Macro to Micro: Understanding z/OS Performance Moment by Moment

Scott Chapman

Enterprise Performance Strategies, Inc.

Scott.chapman@EPStrategies.com

Contact, Copyright, and Trademarks

Questions?

Send email to performance.questions@EPStrategies.com, or visit our website at https://www.epstrategies.com or http://www.pivotor.com.

Copyright Notice:

Trademarks:

Enterprise Performance Strategies, Inc. presentation materials contain trademarks and registered trademarks of several companies.

The following are trademarks of Enterprise Performance Strategies, Inc.: **Health Check®, Reductions®, Pivotor®**

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries: IBM®, z/OS®, zSeries®, WebSphere®, CICS®, DB2®, S390®, WebSphere Application Server®, and many others.

Other trademarks and registered trademarks may exist in this presentation

Abstract (why you're here!)

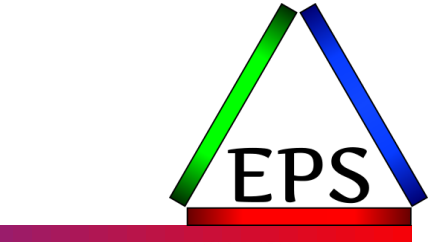
Most customers report on most z/OS performance in 15-minute intervals, which is fine for most use cases. But those 15-minute intervals gloss over the complexity of what happens on time scales more relevant to the CPU: microseconds to seconds. While performance analysts usually don't need to worry about that complexity, having a mental model of what's going on at the micro scale can help explain seemingly confusing results such as having TSO users complain about poor response time when the reports indicate the machine is not very busy. Or why the same work consumes different amounts of CPU at different times. Or why LPAR configurations can impact performance.

In this presentation Scott Chapman will discuss performance from the 15-minute interval down to the microsecond level and discuss how CPUs get virtualized and shared between LPARs and between address spaces within LPARs. You'll leave with a new appreciation for how the micro level can impact the macro level.

Agenda

- Example long term views of performance
- Understanding what's happening on shorter timeframes
- Measurement recommendations and examples

EPS: We do z/OS performance…



- Pivotor - Reporting and analysis software and services
  - Not just reporting, but analysis-based reporting based on our expertise
- Education and instruction
  - We have taught our z/OS performance workshops all over the world
- Consulting
  - Performance war rooms: concentrated, highly productive group discussions and analysis
- Information
  - We present around the world and participate in online forums
    https://www.pivotor.com/content.html
    https://www.pivotor.com/webinar.html

z/OS Performance workshops available

During these workshops you will be analyzing your own data!

- WLM Performance and Re-evaluating Goals
  - February 19-23, 2024
- Parallel Sysplex and z/OS Performance Tuning
  - August 20-21, 2024
- Essential z/OS Performance Tuning
  - October 7-11, 2024
- Also… please make sure you are signed up for our free monthly z/OS educational webinars! (email contact@epstrategies.com)

Like what you see?

- The z/OS Performance Graphs you see here come from Pivotor

- If you don't see them in your performance reporting tool, or you just want a free cursory performance review of your environment, let us know!
  - We're always happy to process a day's worth of data and show you the results
  - See also: http://pivotor.com/cursoryReview.html

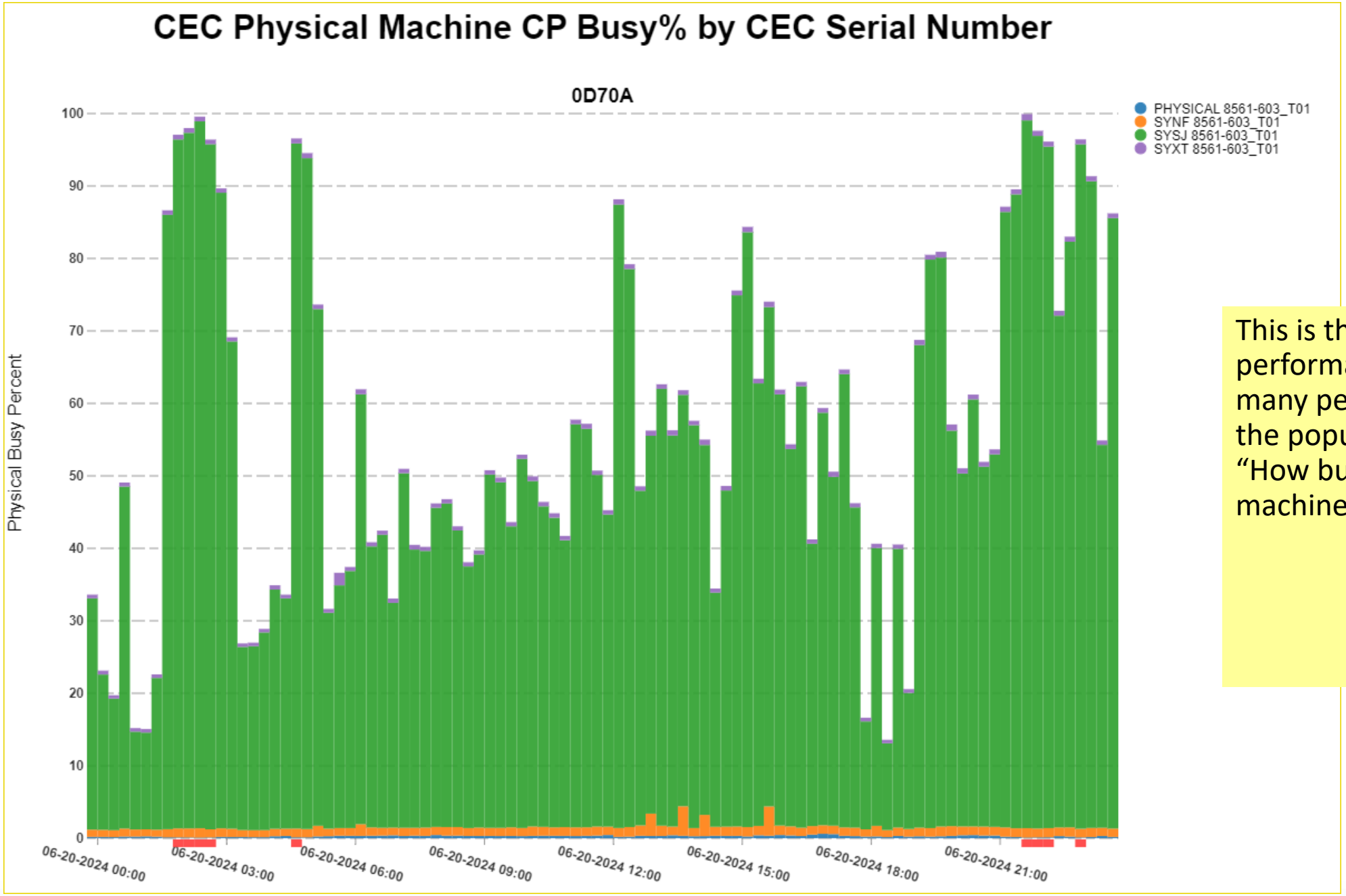
- We also have a free Pivotor offering available as well
  - 1 System, SMF 70-72 only, 7 Day retention
  - That still encompasses over 100 reports!

EPS presentations this week

What	Who	When	Where
60 Years of Pushing Performance Boundaries with the Mainframe	Scott Chapman	Sun 17:00	Neptune D
Introduction to Parallel Sysplex and Data Sharing	Peter Enrico	Mon 13:15	Pomona
Macro to Micro: Understanding z/OS Performance Moment by Moment	Scott Chapman	Mon 15:45	Neptune D
WLM Turns 30! : A Retrospective and Lessons Learned	Peter Enrico	Tue 10:30	Neptune D
PSP: z/OS Performance Spotlight: Some Top Things You May Not Know	Peter Enrico Scott Chapman	Tue 13:00	Pomona
More/Slower vs. Fewer/Faster CPUs: Practical Considerations in 2024	Scott Chapman	Tue 14:15	Neptune D
z16 SMF 113s – Understanding Processor Cache Counters	Peter Enrico	Wed 13:15	Pomona

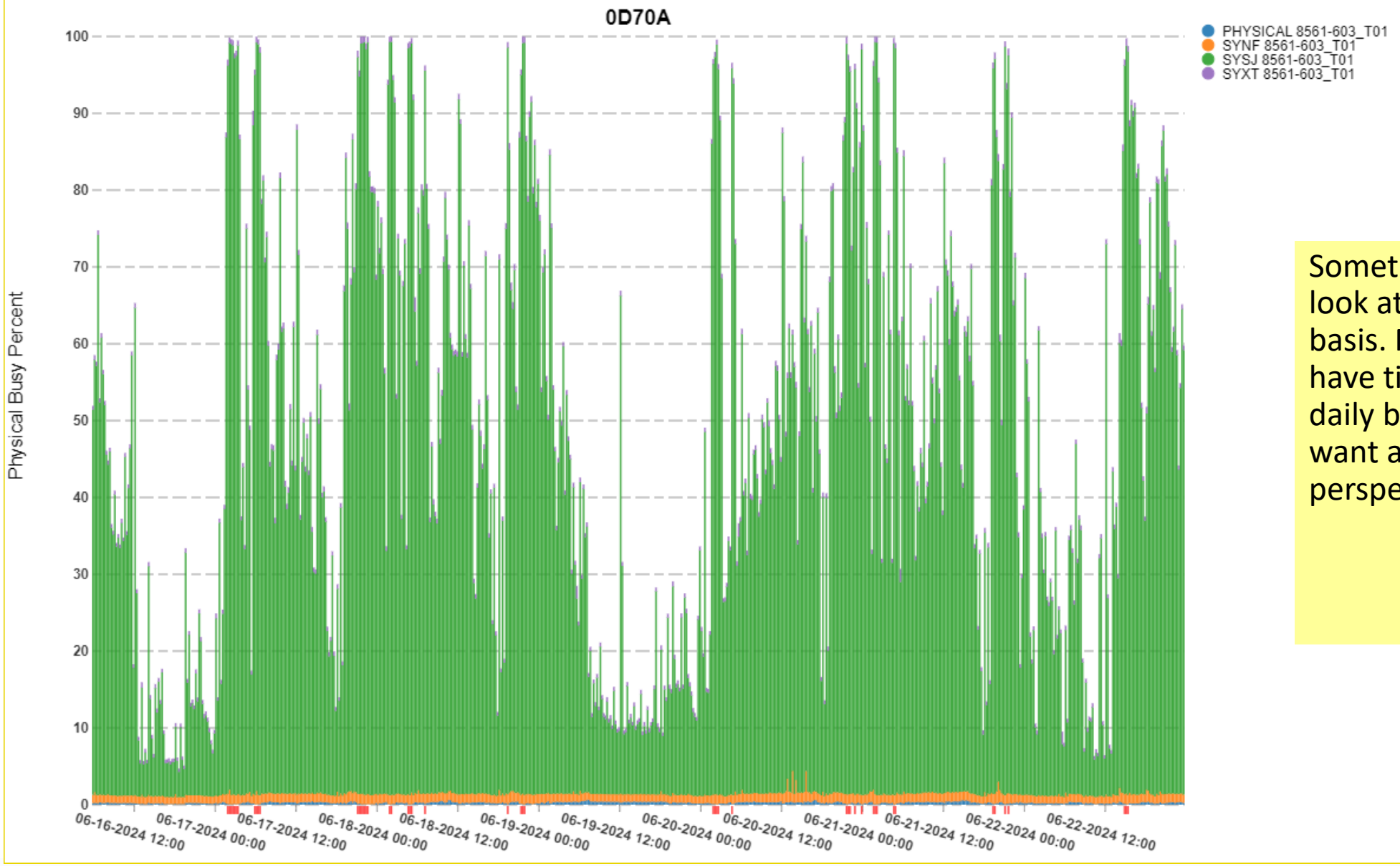
Taking a look at the forest

CEC Physical Machine CP Busy% by CEC Serial Number

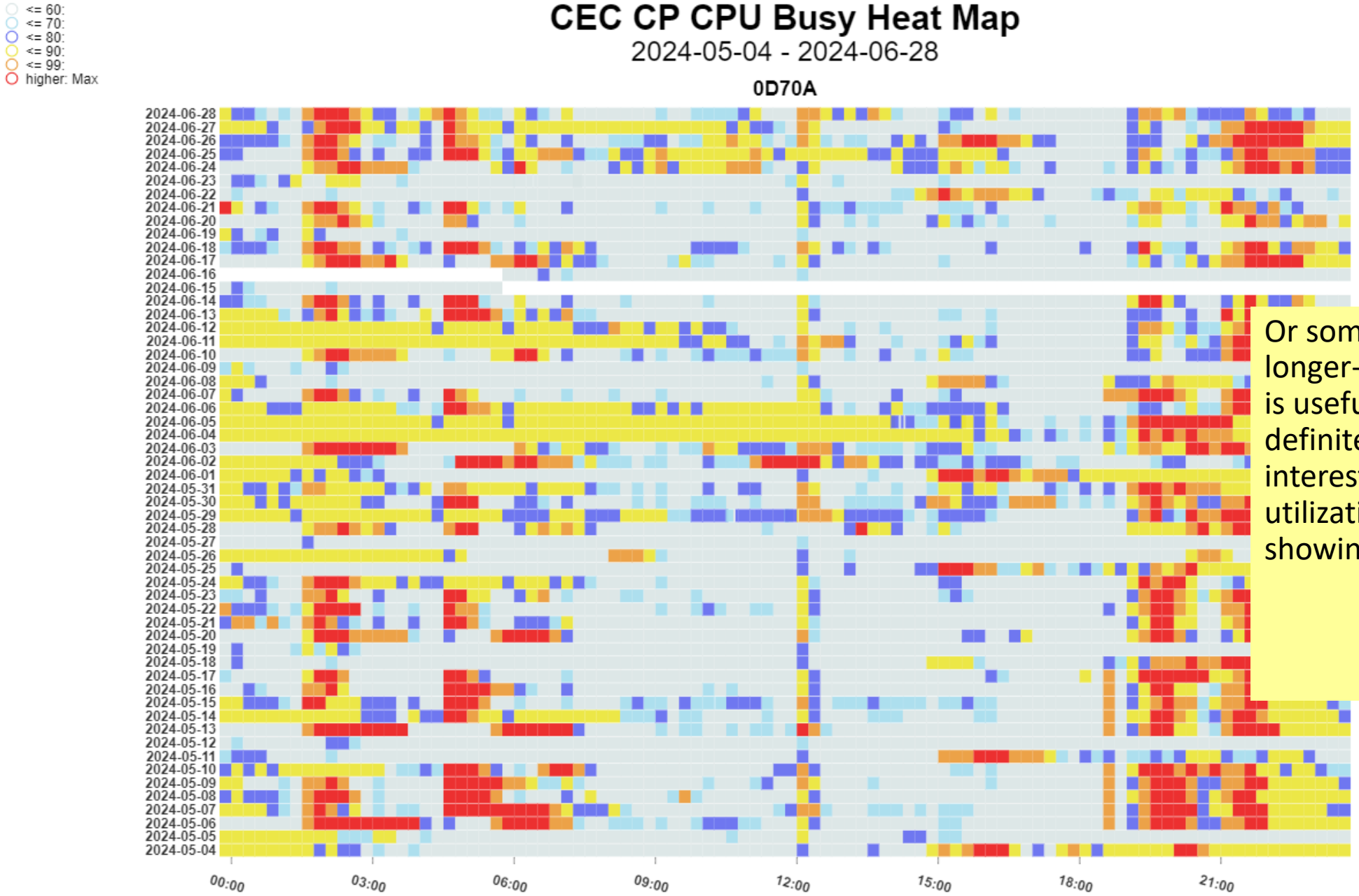


This is the #1 performance graph for many people, to answer the popular question: “How busy was the machine yesterday?”

CEC Physical Machine CP Busy% by CEC Serial Number



Sometimes people might look at it on a weekly basis. Maybe they don't have time to look on a daily basis or they just want a longer term perspective.



Or sometimes an even longer-term perspective is useful. There's definitely some interesting patterns of utilization that are showing up here.

These wide views are good…

- In fact, I'd heartily recommend you look at "how busy was my machine yesterday" every day!
  - As well as a few other things like transaction volumes and response times for your most important applications
  - Business metrics also very handy to have readily at hand!
- A longer term perspective is also very helpful
  - Helps you stay on top of (or even ahead of) long term trends and changes
  - Should really look at these sort of things at least once a month

AI has trouble with cat faces too

Performance analysis is more detailed

- Performance Analysis is typically focused on a very short time period
  - Usually: events that last minutes to hours
- Measurement intervals that are too long can impede your ability to even see that there was a problem
  - 15-minute RMF interval is extremely common, but may be too long at times
- Don’t be mislead into thinking that the capacity of the machine or LPAR is some homogenous bucket of capacity
  - Better to think of it as multiple buckets of varying sizes which are available for variable periods

Better AI-generated cat

Looking under the trees

"System isn't busy" questions

- We often field performance questions where the performance person is confused because there's a perceived performance problem during a time when "the system isn't busy"
- We sometimes are asked why even SYSSTC may suffer delays when "the system isn't even busy"
- People seem to under-appreciate the potential impact of running on a machine with few physical CPs
  - "But they're fast CPs, and we're rarely over 80% busy"

An appreciation for the timescale of common events can help you better explain performance anomalies and make better choices about system configurations

System Event Timescale (Seconds)

Fortunately, there are measurements on these timescales too!

Most people spend a lot of time looking at measurements up here

	1800.0	Default (not recommended) SMF interval
	900.0	Common RMF measurement Interval
	10.0	WLM policy interval
	2.0	HiperDispatch interval
	1.0	1 second (s) (Common RMF sampling interval)
	0.250	SRM sampling interval
	0.2	Faster-than-average human visual reaction time
	0.1	Typical LPAR VH processor time slice
milliseconds	0.0125	Typical LPAR VM/VL processor time slice
milliseconds	0.008	Typical cache miss disk I/O (spinning disk)
milliseconds	0.0032	Typical default zIIPAWMT
milliseconds	0.001	1 millisecond (ms, thousandths)
microseconds	0.0005	Typical average modern I/O
microseconds	0.0002	Typical cache hit I/O
microseconds	0.000030	Possible major z/OS time slice
microseconds	0.000006	Possible minor z/OS time slice
microseconds	0.000005	Typical (approx.) CF sync lock request
microseconds	0.000001	1 microsecond (µs, millionths)
	0.000000001	1 nanosecond (ns, billionths)
	0.0000000002	1 z13 machine cycle (5 Ghz)
	0.00000000019	1 z14/z15/z16 machine cycle (5.2 Ghz)

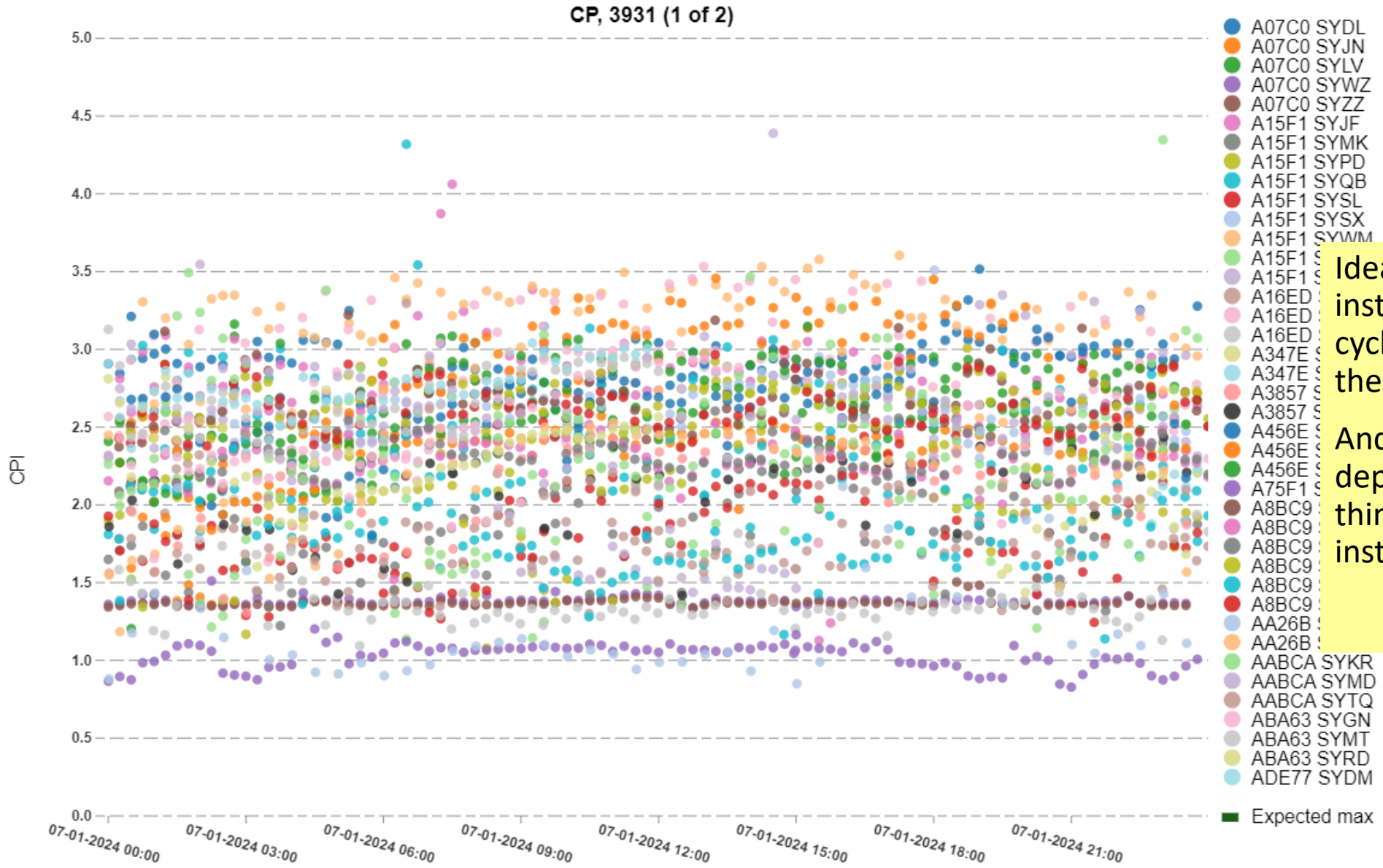
But the work is happening down here on these timescales

Clock Speed and Cycles

- In one z16 clock cycle, light in a vacuum can only travel just over 2 inches!
  - Electrical signal in a circuit is much slower (40-70% of c)
  - 1 meter in fiber ~ 5 ns (>25 clock cycles!)
- Need to make a round trip
- Signal paths aren't as the mosquito flies
  - IBM's "Miles of wire in the chip" numbers:
    - zEC12 – 7.7 miles
    - z13 – Over 13 miles
    - z14 – 14 miles
    - z15 – 15.6 miles
    - z16 – 19 miles
- Physical distance matters!

Cycles Per Instruction

By z/OS Hardware Model



Ideally one would have 1 instruction complete per cycle, but reality is not there for most systems.

And this of course depends on many things, including the instruction mix.

Instructions per interval

Just based on an average CPI of 2.5

	Cycles	Instructions
CF Sync Lock Rqst	26,316	10,526
Minor Time Slice	31,579	12,632
Major Time Slice	1,578,947	631,579
Cache Hit I/O	1,052,632	421,053
LPAR VM/VL Slice	65,789,474	26,315,789
LPAR VH Slice	526,315,789	210,526,316

Instructions that couldn't happen while the CP was "spinning"

Essentially a minimal run time for a work unit

These are mostly gee-whiz numbers: there's a wide range of instruction complexity!

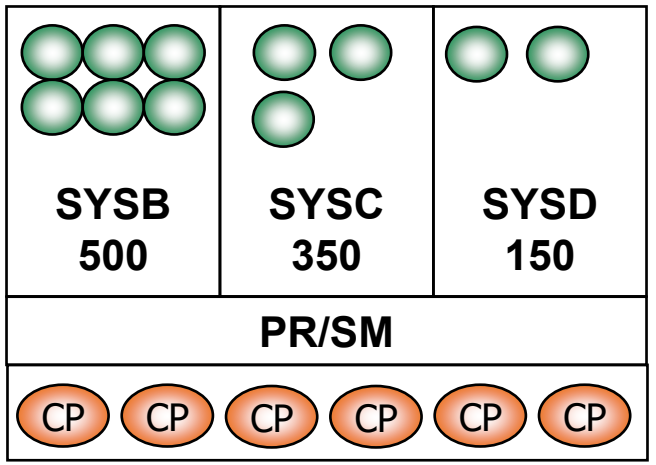
Note that cycles are a measure of time

Background: Logical and Physical CPUs

- Processor = CP = CPU = GCPU or zIIP or any other processor type
  - All the same bit of silicon: a core on a physical chip
- You pay for a certain number of physical processors (CPs)
  - A processor can only be processing one stream of instructions at a time
    - Absent SMT, which don't apply to GCPs and which we're not going to discuss here
- You define LPARs, each with a certain number of logical, shared CPs
  - For each LPAR Logical CPs <= physical CPs, although can have reserved CPs
  - Most machines have multiple LPARs
- z/OS dispatches work to its (logical) CPs
- PR/SM dispatches logical CPs to physical CPs
  - A logical CP can't do any work when it's not dispatched to a physical CP

Weights and logical CPs

- Each LPAR is guaranteed to get at least its share
  - $LPAR\ Share = 100\ * \frac{LPAR\ Weight}{\sum Weight\ of\ activated\ LPARS}$
- In below example:
  - SYSB – guaranteed 50% of capacity of the 6 CPs (3 CPs worth of capacity)
  - SYSC – guaranteed 35% of capacity of the 6 CPs (2.1 CPs worth of capacity)
  - SYSD – guaranteed 15% of capacity of the 6 CPs (0.9 CPs worth of capacity)



Each system has some number of logical CPs

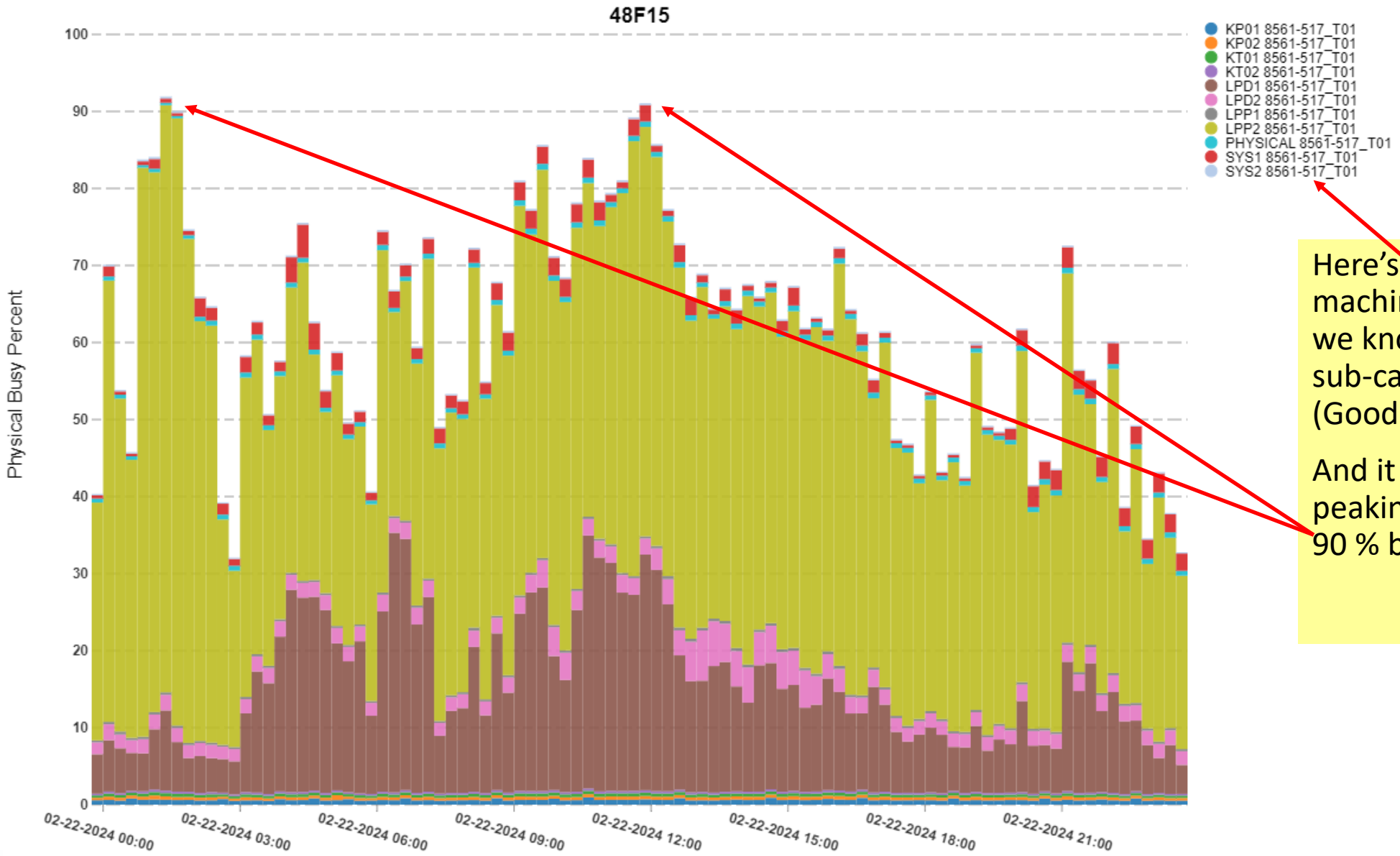
For ease of use, try to make weights add up to 1000 (like they do here).

Physical CPs shared by SYSB, SYSC, SYSD

HiperDispatch CP Management

- HiperDispatch manages CPs “vertically”, meaning it endeavors to make the logical CPs a larger percentage of a physical
- Logical processors classified as:
  - High – The processor is essentially dedicated to the LPAR (100% share)
  - Medium – Share between 0% and 100% (often 50-100% unless small LPAR)
  - Low – Unneeded to satisfy LPAR’s weight
- This processor classification is sometimes referred to as “vertical” or “polarity” or “pool”
  - E.G. Vertical High = VH = High Polarity = High Pool = HP
- Parked / Unparked
  - Initially, VL processors are “parked”: work is not dispatched to them
  - VL processors may become unparked (eligible for work) if there is demand and available capacity

CEC Physical Machine CP Busy% by CEC Serial Number



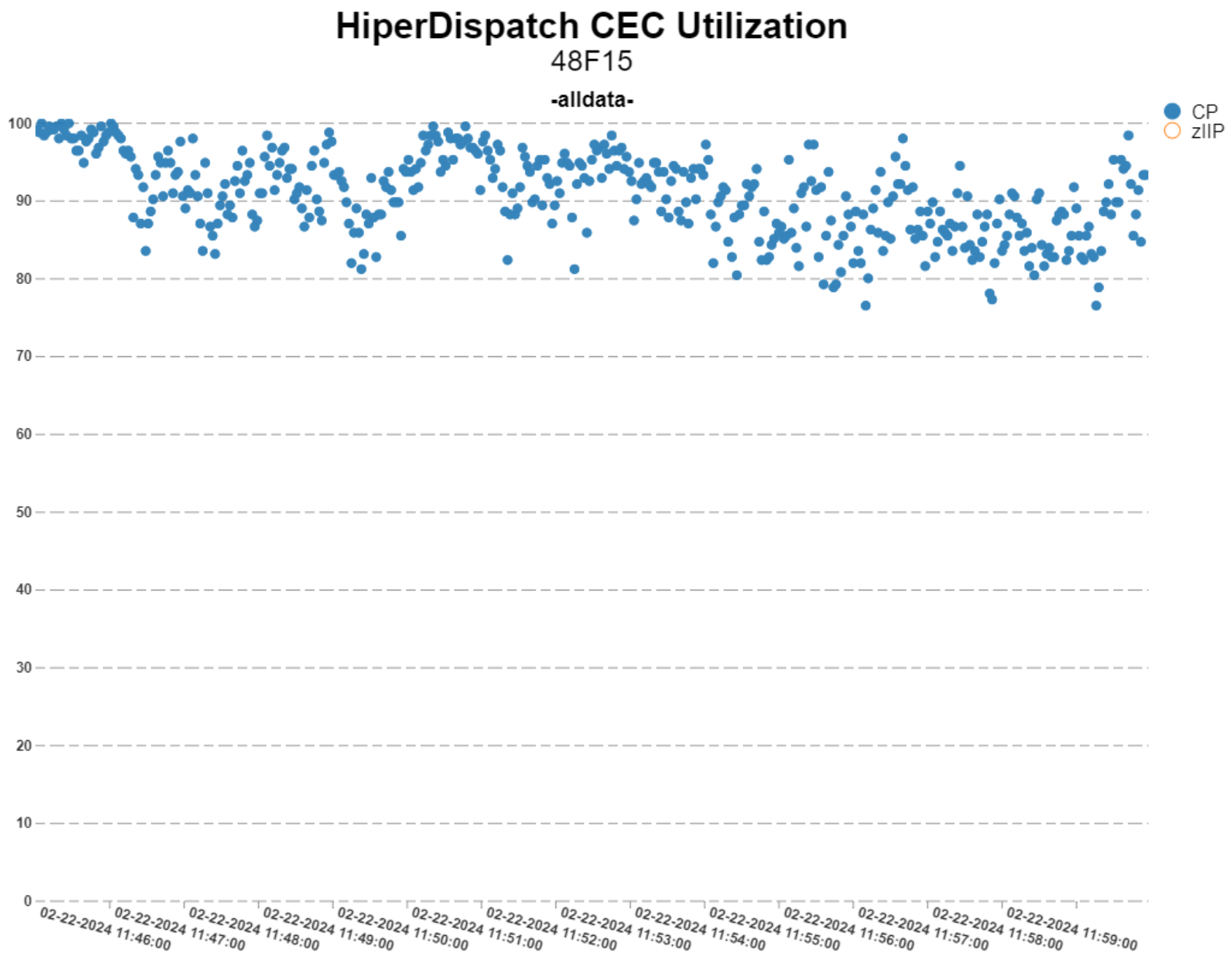
Here's another example machine: a z15 517 so we know that there's 17 sub-capacity GP engines. (Good choice!)

And it looks like it's peaking out at just over 90 % busy.

What does the 517 is 90% busy *mean*?

- Effectively that's an average utilization of the 17 GP engines over the course of the 15 minute (900 second) interval
  - So averaged over space (engines) and time (seconds)
- Really, it's total CPU time / CPUs * interval
  - E.G. 13770 / (900 * 17) = 0.9 = 90%

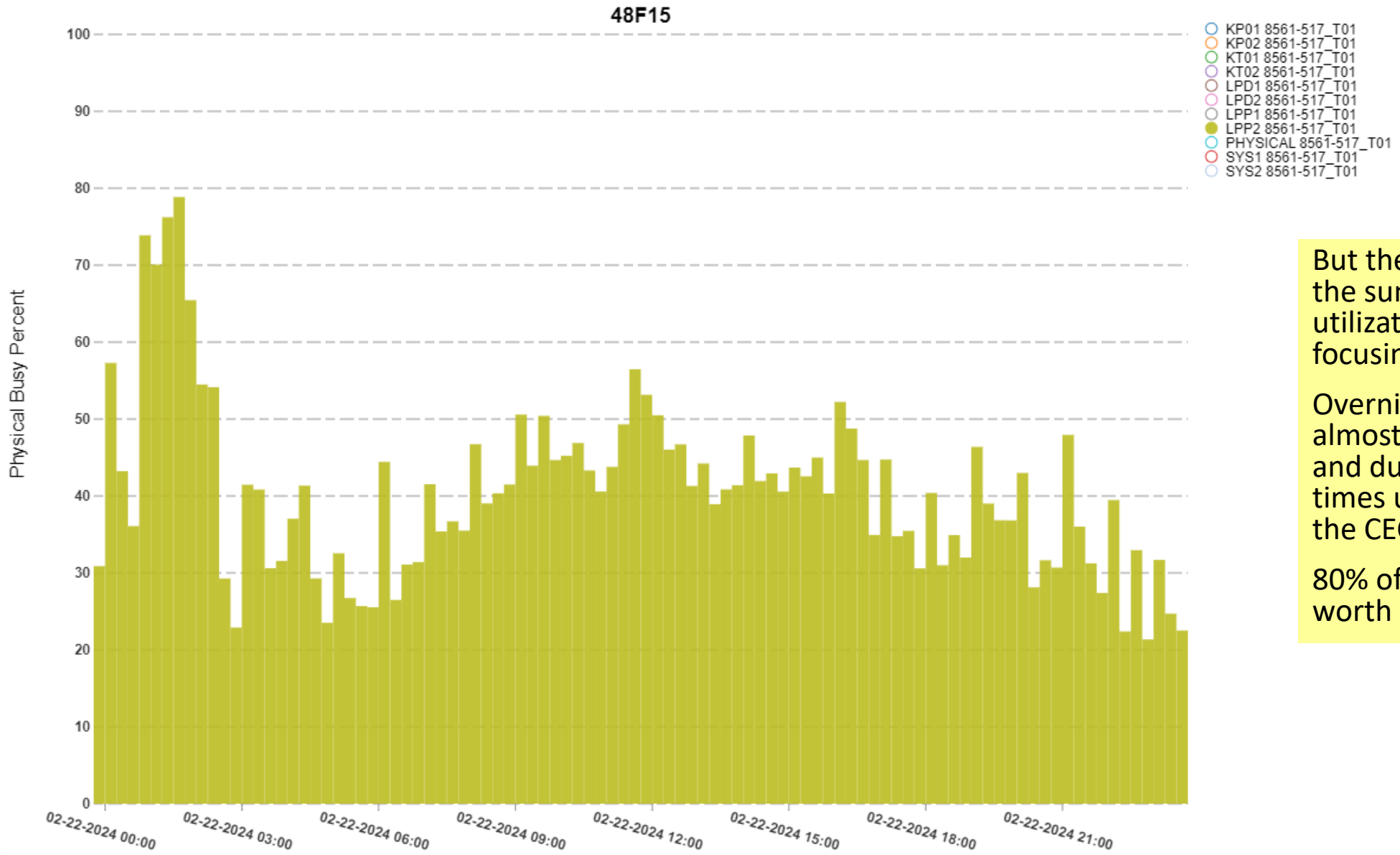
- Important notes:
  - At any given moment a CPU is either being used (CPU time) or is not being used
  - Averages can hide peaks within the interval



Here's that peak 15 minute interval that was showing just over 90% busy, but with observations every 2 seconds.

You see how the average was ~90%, but there were a few minutes where the utilization was more like 99%+.

CEC Physical Machine CP Busy% by CEC Serial Number

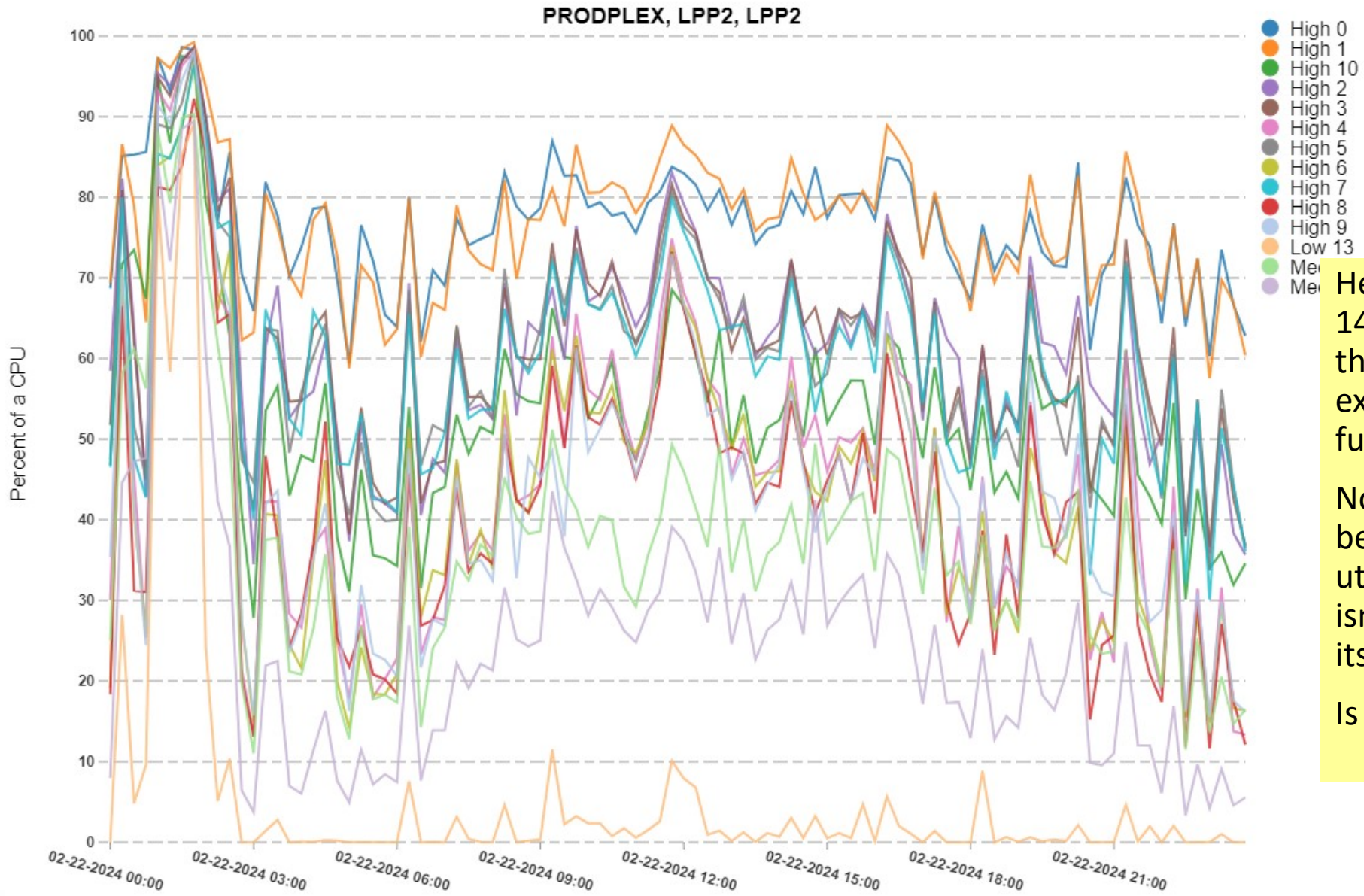


But the CEC utilization is the sum of the LPAR utilizations. Here we're focusing on just one.

Overnight it peaks at almost 80% of the CEC, and during the day at times uses around 50% of the CEC.

80% of 17 = 13.6 CP's worth of capacity.

LPAR Per-CPU CP Busy%

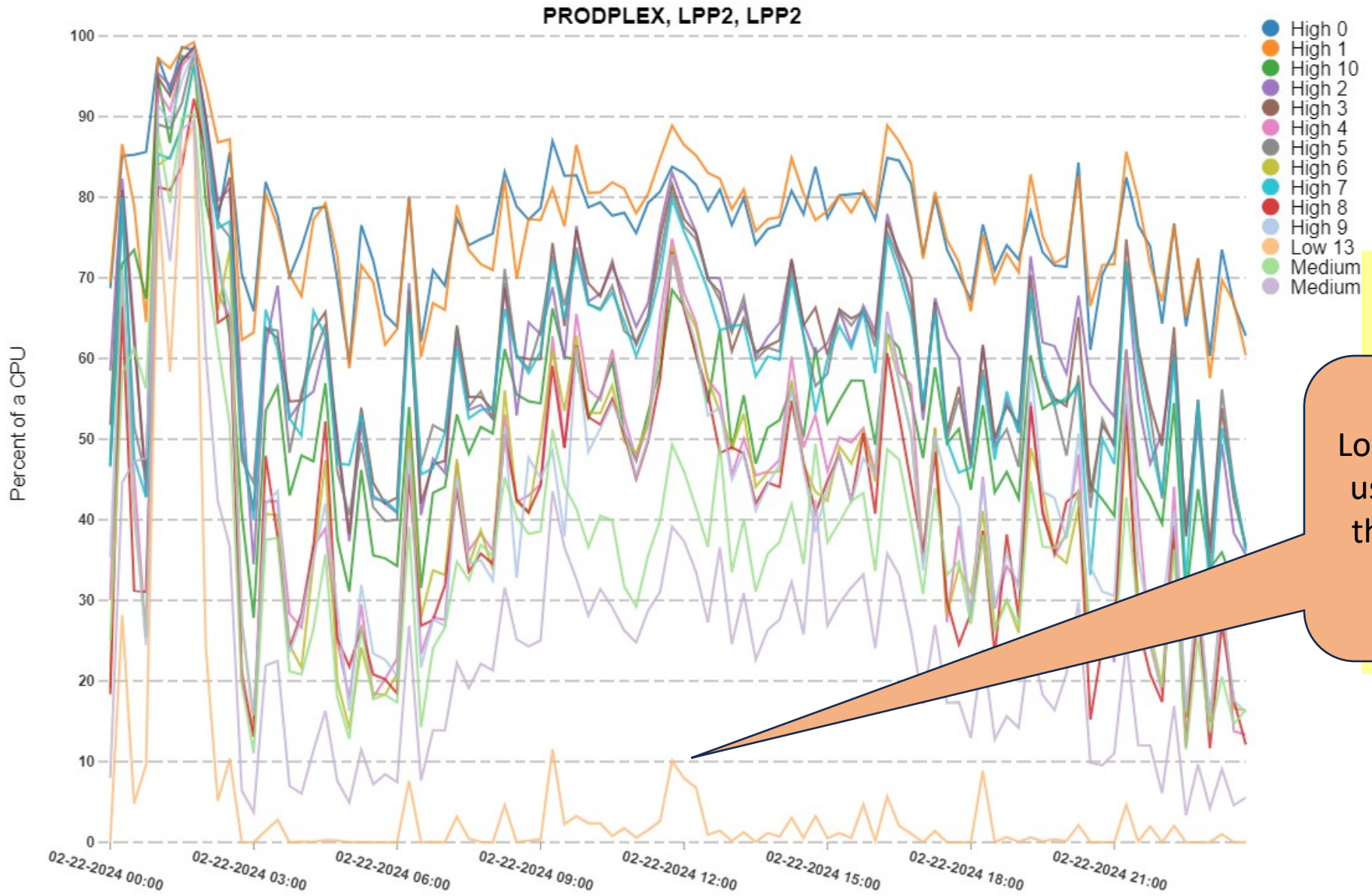


Here's the utilization of the 14 logical CPs defined to the LPAR, with utilization expressed as a percent of a full physical CP.

Note that there seems to be bands of processor utilizations when the LPAR isn't trying to consume all its possible capacity.

Is this surprising?

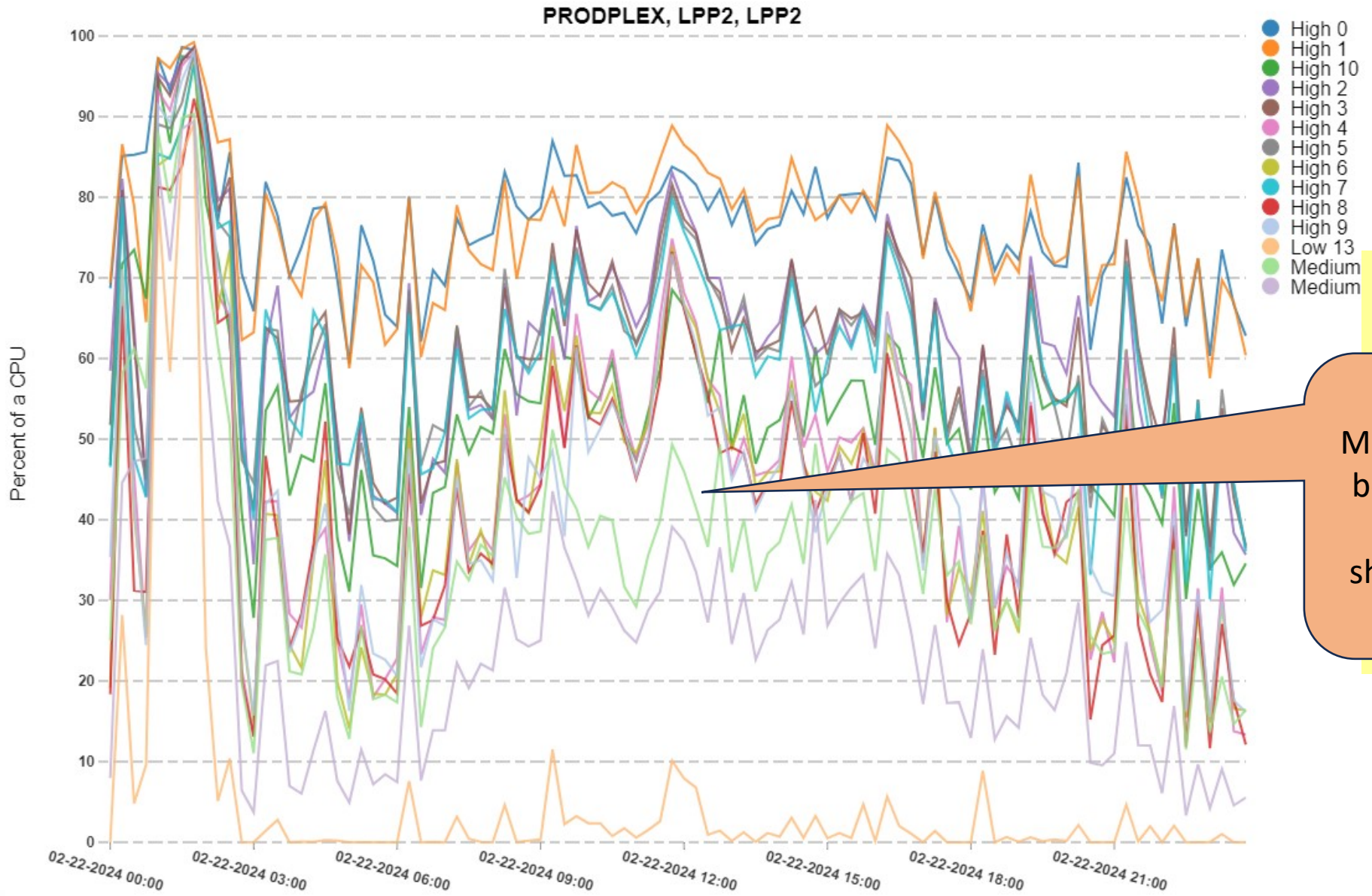
LPAR Per-CPU CP Busy%



This pattern of utilization is not at all surprising!

Low pool CPs will naturally use less, especially when they're not unparked for the entire interval.

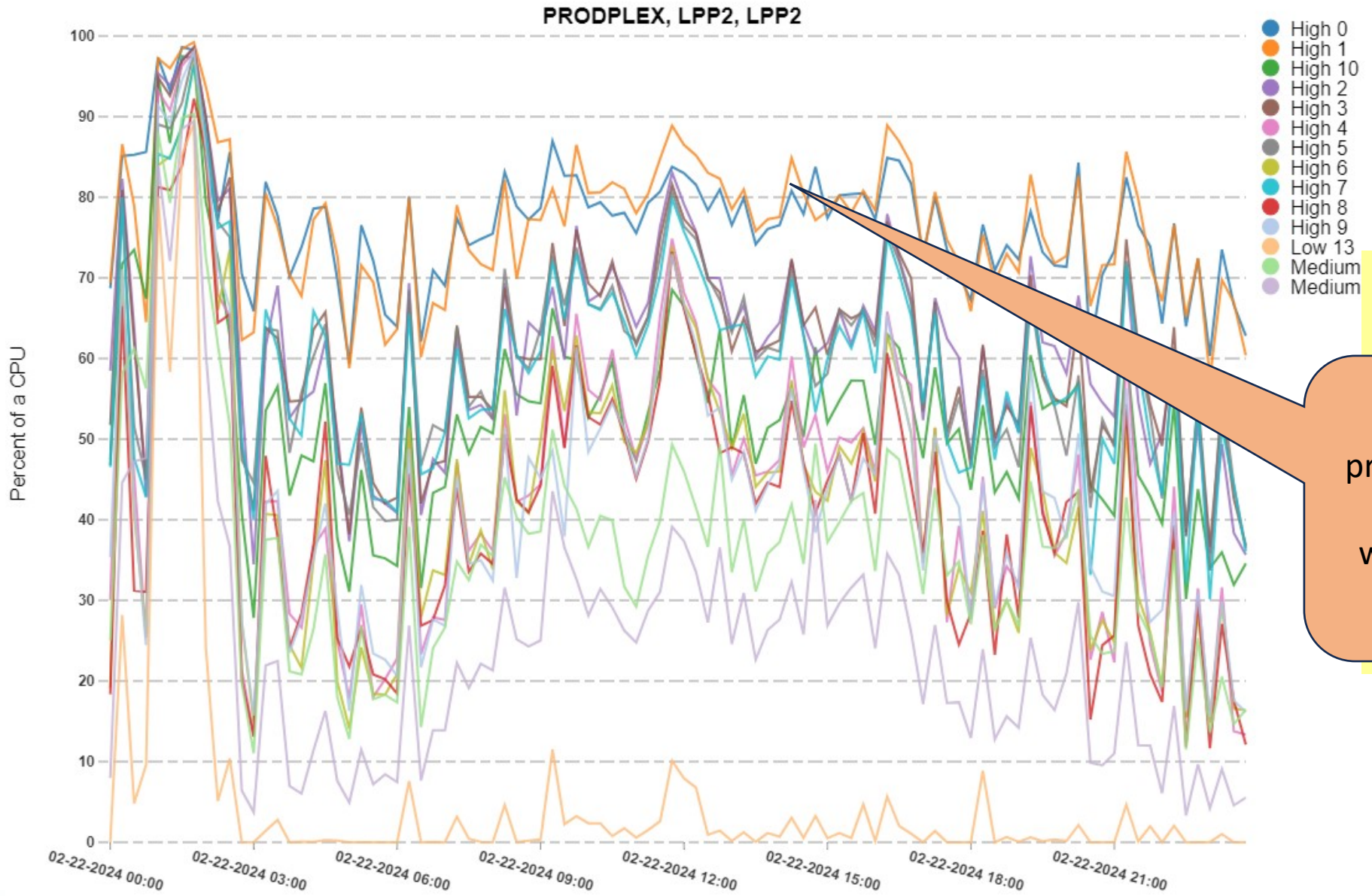
LPAR Per-CPU CP Busy%



This pattern of utilization is not at all surprising!

Medium pool CPs will also be expected to consume less because they're shared with other LPARs.

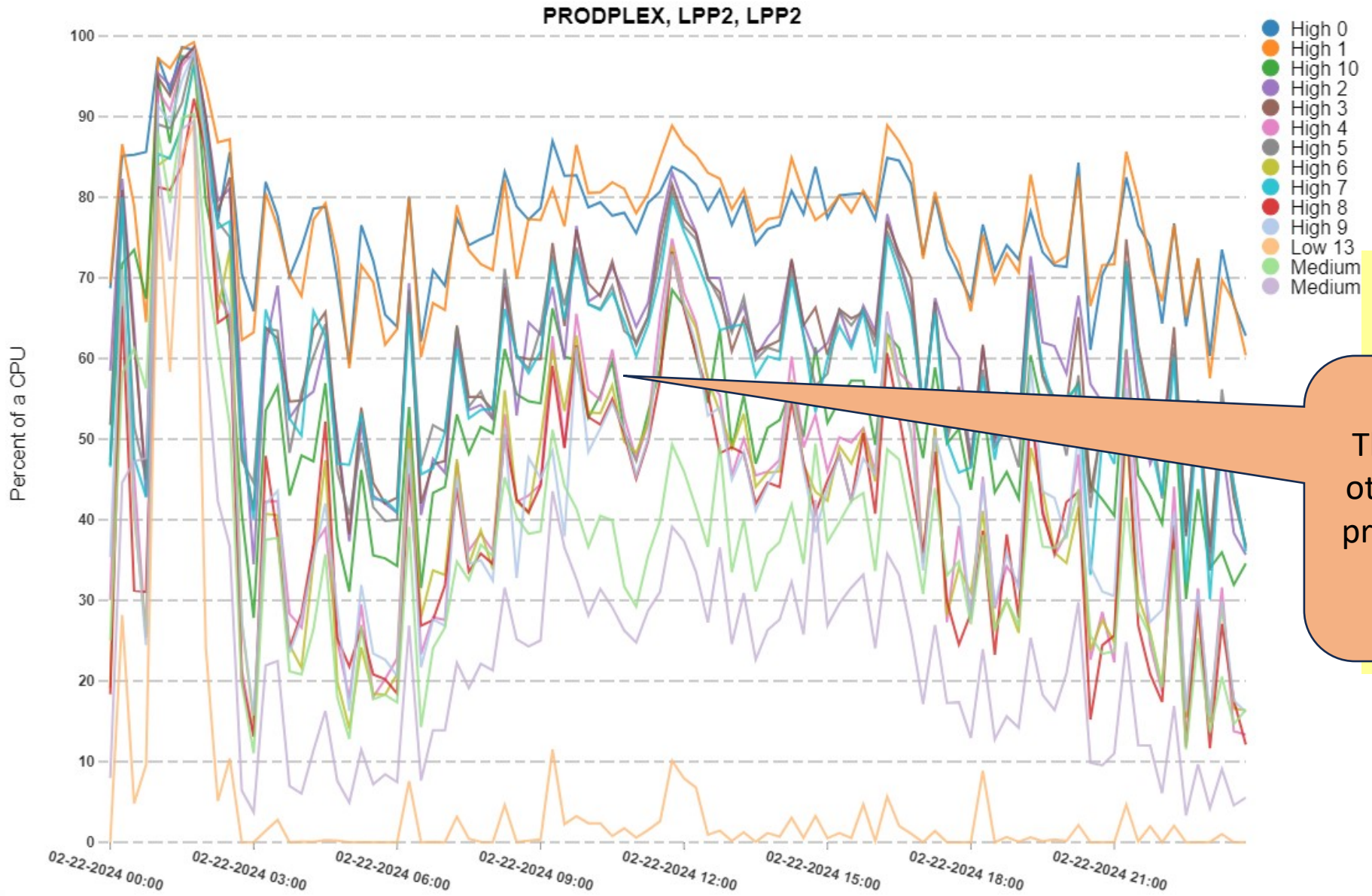
LPAR Per-CPU CP Busy%



This pattern of utilization is not at all surprising!

These high pool processors were handling I/O interrupts so that would explain why they were using more.

LPAR Per-CPU CP Busy%



This pattern of utilization is not at all surprising!

The fact that there's two other groups of high pool processors is explained by affinity nodes.

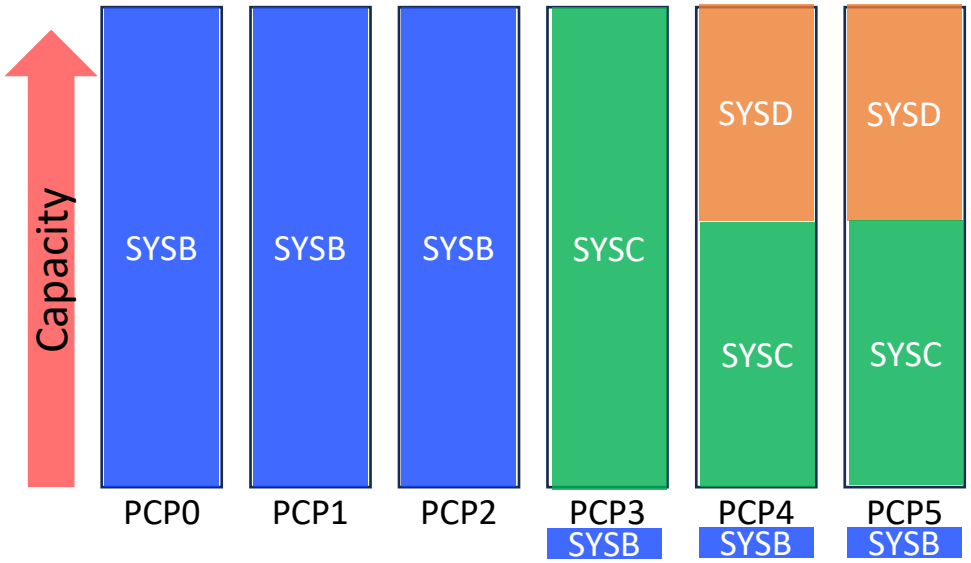
z/OS Dispatcher Affinity Nodes

- System creates nodes of logical processors
  - Originally said to be “ideally 4 high-pool processors”
  - But on recent machines, 2-3 high pool processors seems quite common
    - This makes more sense to me!
  - May have many low pool processors in one node
- Each node gets its own queue
  - Work units assigned to a particular node
  - Separate high performance work unit queue for SYSSTC/SYSTEM SRBs crosses nodes
- Nodes have list of helper nodes
  - Node needs help when it can’t run all the work assigned to it
    - Low pool processor in the node used before signaling another node
  - “Needs help” frequency controlled in part by CCCAWMT and ZIIPAWMT in IEAOPTxx

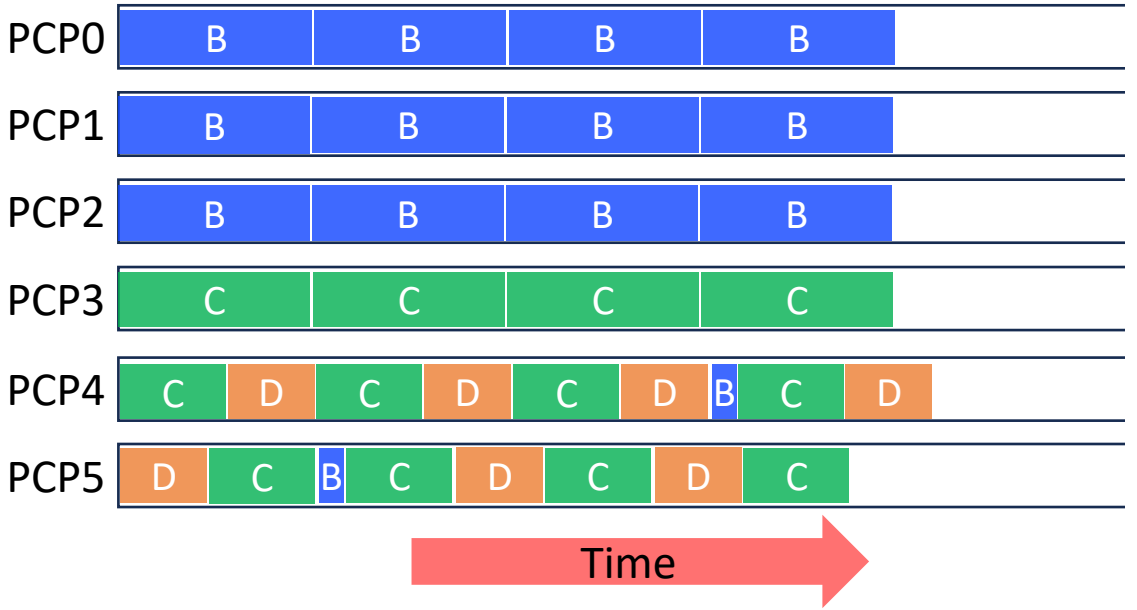
PR/SM Affinity

- PR/SM also enforces affinity
  - High Pool logical CPs have very strong affinity to a particular physical CP
  - Mediums will try to stay in the same area in the nest (especially at book level)
  - Low pool CPs have little affinity as their capacity is not guaranteed by their weight
- See “The Highs and Lows: How Does Hyperdispatch Really Impact CPU Efficiency?” at https://www.pivotor.com/content.html
  - While tweaking weights to convert 1 medium to 1 high probably won’t have a significant impact, choosing more/slower CPs so you have a number of high pool processors instead of all mediums can be significant

Physical to Logical: Vertical Mgt

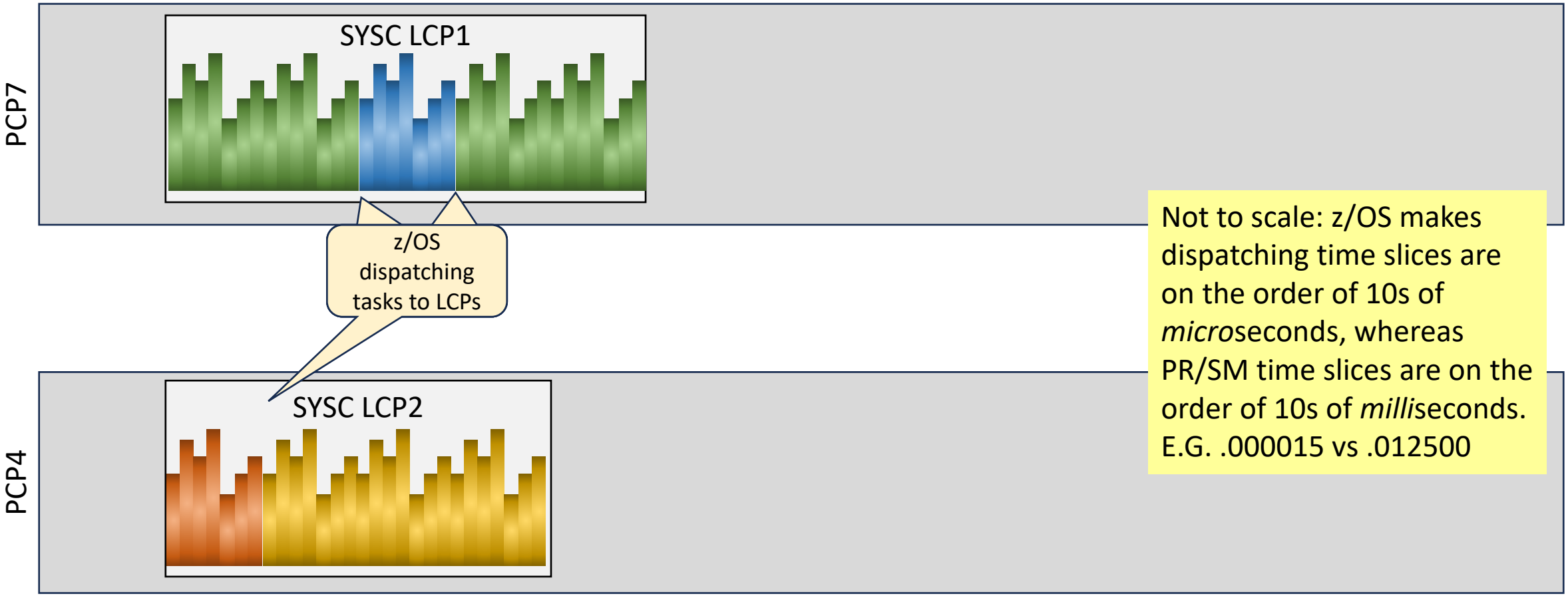


With HiperDispatch, vertical high CPs are quasi-dedicated to an LPAR. Note that SYSB's VLs will only come into play when there's both demand from SYSB and the other LPARs aren't using the capacity.



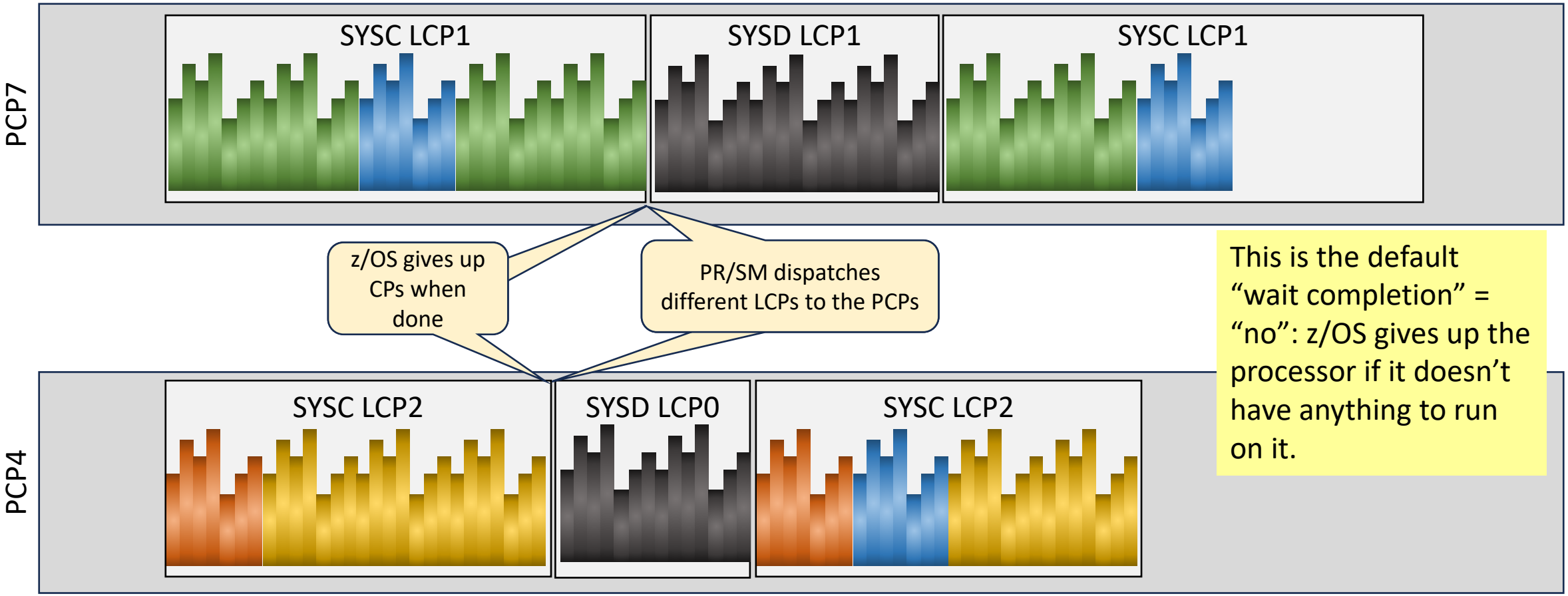
Note that while reality may be a bit messier, vertical CPU management does greatly reduce the movement of logicals to different physicals. Also note VH CPs get longer dispatch intervals.

PR/SM Dispatching LCPs



Not to scale: z/OS makes dispatching time slices are on the order of 10s of *micro*seconds, whereas PR/SM time slices are on the order of 10s of *milli*seconds. E.G. .000015 vs .012500

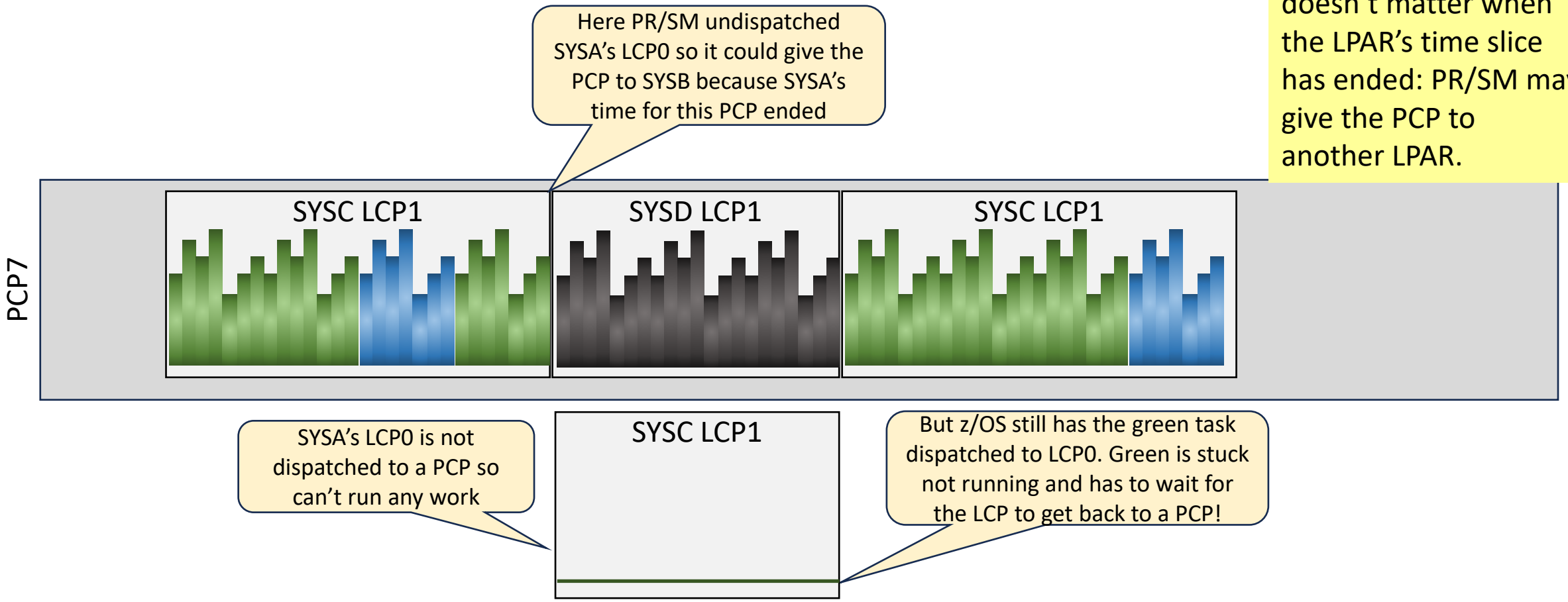
PR/SM Dispatching LCPs



This is the default "wait completion" = "no": z/OS gives up the processor if it doesn't have anything to run on it.

What if z/OS task wasn't done?

Wait completion doesn't matter when the LPAR's time slice has ended: PR/SM may give the PCP to another LPAR.



Involuntary Wait Issue

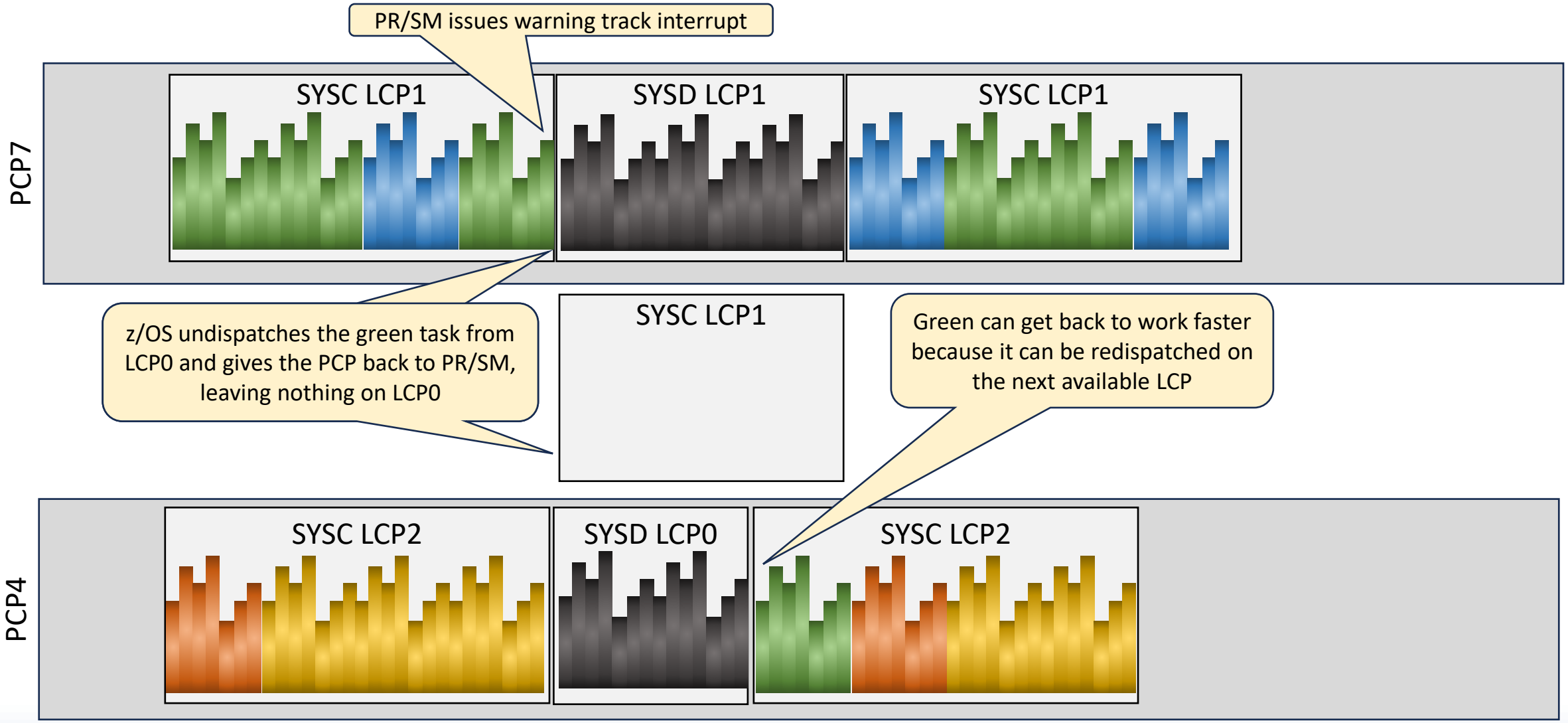
- When PR/SM steals an PCP from a z/OS LPAR when z/OS is still actively using it, the active task remains dispatched to the logical processor but is effectively suspended because it has no hardware to run on
- Note that with HiperDispatch this generally would only be expected to happen for Vertical Medium and Low processors
  - Vertical Highs are quasi-dedicated to the LPAR so if the LPAR's time slice ended but still has demand PR/SM would be expected to give the PCP back to the LPAR
- For Vertical Medium and Low processors, the PR/SM dispatch interval is between 12.5 and 25ms (often 12.5)
  - This can be a long time for a task to be involuntarily stranded
  - Worse: VLs might not come back for seconds (or longer) if they get parked
  - Can be especially painful if an important task gets stuck this way!

Warning Track Benefit

- Starting with the zEC12, PR/SM issues a warning track interrupt (WTI) to z/OS that it's about to take away the processor
- z/OS gets a grace period to un-dispatch the running task from the LCP and return the PCP to PR/SM
  - z/OS can then redispatch the task to an active LCP
  - "Successful" Warning Track Interrupt
- If z/OS doesn't return the processor in time, PR/SM takes it anyways
  - "Unsuccessful" Warning Track Interrupt
- Goal is to avoid having work hung on an LCP that's not going to get redispatched for some time

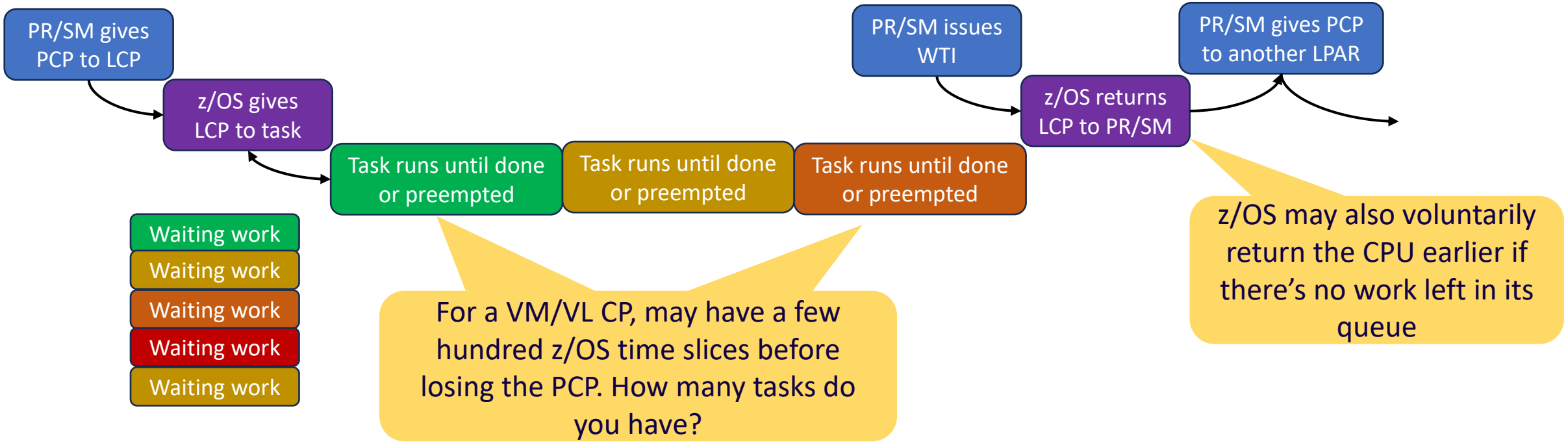
See also "Understanding and Measuring Warning Track on z/OS" at https://www.pivotor.com/content.html

What if z/OS task wasn't done?

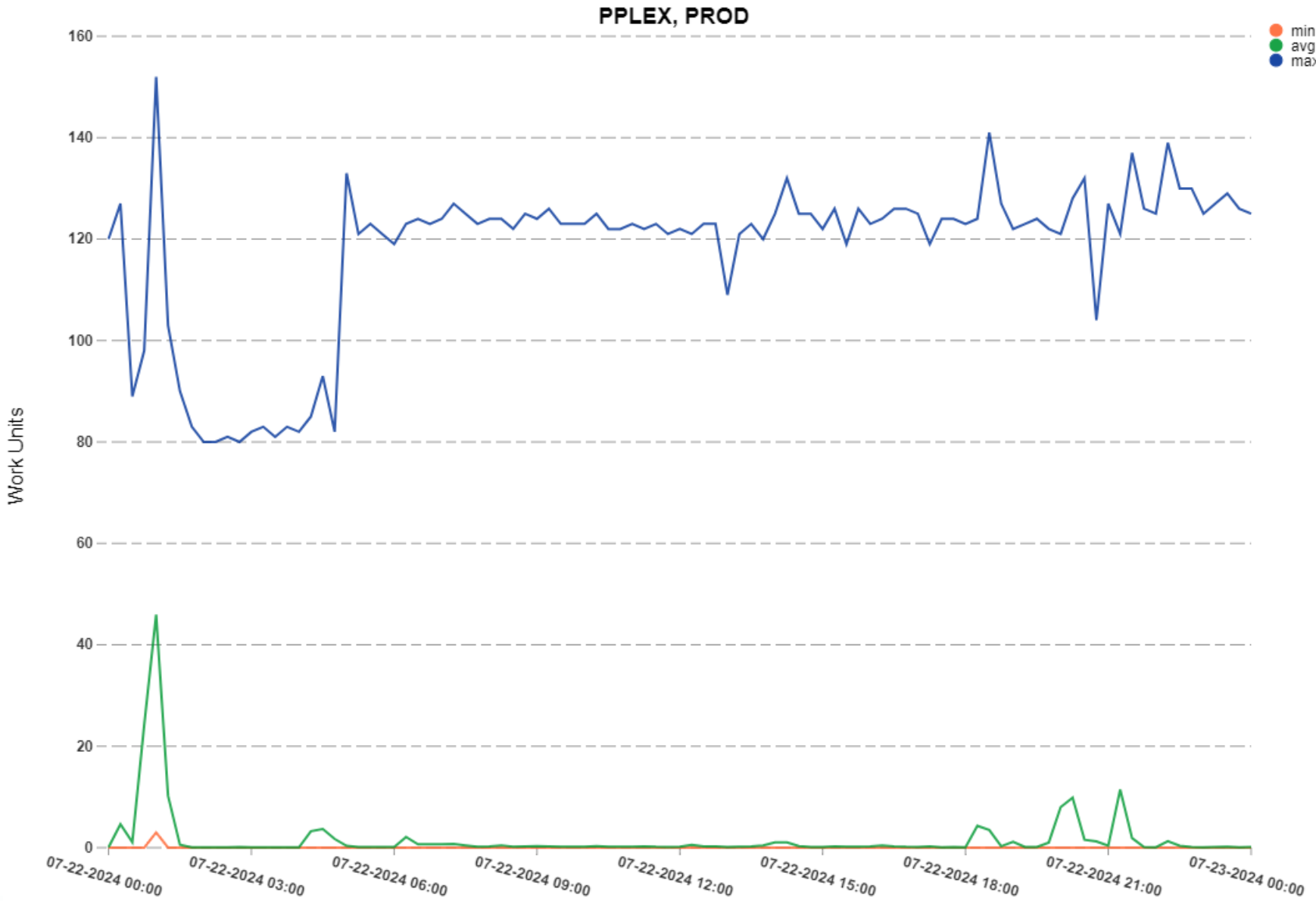


Why are we down in the weeds?

- Sometimes people forget how much is happening within a 900 second interval
- Having a mental model of how much sharing and time slicing is going on can help make sense of unexpected performance numbers



CP Work Units - Min, Avg, Max



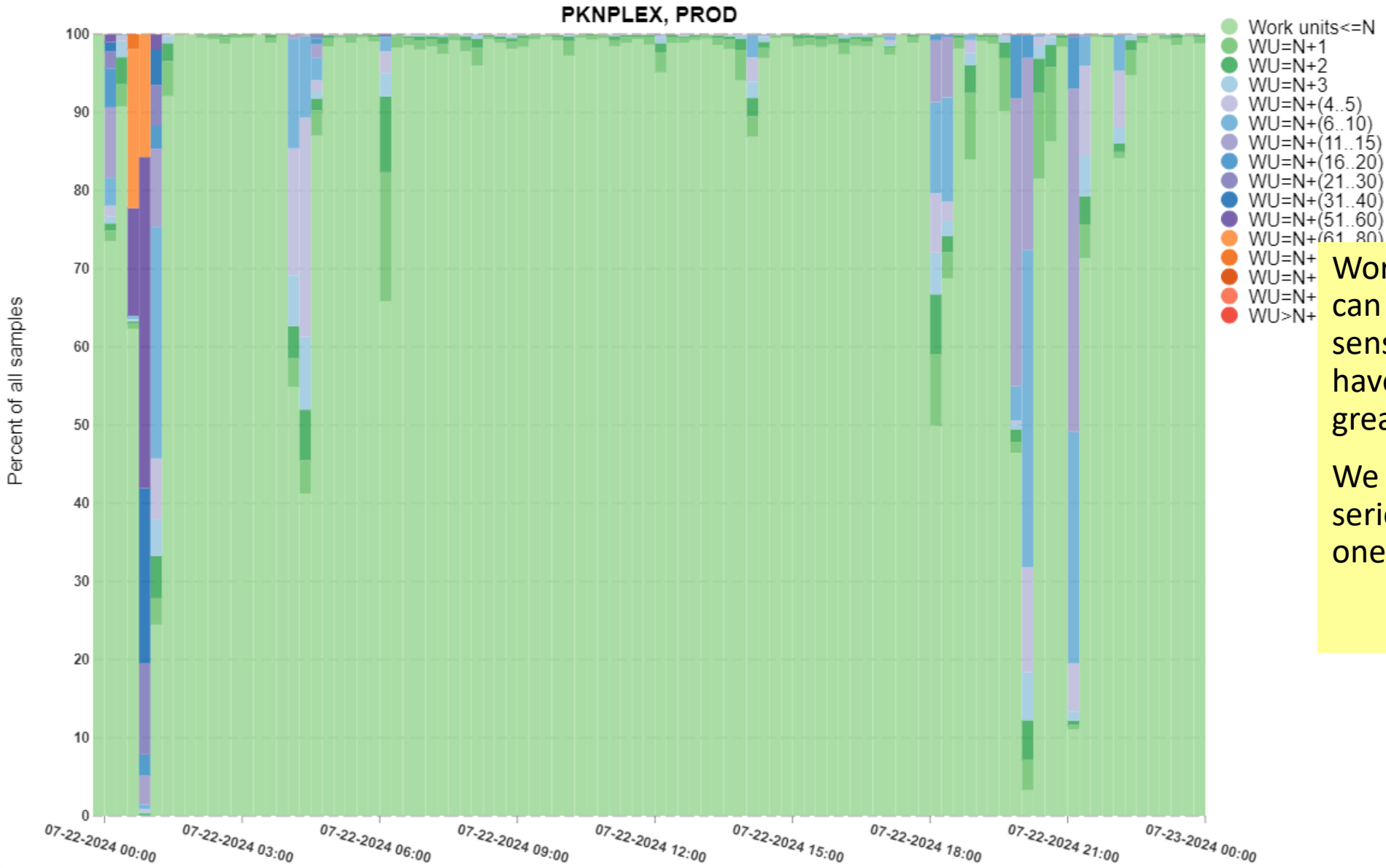
A pretty small production LPAR on a z14 K02.

This is the number of work units running or waiting for a CPU.

Those maximums are not uncommon and likely represents arrival patterns and dispatching waits between LPARs.

CPU Work Unit Distribution

(N = Average of Unparked Engines Regardless of Engine Type)

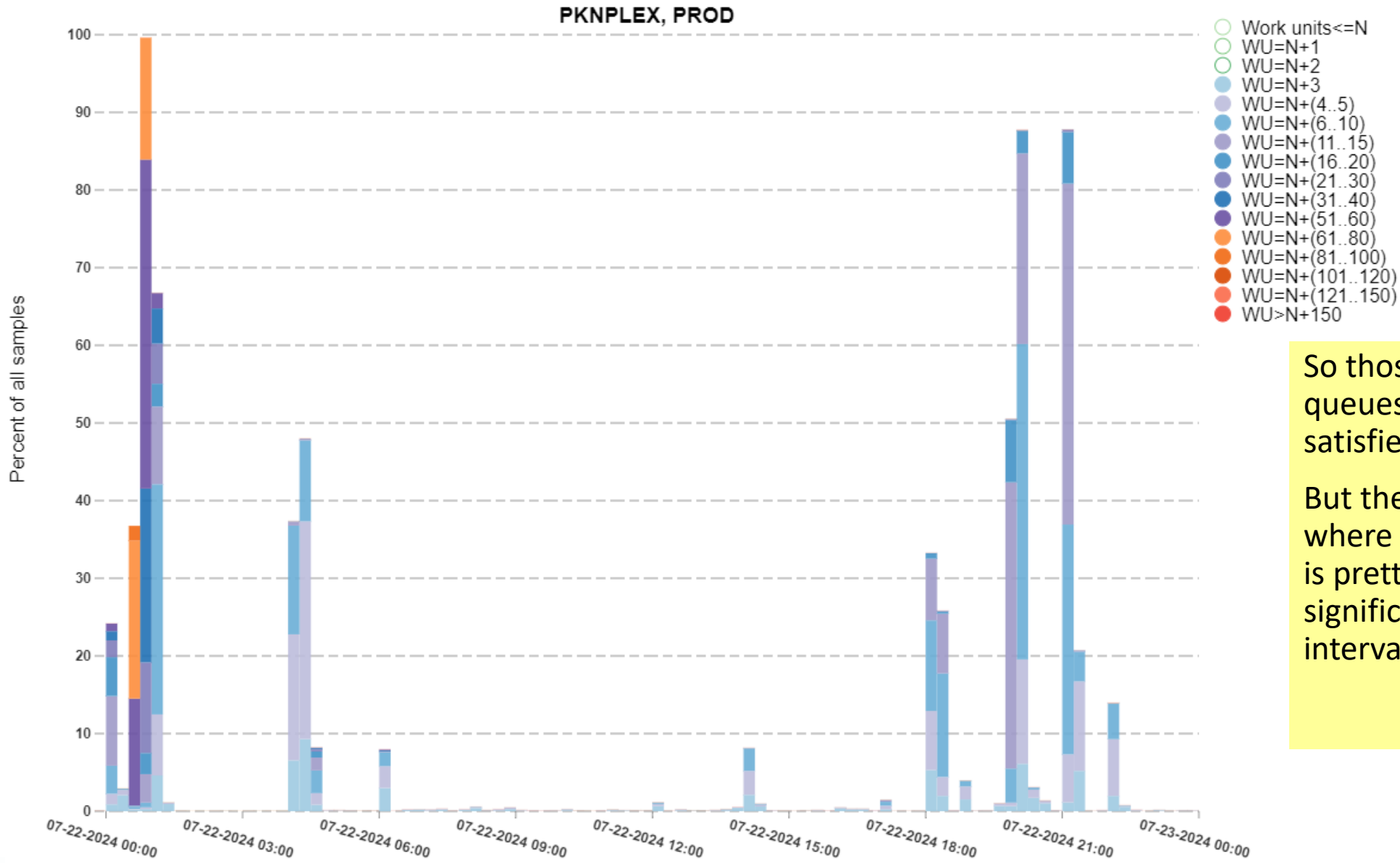


Work unit distribution can give us a better sense of how often we have a queue of how great a depth.

We can turn off some series to just see the ones we want…

CPU Work Unit Distribution

(N = Average of Unparked Engines Regardless of Engine Type)



So those really long queues are usually satisfied quickly.

But there are intervals where the queue depth is pretty significant for a significant portion of the interval.

Measurements

SMF/RMF Interval

- ~90% of the systems we see have 900 second RMF intervals
  - This is fine for most customers, most of the time
  - I might choose to use 300 second intervals today if I was running a z/OS system
  - We don't think 60 second intervals make sense in most cases
    - In part because there are measurements that go below 1 minute that should be collected
    - In environments with lots of "things" (DASD, CF structures, CPs, RCs, etc.) the data volume can get to be significant
      - Most of the time we don't need performance measurements for those things on a minute by minute basis
- SMF interval default is 1800 seconds
  - This is too long!
  - And this should be synced to your RMF interval

```
In SMFPRMxx:

INTVAL(15)  - 15 minute SMF intervals
SYNCVAL(15) - Sync at 15 minutes after hour
```

Some products may require SYNCVAL(59)

```
In ERBRMFxx:
SYNC(SMF) - Sync with and use SMF intervals

In CMFCPMxx:
On REPORT statement:
… SYNC=SMF
```

Key Sync Problem

- There are usually subsystem-specific options in your SMFPRMxx and those need to be set correctly too.
  - Sometimes there will be a different INTERVAL set there or NOINTERVAL
  - Default is NOINTERVAL which (I think) overrides the global interval
  - Easy answer: specify INTERVAL(SMF,SYNC) on the SYS and SUBSYS statements

```
SYNCVAL(00)                    /* SYNCRONIZE ON THE HOUR          */
INTVAL(15)                     /* STANDARD RECORDING INTERVAL     */
…
SYS(TYPE(0:125,127:255),INTERVAL(SMF,SYNC),DETAIL)
SUBSYS(STC,EXITS(IEFU29,IEFU83,IEFU84,IEFUJP,IEFUSI,IEFUSO),INTERVAL(SMF,SYNC))
```

If you aren't syncing your SMF intervals, you won't get new interval records coming in/out of system recovery boost, making those records that include boost periods problematic!

Short-interval records

- SMF 98 and 99 are the primary records for investigating transient performance problems
  - "transient" meaning lasting less than an RMF interval: seconds to minutes
- SMF98 = High Frequency Throughput Statistics
  - Enable with HFTSINTVL(n) in SMFPRMxx
    - Sets HFTS interval, IBM recommendation is 5 seconds, can be up to 60 seconds
  - Additional records (subtype 2) cut with IBM Workload Interaction Correlator
    - Specified with WIC|NOWIC in SMFPRMxx (forces interval to 5 seconds)
- SMF 99 = System Resource Manager Decisions
  - AKA WLM Interval records (but more than WLM today)
- Both have to be selected for recording in SMFPRMxx via TYPE/NOTYPE

Isn’t that a lot of data?

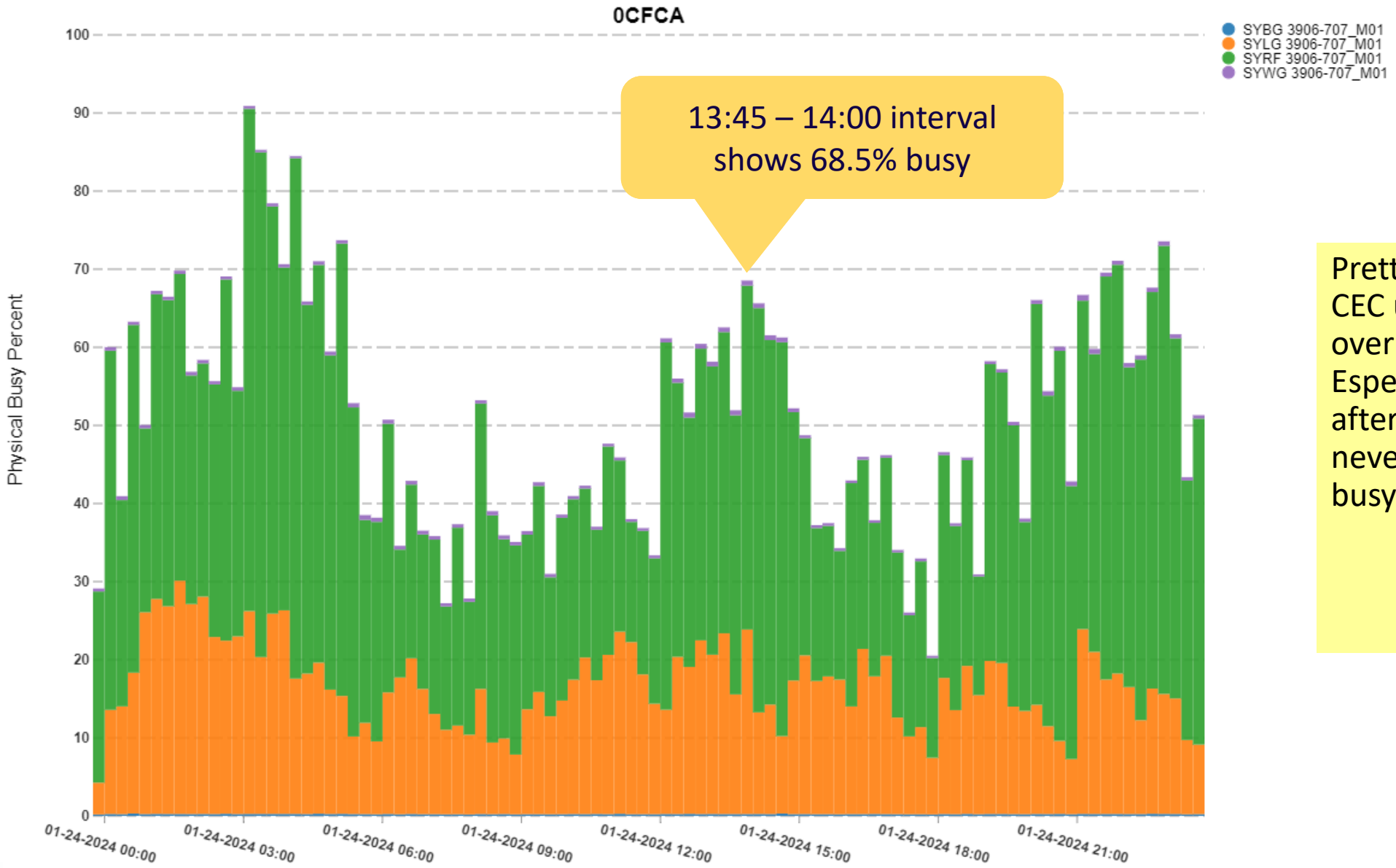
- Have you seen the Db2 records? Or CICS? Or TCPIP? Or…
- SMF 98.1 with 5 second interval is probably < 500MB/system/day
  - 98.2 probably more voluminous but depends on what subsystems are recording to it
    - E.G. CICS metrics every 5 seconds
- SMF 99(6,10,11,12,14) is probably 50 - 150 MB/system/day
  - Dependent on (e.g.) number of active service class periods
- SMF 99(1,2,3) could be 200-400MB/system/day
  - All 99s probably total <500MB/system/day
  - But 99.13 is undocumented IBM only and so we do recommend not recording that unless instructed so by IBM

Interesting stuff in an extra GB of data

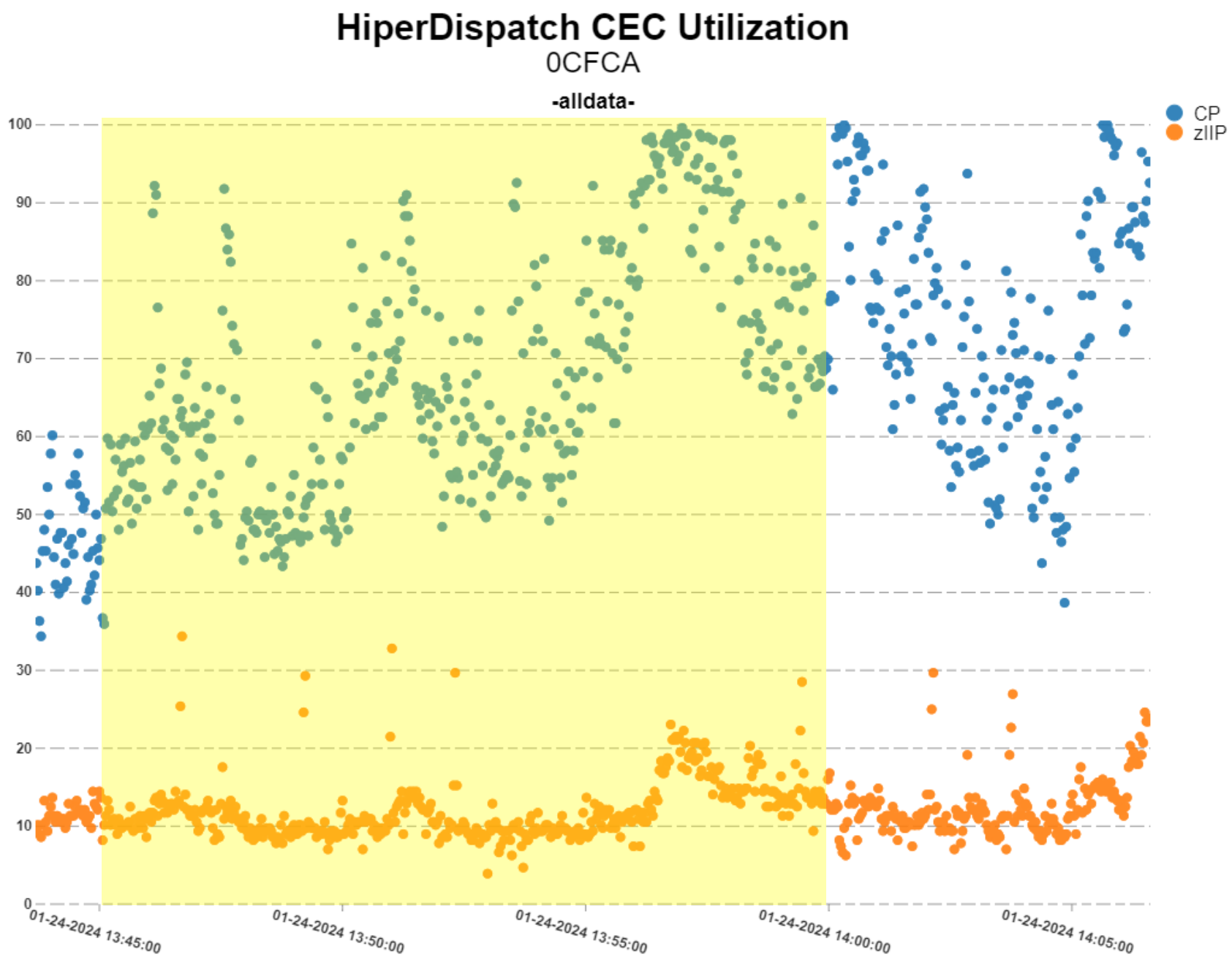
- Every 2 seconds:
  - Overall CEC utilization every 2 seconds
  - MVS Busy at the LPAR level
  - HiperDispatch parking/unparking
- Every 5(?) seconds:
  - CP Utilization by HD pool
  - Work Queue Depths by Priority Bucket Range
  - Dispatch Delays, Preemptions, Spin lock times
- Every 10 seconds:
  - WLM decisions and predictions
  - Service Class Period CPU consumption
  - Service Class Period dispatching priority

These are the highlights, there is other data in the 98s and 99s as well.

CEC Physical Machine CP Busy% by CEC Serial Number

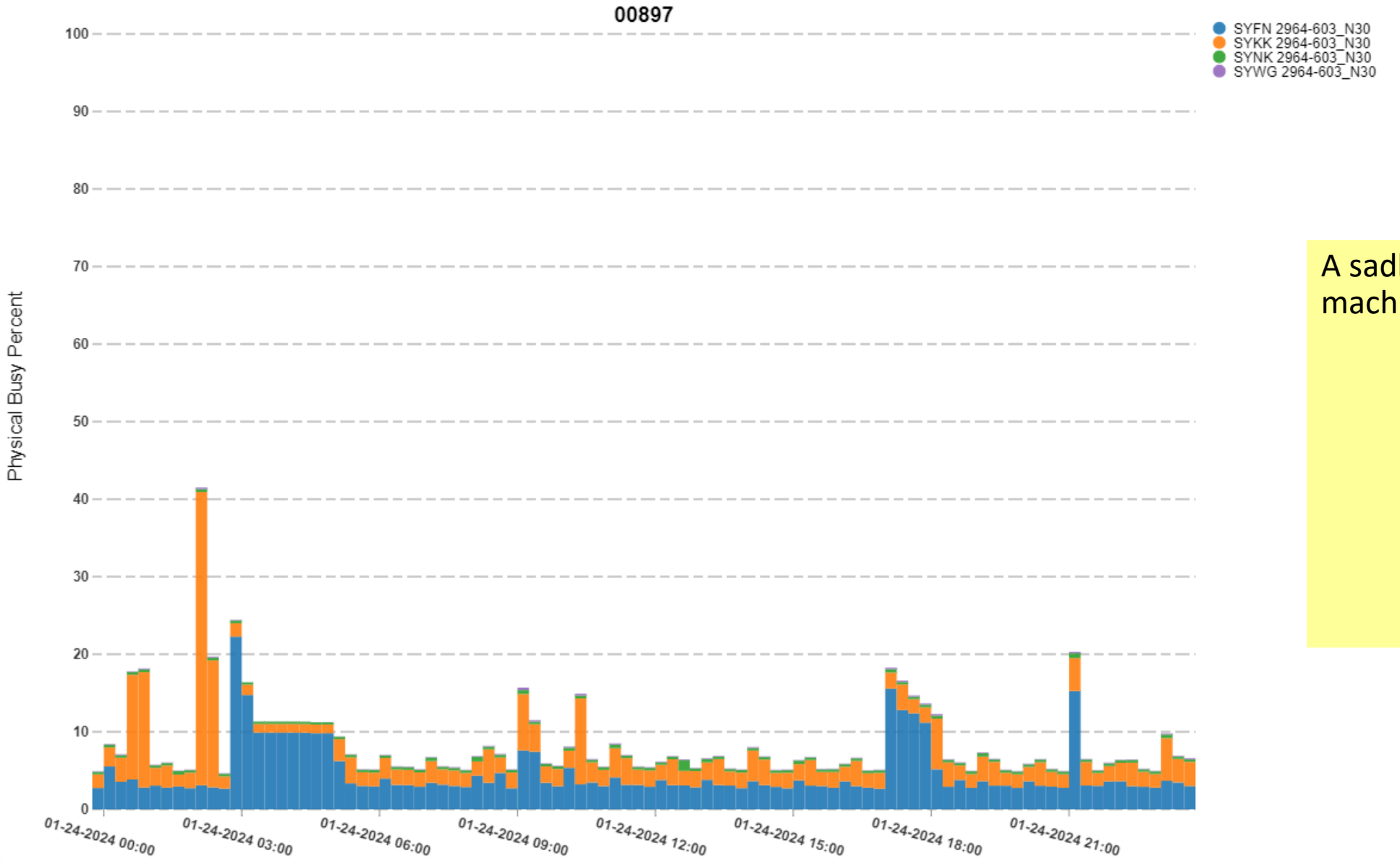


Pretty standard looking CEC utilization for a not overly busy machine. Especially in the afternoon it appears to never be above 70% busy.



But when we zoom into 2 second intervals, we see a completely different story for certain periods of time withing that 15 minute interval.

CEC Physical Machine CP Busy% by CEC Serial Number



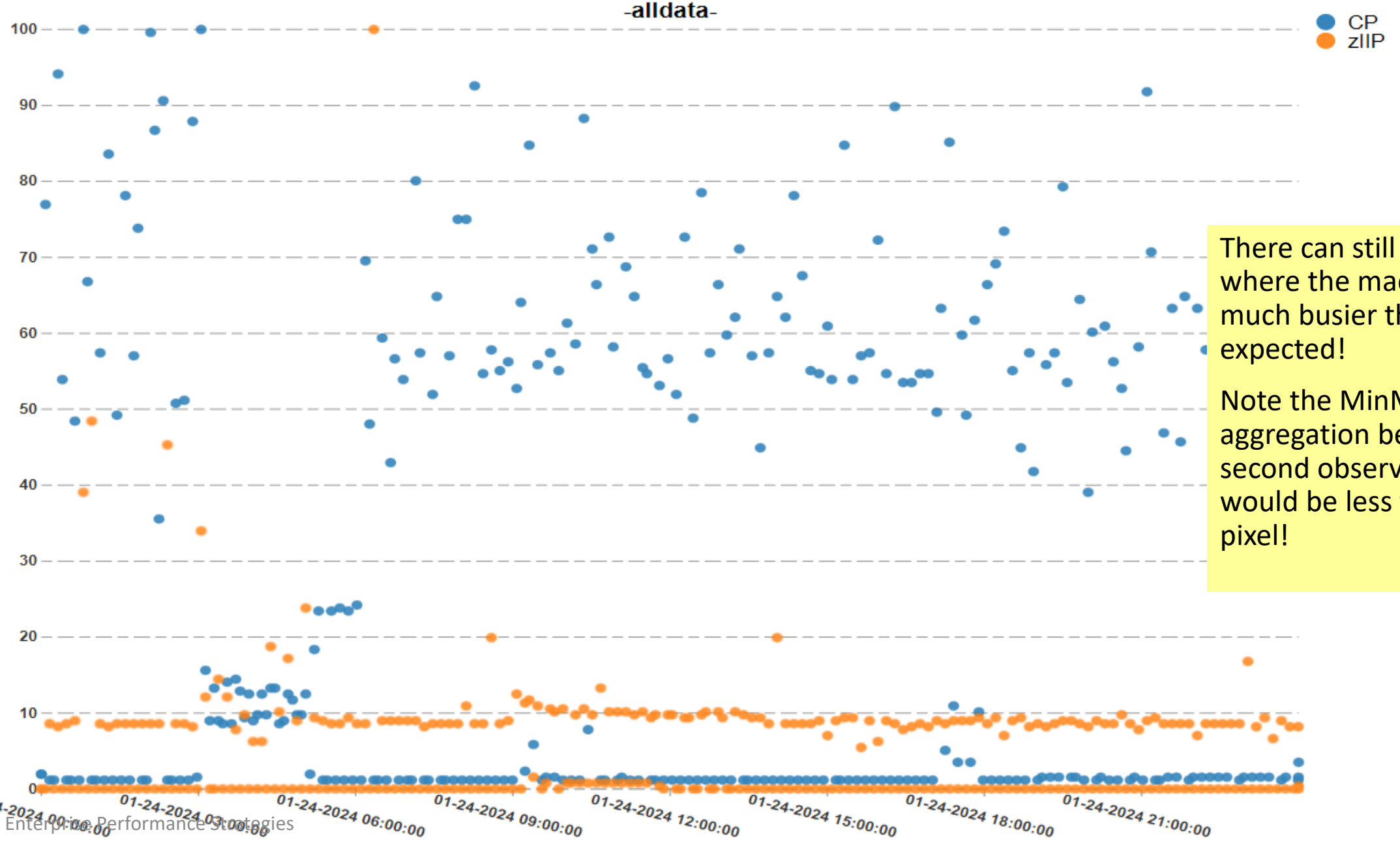
A sadly underutilized machine

HiperDispatch CEC Utilization

00897

-alldata-

MinMax Aggregation Active



There can still be times where the machine is much busier than expected!

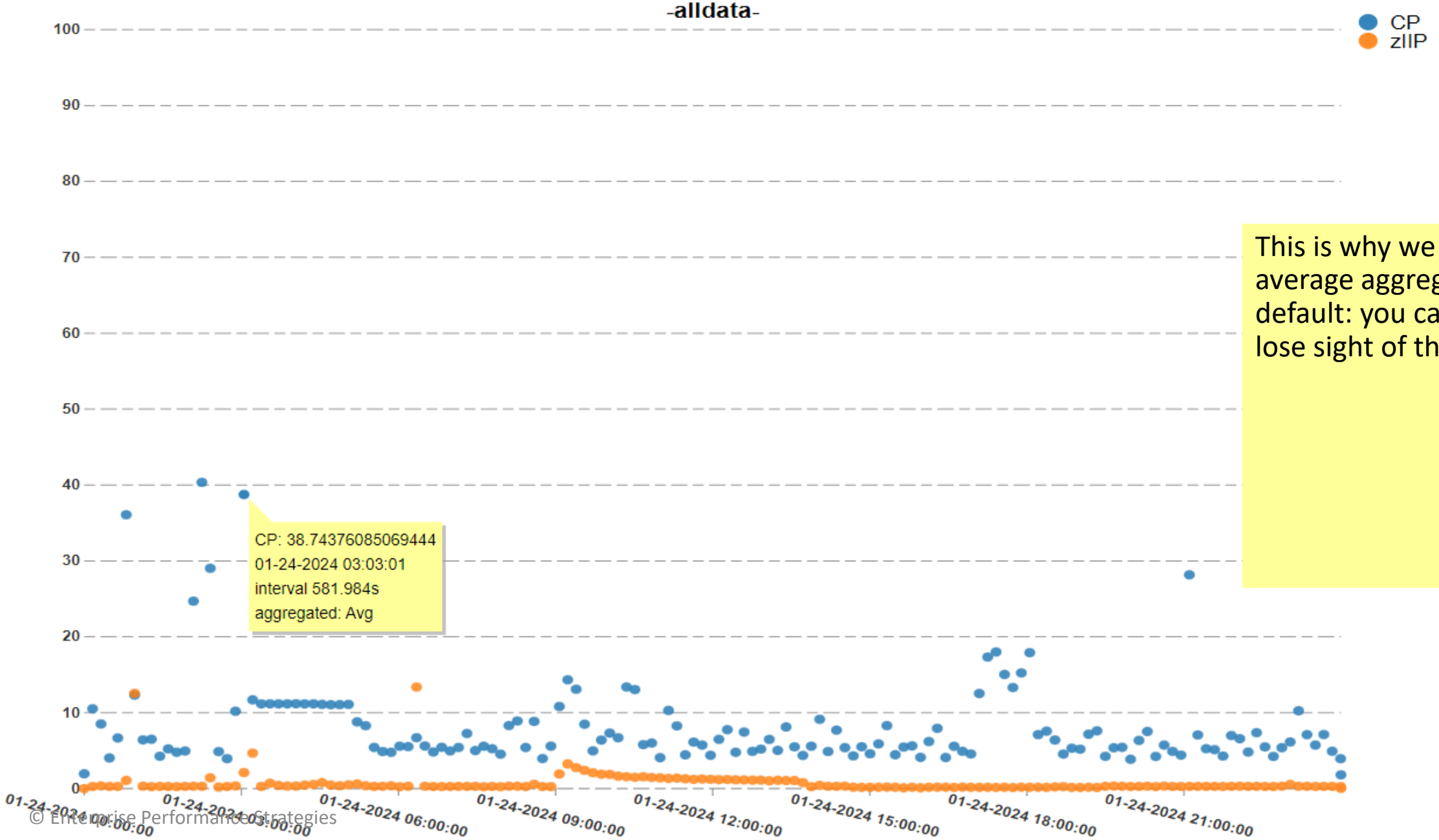
Note the MinMax aggregation because 2 second observations would be less than 1 pixel!

HiperDispatch CEC Utilization

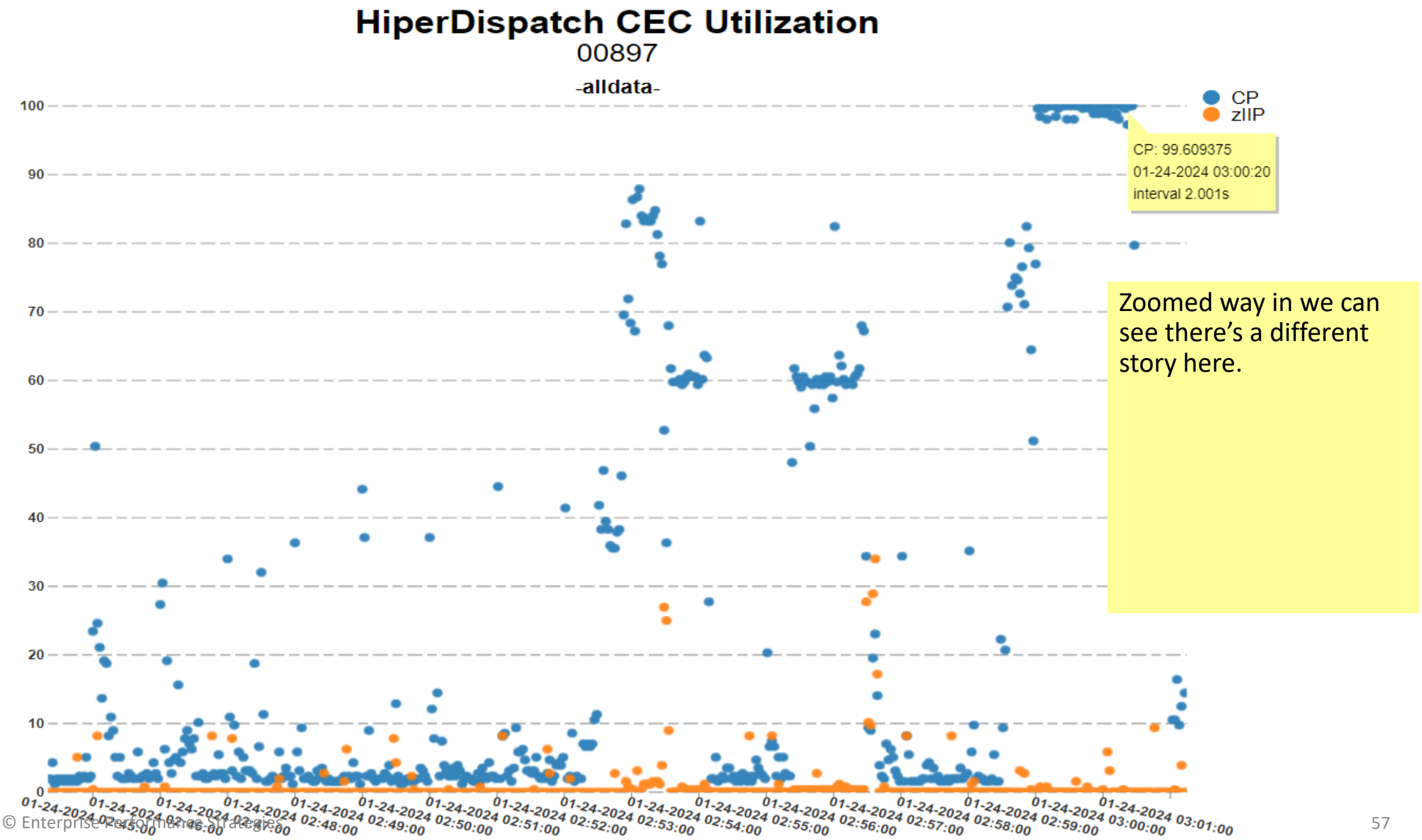
00897

-alldata-

Avg Aggregation Active



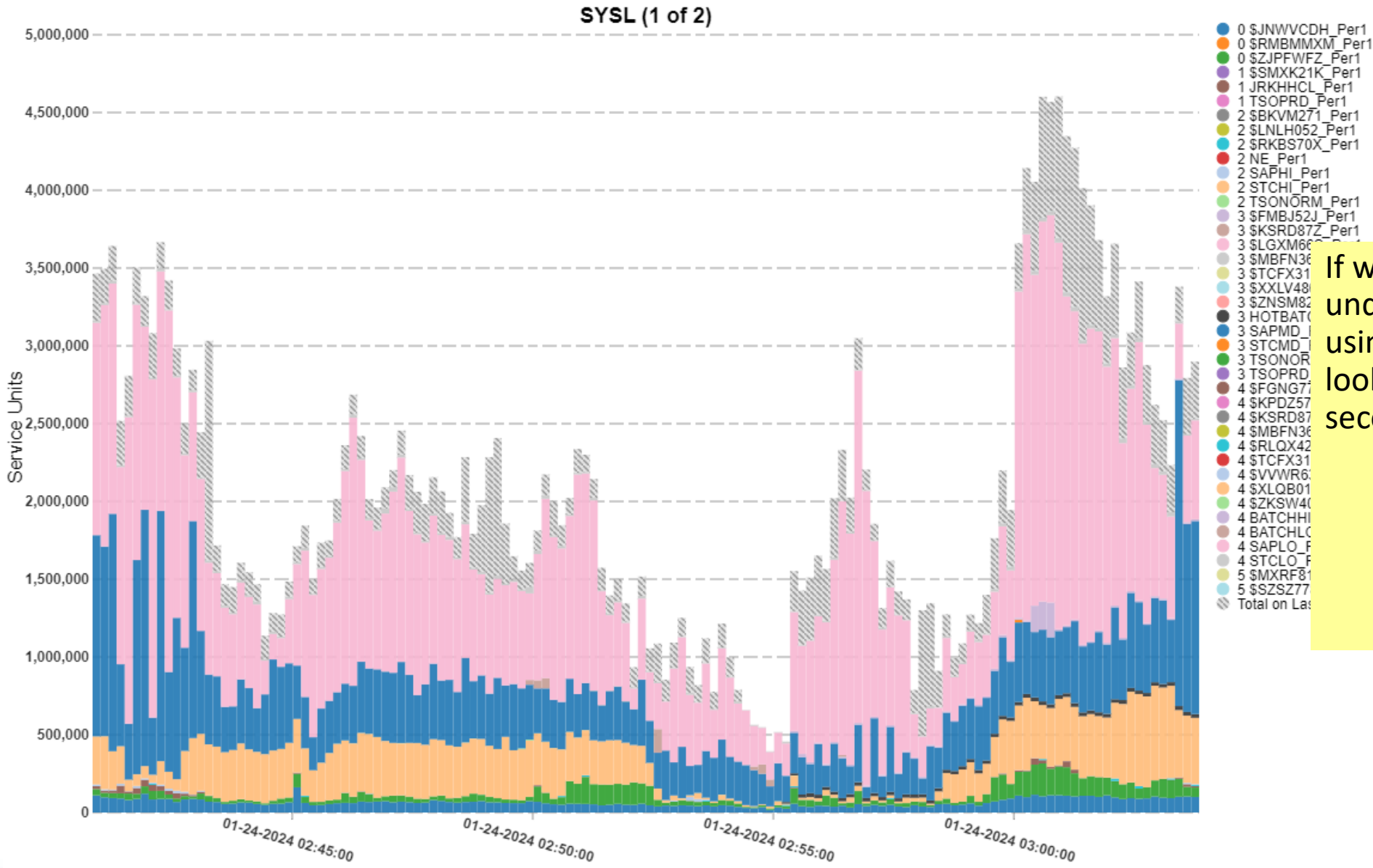
This is why we don't do average aggregation by default: you can easily lose sight of the peaks!



Zoomed way in we can see there's a different story here.

CPU Accumulated by Service Class Period

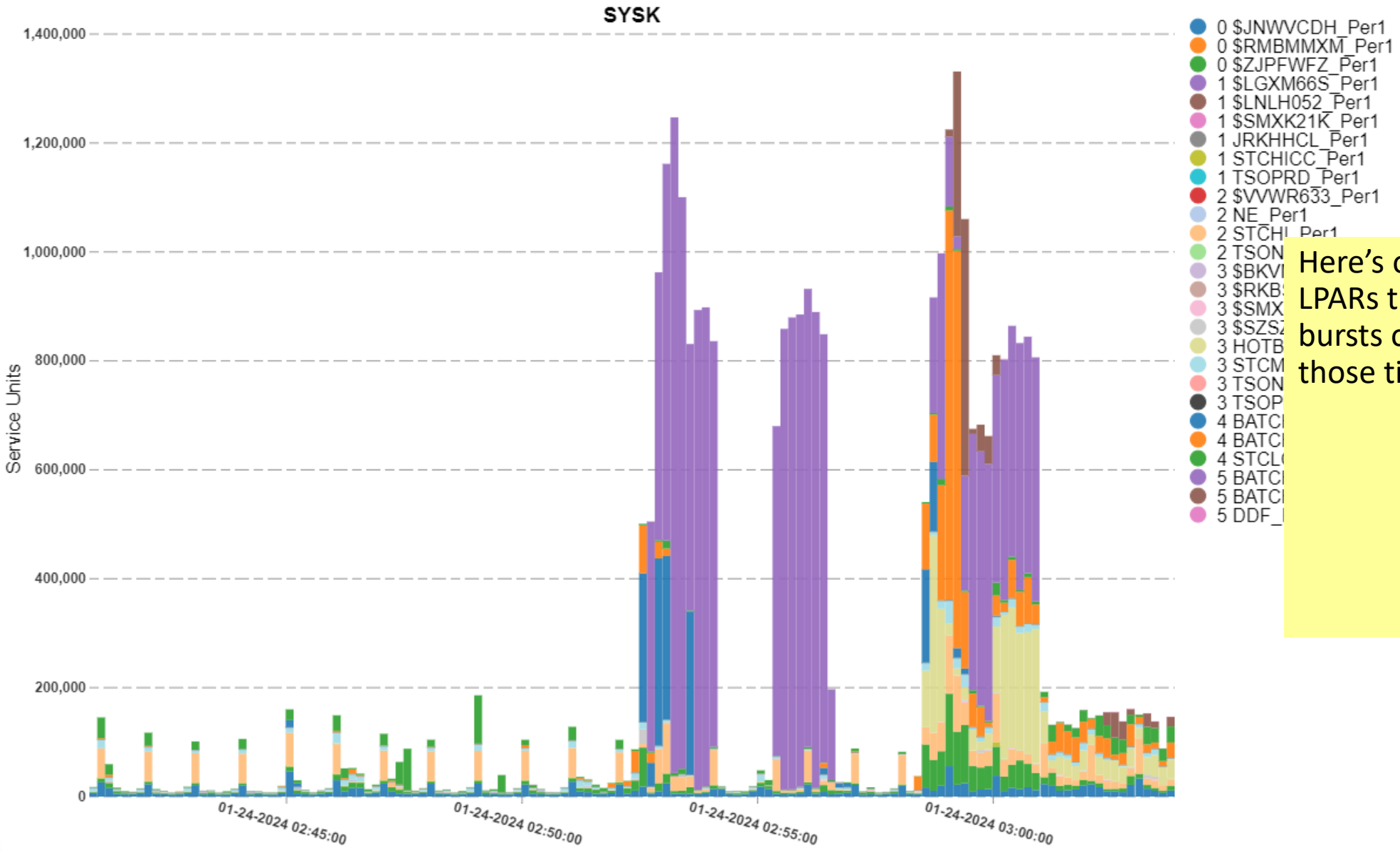
From SMF 99.6



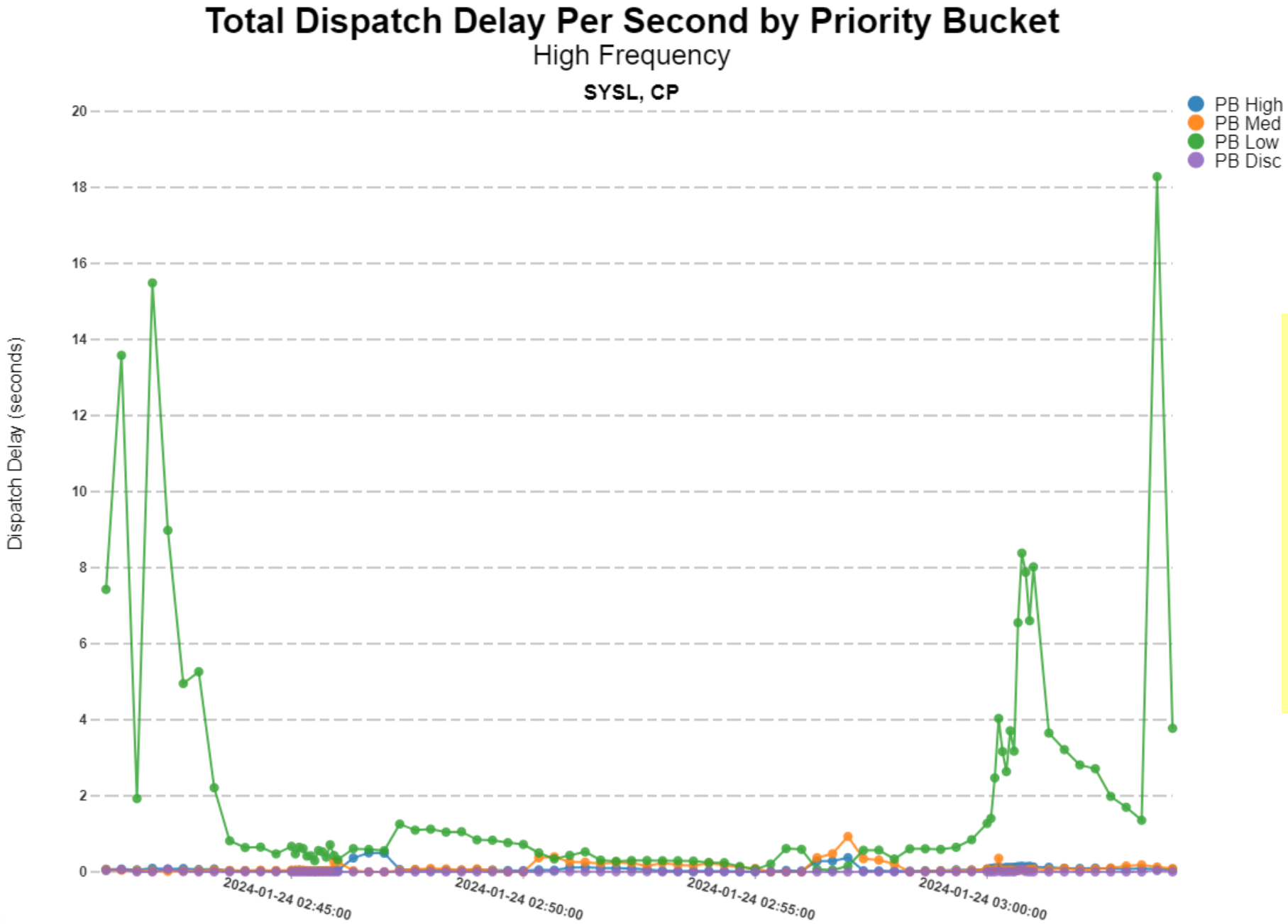
If we want to understand what was using the CPU we can look at the SCPs on a 10 second basis.

CPU Accumulated by Service Class Period

From SMF 99.6



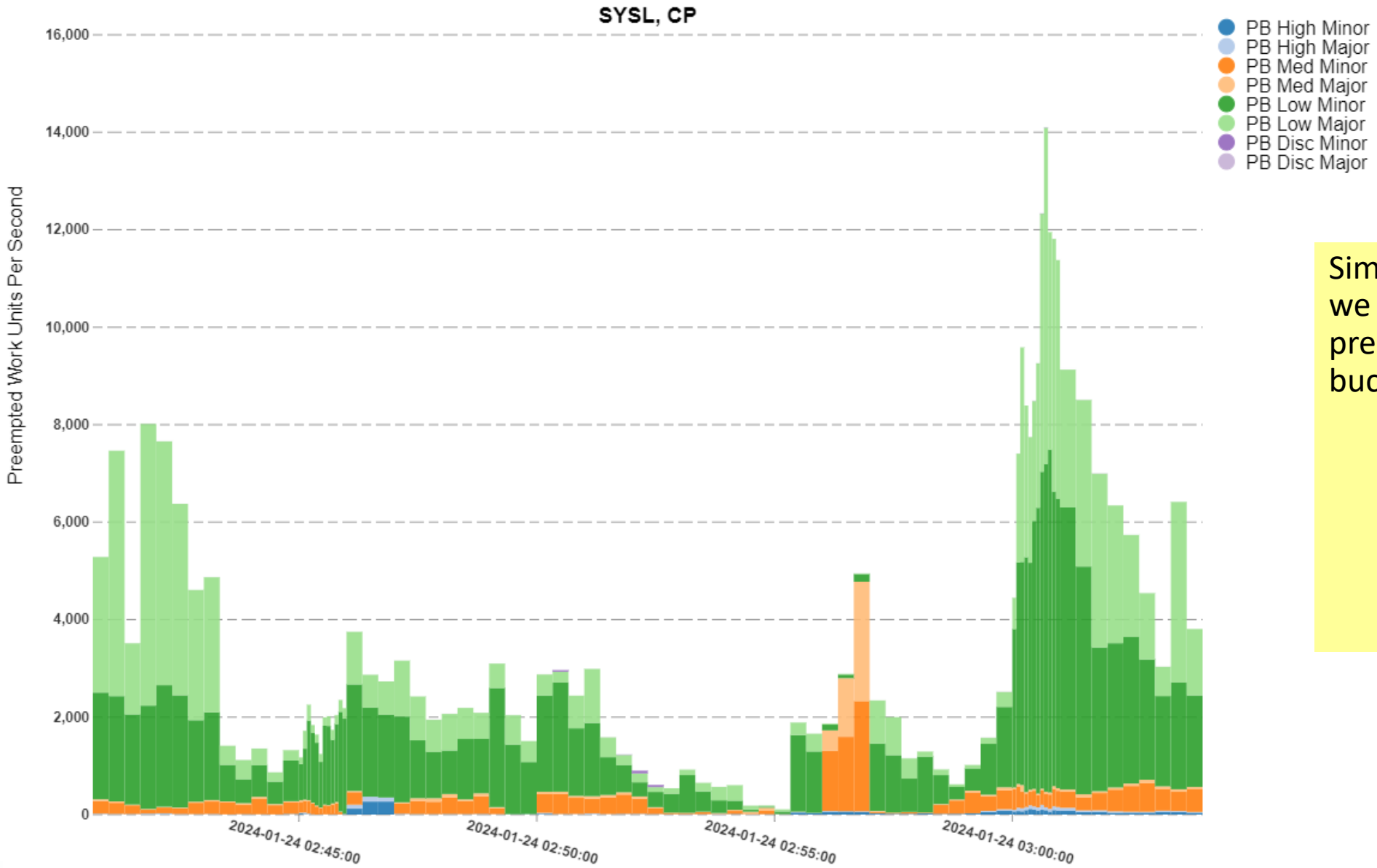
Here's one of the other LPARs that shows some bursts of activity around those times.



The 98s show dispatch delays are primarily for work in the low priority "bucket".

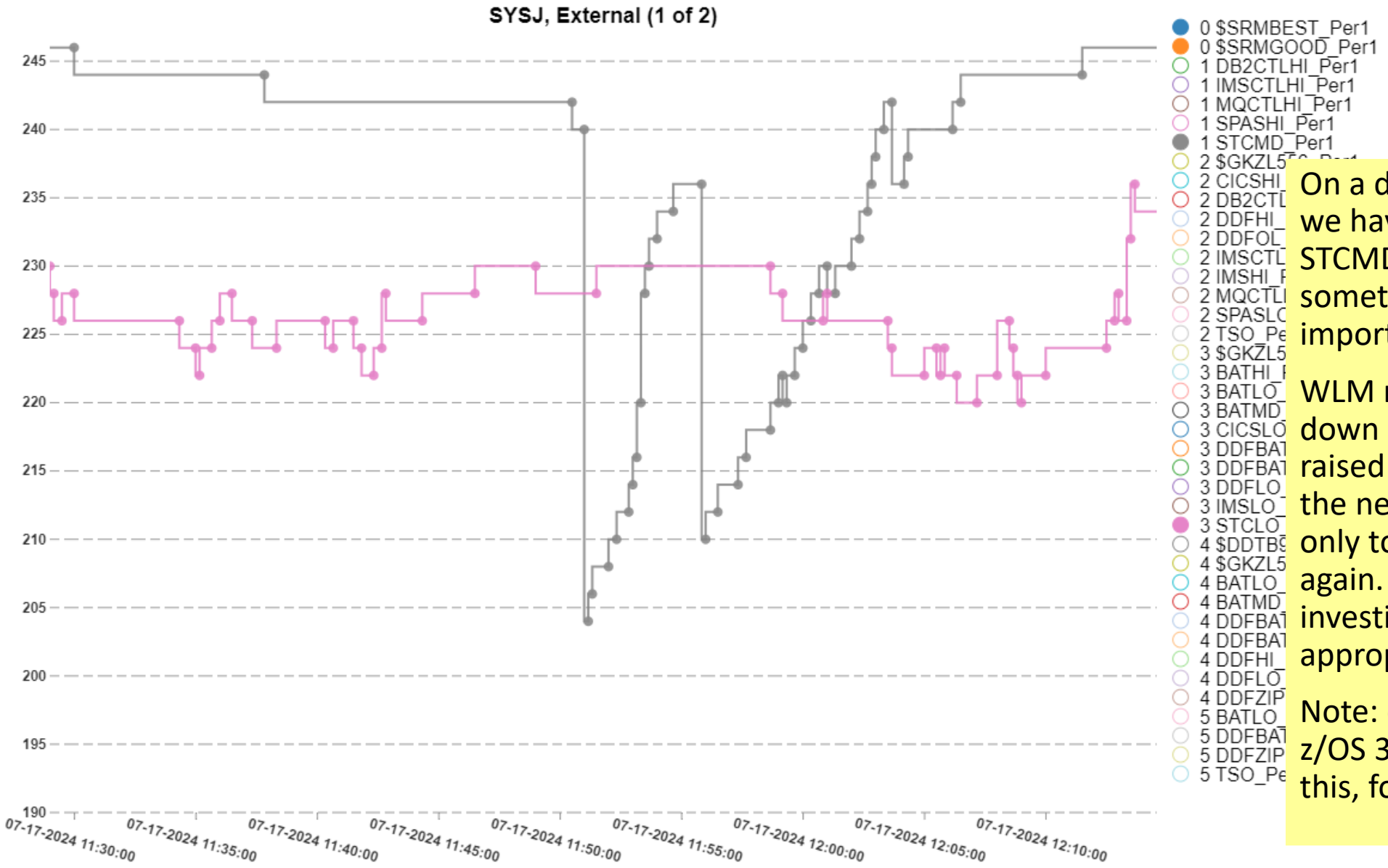
Preemptions per Second by Time Slice

High Frequency



Similarly, from the 98s we can see the preemptions by priority bucket.

WLM SMF 99.6 - CPU Dispatching Priority



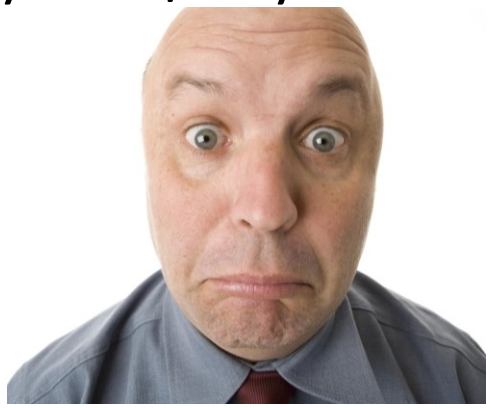
On a different system, we have importance 1 STCMD is running sometimes below importance 3 STCLO.

WLM moved STCMD way down at 11:51:09, then raised it back up over the next ~5 minutes, only to drop it down again. We should investigate if this goal is appropriate.

Note: default settings in z/OS 3.1 will prevent this, for better or worse.

Summary

- A CPU can only be executing 1 work unit on one LPAR at any given moment
- The wide view of 15-minute intervals is fine most of the time
- Remembering that things are happening on much shorter timescales can help explain some performance puzzles
  - You may be much busier than you expect on shorter than 15 minute timescales!
- The SMF 98 and 99 records can help you dig deeper to sub-minute intervals
  - If your mainframe can't handle recording another 1GB of data / system / day …



What Copilot thinks should go here

Don’t forget Your Session Evaluations!!

Questions?