What I Learned This Month: More zIIPage May Not Be Better (generosity)

Scott Chapman
American Electric Power

Actually, I didn't just learn this lesson this month, but had held off writing about it on the theory that many people understood the situation and that IBM would soon produce a satisfactory solution. When we last tested IBM's solution it didn't seem to change the situation and I recently ran into somebody else that was unaware of the situation, so I decided perhaps I'd share my experiences here.

In May 2010, IBM released what I call the 60% PTF, PM12256, for DB2. This ostensibly raised the percentage of the DDF work that is offloaded to a zIIP from 55% to 60%. We applied it and found that the actual impact was undesirable for us. We contacted IBM who confirmed the functionality was as designed and suggested that if we didn't like it, we could choose to not apply the PTF. And so we did not apply it to our version 9 DB2 subsystems. Late last year however, it rolled into v9 with RSU maintenance, which makes it a bit uncomfortable to pull out; after all, RSU maintenance is supposed to be a whole package that was tested together. We again contacted IBM and went through a few months of back and forth and testing some early release PTFs that didn't fix the situation before finally closing the ETR in frustration. My opinion at the moment is that it's BAD: "Broken As Designed".

So why do I think that and why do some shops think the 60% PTF is great?

When an authorized / licensed product creates an enclave to run zIIP-eligible work, it can tell z/OS what percentage of the CPU time it wants to make eligible to run on the zIIP. Peter Enrico has referred to this as the "generosity factor". DB2 has historically been 55% generous. It seems most ISVs that are using the zIIP interface are setting the value to 100%, but I know of one that allows the customer to set it to any value they want.

Unfortunately, PM12256 does not have DB2 simply setting the value to 60%. Instead it marks 3 out of every 5 (60%) of the enclaves to 100%, and the remainder to 0%. If your DB2 DDF threads are relatively similar to each other, then it will work out that about 60% of your DDF CPU time will be offloaded instead of 55%. And generally speaking, more zIIP offload is better.

But if you have something like QMF or other ad hoc queries coming into DB2 via DDF, your transaction mix is almost certainly not homogenous. We have DDF queries that take from single-digit milliseconds to tens of minutes of CPU time. In that sort of situation, you'll find that the percentage of the DDF work that is offloaded will be highly variable. For example, assume you have a situation where you have 1 thread per second that takes 10ms of CPU time. And one thread every 15 minutes that takes 60 CPU seconds. That's 69 seconds total of CPU time every 15 minutes. Previously, about 38 seconds would have been offloaded to the zIIP. With PM12256, how much gets offloaded depends entirely on how "lucky" the 60 second thread gets: it will be either all offloaded or not. So in total, you might get 5 seconds offloaded, or you might get 65 seconds offloaded. This issue gets compounded when you have queries that take tens of CPU minutes. If you have users doing adhoc queries, it's probably a fair guess that you have some of those queries.

This poses an obvious capacity planning problem: it's hard to plan for randomness. It also makes for unhappy users, unless your GCPs (general-purpose CPUs) run at full speed (same as the zIIPs) and you have similar free capacity on both types of CPUs. If either of those is false, then the elapsed time difference between running 100% on the GCP and 100% on the zIIP may be considerable. It is a very real possibility that a query's elapsed time may vary by 100% based simply on whether it was lucky enough to get on the zIIP or not.

To add to the complexity: WLM counts specialty engine CPU time towards period aging, but not to resource groups. So if you have WLM resource groups for DDF work, they will now only affect 60% of the workload.

It's not all bad, though. If your DDF workload is homogenous, it's probably a fine deal. And if you have giant hour-long queries that you want to ensure run on the zIIP, you could always check if they're running on the GCP, and if so cancel them and resubmit them until they get on the zIIP. That's not really practical in a general way, but in exceptional cases, it's something to consider.

Things also get interesting when the GCPs are constrained and higher importance work is keeping the DDF work off from them. Assuming there's zIIP capacity available, previously all DDF work would have been impacted the same. But since 55% of every thread was offloaded, if they trickled through on the GCP, they'd eventually fall over to the zIIP, and maybe suddenly finish. Now 60% of the threads will run fine on the zIIP and the other 40% will likely be impacted even worse than they were before. Whether you consider this good or not depends on whether your thread is in the 60% or 40% group.

Two APARs (PM28626 and OA35146) are now available to address the situation. I believe those are the APARs we tested to no avail. Hopefully they work better now in their final release, but we haven't put them on yet so I can't say. If you have PM12256 installed and are experiencing issues as outlined above, then certainly I would try those. And if they don't solve your issue, let IBM know about it. From my perspective, they took a step backwards and they need to hear from customers how it's impacting them.

A brief editorial: the specialty engines are a marketing solution to help lower the cost of running certain workloads on the mainframe, presumably to help keep work on the mainframe and maybe even attract new workloads. The current DDF zIIP situation, with unpredictable performance, is not helpful to that end. How about handling DDF on zIIPs like Java on zAAPs and making the generosity factor 100% and getting rid of the randomness? It seems that this would be more useful in terms of attracting (or keeping) work on the platform. From an IBM revenue perspective, it could only reduce IBM's revenue to the degree that customers (using sub-capacity pricing) have monthly peaks that are being driven by DDF work. In those cases, PM12256 could have very well raised those customers' costs if their peaks are driven by long running queries. My belief is that many customers would see great value in 100% DDF zIIP offload and move all sorts of work to DB2 on z/OS. It seems that this would be good for both IBM and customers. I know it would help me when having the discussions about copying data out of DB2 vs. simply leaving it in DB2 and accessing it in-place.

As always, if you think I've got it all wrong, please email me at sachapman@aep.com and let me know!