AIX Micro-Partitioning (part 1)

Mark Cohen Austrowiek - EPV Technologies

1 Introduction

Starting from year 2000 IBM announced the possibility to partition the pSeries family systems.
Initially it was only possible to run dedicated processors in a logical partition. Now p5 series has integrated new virtual engine system technologies into their hardware and software. This introduced new features as Micro-Partitioning which provides the ability to share physical processors among logical partitions, Virtual Lan which provides network utilization features capabilities that permit you to prioritize traffic on shared networks and allows secure communication between logical partitions without the need of a physical network adapter, Virtual I/O which provides the ability to dedicate I/O adapters and devices to a virtual server. It allows a single physical I/O adapter to be used by multiple logical partitions on the same server. This allows consolidation of I/O servers and minimizes the amount of I/O adapters required.
New concepts of shared partitioning have been implemented in the p5 series compared to the IBM mainframe. In this paper I will provide a detailed overview of shared partitioning and its performance metric considerations .
Before starting to measure shared partition activity it's essential for the performance analyst to obtain a general overview of the new terms introduced and understand how logical partitioning is configured and works in the P5 series environment.
The following arguments will be discussed in this paper: Power Hypervisor, Hardware Manager Console (abbreviated as HMC), partition profiles, Micro-Partitioning, entitled capacity, virtual CPUs, logical CPUs, Symmetric Multi Processing (abbreviated as SMT), Partition Load Manager (abbreviated as PLM), performance tools as *vmstat, lparstat, mpstat* and performance case examples.

Due to the complexity of the topic the paper has been divided in two parts.

2 Concepts

2.1 Power Hypervisor

The Power Hypervisor (Hypervisor in the following) is a component of system firmware that permits partitioning and dynamic resource movement across multiple operating systems. It performs the time slicing of the physical processors between the logical partitions (LPARs). Equivalent to the IBM mainframe PR/SM software, it acts as a hidden partition with no processors assigned to it.
The software lays between the system hardware and LPARs. The communication between the LPAR and the Hypervisor is done on behalf of functions generated by the operating system kernel running in the partition. The Hypervisor is scheduled to generate an interrupt every 10ms as a timing mechanism for controlling the dispatch of physical processors between LPARs. Each LPAR is guaranteed to get its share of processor cycles during each 10 ms dispatch window. The operating system kernel software generates hypervisor calls that permit LPAR integrity.

The number and type of calls can be analysed by the *lparstat* utility.

2.2 HMC - Hardware Management Console

The HMC is a dedicated Linux black-box computer for configuring and managing LPARs on a system. As up to date the HMC can manage up to 254 LPARs. It provides a set of functions to manage the LPARs, perform capacity on demand functions, create LPAR and system profiles, display system and LPAR status, act as a virtual console for each LPAR and provide cluster support. HMC also supports dynamic partitioning (the ability to add or remove resources while the LPAR is running).

2.3 Virtual CPUs

A virtual CPU is the operative system view of a physical processor.
Every LPAR is assigned a minimum, desired and maximum virtual CPUs. The more virtual CPUs defined the more parallel work can be executed. The total assigned capacity divided by the virtual CPUs provides the amount of capacity each virtual CPU can execute each time it's dispatched. A maximum number of 10 virtual cpus can be allocated for each physical processor. The Hypervisor firmware schedules the virtual CPUs on the physical processors.

2.4 Partition profiles

Partition profiles are created by the HMC. Each LPAR can have one or more partition profiles. Depending on the type of workload you can define a profile that assigns the resources which with the LPAR should boot. In order to refresh the profile the operating system needs to be shutdown and the LPAR deactivated.
In a partition profile you assign all the resources as virtual CPUs, memory, physical devices and virtual devices. When activating the LPAR you need to select the partition file otherwise it will use the default profile.
It's also possible to create a system profile which can manage more then one partition profile.
If you use a system profile all the partition profiles are activated in the order specified by the system profile.

At the moment mixing shared and dedicated processors in the same LPAR is not possible but both types of partitions can exist on the same system: one with dedicated CPUs or one with virtualized processors that share a pool of physical processors (see the Micro-Partitioning chapter). Dedicated CPUs go in the shared pool if their partitions are not activated[1].

The maximum number of virtual processors supported by a LPAR is 64.

2.5 Micro-Partitioning

Micro-Partitioning provides the ability to share physical processors among LPARs.
This feature allows LPARs to allocate fractions of processors. The minimum amount of processor units that can be assigned to a LPAR is 1/10 of a physical processor. In the partition profile it is represented as 0.10.

[1] It's possible to set an option in the HMC to avoid this behaviour.

When creating a partition file you establish the amount of maximum, desired and minimum processing capacity for each LPAR. When the LPAR boots it tries to obtain the desired amount; if this amount is not available it will try to obtain the maximum amount it can get from the minimum and desired amount defined. This amount of capacity is called "entitled capacity".

If the sum of the entitled capacity from the active LPARs and the one required to start the next LPAR over-commits the number of physical processors the LPAR will not boot[2].

The entitled capacity is always available for the LPAR if it needs it irrespective to the load of the system. The Hypervisor does not change the entitlement of a LPAR, it just controls the number of cycles a LPAR can use. However the entitled capacity can be changed by dynamic logical partition operations manually or by the PLM. The minimum and maximum capacity entitled values set in the partition profile set the limits for dynamic partition operations.

Unallocated CPU cycles are those cycles not included in the entitled capacity.
Furthermore a LPAR can use less then its entitled capacity and donate the **unused CPU** cycles to the Hypervisor. The sum of unallocated and unused capacity is the **available** capacity.

A LPAR can be "capped" or "uncapped".
A capped LPAR can only consume up to its entitled capacity.
You have to assign each uncapped LPAR a weight value which can range from 1 to 255[3].

Micro-Partitioning provides additional CPU capacity taken from the available capacity in the shared pool to requesting uncapped LPARs based on their weight value.
An uncapped LPAR is only limited from the amount of virtual processors assigned to the LPAR and from the available capacity.
The additional capacity used above the entitled is known as the **excess capacity**.

When more uncapped LPARs need additional capacity at the same time the Hypervisor will distribute available CPU cycles based on the LPAR priority.
The LPAR priority is calculated by dividing the current excess capacity (used capacity – entitled capacity) by the weight. A lower value indicates an higher priority.
An example of LPAR priorities calculation is showed in Table 1.

	Used capacity	Entitled capacity	Excess capacity	Weight	Lpar priority
Lpar1	0.8	0.5	0.3	100	0.003
Lpar2	0.7	0.5	0.2	200	0.001 (higher priority)

Table 1

2.6 SMT – Symmetric Multi Processing

Symmetric Multi Processing allows applications to increase there overall resource utilization by "virtualizing" virtual CPUs through the use of multi threading. Two separate instruction streams can run concurrently on the same virtual CPU. Each thread is treated as a logical processor.
This feature can be enabled or disabled dynamically or after reboot by means of the *smtctl* command or from the *smit* panel. The *smtcnt* command shows the threads allocated to the virtual

[2] The entitled capacity is guaranteed to the logical partition and can not be used to start another partition.
[3] A capped logical partition has a weight value equal to 0.

processors. A LPAR can be booted in single thread mode (ST) or multi thread mode (SMT) which is the default. The *mpstat* reports the usage of the virtual and its associated logical CPUs. The *mpstat* utility also shows the partitions entitled capacity. The entitled capacity is broken down into the virtual CPUs and then logical CPUs. Setting the *smt_nooze_delay* to 0 by the *schedo* command maximizes the speed of work of the processors because it gives the processor capacity back to the hardware when the thread is in an idle loop. The evaluated performance increase is 30%.

This feature supports dedicated and shared CPUs and provides advantages to application where the number of transactions outweighs the importance of speed.
Not in all cases the performance is better. In some cases it would be better to disable the SMT feature and provide dedicated processors where workloads can not tolerate the variability due to resource sharing.

2.7 Logical CPUs

When SMT is enabled each virtual processor is assigned two threads that are seen by the operating system as two logical CPUs. This means the number of logical CPUs seen by the operating system is always double the number of virtual CPUs when SMT is enabled.
The first output record created by the standard native utilities as *vmstat* and *mpstat* always shows the number of logical CPUs. It is presented by the *lcpu* field.

2.8 DLPAR – dynamic logical partitioning

Dynamic partitioning allows resources to be moved from different LPARs.
For dedicated processors it's only possible to move and take whole processors. For shared LPARs it's possible to change the entitled capacity, the weight of the partition, the mode (capped or uncapped), the number of virtual processors.
The highest possible values for entitled capacity and number of virtual CPUs are dependant on the maximum values defined in the partition profile.

To verify the changes the *lparstat -i* command can be used.

2.9 PLM - Partition Load Manager

Partition Load Manager is part of the advanced virtualization feature; it helps maximize the resource utilization for LPARs exploiting dynamic logical partition capabilities by making automatic adjustment.
Based on defined policies, resources as memory and CPU are moved to LPARs with higher demands from LPARs with low demands.

Partition Load Manager has to be installed on a LPAR but that LPAR can be shared with other workloads. It uses resource monitoring to control and communicate with the managed partitions. To communicate with the HMC station, a SSH (secured socket) connection needs to be configured.

The policy defines the minimum and maximum resource thresholds[4] and other specific parameters for each LPAR.

[4] This values have to be inside the HMC minimum and maximum values.

LPARs that have reached an upper threshold become resource requesters and LPARs that have reached minimum thresholds become resource donors.
Partition Load Manager uses the load average values to decide when to add or remove resources.

However the resource increments and decrement changes[5] can not go out of the minimum and maximum values defined in the HMC partition policy.

The LPAR that needs the resource and still didn't arrive at its maximum resource defined can get the resource from one of the following areas:
1) a pool of free unallocated resources
2) a resource donor
3) a lower priority LPAR

If a request can not be honoured it is queued until it can find resources.

When PLM needs to distribute the resources between more LPARs requesting them at the same time, it uses the partitions shares (a PLM parameter similar to the HMC weight) and the guaranteed entitlement (another PLM parameter by default set equal to HMC desired entitlement) to determine the LPAR priority.

The calculation is similar to the one showed in Table 1 but in this case the LPAR priority is calculated dividing the current excess entitlement (used capacity – guaranteed capacity) by the shares. Also in this case a lower value indicates an higher priority.

It's important to note that Hypervisor uses the weights to distribute the excess CPUs in addition to each LPAR entitled CPUs while PLM uses the shares to increase or decrease the LPAR entitled CPUs.
A consequence of this different behaviour is that the distribution of resources at peak times can be different when using Hypervisor or PLM.

Here is an example provided from one of the IBM's redbook .

On a system with 6 physical processors we have the following configuration:

Lparname	Entitled capacity	weight	shares
Lpar1	1	20	20
Lpar2	2	10	10

Table 2

We have 3 excess CPUs that can be used (6 physical - 3 entitled = 3 excess).

When PLM is not activated the number of excess CPUs to be assigned to each LPAR is calculated dividing the LPAR weight by the total sum of the weights and multiplying the result by the number of excess CPUs.
The total number of CPUs each partition can get is:

- Lpar1 - **1** entitled CPU + **2** excess CPUs (20/30*3)
- Lpar2 - **2** entitled CPUs + **1** excess CPU (10/30*3)

[5] The capacity moved is a fixed percentage defined as the delta value in the PLM policy.

When PLM is activated the number of entitled CPUs to be assigned to each LPAR is calculated dividing the LPAR shares by the total sum of the shares and multiplying the result by the number of total CPUs.
In this case the total number of CPUs each partition can get is:

- Lpar1 - 4 entitled CPUs (20/30*6)
- Lpar2 - 2 entitled CPUs (10/30*6)

If running workloads are very variable it could be necessary to define LPARs with many virtual processors (much more than the LPARs normally need). This of course can cause a lot of overhead.

One of the main advantages of PLM is the possibility to define a small amount of virtual processors and let PLM manage the virtual CPUs towards workload needs. This way there is a higher utilization for the virtual CPUs and lower overhead.