The many effects of the LPAR weight

Fabio Massimo Ottaviani – EPV Technologies

October 2016

1 Introduction

For many years z/OS customers have been used to setting a weight for each logical partition (LPAR) just to tell PR/SM how to manage shared CPs when they are in contention.
The assigned weight expresses the importance of LPARs workloads to be considered in the distribution of the CPU and zIIP capacity.

Generally speaking weight is enforced by PR/SM only when there is not enough power to satisfy all LPARs' demands. As long as the physical CPs have some spare power, all LPARs can use whatever they want. However, if initial capping is set for an LPAR its weight becomes an insurmountable limit to the usable CP capacity. No matter how much capacity is available and not being used by the other LPARs.

A not well known effect of the weight is that, if it's not coherent with the number of logical CPs assigned to the LPAR, it may negatively influence PR/SM dispatching and application performance leading to the so-called "short CP effect".

More recently, with the advent of HiperDispatch, the LPAR weight has also become a key element in the "vertical polarization" of the logical CPs which has again a very strong effect on workload performance.

Finally, LPAR weight is also used when group capacity is used to determine the LPAR minimum entitlement, which is the amount of CPU it can use in the 4-hour rolling average without being soft-capped when the group capacity limit is reached.

In this paper we will discuss the many effects of LPAR weight providing examples and suggestions.

2 Partition weight and CP contention

As mentioned, the first goal of LPAR weight is to tell PR/SM how to manage shared CPs when all the resources in a CP pool (CPU, zIIP) are saturated and LPARs are contending for them. Different weights can (and normally have to) be specified for each CP pool.

Weight is set as an absolute number but it's internally converted to a percentage which expresses the relative importance of LPARs workloads in the distribution of CPU and zIIP capacity.

In the table in Figure 1 you will find an example of very simple weight definition. Values in the Weight column are set by customers while PR/SM uses the values in %Weight.

LPAR	Weight	%Weight
LPAA	990	60%
LPAB	495	30%
LPAC	165	10%
	1650	100%

Figure 1

To make life easier and have an immediate understanding of the %Weight values, it's strongly suggested the sum of the weights to be equal to 1000[1].

The values in Figure 2 provides exactly the same indications to PR/SM than the ones in Figure 1 but they allow you to immediately understand what is the percentage of the CP pool capacity you assigned to each LPAR.

LPAR	Weight	%Weight
LPAA	600	60%
LPAB	300	30%
LPAC	100	10%
	1000	100%

Figure 2

If initial capping is not active (see next chapter), weight is enforced by PR/SM only when there is not enough power to satisfy all LPARs' demands. As long as the physical CPs have some spare power, all LPARs can use whatever they want.

[1] You could also use 100 but 1000 provides more granularity which can be needed in complex configurations with many small partitions.

An important check you have to do is comparing the target CPs, each LPAR can use in contention, with the number of logical CPs assigned to it.

The example in Figure 3 shows the CEC usage view[2] for the LPAA LPAR. It can theoretically use up to 60% of the CPs when in contention as shown in the % TARGET row.
However, its maximum usage of the CEC CPs never trespasses 38,5% in the peak hours (see values in red).

METRIC	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
% USED	10,2	13,4	11,6	7,6	9,2	10,4	9,5	14,7	28,9	38,5	38,5	38,3	38,4	28,6	19,5	21,5	22,9	17,7	13,6	23,7	23,1	17,7	15,6	17,5
% TARGET	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0	60,0
% LIMIT	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5	38,5

Figure 3

The reason is the number of logical CPs assigned to this LPAR. The last row of this simple but important view says that the maximum CEC capacity, which can be used by LPAA even when there is a lot of capacity unused by the other partitions, is 38,5%.

CEC	LPAR	Weight	%Weight	LCP	PCP
CEC1	LPAA	600	60%	5	
CEC1	LPAB	300	30%	5	
CEC1	LPAC	100	10%	5	
	TOTAL	1000	100%	15	13

Figure 4

As reported in Figure 4 the number of logical CPs assigned to LPAA is 5 which is only 38,5% of the physical CPs available in the CEC (see PCP). So LPAA can't use more than that.

[2] From EPV for z/OS Workloads vision.

3 Initial capping

Initial capping is the oldest available capping technique. It exploits the PR/SM capping function and it can be used to limit the capacity which can be used by an LPAR on one or more CP pools independently.

Initial capping is based on LPAR (relative) weight.
A capped LPAR running at its cap does not have access to the resources that are not utilized by other LPARs while resources that are not used by a capped LPAR can be used by other LPARs.

Initial capping can be used to limit:

- effect of loops in development LPARs;
- capacity used during stress tests;
- capacity specified in an outsourcing service contract;
- etc.

In the example below LPAA workloads demand is using only 50% (instead of 60% as guaranteed by its weight) so LPAC can use this free capacity exceeding the 10% it should use in contention.

LPAR	Active	Weight	%Weight	%Demand	CAP	%Used
LPAA	Yes	600	60%	50%	No	50%
LPAB	Yes	300	30%	30%	No	30%
LPAC	Yes	100	10%	20%	No	20%
	TOTAL	1000	100%	100%		100%

Figure 5

If initial capping would be activated for LPAC the results would be what is shown in Figure 6.

LPAR	Active	Weight	%Weight	%Demand	CAP	%Used
LPAA	Yes	600	60%	50%	No	50%
LPAB	Yes	300	30%	30%	No	30%
LPAC	Yes	100	10%	20%	YES	10%
	TOTAL	1000	100%	100%		90%

Figure 6

The major issue of initial capping is that, being based on relative weight, it's sensitive to LPAR activation/deactivation.

As you can see in the example below if LPAB will be deactivated LPAC will be capped at 14% of the CEC capacity instead of 10% as desired.

LPAR	Active	Weight	%Weight	%Demand	CAP	%Used
LPAA	Yes	600	86%	50%	No	50%
LPAB	NO					
LPAC	Yes	100	14%	20%	YES	14%
	TOTAL	700	100%	70%		64%

Figure 7

To overcome this kind of issues IBM released the Absolute Capping and, more recently, the Absolute Group Capping functionalities which allow you to cap an LPAR at a fixed amount of capacity independently from the LPAR weight.
Unlike initial capping these new functionalities can also be used together with Defined and Group Capacity.

4 Short CP effect

In order for an LPAR to consume additional capacity that's not used by other LPARs, it needs to have sufficient logical processors. This means that its number of logical CPs assigned should be sometimes higher what would be needed to consume only the guaranteed share.

Over-configuring the logical processors decreases the percentage of time each of them will actually be running. This is absolutely true before HiperDispatch when PR/SM traditionally distributed the capacity share of an LPAR equally across all of its online logical processors.

CEC	LPAR	Weight	%Weight	LCP	Target CPs	% of 1 CP	PCP
CEC1	LPAA	600	60%	5	7,8	156%	
CEC1	LPAB	300	30%	5	3,9	78%	
CEC1	LPAC	100	10%	5	1,3	26%	
	TOTAL	1000	100%	15			13

Figure 8

In the above example LPAC relative weight is 10% so it will give this LPAR a guaranteed number of physical CPs equal to 1,3 (Target CPs); because LPAC has 5 logical CPs each of them will be dispatched 26% of the time (% of 1 CP).

The resulting percentage of 1 physical CP per logical CP can be calculated with the following simple formula:

$$\% \text{ of 1 CP} = \% \text{ Weight} * PCP / LCP$$

In this case after having run for its time slice, a logical processor must wait its turn for 74% of the time. From the perspective of a workload running on that logical processor, unaware of PR/SM activity, it would be similar to using less powerful (short) CPs.

The short CP effect can lead to poor response time especially for CPU intensive workloads that can be stranded on a logical processor that won't run again for a long time.
It can also cause a waste of cycles on each running processor spinning for system locks held by the not running processors.

HiperDispatch can manage the number of online logical processors to mitigate the short CP effect.
However LPARs can be still exposed to the "vertical" short CP effect.
We will discuss that in the next chapter.

5 HiperDispatch vertical polarization

In the last few years, the power and number of the physical processors available on mainframe hardware has greatly increased, allowing the concentration of a much higher number of LPARs than before on a single machine.
A side effect of this high number of LPARs is an increase of the number of defined logical processors compared to the number of physical CPs.
These factors tend to reduce the probability for a logical processor to be re-dispatched to the same physical processor and therefore re-use instructions and data previously loaded in the Level 1 cache. A L1 cache miss will cause data and instructions to be loaded from the other cache levels.
Performance degradation and CP overhead occurs when this happens because of the access to the higher cache levels. The more distant the cache level the higher performance degradation and CP overhead.

HiperDispatch has been designed to work both with PR/SM and z/OS to solve this issue and also to maximize the probability for a workload to be re-dispatched to the same logical processor or group of processors.
HiperDispatch functionalities are many and complex. In this paper, we will only discuss the effect of the LPAR weight on HiperDispatch vertical polarization[3].

Vertical polarization splits the LPAR logical processors in three pools:

- High polarity (high processor share); they will have a target share corresponding to 100% of a physical processor which will be pseudo-dedicated to each of them;
- Medium polarity (medium processor share); they will normally have a target share greater than 0% and less than 100% of a physical processor; these medium logical processors have the remainder of the LPAR's shares after the allocation of the logical processors with the high share; they will compete for the physical processors as before HiperDispatch;
- Low polarity (low processor share); they will receive a target share corresponding to 0% of a physical processor; these are considered discretionary logical processors which are not needed to allow the LPAR to fully utilize the physical processor resource associated with its weight; if there is not unused capacity left from other LPARs in the CEC they will be parked and not used.

Based on the LPAR weight and the number of shared physical processors in the CEC the number of LPAR target CPs has to be calculated as described in the previous chapter.

CEC	LPAR	Weight	%Weight	LCP	Target CPs	PCP
CEC1	LPAA	600	60%	5	7,8	
CEC1	LPAB	300	30%	5	3,9	
CEC1	LPAC	100	10%	5	1,3	
	TOTAL	1000	100%	15		13

Figure 9

[3] A complete discussion of HiperDispatch is beyond the scope of this paper.

Starting from the number of LPAR target CPs, in the hh.mm format, the general rule is that:

- hh is the number of high share LPs;
- mm is the fraction of a CP which has to be used by medium share LPs.

In the example in Figure 9, LPAB has 3.9 target CPs that will become 3 high polarity with 100% share and 1 medium polarity with 90% share. Because LPAB has 5 logical processors there is a logical processor left which will become a low polarity processor with an initial 0% weight.
There are some exceptions to the general rule; in the example in Figure 10 we find two of them[4]:

- if hh is greater than LCP (LPAR logical processors) then hh=LCP; no medium and low polarity processors; LPAA has 7.8 target CPs but only 5 LCPs; so it will get only 5 high polarity processors.
- if mm is lower than 0,50 then hh=hh-1 and mm=1+mm; LPAC has 1.3 target CPs which will become 2 medium polarity processors with 65% share each. Because LPAC has 5 logical processors there are 3 logical processors left which will become 3 low polarity processors with an initial 0% weight.

CEC	LPAR	Weight	%Weight	LCP	Target CPs	High	Med	Low	PCP
CEC1	LPAA	600	60%	5	7,8	5	0	0	
CEC1	LPAB	300	30%	5	3,9	3	1	1	
CEC1	LPAC	100	10%	5	1,3	0	2	3	
	TOTAL	1000	100%	15					13

Figure 10

LPAA weight and number of LCPs are incoherent; they will provide LPAA workloads high performance because all the LCPs will run on a pseudo-dedicated processor but no scalability because, even if all the other LPARs in the CEC will not use any CPU cycle, LPAA cannot take advantage of the free capacity.

The problem is different for LPAC; if there will be free CEC capacity, HiperDispatch will unpark the low polarity processors; because the initial weight of these processors is 0, the weight of the medium polarity pool will be shared; each of the 5 logical processors will get 26% weight.
So even with HiperDispatch, LPAC is still exposed to the short CP effect, that in this case it's called "vertical" short CP effect.
The bottom line is that the LPAR weight and the number of LCPs assigned to it must be carefully set taking into account their effect on HiperDispatch polarization.

The following "best practices" should be used:

a) Important LPARs should get as much high polarity processors as possible;
b) Workloads should run mostly on vertical high and medium polarity processors;
c) Low polarity processors should be used only for occasional workload spikes
d) The number of low polarity processors should be limited to the ones really needed to reduce the risk of the vertical short CP effect.

[4] More exceptions exists; IBM provides the LPAR Design free tool taking into account all of them; you can download it at http://www-03.ibm.com/systems/z/os/zos/features/wlm/WLM_Further_Info_Tools.html#Design

6 Group capacity entitlement

Group capacity is an extension of the defined capacity concept that allows capacity groups to be created by putting together a set of LPARs running on the same CEC and assigning a group limit to the amount of MSUs the group of LPARs can use in the 4-hour rolling average.

WLM uses the weight definitions to set the MSU entitlement value of each LPAR:

- Minimum Entitlement; it is the guaranteed MSU share the LPAR can get when all the LPARs in the CEC want to get their share;
- Maximum Entitlement; it is the maximum MSU share the LPAR can get when there are MSUs not used by the other LPARs in the CEC.

Minimum entitlement is calculated as the minimum value between the LPAR defined capacity limit (if set) and the value resulting multiplying the LPAR relative weight (%Weight) by the group capacity limit.
Maximum entitlement, is independent from the LPAR weight; it is calculated as the minimum value between the LPAR defined capacity limit (if set) and the group capacity limit.

In the table below, we show the minimum and maximum entitlement of the three LPARs of our example in the hypothesis that:

- no defined capacity limits have been set for the LPARs;
- CEC capacity is 2011 MSUs (2964-713);
- all the LPARs belong to the same group (GROUP1);
- the group capacity limit is 1500 MSUs.

CEC	LPAR	Weight	%Weight	Group	Group Capacity Limit	Min Entitlement	Max Entitlement
CEC1	LPAA	600	60%	GROUP1		900	1500
CEC1	LPAB	300	30%	GROUP1		450	1500
CEC1	LPAC	100	10%	GROUP1		150	1500
	TOTAL	1000	100%		1500	1500	

Figure 11

It’s interesting to note that minimum and maximum entitlement don’t take into consideration the number of logical processors assigned to each LPAR.

Because of the number of logical processors assigned, LPAA will never be able to use its theoretical minimum and maximum entitlement.
LPAB and LPAC can use their minimum entitlement but not their maximum entitlement.

7 Defined capacity

When a defined capacity limit is set for an LPAR, different techniques are used to cap it (soft capping) when the limit is reached.

WLM uses the standard PR/SM initial capping mechanism, so the capping technique depends on the relation between the limit and the MSU the LPAR can use based on his weight (MSU AT WEIGHT) calculated as:

MSU AT WEIGHT = %Weight * CEC MSU

If the value of MSU AT WEIGHT is equal to the defined capacity limit the standard initial capping technique is used.

If the value of MSU AT WEIGHT is lower than the defined capacity limit, a negative phantom weight, to increase the MSU AT WEIGHT value and match the defined capacity limit, is internally created and used[5];

If the value of MSU AT WEIGHT is greater than the defined capacity limit, a phantom weight, to reduce the MSU AT WEIGHT value and match the defined capacity limit, is internally created and used.

In the last case, the result will also be a change of the logical processors vertical polarization.

An example is reported in the next figure.

CEC	LPAR	Weight	%Weight	LCP	Target CPs	MSU AT WEIGHT	DEF MSU	High	Med	Low
CEC1	LPAB	300	30%	5	3,9	603,3		3	1	1
CEC1	LPAB	300	20%	5	2,6	400	400	2	1	2

Figure 12

If a defined capacity limit of 400 MSU is set for LPAB when this limit is reached the MSU at weight are reduced to allow WLM to soft cap the LPAR to the same value by using a phantom weight.
This will also reduce the %Weight, re-calculated dividing the MSU AT WEIGHT value by the CEC MSUs (2011 MSUs in our example). Based on that, the Target CPs will be reduced to 2,6 reducing the high polarity logical processors from 3 to 2 and increasing the number of low polarity processors from 1 to 2.

8 Conclusions

LPAR weight is not just used to tell PR/SM how to manage shared CPs when they are in contention anymore. It has pervasive effects on many new extremely important z/OS functions.
All these effects, described in this paper, have to be deeply understood in order to avoid unexpected and undesired results.

[5] Before z/OS 2.1 and zEC12 GA2, WLM defines a cap pattern that repeatedly applies and removes the cap at LPAR weight.