close window

e-Newsletter Exclusive

Print

Entitlements and VPs—Why You Should Care

May 2013 | by Jaqui Lynch

With the POWER7 processor, many changes were integrated into the chip that must be taken into account when defining LPARs. First, understand that POWER7 technology and simultaneous multithreading (SMT4) are all about throughput, and rPerf is a throughput-based performance measurement. Therefore, to take advantage of the server's full rPerf it's important to drive all of the threads. However, the way the threads get kicked off has changed with POWER7.

SMT and Threading

Threads are dispatched via a virtual processor (VP). When a VP is dispatched to a physical core, all four of its associated threads are dispatched with it. In POWER5 and POWER6, the primary and secondary threads are loaded to 80 percent before another VP is unfolded. With POWER7 and POWER7+, the primary threads get loaded to 50 percent, before a VP is unfolded. Once the primary threads are spread across the VPs, then secondary threads are used. When those secondary threads are loaded to 50 percent, the remaining two tertiary threads are loaded. This means VPs are unfolded more often in POWER7/POWER7+ than in POWER5 or POWER6, and it gives the appearance that more physical cores are being used as the workload is spread out. However, you'll see more idle time because the secondary and tertiary threads might not be in use. If the VPs were reduced, it would force the VPs to load the secondary and tertiary threads and allow you to attain full rPerf. So think of POWER5/6 as stack then spread and POWER7/POWER7+ as spread then stack.

POWER7 servers run in “raw throughput mode,” which is intended to provide the highest per thread throughput and the best response time, but it's at the expense of activating or using more physical cores. With AIX V6.1 TL08 and AIX V7.1 TL02, it's possible to switch the LPAR such that it behaves more like POWER5/POWER6, loading the secondary threads before unfolding VPs. This is called “scaled throughput mode,” which provides the highest system-wide throughput per VP, because it ensures that the secondary and tertiary threads actually get used. Scaled throughput is a dynamic tunable that can be enabled as follows:

```
schedo –p –o vpm_throughput_mode=

0       Legacy Raw mode (default)
1       Enhanced Raw mode with a higher threshold than legacy
2       Scaled mode, using primary and secondary SMT threads
4       Scaled mode, using all four SMT threads
```

VPs and Memory

However, you might avoid needing to change from raw throughput to scaled throughput by paying attention to VP entitlement ratios. This requires understanding how dispatching works. In the shared pool, a VP gets dispatched to a core. The first time this happens that core becomes the home node for that VP.

The VP gets assigned a dispatch window equal to its entitlement/VPs—i.e., if entitlement is 0.2 and there are 2 VPs then each is assigned 1 millisecond (ms) to run. During that 1ms, that VP's four threads do as much work as they can. At the end of the entitlement or if the VP cedes the core (could be an I/O or just runs out of work to do), the system dispatches the next priority VP that has work to run. The old VP gets context switched and put back on the home node run queue if it still has work to do. If at all possible, the dispatcher will return that VP to the same core (home node) to ensure it can use the data it already has in the cache for the core and in the memory DIMMs attached to the core. Otherwise, it will go on the global run queue and get dispatched to a different core. That core could be on the same chip (local), the same book (near) or a different node (far). Two areas of concern, however, involve:

1. VP-to-entitlement ratios
2. Memory bus

It's possible in POWER7 to go down to 1/10 of a core (1ms) for an LPAR and in POWER7+ to 1/20 of a core (0.05ms) for an LPAR. Accordingly, in a 10ms dispatch window on a very busy system with lots of LPARs, a different LPAR could get dispatched every 0.5 or 1ms. This can lead to LPAR shredding and slower execution because of context switches and continually warming up the cache.

Take a workload with three LPARs, each with an entitlement of 0.1 and 1 VP. To keep the example simple, we'll work with one core for all three LPARs. Assuming each workload really needs 3ms to run, we need 9ms out of the 10ms dispatch window to complete the work. The first LPAR will be dispatched and run for 1ms loading its data into the DIMMs and then into the cache for the core. It then does an involuntary context switch (ICS) and the core gets assigned to the next LPAR. After 1ms more, the third LPAR gets its 1ms. Each time a new LPAR starts, it has to warm up the CPU cache, loading its data into it. After all three workloads are given their entitlement the first time, each still has 2ms of work left. So they request more CPU and the system evaluates their weights and assigns them processor resources accordingly to determine their share.

So during our 10ms window, each LPAR has run three times and has had to warm up the cache each time, all of which slows down the workload. If the entitlement had been set to 0.3 for each workload, they would have run once for the full 3ms and the cache would have only had to be warmed up once.

It's also important to look at VP-to-entitlement ratios. Ideally the ratio should be 2.5 or less. Anything above 4.0 is performance unfriendly, especially on multi-node systems (770 and above). When the VP is dispatched but its home node is busy, the system uses scheduler resource affinity domains (SRADs) at run time to determine the best core to dispatch it to. This core could be local, near or far. However, the memory might still be allocated on DIMMs attached to the home or some other core. As the LPARs get busy, this is more and more likely to happen. If memory is local, the bandwidth on the POWER7 is 68GB/s per memory controller. If it's near, it goes to 40-50GB/s, and far memory is about 23-26GB/s.

Clearly, you take a performance hit if memory isn't on the DIMMs attached to the home core. On a very busy system with lots of LPARs, it can lead to thrashing of the memory subsystem as well as affinity issues. This is more likely to happen with lots of VPs and low entitlement in the LPAR because the LPAR will be spread further in raw throughput mode. The lssrad command can show if this is happening. You can also run the Hardware Management Console (HMC) scanner to get information on servers and calculate the VP-to-entitlement ratios, keeping them within the values recommended by IBM's Nigel Griffiths

Redefined

When defining LPARs, changes in POWER7/POWER7+ technology now require you to:

- Reduce VPs so the system is forced to drive the threads giving full throughput.
- Pay attention to entitlements, setting them to something more like the peak average that the workloads need to run. This will help reduce LPAR shredding and ensure that VPs run at full

speed as long as possible.

- Look at the VP-to-entitlement ratios to reduce stress on the memory subsystem and to ensure that workloads have sufficient VPs to run at peak, but also that their entitlements are sensible.

Resources

< Return to main article

Print Email

Virtual User Group: “Capacity and Entitlements and Virtual Processers,” Parts 1 and 2 with Rosa Davidson

http://www.tinyurl.com/ibmaixvug

PowerVM User Group: Sessions 19 and 20 on “POWER Affinity and Performance” by Nigel Griffiths

http://tinyurl.com/newUK-PowerVM-VUG

“Simultaneous Multi-Threading on POWER7 Processors” by Mark Funk

http://www.ibm.com/systems/resources/pwrsysperf_SMT4OnP7.pdf

“Processor Utilization in AIX” by Saravanan Devendran

http://www.redbooks.ibm.com/redbooks/pdfs/sg247940.pdf

“IBM PowerVM Virtualization—Managing and Monitoring”

http://www.redbooks.ibm.com/redbooks/pdfs/sg247590.pdf

“IBM Power Systems Performance Guide—Implementing and Optimizing”

http://www.redbooks.ibm.com/redbooks/pdfs/sg248079.pdf

“Maximizing the Value of an IBM POWER7 and IBM POWER7+ Environment Through Tuning and Optimization”

http://www.redbooks.ibm.com/technotes/tips0956.pdf

< Return to main article