you could set each partition to 8.0 cores, 16 VPs and 230 GB. In this situation when the hypervisor tries to place the partitions, the memory requirements of A2 and B2 force the hypervisor to place the partitions in separate drawers. Allocation of memory beyond 50% of the capacity of the drawers would force this split.

Some users may consider using the Dynamic Platform Optimizer after a failover as a placement strategy but there is a performance cost involved in running the optimizer. The partitions, as the resources are moved between domains, run a bit slower and overall there is more CPU and memory bandwidth demand for the system. If there is sufficient unused capacity in the system or

the optimization can be delayed to a time when there is unused capacity, then the Dynamic Platform Optimizer may have a role to play in some failover scenarios.

6.10 SERVER EVACUATION USING PARTITION MOBILITY

There may be situations where there is a need to move multiple partitions from one server to another server or multiple servers using partition mobility, for example in the case of hardware maintenance or firmware upgrade. In such a scenario, the order in which the partitions are migrated will influence the resources assigned to the partitions on the target server. One option to handle this situation is to run the dynamic platform optimizer after all partitions are migrated to the target server. Another option would be to move the partitions in order from most important to least important. Moving partitions in this order allow the hypervisor to first optimize the allocation of the CPU and memory resources to the most important partitions.

6.11 POWERVM RESOURCE CONSUMPTION FOR CAPACITY PLANNING CONSIDERATIONS

PowerVM hypervisor consumes a portion of memory resources in the system; during planning stage take that into consideration to lay out LPARs. The amount of memory consumed by the hypervisor depends on factors such as the size of the hardware page tables for the partitions, the number of I/O adapters in the system, and memory mirroring. The IBM system planning tool (<http://www.ibm.com/systems/support/tools/systemplanningtool/>) should be used to estimate the amount of memory that will be reserved by the hypervisor.

6.12 LICENSING RESOURCES (COD)

Power systems support capacity on-demand where customers can license capacity on-demand as the business needs for compute capacity grows. In addition to future growth COD resources can be used to provide performance improvement.

Licensing for COD memory is managed as an overall total of the memory consumed and there is no specific licensing of individual logical memory blocks (LMBs) or DIMMs. If there is COD memory installed on the system, this can provide performance benefits because the hypervisor can use all the physical memory installed to improve partition resource assignments. For example, on a two-drawer system, each drawer has 256GB of installed memory but only 400GB of licensed memory, a 240GB partition can be created on one drawer and a 120GB partition on the second drawer. In this case, all 400GB is available and can be divided between the drawers in a manner that provides the best performance.

Licensing for COD processor is managed on an individual core basis. When there are fewer licensed cores than installed cores, the hypervisor identifies cores to unlicensed and puts these cores into a low power state to save energy. Note, when activating additional cores, the hypervisor may dynamically change which cores are licensed and unlicensed to improve partition resource assignment.

The dynamic platform optimizer can be utilized when configuration changes are made as a result of licensing additional cores and memory to optimize the system performance.

7 ENERGY SCALE™

There are four power management mode available on POWER9 processor-based servers. The power management mode controls the tradeoff choices between the energy efficiency and performance capabilities. It can be set from ASMI. The setting is dynamic so there is no need to reboot system or partitions. The power mode of partition is running can be checked by AIX command “lparstat -E 1 1”, if the mode is set to Dynamic or Maximum Performance mode, the reported mode can be changed based on the utilization. The processor frequency can be monitored by AIX command “lparstat -E “or Linux command of “ppc64_cpu -frequency”. For IBM i, the relative frequency can be displayed using WRKSYSACT. More details can be found at the following links:

[IBM EnergyScale for POWER9 Processor-Based Systems](#)

[POWER9 EnergyScale - Configuration & Management](#)

The four power management modes are:

Disable all modes

The processor clock frequency will be set to its fixed, nominal value.

This mode does not maximize the performance of the system.

Enable Static Power Saver mode

Enabling this feature reduces power consumption by lowering the processor clock frequency and voltage to fixed values. This reduces the power consumption of the system while delivering predictable performance.

This mode can be good if there is no critical workload which requires running at higher frequency.

Enable Dynamic Performance mode

Enabling this feature causes the processor frequency to vary based on workload and active core count. As the workload/active core count decreases, the processor uses less power, which enables the frequency to be increased above nominal. During periods of very low utilization, the processor frequency will be reduced in order to save energy. This mode provides consistent performance across all environmental operating conditions.

If the system is not fully busy or some cores are not being used, this mode can help maximize the busy cores performance, while conserving energy when the system is idle. It is default mode for the Power S914 system.

Enable Maximum Performance mode

Enabling this feature causes the processor frequency to vary based on workload and active core count. As the workload/active core count decreases, the processor uses less power, which enables the frequency to be increased above nominal. In this mode, the allowed socket power is increased to the maximum value, which results in top performance along with increased fan noise and higher power consumption. In more stressful environmental conditions, performance may vary.

This mode is the default for most POWER9 processor-based systems; it delivers maximum possible system performance, but it increases power draw.

8 PROCESSOR COMPATIBILITY MODE

There are five processor compatibility modes available on POWER9 processor-based system with firmware level 940. Those different compatibility modes help continue successful logical partitions operation with different operating system version or application level support. It enables the migration of the logical partitions from servers with different processor types. The different mode choice can be done on HMC through logical partition profile in the processor section. It requires power cycle the logical partition to change the processor compatibility mode.

Check out the detail description of each mode by visiting

[Processor compatibility mode definitions](#)

9 CONCLUSION

This document has described virtualization best practices for Power systems in order to configure your system and partition to help improve performance.

Additional references are included in the next section. If you need additional help in assessing the potential impact of implementing LPARs, testing virtualization changes, or identifying ways to improve the performance of LPARs in your environment, please contact IBM Lab Services at ibmsls@us.ibm.com.

10 REFERENCES

The following is a list of IBM documents that are good references:

- AIX on POWER – Performance FAQ
https://www14.software.ibm.com/webapp/set2/sas/f/best/aix_perf_FAQ.pdf
- IBM i on POWER – Performance FAQ
<https://www.ibm.com/downloads/cas/QWXA9XKN>
- Service and Support Best Practices
<https://www14.software.ibm.com/webapp/set2/sas/f/best/home.html>

IBM Power Virtualization Best Practices Guide

- IBM System p Advanced POWER Virtualization Best Practices Redbook:
<http://www.redbooks.ibm.com/redpapers/pdfs/redp4194.pdf>
- Power Developer Portal
<https://developer.ibm.com/linuxonpower/perfcol/>
- System Planning Tool
<http://www.ibm.com/systems/support/tools/systemplanningtool/>
- An LPAR Review:
<http://www.ibmssystemsmag.com/aix/administrator/lpar/An-LPAR-Review/>
- IBM EnergyScale for POWER9Processor-Based Systems
<https://www.ibm.com/downloads/cas/6GZMODN3>
- Power EnergyScale – Configuration & Management
[POWER9 EnergyScale - Configuration & Management](#)
- Processor compatibility mode definitions
https://www.ibm.com/support/knowledgecenter/9009-22A/p9hc3/p9hc3_pcmdefs.htm

IBM Power Virtualization Best Practices Guide



The Power Architecture and Power.org wordmarks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© IBM Corporation 2020
IBM Corporation
Systems and Technology Group
Route 100
Somers, New York 10589

Produced in the United States of America May 2020
All Rights Reserved
This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries.
The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features and services available in your area. All statements regarding IBM future directions and intent are subject to change or withdrawal without notice and represent goals and objectives only.
IBM, the IBM logo, ibm.com, AIX, EnergyScale, Power Systems, POWER, PowerVM, POWER8, and POWER9 are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml
Other company, product, and service names may be trademarks or service marks of others.
IBM hardware products are manufactured from new parts, or new and used parts. In some cases, the hardware product may not be new and may have been previously installed. Regardless, our warranty terms apply.
Photographs show engineering and design models. Changes may be incorporated in production models. Copying or downloading the images contained in this document is expressly prohibited without the written consent of IBM.
This equipment is subject to FCC rules. It will comply with the appropriate FCC rules before final delivery to the buyer. Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with those suppliers.
All performance information was determined in a controlled environment. Actual results may vary. Performance information is provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Buyers should consult other sources of information, including system benchmarks, to evaluate the performance of a system they are considering buying.
When referring to storage capacity, 1 TB equals total GB divided by 1000; accessible capacity may be less.
The IBM home page on the Internet can be found at: <http://www.ibm.com>.
A full list of U.S. trademarks owned by IBM may be found at: <http://www.ibm.com/legal/copytrade.shtml>.
The IBM Power Systems home page on the Internet can be found at: <http://www.ibm.com/systems/power/>

39031139USEN-01