Datacenter Computing @Microsoft

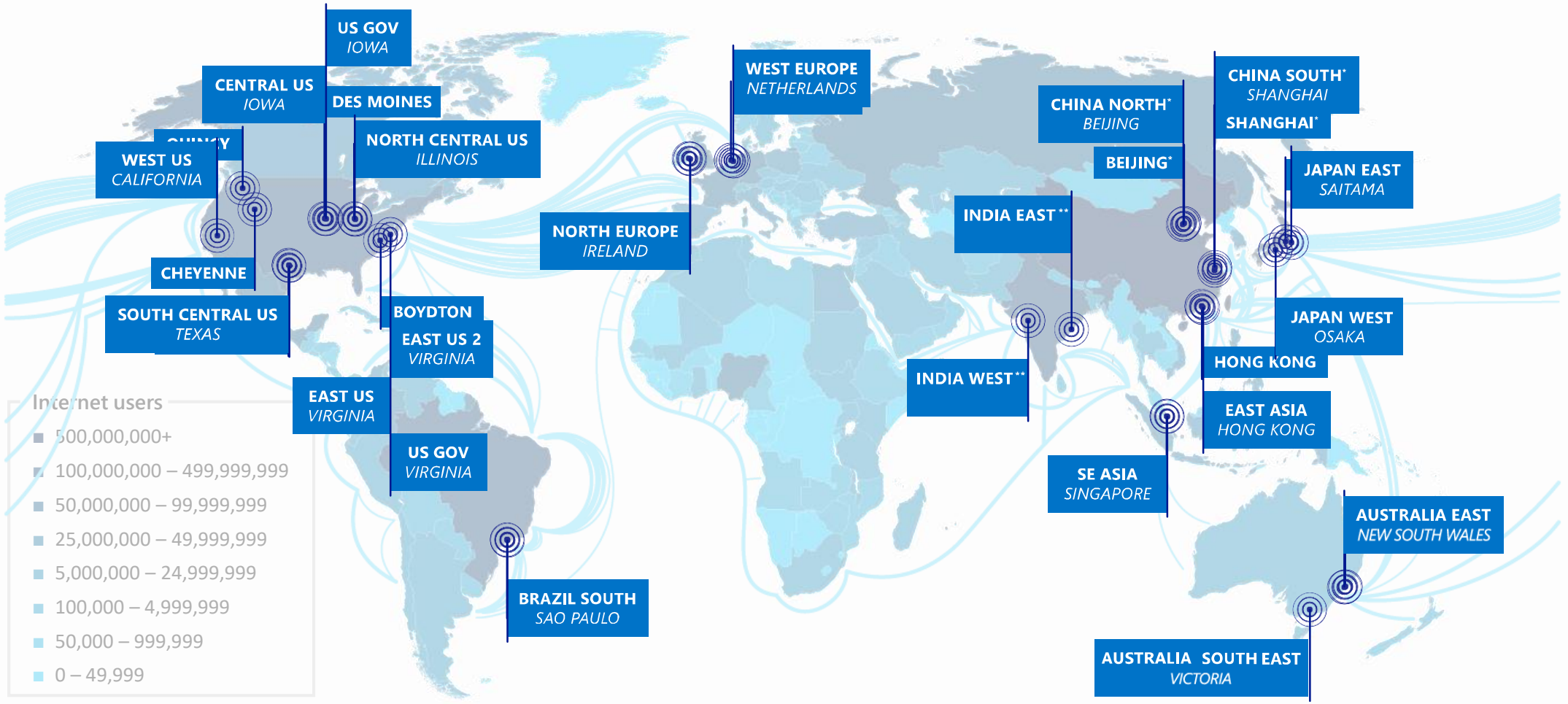
David Levinthal

Types of data center computing are diverse

- **Azure compute (classic concept of cloud computing)**
- **Exchange**
- **Data analytics (aka Cosmos or Azure Datalake)**
- **Azure storage (has distinct properties from Azure Compute)**
- **Bing (Search)**
- **Web Crawling (building search index updates)**
- **SQL**
- **Machine learning (Cortana)**
- **Azure HPC**

Microsoft's global datacenter footprint

Microsoft's network is one of the two largest in the world



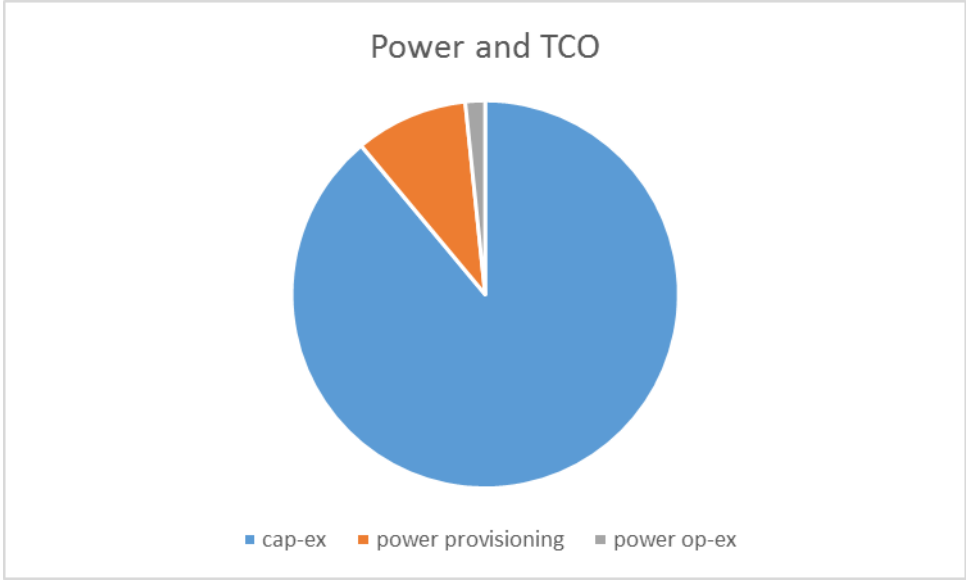
*Operated by 21Vianet

** Announced/Not Operational

1 million+ servers • 100+ Datacenters in over 40 countries

TCO and datacenter planning

- TCO modelling needs to include everything
  - Share of rack, network, management infrastructure
- Power provisioning (~$12/watt with 15 year amortization) dominates real power cost
  - Power provisioning has to be decided years before blades are designed
- If actual demand/utilization exceeds planned power & cooling capacity then power and cooling become dominant constraints



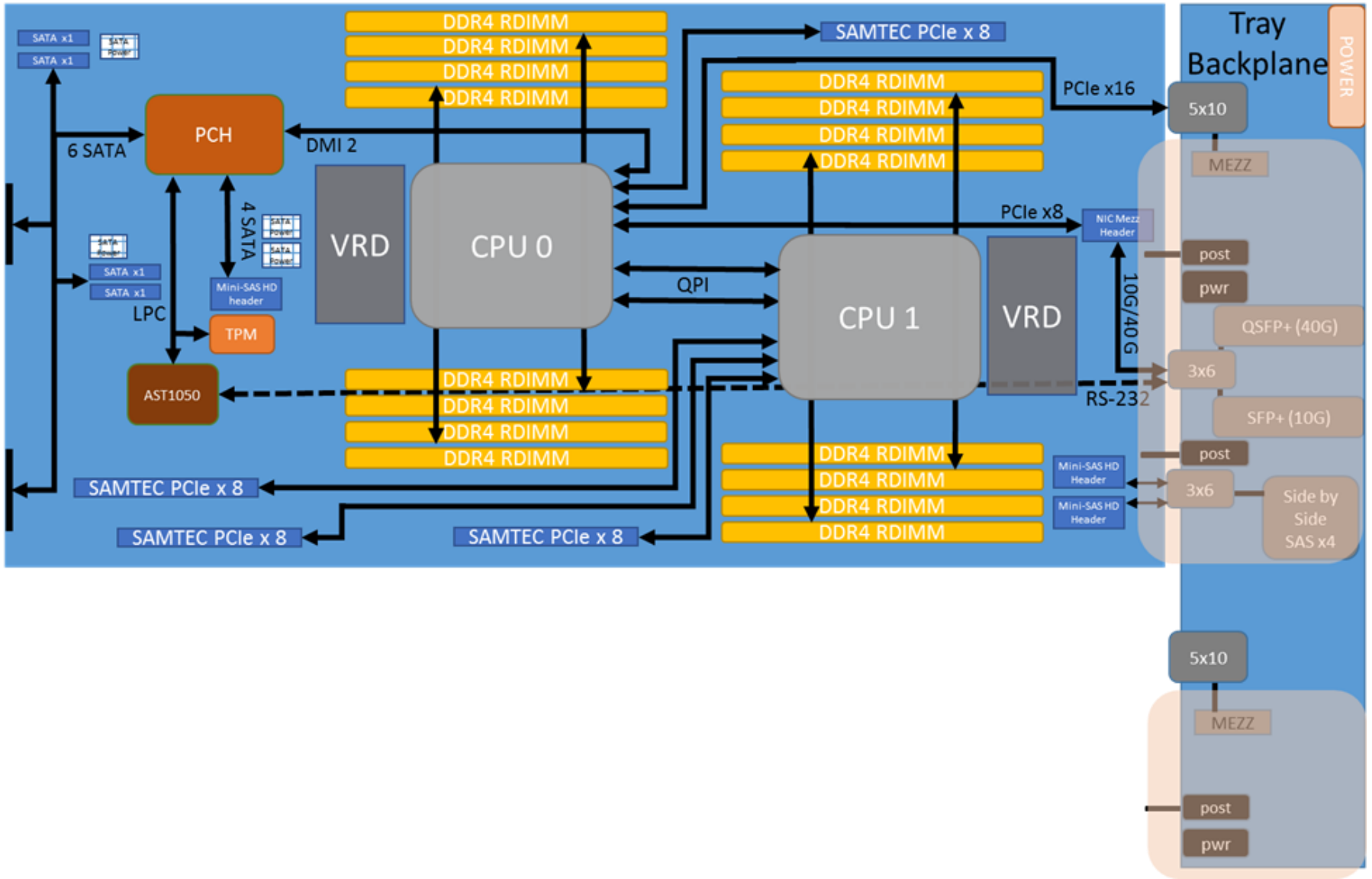
Azure-CSI Performance team objectives

- **Usage performance sensitivities addressed in next generation designs**
- **Base component performance validation**
  - **early stepping silicon testing**
  - **First pass component testing (dimms, storage devices)**
- **Configuration tuning**
  - **Components and Bios settings**
- **In band system health and performance monitoring at scale**
- **Early testing at mini cluster scale**
  - **~100-200 machines tune apps/schedulers**
- **Application and infrastructure enabling in production**

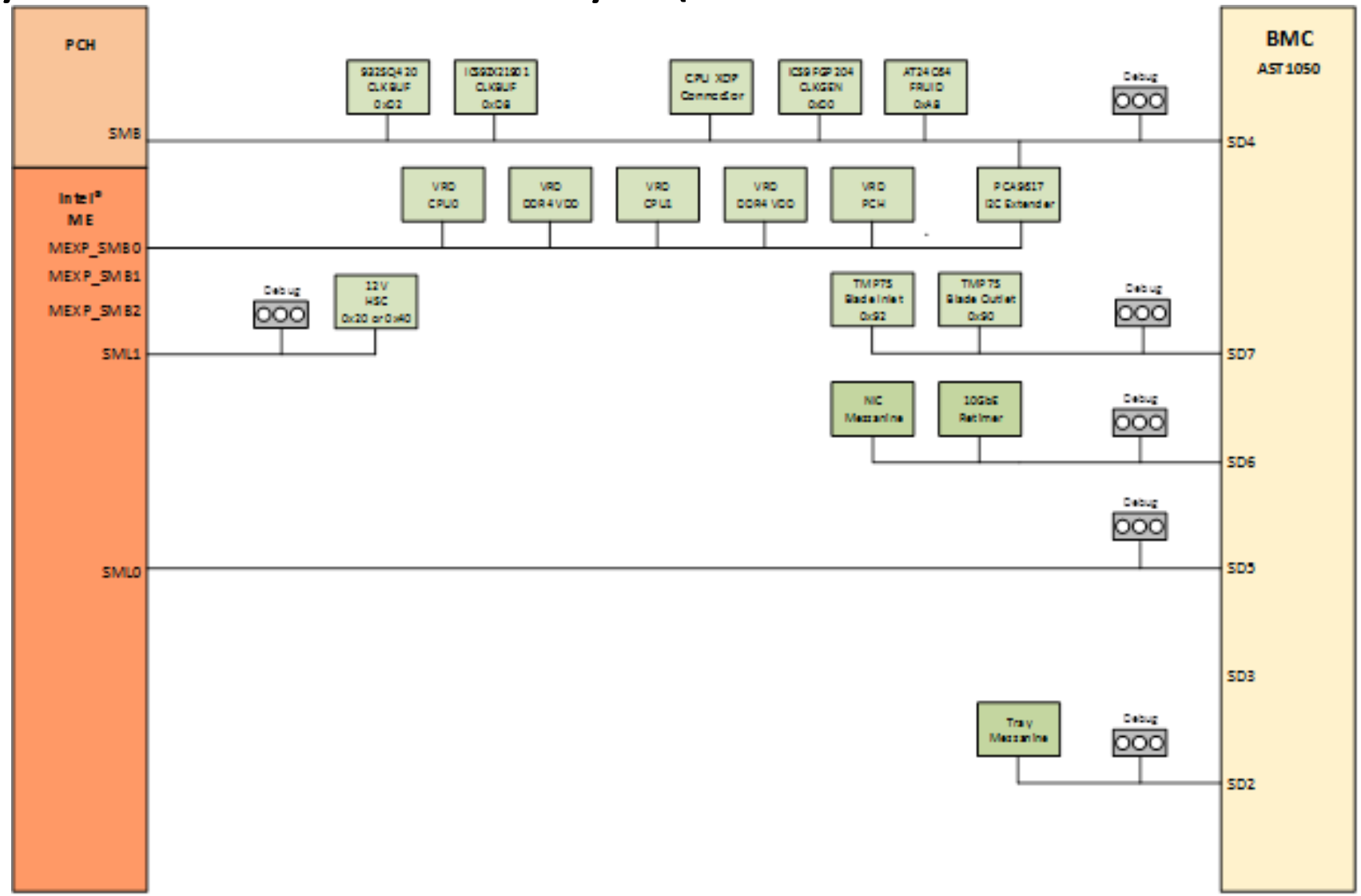
What is performance work at scale?

- **System health/usage telemetry at scale**
- **Characterization by workload/sector to platform performance limits**
  - **Feeds tuning and next generation design considerations**
- **Platform configuration tuning by workload/sector**
- **Future component feature tuning**
- **Early platform performance debug**
- **Selected application tuning/feature enabling (major in house users)**
- **Scheduler tuning**
- **HW based PGO/FDO at scale**

Real servers have a lot of components and many have telemetry (OCP standard blade)



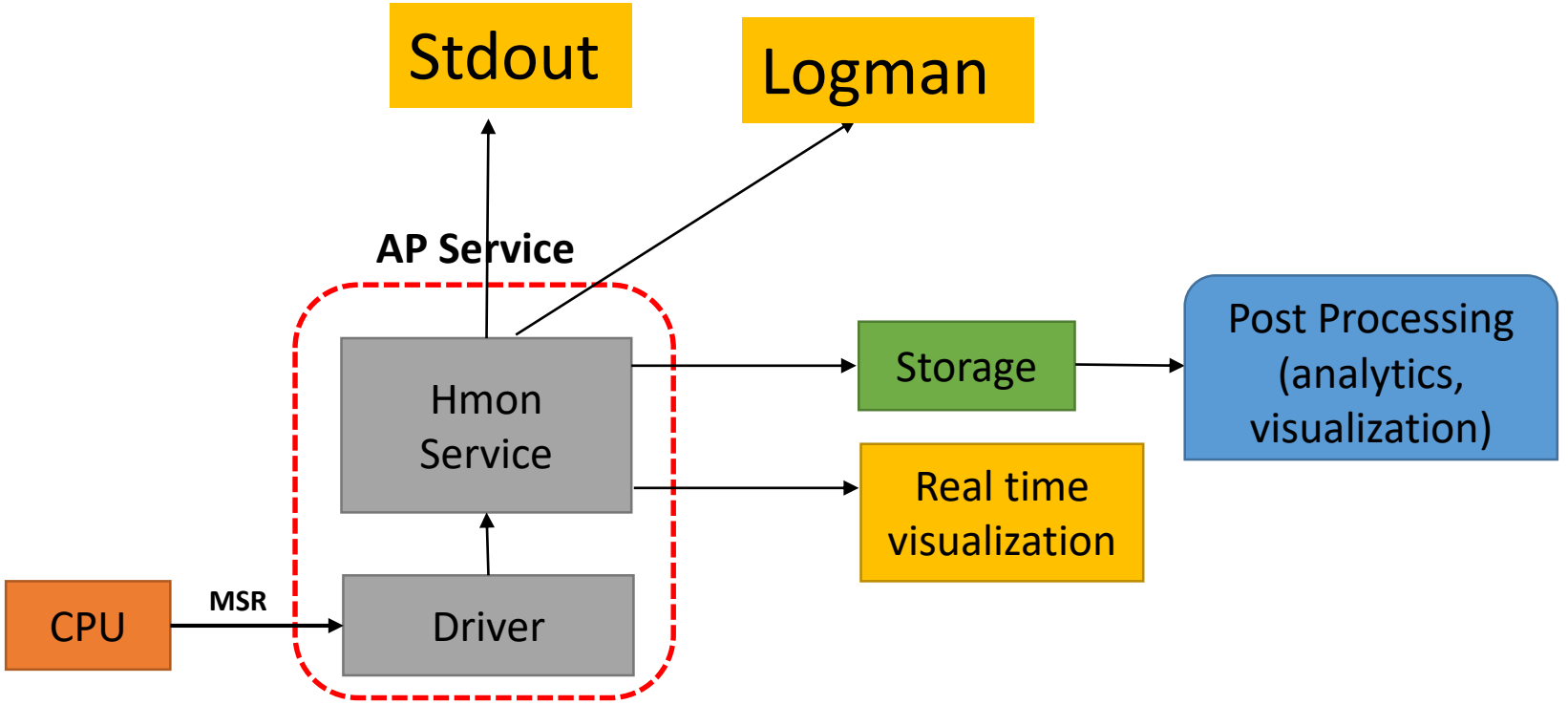
Real servers have a lot of components and many have telemetry (BMC I2C connections)



Sources of telemetry

- **CPU**
  - **MSRs and CSRs collect temp, power, freq., FW ver., errors, controls, perf.,PPIN**
  - **Requires (at least) 2 different drivers (MSR, PCI config, MMIO?)**
- **BMC through IPMI**
  - **SEL, I2C sensors (air temp, platform power, etc) FW ver., dimm serial #**
- **SSDs and Disks**
  - **Smartdata, FW ver., manufacturer, model, serial #**
- **PCIe add in cards (NIC, FPGA)**
  - **Card specific usage and performance, fw ver., serial #**
- **Perfmon: OS level performance data and sensors know to the OS**
- **WMIC: OS cached system configuration**

Example Service Architecture



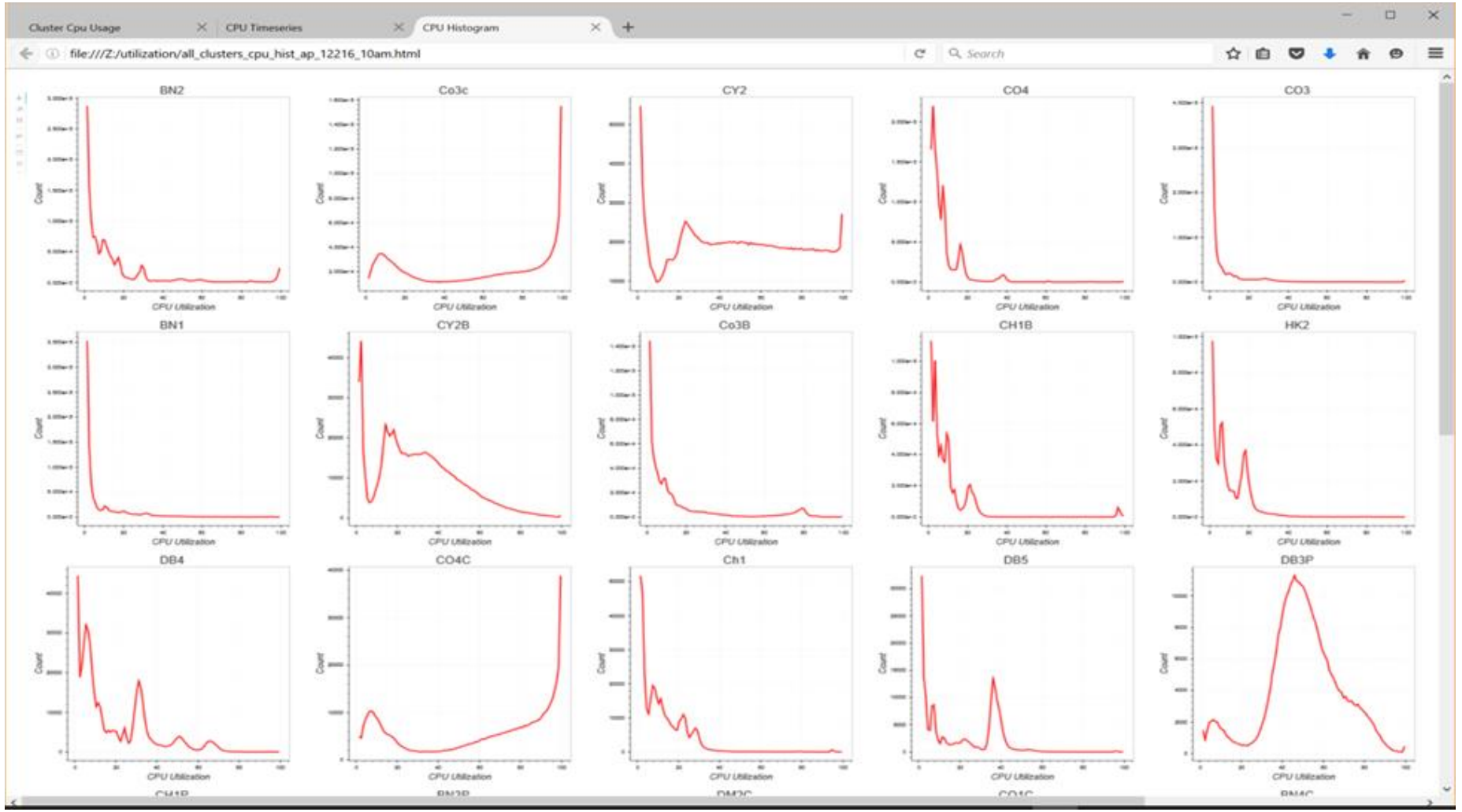
Service Design Objectives

- **Service is extensible to future CPU platforms**
  - **Config. files driven (msr list, def, and metric def)**
    - **Msr list defined per architecture in config file**
      - **Adding MSRs requires no code changes**
    - **Msr file name associated with family model through config file**
      - **Adding an architecture requires no code change**
    - **Derived metrics are defined through a config file**
      - **Adding new metrics requires no code changes**
- **Light-weight (< 1% CPU overhead)**

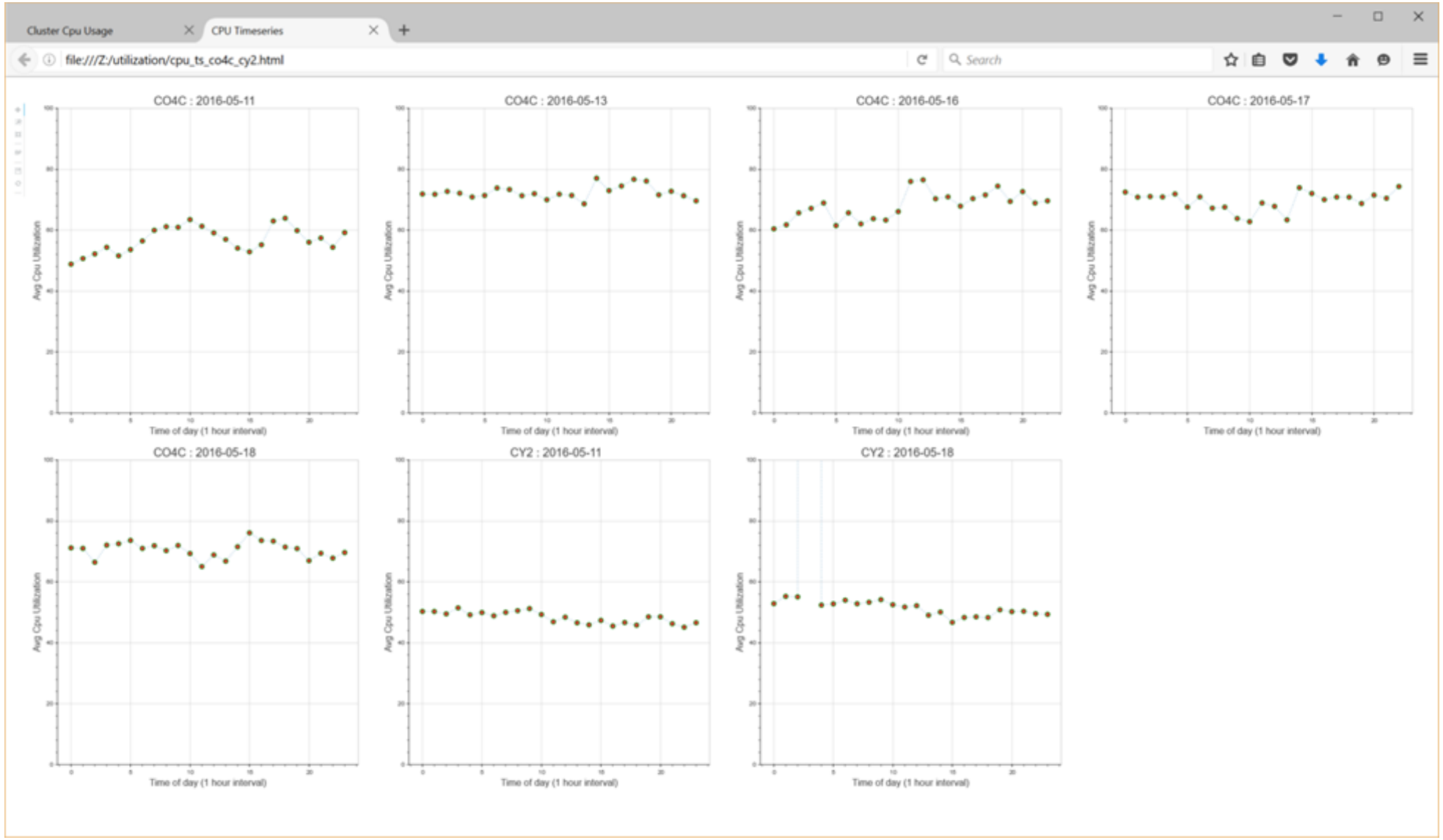
Examples of desired data

infrequent collection		Frequent collection	
freq cap/core	**thread count**	**TSC**	**Mperf**
Turbo setting + other	**uncore freq cap**	**Core C3 residency**	**Pkg RAPL Status**
Turbo curve	**Thermal interupt thresholds**	**Core C6 residency**	**Current Uncore Freq**
Turbo curve	**Thermal interupt offset**	**Pkg C3 residency**	**Package Energy Counter**
Turbo curve	**Thermal interupt control**	**Pkg C6 residency**	**DDR Energy**
HW prefetchers	**Package power limit times**	**current Freq**	**Pkg C7 residency**
Microcode signature	**Power/energy/time units**	**Pkg Thermal Status (temp)**	**Core C7 residency**
Processor Inventory number	**Frequency limits/part status**	**Core Thermal Status (temp)**	**Pkg C2**
	HW Energy policy	**Thermal Fan Control**	**SMI since boot**
		Aperf	**Perf Limit Reason**

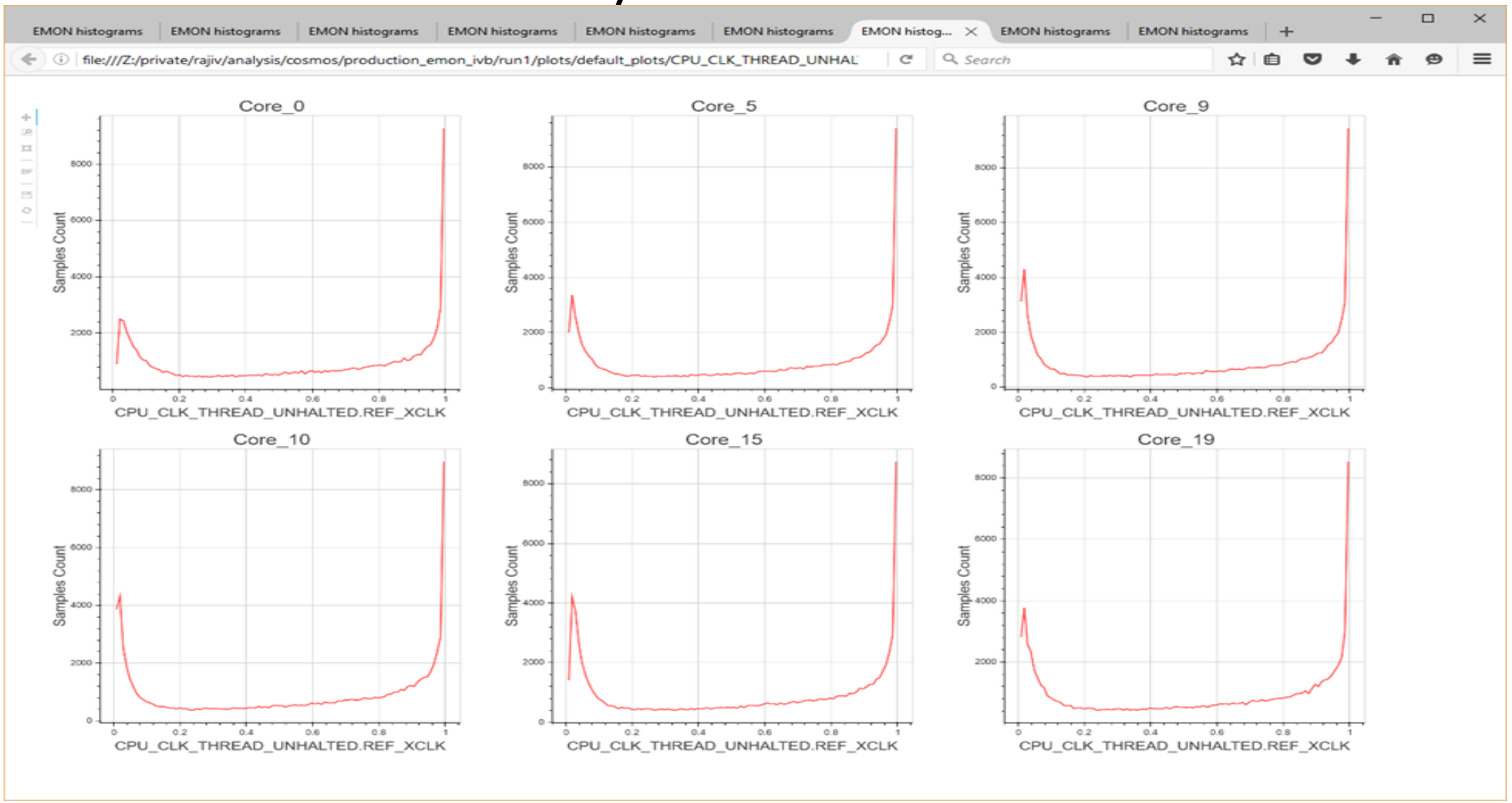
Cluster utilization varies wildly as do the workloads



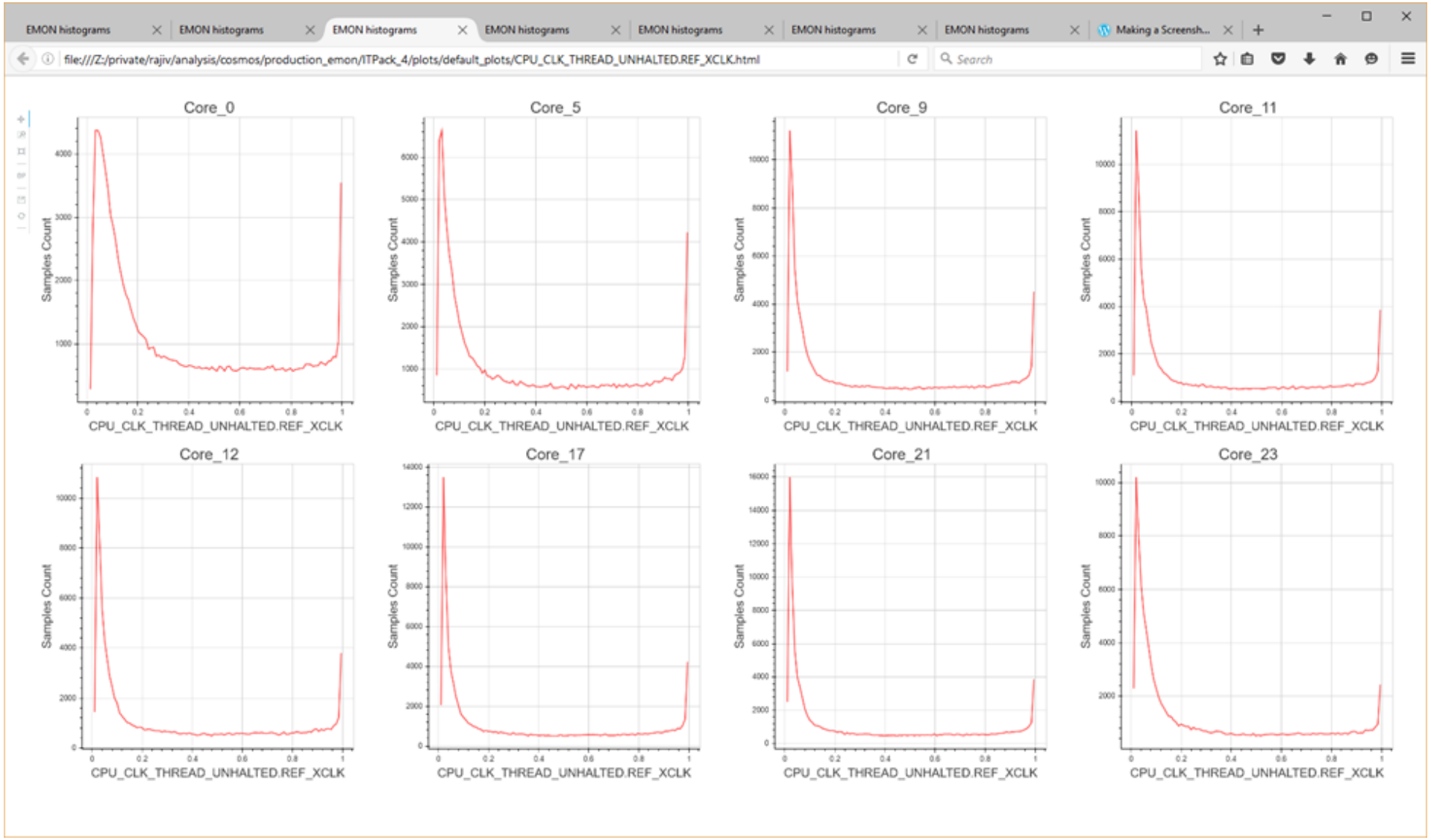
Some clusters work all the time



Some are uniformly loaded



But some are not



Utilization is driven by schedulers

- **Improvements in scheduling add value to a datacenter**
- **Load balancing across machines is not obvious**
  - **Scheduler does not know the application activity level until it is running**
    - **Large numbers of small jobs per machine**
  - **Moving applications across machines is very expensive**
    - **VMs can have hundreds of MBs of local data**
    - **@40 Mbits/sec 500MB takes 100 sec**
    - **There can be a lot of VMs on a large server**
  - **Data analytics schedules the calculation on the machine with the data**
    - **Map reduce**
    - **Movement if not really an option**

Performance@scale
The cloud as statistical ensemble

- **How do you think about millions of machines, distributed worldwide, in large numbers of clusters, running a huge variety of applications?**
- **Proposal:**
  - **Explore distributions of hierarchical cycle decomposition**
    - **A machine at a given time is at a point in a multidimensional space**
    - **Can the cloud be described as a density function in that space?**
    - **Is the distribution stable over time?**

Hierarchical cycle accounting

- **Decompose cycles into processor independent categories**
  - **Not all categories supported on all processors (not even most)**
  - **Load latency**
  - **Branch misprediction**
  - **Instruction starvation**
  - **Bandwidth saturation**
  - **Store resource saturation**
  - **Function call overhead**
  - **Exception handling**
  - **Port saturation**
  - **Serialization**
  - **+ a few others**

Stalled

Unstalled

Perf_win: best of linux perf and Intel emon

D:\app\Pmon>perf_win.exe -h

Argument processing for perf_win

process arguments after mode = stat or record

-tXXX XXX = time in seconds

-mXXX XXX = multiplex time in milliseconds

-iXXX XXX = number of multiplex iterations

-v disable verbose printout of each core's data for each multiplex iteration

-d disable detailed printout of each core's data summed over multiplex iterations

-s disable summary printout of each event's data summed over cores and multiplex iterations

-CX,Y,Z,A-B X,Y,Z individual cores, A-B is core range inclusive of A & B

-XC C = field separation character used for stat output lines, default is tab

-F add fixed counters to every collection group (stat only)

-h call usage, print these comments and terminate

-eS1,S2 S1 and S2 are event definition strings

S1 s1.s2.s3:c=X:i=Y:u:k:p=P:L=SL:P=N

s1 is event name

s2,s3..sN are umask names, programming fields are OR'd together

c=X X is cmask < 0xFF

i=Y Y = [0,1]

u user mode (default set=1)

k kernel mode (default set=1)

when only u or k is present, other is set to 0

p=P P = [0,1] default is 0, for stat mode option is ignored

L=SL SL is a string defining the LBR filtering mode, ignored in stat mode

P=N N is the sampling period, ignored in stat mode

-- AS AS is a string defining application to be launched by utility

if this field exists, collection time is set by duration of application

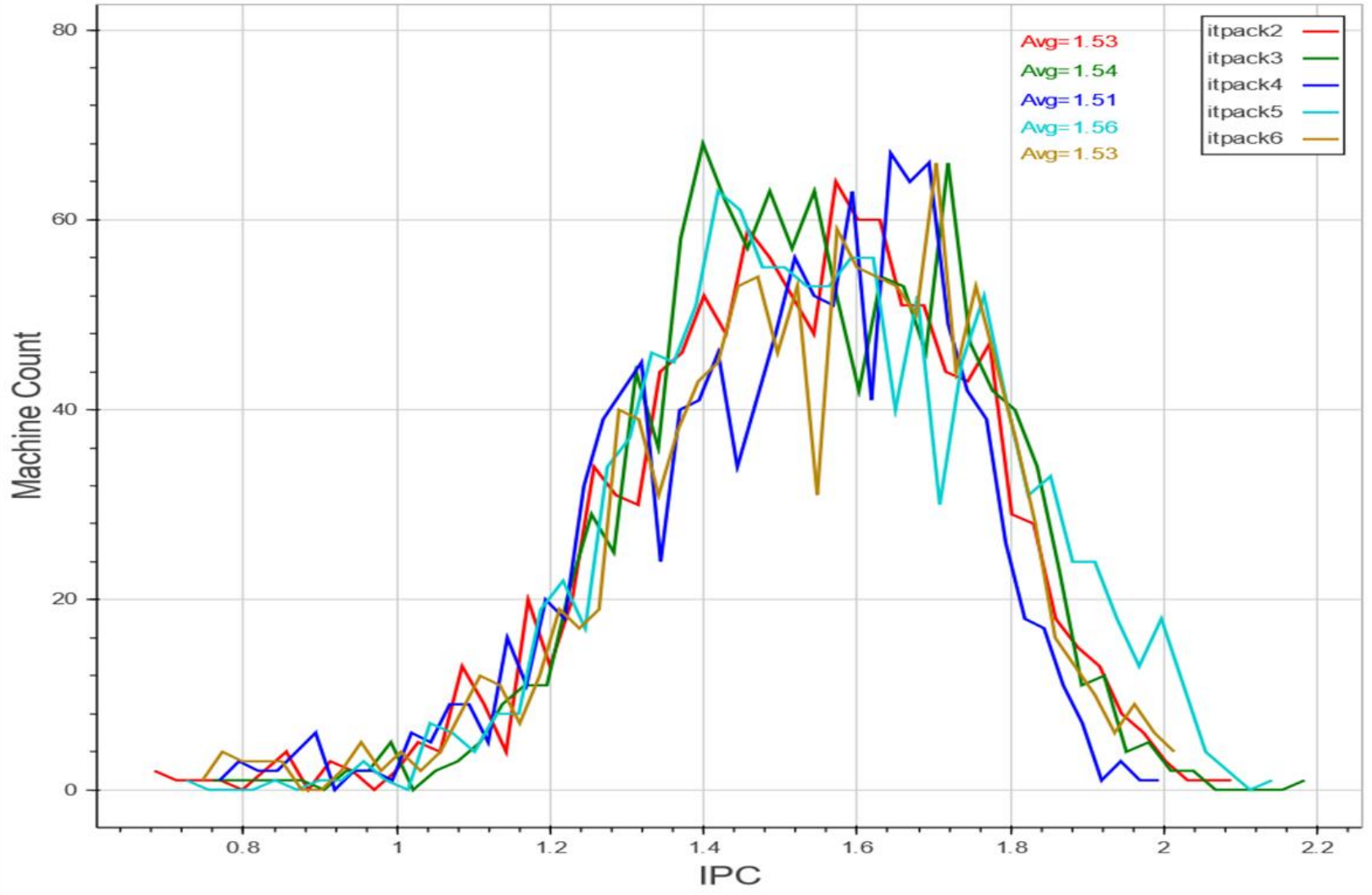
-finfile infile is a full or relative path to a file that contains all the arguments desired for the run. If this option is used it must be the only option other that output redirection

this option is required for cases where the command line exceeds 8191 characters

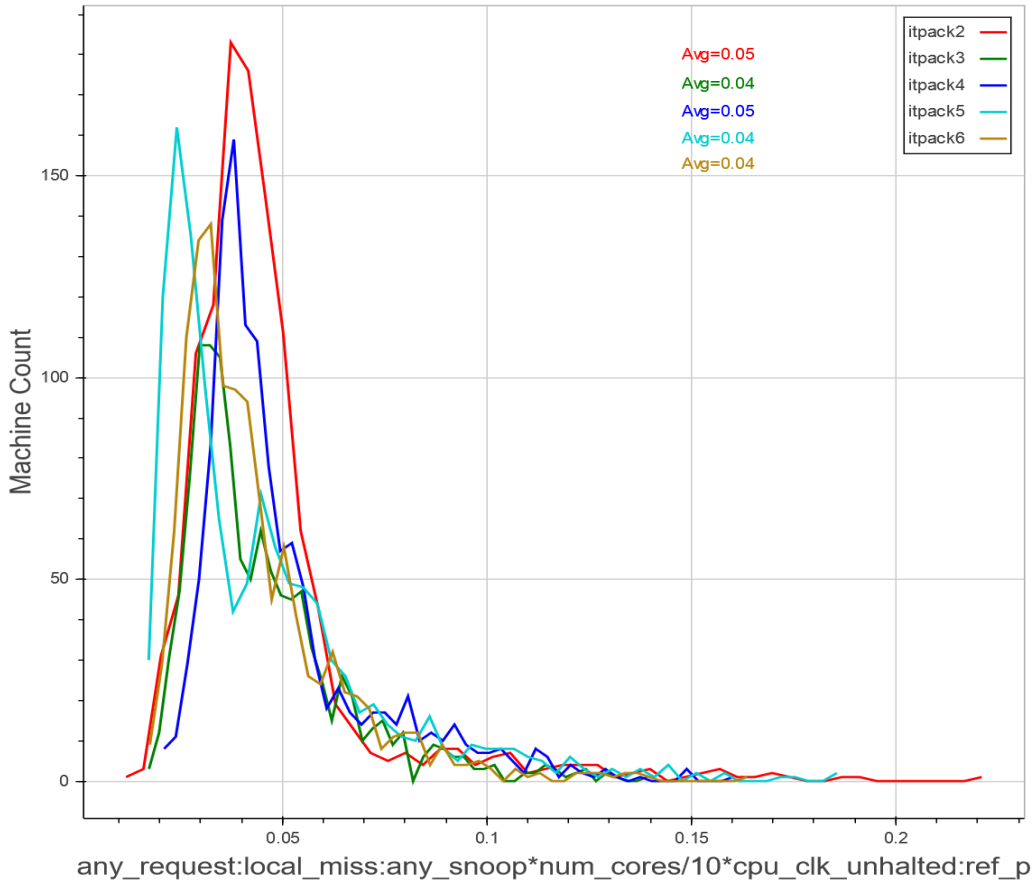
-ooutfile By default the output is sent to stderr and stderr is set to be unbuffered

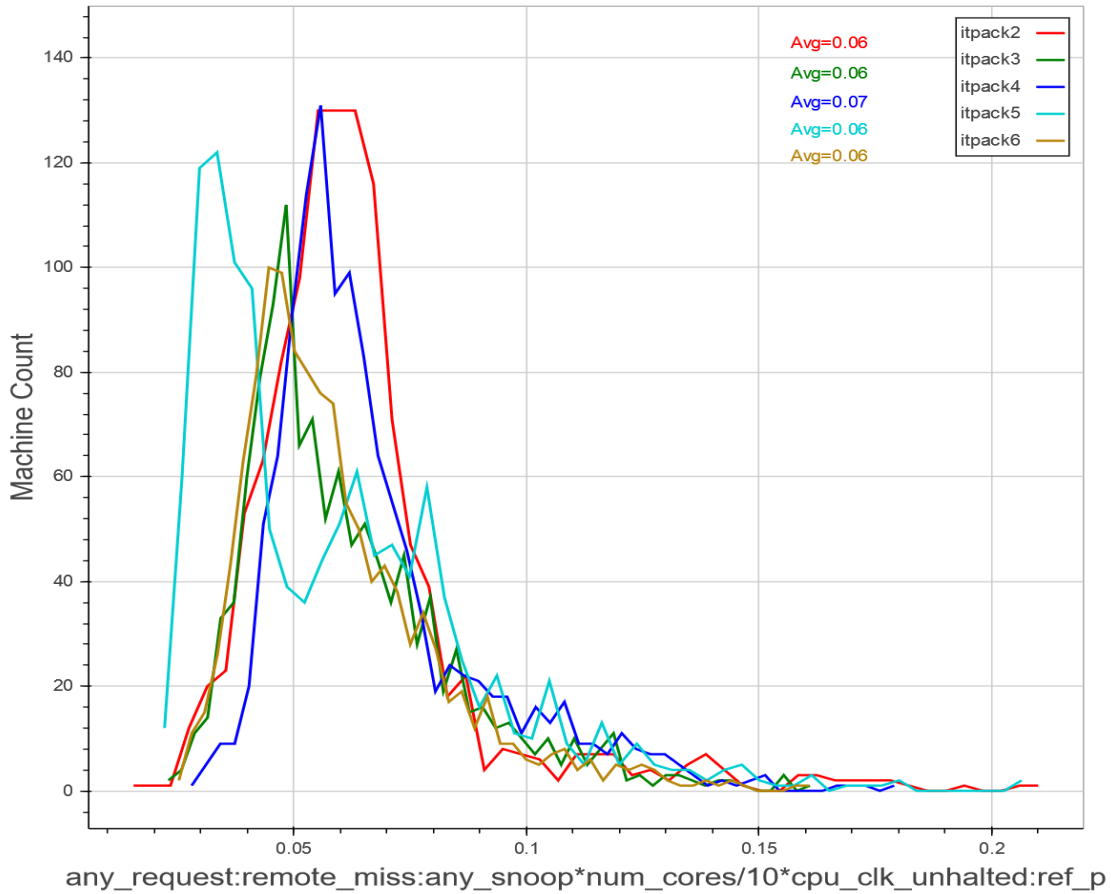
the data will be written to the file defined by outfile

Are distributions stable over time
Data analytics as an example



Memory Bandwidth $L/ns (limit is ~1)
total is ~ 0.1





Counting mode demo

- Show spreadsheet

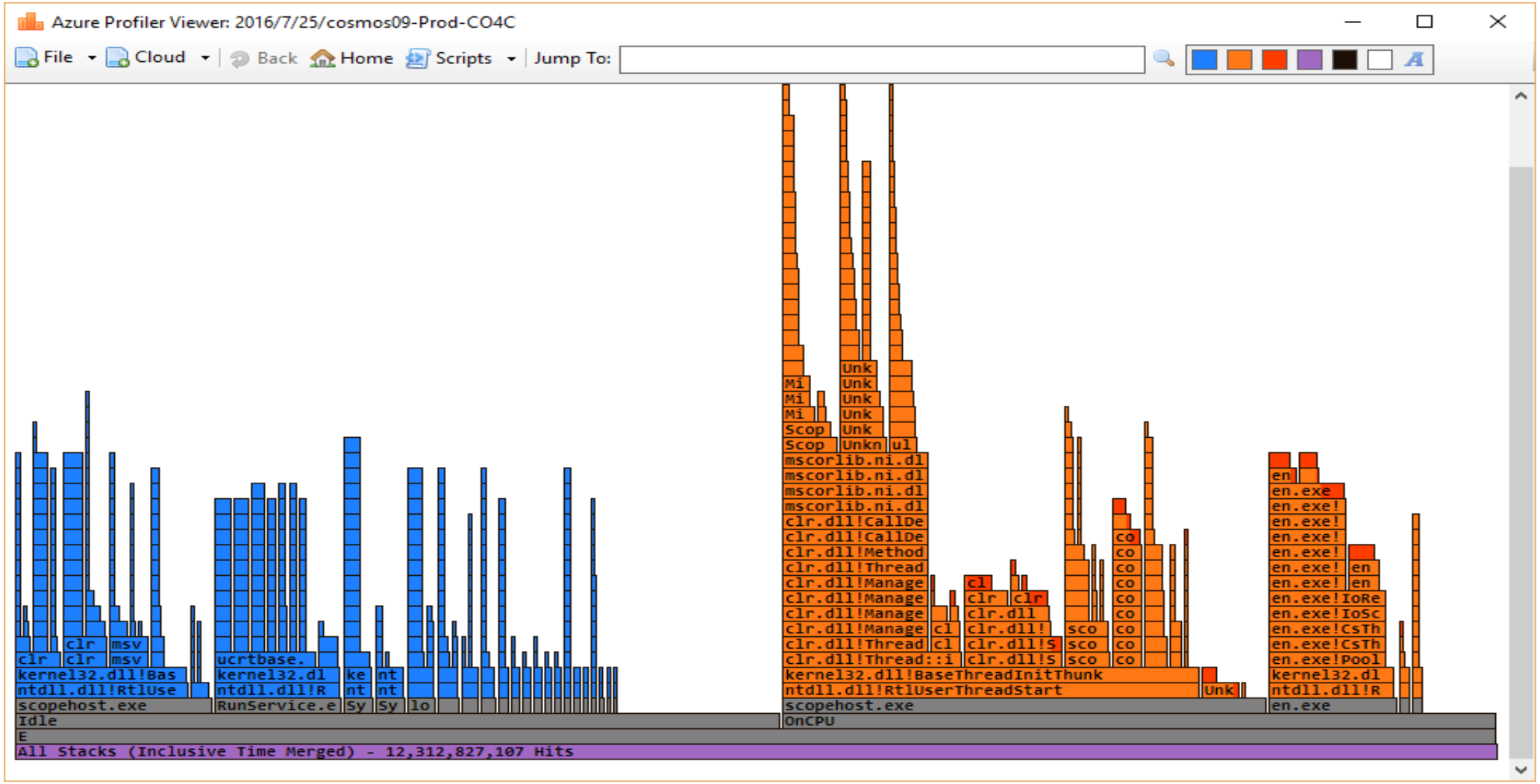
Cycle breakdown + metric averages for a few usages

	data analytics	web crawling
CPU Util.	0.36	0.77
IPC	1.53	1.24
stalled_cycles/cycles	51%	55%
instructions/unstalled_cycle	3.13	2.75
instructions/call	122	89
instruction starvation	11.70%	14.10%
br_misprediction	11.70%	7.40%
load latency	30%	29.4
resource_stalls:st/cycles	5.90%	6.60%
local+remote data lines/ns	0.1	0.137
microcode_uops/all_FE_uops	6.20%	8.30%
walk cycles/dtlb_walk	40.7	33.8
ring0/(ring0 + ring123)	14.40%	9.80%

Profiling @ Scale

- **Rare, short duration collection/machine**
- **Integration of data over huge machine counts**
  - **Merging of symbol data**
  - **Identifying common applications across machines**
- **ETW Call stack based collection**
  - **Time based HW sampling**
  - **Context switch OS event based sampling**

Time based Profiling at scale



Compression

- Used to reduce network bandwidth and disk/SSD space requirements
- Low compression/fast mode (XPRess9 level 3/6) for hot data
- High compression (LZMA) for cold data

cluster	low compression % time	high compression % time
cosmos08-co4	6.4	10.6
cosmos08co4c	6.5	8.8
cosmos08co3c	6.1	9.6
cosmos09co3c	1.3	4.7
cosmos11a-CY2	5.3	9.9
cosmos11b-cy2	6.6	10.1
cosmos09CO4C	1.6	5
cosmos11c-cy2b	6.38	6.67
cosmos14-cy2b	4.1	11.6
cosmos14-cy2	4.3	14.4

Conclusions

- **Doing thing at scale fundamentally changes the nature of the problems.**
- **System Health becomes critical**
  - **1 in ten thousand issues result in hundreds of machines that must be individually debugged**
  - **Automation of problem identification is critical**
- **Optimizing physical and SW configuration aims at average**
  - **But must also take outliers and spikes into account**
- **Optimized scheduling is paramount**
- **Application optimization needs automation to work at scale**
  - **Auto fdo/pgo**
- **Data collection and tool deployment at scale require different approaches**

Backup