Long Running Grid Submissions: A basic study of job duration

James Mott

FIFE Meeting

09/29/22

Long Running Grid Submissions

- We see that some grid nodes take longer for each file than others.

Typical example from https://samweb.fnal.gov:8483/station_monitor/gm2/stations/gm2/projects/gm2pro_2022091817_23968

Bottom node is running twice as many files as top one.

- Grid nodes have different hardware configurations.
  - Different CPU types with different specs.
  - AMD vs Intel chips can behave differently too.
  - Other differences not related to CPUs?

- This talk will try and show how much of a difference the CPU type makes to our job length.

Current Full Production Workflow 1

- We now long jobs, submitting slices of 20k files into 2.5k slots:

submission jobsub_jobid	project	dataset	date	available output	submit-ted	deliv-ered SAM
38710309@ jobsub03.fnal.gov	gm2pro_2022091509_22911	poms_recover_1323316_1	2022-09-15 09:35	104	141	100
61040689@ jobsub02.fnal.gov	gm2pro_2022091505_10856	poms_recover_1323391_1	2022-09-15 05:05	104	129	95
38673062@ jobsub03.fnal.gov	gm2pro_2022091317_19975	poms_recover_1322427_1	2022-09-13 17:35	137	163	147
60983321@ jobsub02.fnal.gov	gm2pro_2022091311_3307	poms_recover_1322000_1	2022-09-13 11:35	97	141	107
38647458@ jobsub03.fnal.gov	gm2pro_2022091308_7889	gm2pro_daq_raw_run4_Prod_PQ_notest_slice4_files20000	2022-09-13 08:15	14502	14502	14377
60943441@ jobsub02.fnal.gov	gm2pro_2022091306_13240	gm2pro_daq_raw_run4_Prod_PQ_notest_slice3_files20000	2022-09-13 06:15	20000	20000	19861
60904086@ jobsub02.fnal.gov	gm2pro_2022091206_8811	gm2pro_daq_raw_run4_Prod_PQ_notest_slice2_files20000	2022-09-12 06:15	19997	20000	19839
60902727@ jobsub02.fnal.gov	gm2pro_2022091201_10225	poms_recover_1321305_1	2022-09-12 01:05	37	58	57
38595092@ jobsub03.fnal.gov	gm2pro_2022091118_29098	gm2pro_daq_raw_run4_Prod_PQ_notest_slice1_files20000	2022-09-11 18:15	20000	20000	19859
38569707@ jobsub03.fnal.gov	gm2pro_2022091023_15579	gm2pro_daq_raw_run4_Prod_PQ_notest_slice0_files20000	2022-09-10 23:40	20000	20000	19993

4PQ Dataset

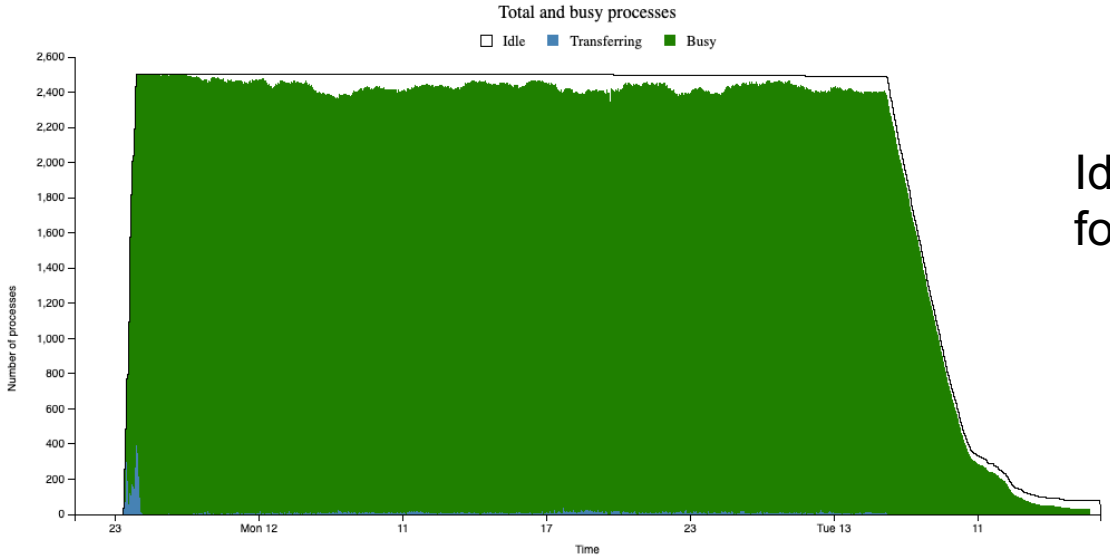
POMS Campaign

- Each slice also has a recovery submission that reruns failed jobs and then auto-launches 3 shorter dependency stages.
- A dataset is typically complete in ~5 main stage submissions

Current Full Production Workflow 2

- With ~8 files per job it takes around 24 hours to process each slice

SAM Project Monitoring: number of nodes that are busy vs time.



Ideally we'd have a box function for this – with no long tails

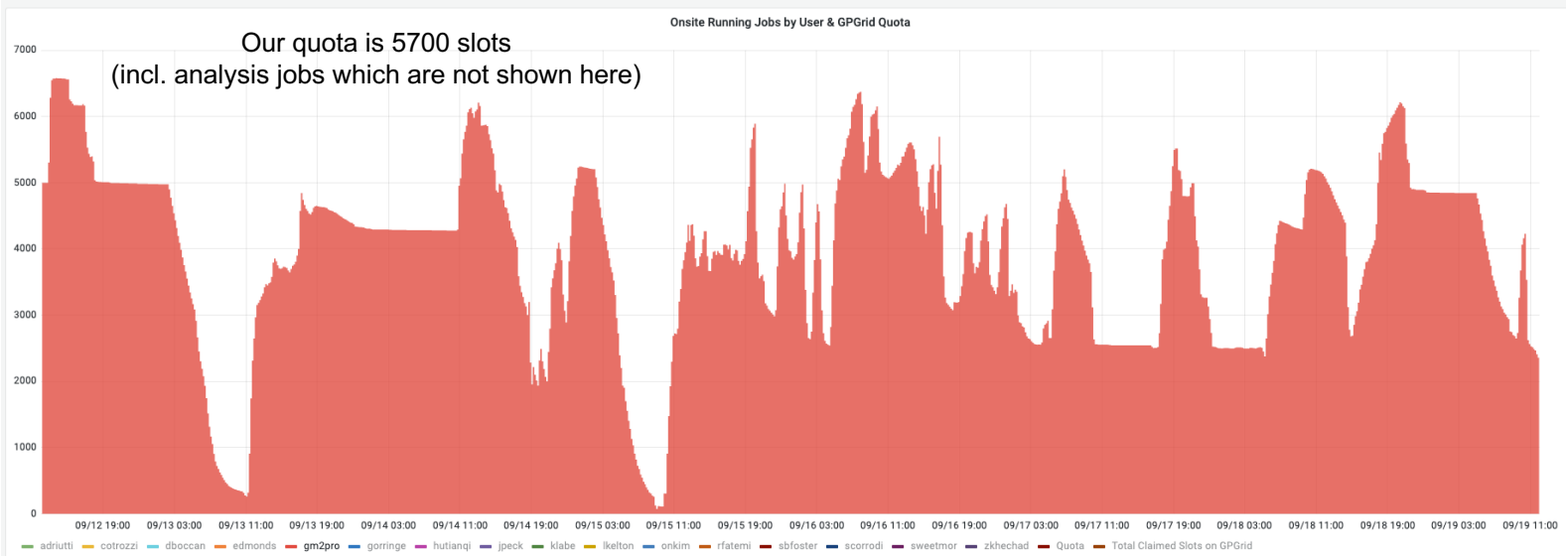
[This is actually one where I killed the jobs to stop the slice.]

- We set a 40h lifetime for each grid job, after which it's removed.
- The nodes keep going until the files run out, so some nodes do more than 8 and some less.

Current Full Production Workflow 3

- Running these big long slices makes a big difference to efficient use of our slot allocation:

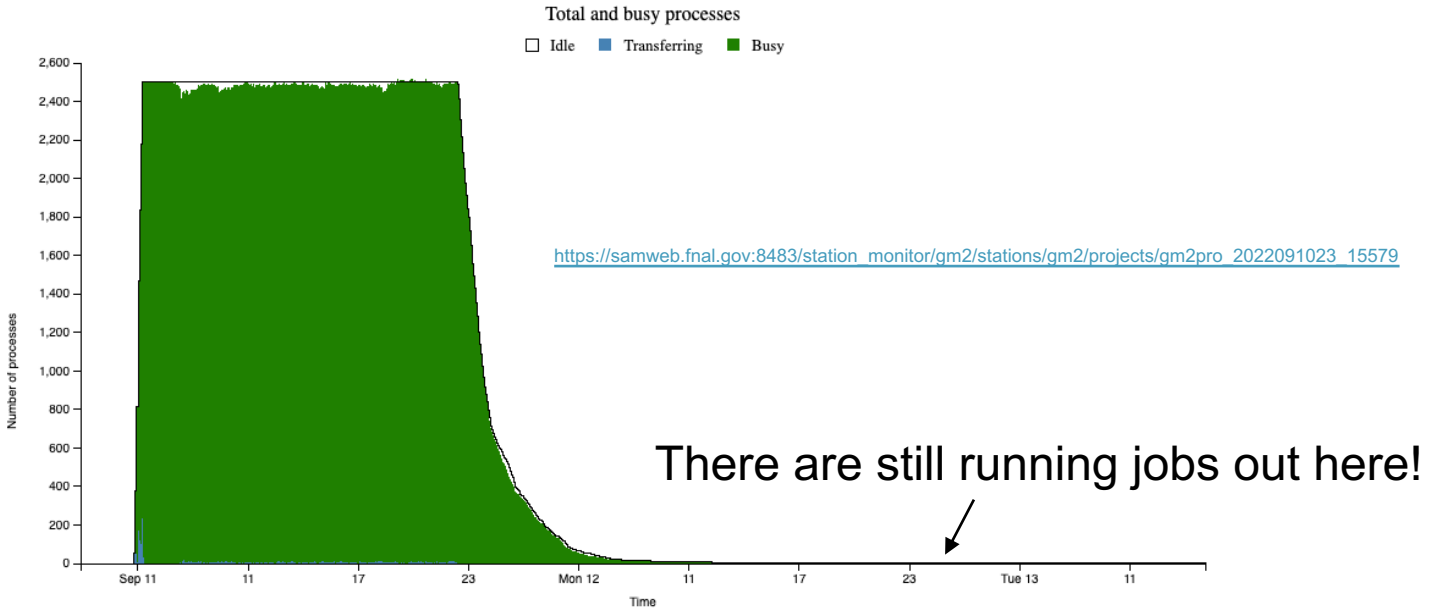
Grid slot allocation for gm2pro for the last 7 days



- We aim for a submission every 24h to keep a baseline of 2.5k slots running jobs and run e.g. preproduction or another dataset on top.
- Shorter and more frequent submissions either mean more gaps or overlaps between submissions and are harder to monitor.

Why do we care about long submissions?

- If we leave a SAM submission alone, this is common behaviour:



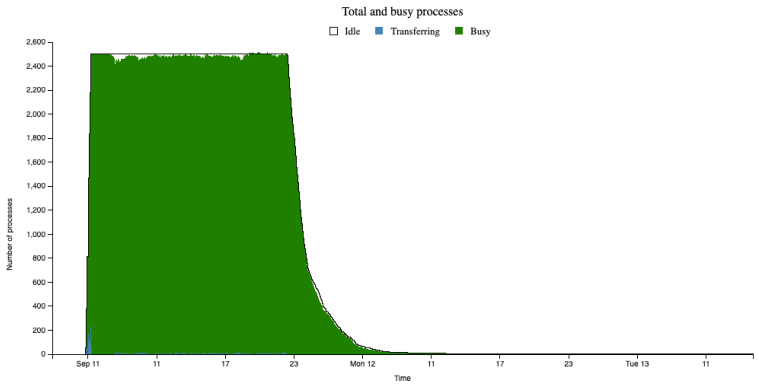
- Long tail goes way out in time, even past the 40h job lifetime.

- Recovery and dependency jobs sometimes aren't launched until these complete or sometimes launch early (depending on success of completed jobs).

Why do we care about long submissions?

- We want to keep launching submissions to keep the grid full, so we end up with overlapping submissions:

Submission 1

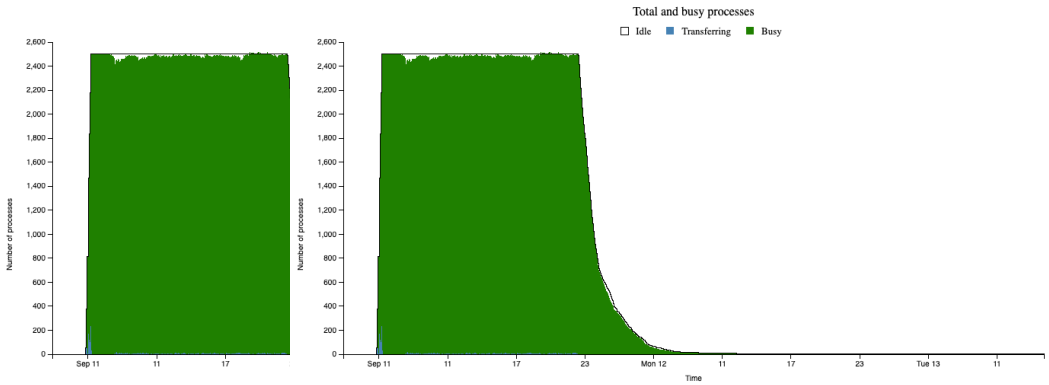


Why do we care about long submissions?

- We want to keep launching submissions to keep the grid full, so we end up with overlapping submissions:

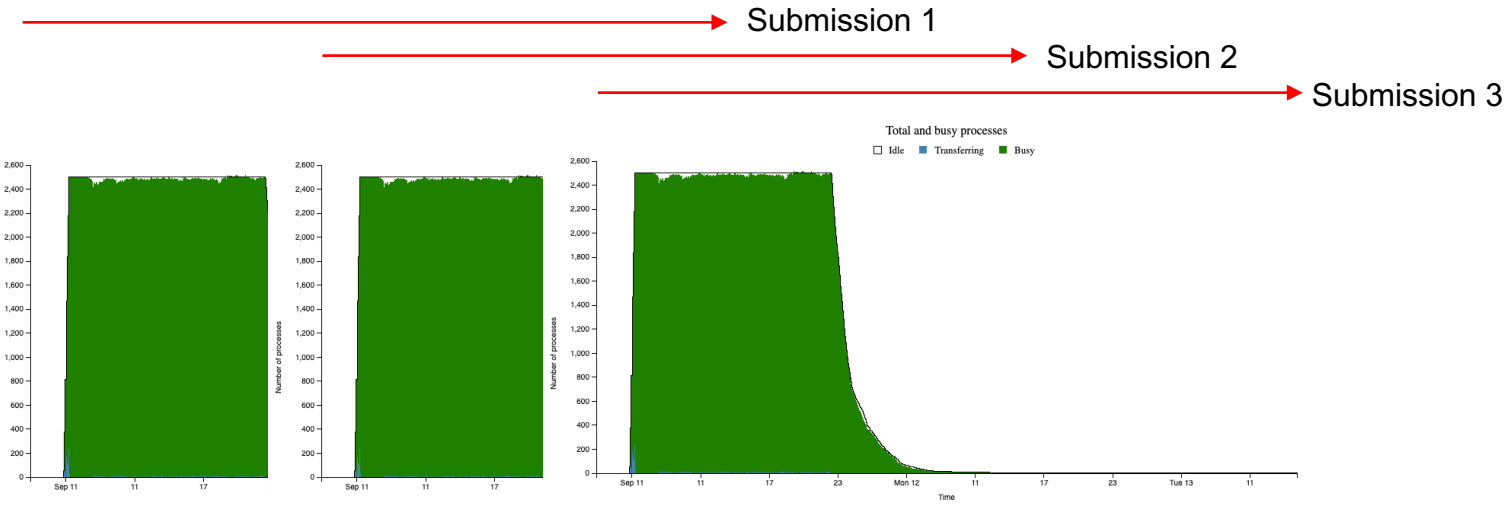
Submission 1

Submission 2



Why do we care about long submissions?

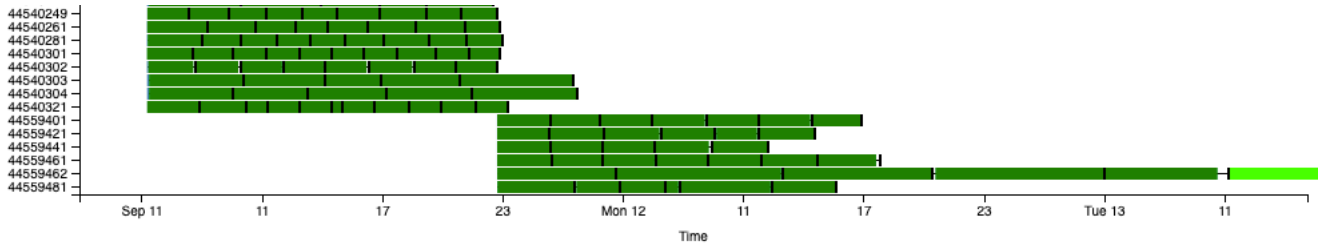
- We want to keep launching submissions to keep the grid full, so we end up with overlapping submissions:



- As a shifter, you end up monitoring multiple slices at once
- If dependencies launch, there are a large number of POMS campaign stages on-going (up to 9 per dataset + recoveries).
- It's hard and stressful to monitor, so we can miss issues where some parts of the campaigns fail or get stuck.

Two main causes for long tails

1. Restarting jobs



- Often “Job disconnected” error, followed by re-submission of jobs and reset of lifetime back to 40h. This project kept grabbing files past 20k.

- First attempted fix failed:

```
--lines=‘max_retries=0’
```

  - Apparently a disconnect doesn’t count as a job completion/retry.

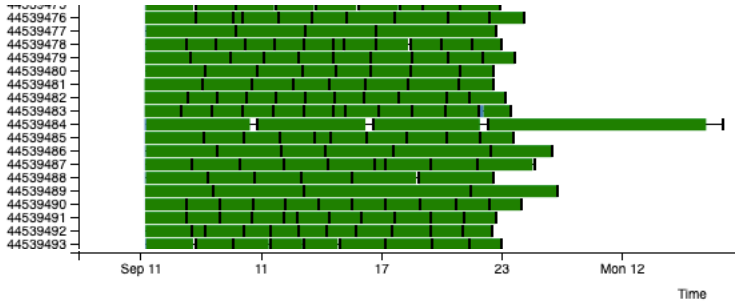
- Next attempt (in progress – efficacy as yet undetermined):

```
--lines='periodic_hold=(NumJobStarts>1)&&((time()-EnteredCurrentStatus)>1800)’
```

  - NumJobStarts does increment with reconnect, and want 30 min duration to make sure we don’t kill e.g. samend.sh scripts.

Two main causes for long tails

2. Slow running jobs



- Sometimes get unlucky and a slow node starts a file right at the end.
- On slow nodes, jobs can take 12+ hours: not on the same long time scale as restarts issue, but there are a lot more of these
- Hard for shifters to guess when submission will end
- Annoying if it's a recovery where we run 1 file per job with only a few input files – can slow up the whole process a lot as it drags over multiple days

Getting CPU information from the grid

- At Adam's suggestion, Liang added **`lscpu`** to our grid jobs.

```
gm2gpvm04:~$ head -n 43 /pnfs/GM2/scratch/daq/2022-09-18-17-05-13/38852815/logs/env_38852815_run43359.00467.log

>>>Here is the your environment and a few debug statements in this job:

>>>gm2 RELEASE=v10_06_00

Sun Sep 18 23:08:30 UTC 2022

>>>setup cvmfs common products

>>>Job started on Linux gm2pro-38852815-0-fnpc17107.fnal.gov 3.10.0-1160.71.1.el7.x86_64 #1 SMP Tue Jun 28 08:1
9:35 CDT 2022 x86_64 x86_64 x86_64 GNU/Linux

OS=SL7

OFFSITE=1

>>>Site is FermiGrid

Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                72
On-line CPU(s) list:   0-71
Thread(s) per core:    2
Core(s) per socket:    18
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz
Stepping:              4
CPU MHz:               2535.699
CPU max MHz:           3700.0000
CPU min MHz:           1000.0000
BogoMIPS:              4600.00
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              1024K
L3 cache:              25344K
NUMA node0 CPU(s):     0-17,36-53
NUMA node1 CPU(s):     18-35,54-71
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi
 mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good no
pl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma
 cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm
 abm 3dnowprefetch epb cat_l3 cdp_l3 invpcid_single intel_ppin intel_pt ssbd mba ibrs ibpb stibp tpr_shadow vnm
i flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm cqm mpx rdt_a avx512f avx5
12dq rdseed adx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1 cqm_llc cqm_occup_llc c
qm_mbm_total cqm_mbm_local dtherm ida arat pln pts pku ospke md_clear spec_ctrl intel_stibp flush_l1d arch_capa
bilities
gm2gpvm04:~$
```

First env*.log file from each node now contains the CPU information

- I've taken CPU information from the env files and timing information from our gm2 log files.
- Here I'll show preproduction of **5I** and full production of **4PQ** (both running week of 9/12 – 9/19).

What CPUs are there on the grid?

- Here's where the jobs from **5I** and **4PQ** landed*:

CPU Model Name	Preproduction (2 files per job)		Full Production (8 files per job)	
	# Nodes	%	# Nodes	%
AMD_EPYC_7543_32-Core_Processor	9048	20.0	2319	19.6
AMD_EPYC_7502_32-Core_Processor	8955	19.8	2289	19.4
AMD_Opteron(tm)_Processor_6376	1299	2.9	371	3.1
AMD_EPYC_7413_24-Core_Processor	318	0.7	-	-
AMD_Other	36	0.1	-	-
Intel(R)_Xeon(R)_CPU_E5-2680_v4_@_2.40GHz	8104	17.9	2554	21.6
Intel(R)_Xeon(R)_CPU_E5-2650_v2_@_2.60GHz	6236	13.8	1245	10.5
Intel(R)_Xeon(R)_CPU_E5-2670_v3_@_2.30GHz	1643	3.6	503	4.3
Intel(R)_Xeon(R)_CPU_E5-2650_v3_@_2.30GHz	978	2.2	286	2.4
Intel(R)_Xeon(R)_Gold_6140_CPU_@_2.30GHz	8500	18.8	2243	19.0
Intel(R)_Xeon(R)_Other	155	0.2	-	-
Total	**45272**	**100**	**11810**	**100**

- Given this was over a few days with ~10 slices, this is presumably a fair representation of what our jobs normally use.

Job Details: Preproduction

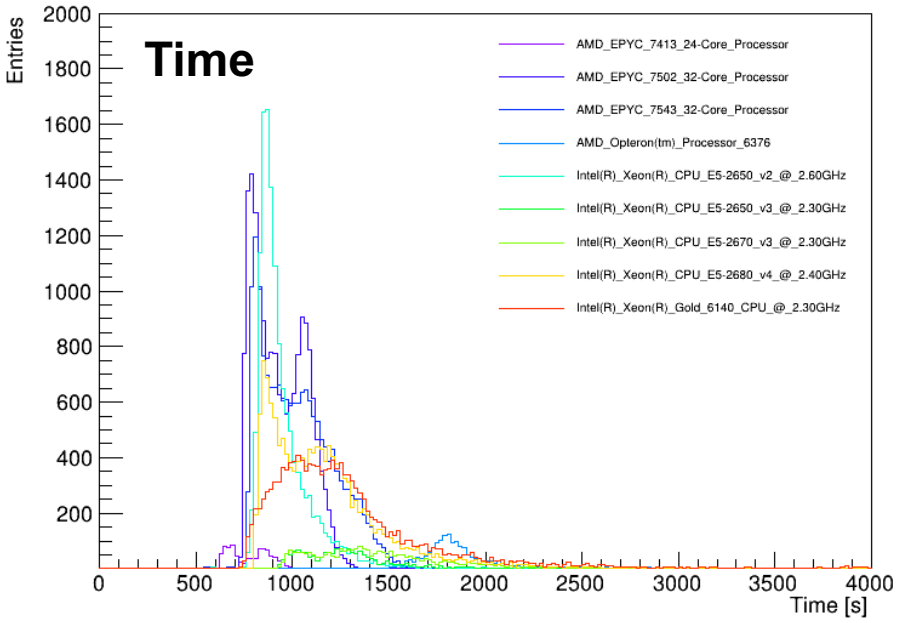
- Preproduction is a single gm2 process (15 – 30 mins):

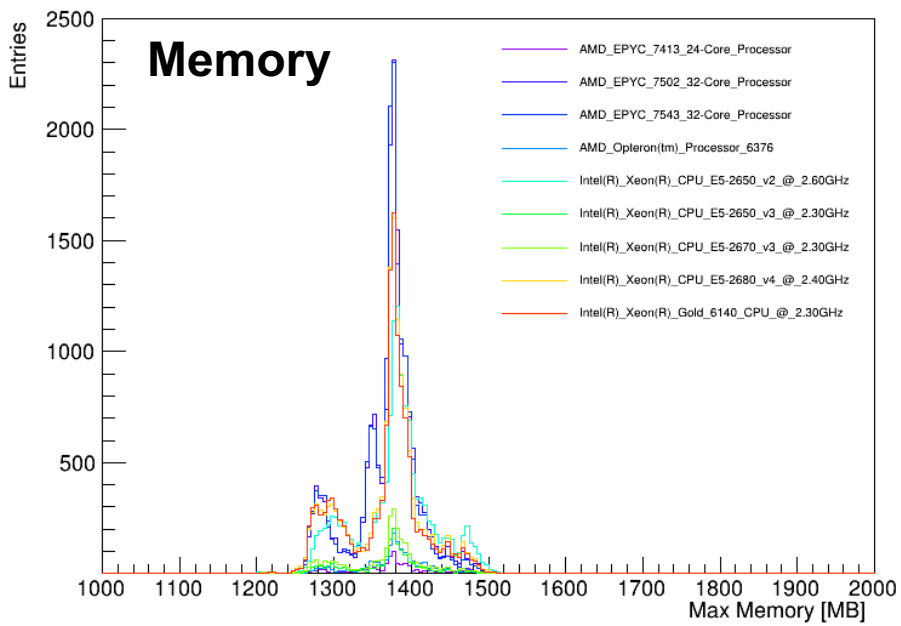
CPU Model Name	% of Fermigrid	Real Time (secs)		Max. Mem. (MB)	
		Mean	RMS	Mean	RMS
AMD_EPYC_7543_32-Core_Processor	20.0	1026	185	1367	41
AMD_EPYC_7502_32-Core_Processor	19.8	935	133	1369	42
AMD_Opteron(tm)_Processor_6376	2.9	**1904**	310	1378	45
AMD_EPYC_7413_24-Core_Processor	0.7	780	103	1379	53
AMD_Other	0.1	921	117	1431	49
Intel(R)_Xeon(R)_CPU_E5-2680_v4_@_2.40GHz	17.9	1166	**319**	1367	50
Intel(R)_Xeon(R)_CPU_E5-2650_v2_@_2.60GHz	13.8	947	142	1379	51
Intel(R)_Xeon(R)_CPU_E5-2670_v3_@_2.30GHz	3.6	1447	**372**	1367	49
Intel(R)_Xeon(R)_CPU_E5-2650_v3_@_2.30GHz	2.2	1281	**263**	1368	50
Intel(R)_Xeon(R)_Gold_6140_CPU_@_2.30GHz	18.8	1277	**422**	1365	50
Intel(R)_Xeon(R)_Other	0.2	1228	**429**	1427	57

- AMD Opteron-6376 is much slower than others
- A lot more variation (larger RMS) in the intel chip times.

Job Details: Preproduction

- Distributions of time and memory from previous slide:

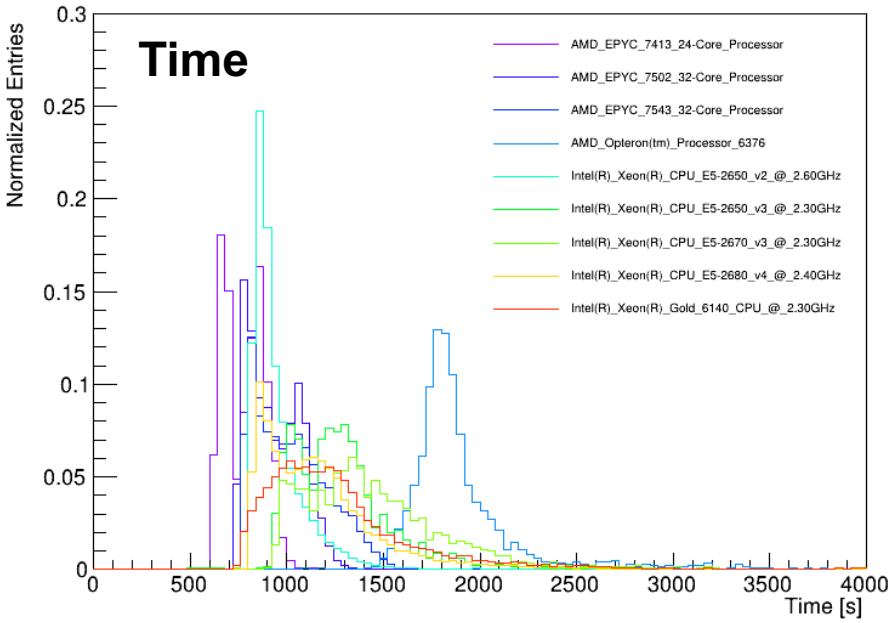


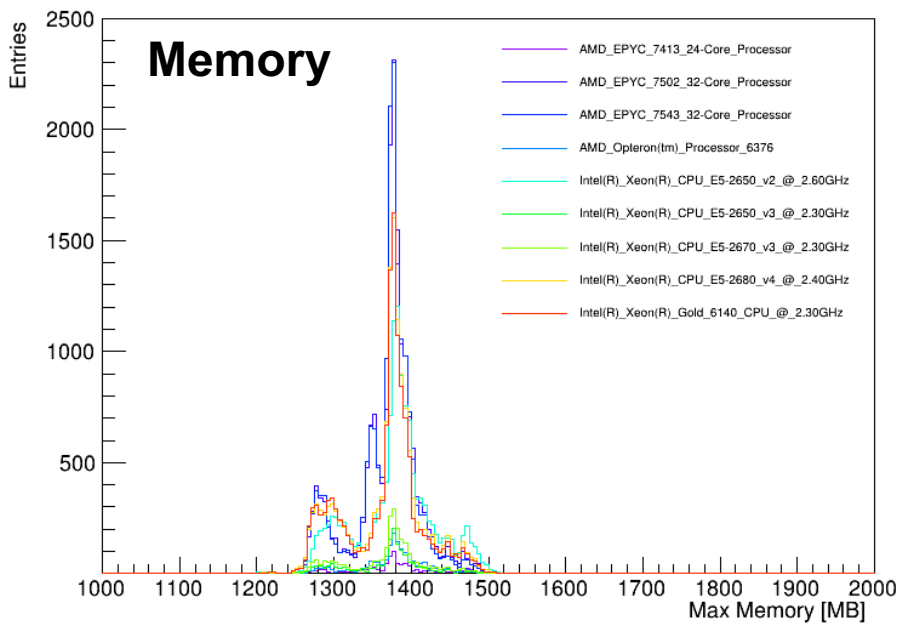


- Memory shows little variation between CPUs
- Time shows some longer tails on the Intel chips
- Slow jobs are deweighted here as there aren't as many completed ones for the plots!

Job Details: Preproduction Normalised

- Distributions of time and memory from previous slide:





- After normalizing, AMD Opteron-6376 shows up as clear outlier.
- Other AMD chips appear to be doing slightly better than Intel
- Fastest are comparable, but long tails for Intel.
- Is there more than just CPU at play here?

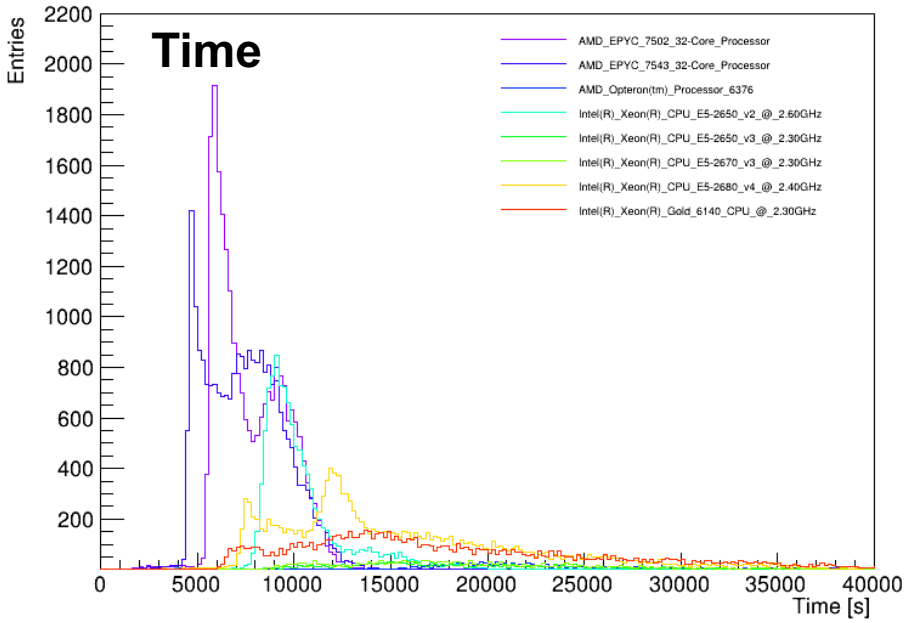
Job Details: Full Production

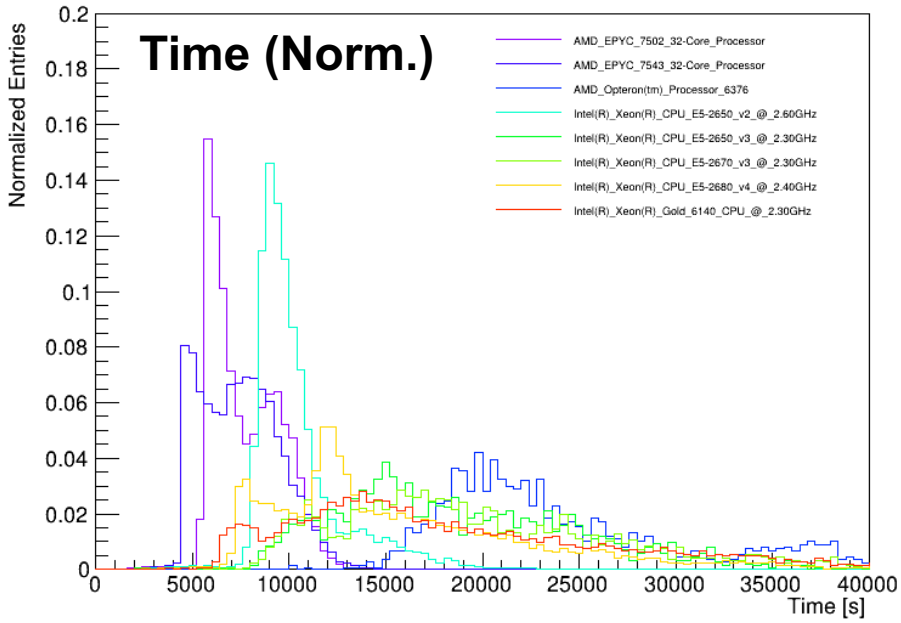
- For full production, here's the total time and highest memory:

CPU Model Name	% of Fermigrid	Real Time (secs)		Max. Mem. (MB)	
		Mean	RMS	Mean	RMS
AMD_EPYC_7543_32-Core_Processor	19.6	7231	2234	1972	15
AMD_EPYC_7502_32-Core_Processor	19.4	7755	1774	1975	5
AMD_Opteron(tm)_Processor_6376	3.1	**26839**	11366	1978	5
Intel(R)_Xeon(R)_CPU_E5-2680_v4_@_2.40GHz	21.6	15915	**7375**	1971	5
Intel(R)_Xeon(R)_CPU_E5-2650_v2_@_2.60GHz	10.5	10418	2322	1979	7
Intel(R)_Xeon(R)_CPU_E5-2670_v3_@_2.30GHz	4.3	20429	**8501**	1972	5
Intel(R)_Xeon(R)_CPU_E5-2650_v3_@_2.30GHz	2.4	18888	**6384**	1973	4
Intel(R)_Xeon(R)_Gold_6140_CPU_@_2.30GHz	19.0	19358	9967	1971	5

- Same patterns as with preproduction:
  - AMD Opteron-6376 is much slower than others
  - A lot more variation (larger RMS) in the intel chip times.
  - Other AMD chips are clearly faster now
  - Nothing to choose between memory usage (tracking stage)

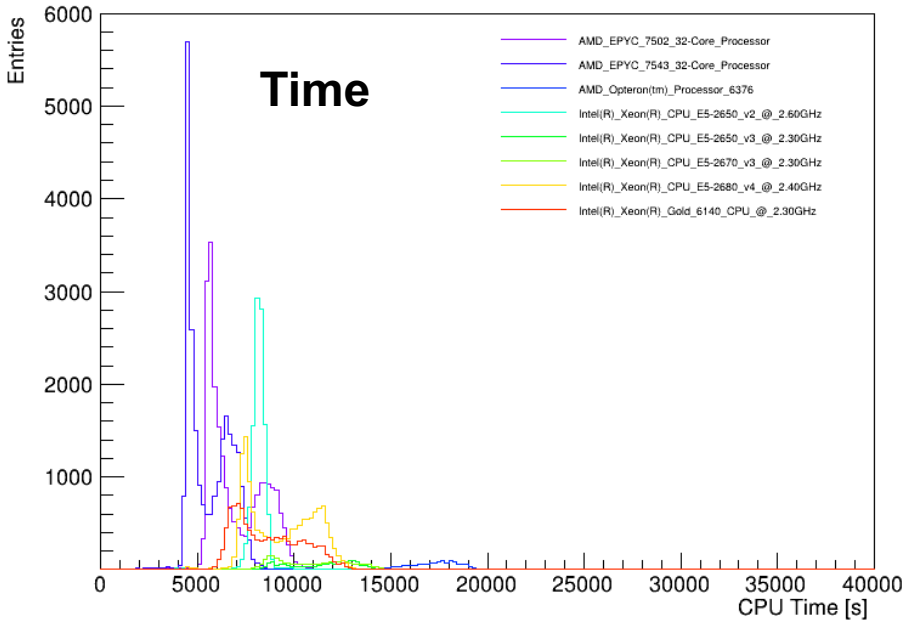
Job Details: Full Production Real Time

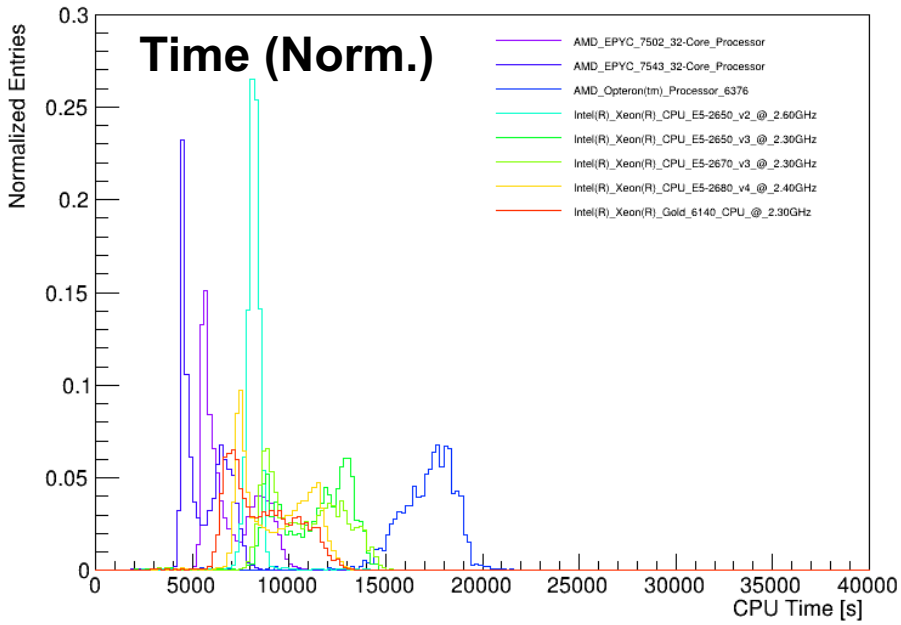




- AMD EPYCs are best, followed by Intel E-2650-v2.
- AMD Opteron-6376 is terrible.
- Other intel chips are sometimes good and sometimes very bad.

Job Details: Full Production CPU Time

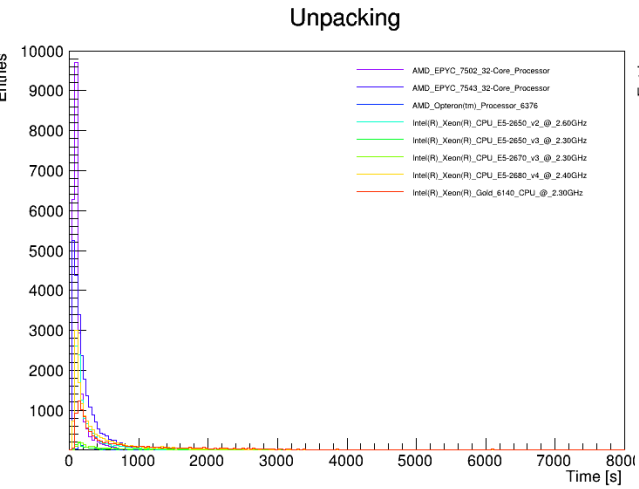


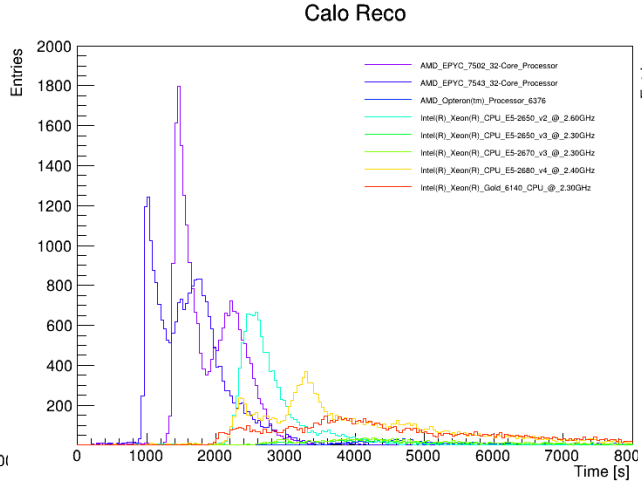


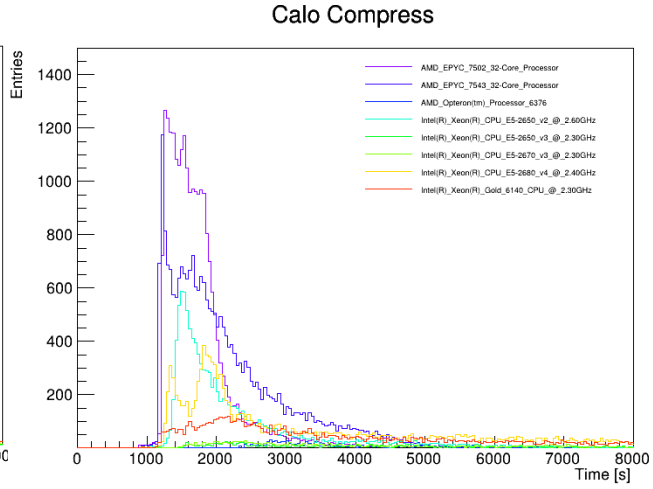
- Most of spread and really long tails are only present in real time, not CPU time.
- Same ordering of chip speed as shown in real time
- Still more spread for some of the intel chips

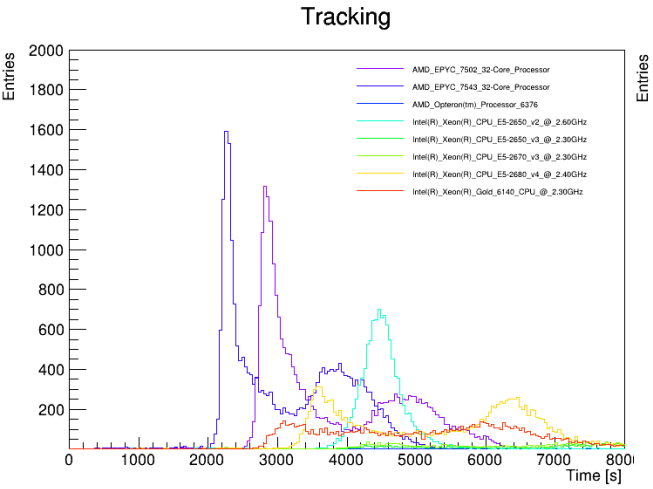
Full Production Stages: Real Time

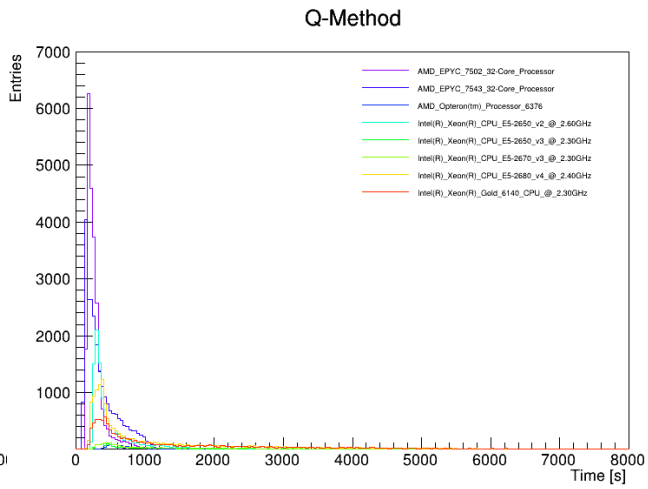
- Breakdown into 5 production stages:









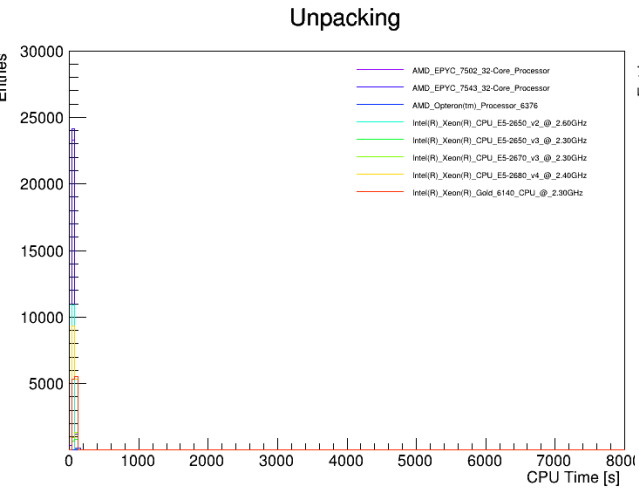


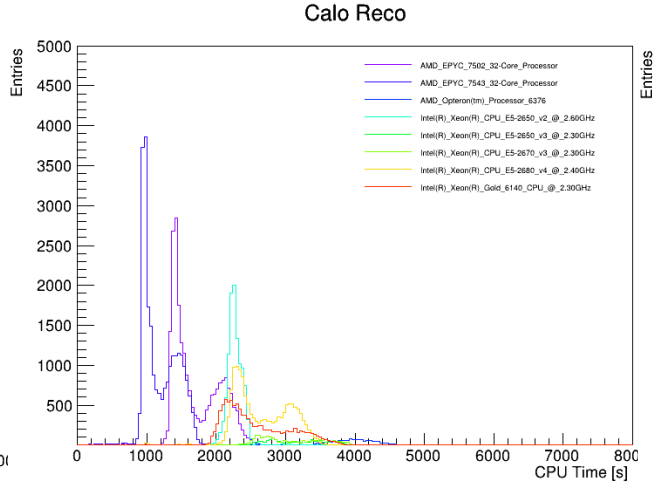
Mostly follows total time.

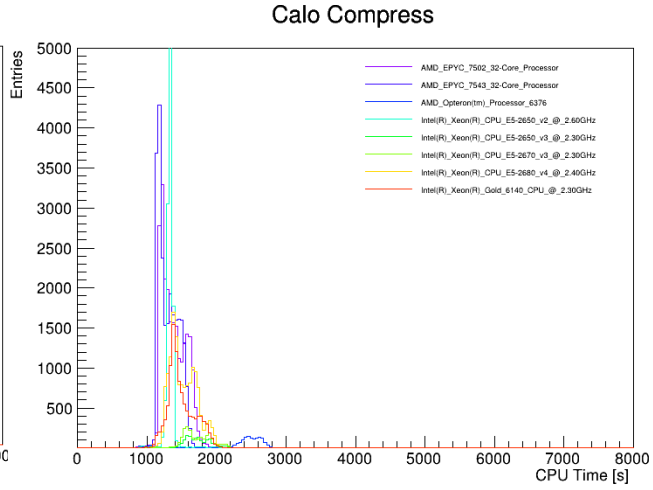
CaloReco and TrackReco show bimodal structures but not for all chip types?!

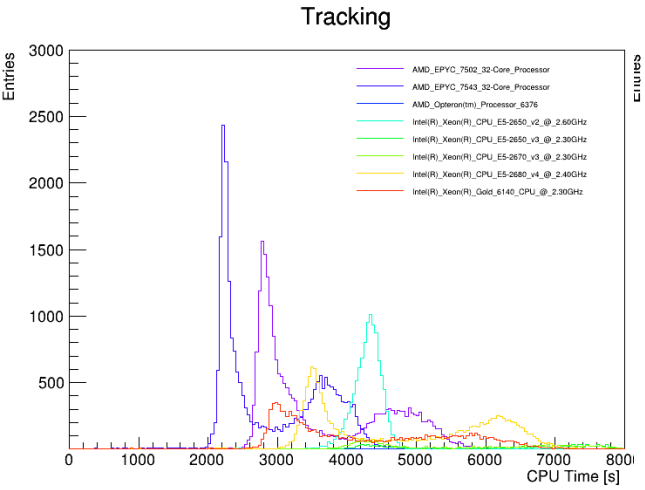
Full Production Stages: CPU Time

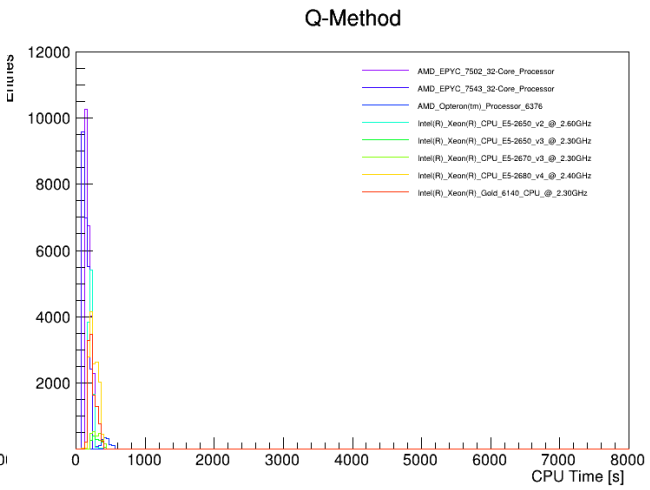
- Breakdown into 5 production stages:











Tails in calo data processing disappear.

CaloReco and TrackReco still show bimodal structure

TrackReco largely unchanged (small amounts of data and lots of CPU)

Why are some so slow?

- Some CPU specs from AMD website:

Opteron 6376	
Platform: Server	**Product Family:** AMD Opteron™
# of CPU Cores: 16	**# of Threads:** 16

AMD EPYC™ 7502	
Platform: Server	**Product Family:** AMD EPYC™
# of CPU Cores: 32	**# of Threads:** 64

Looks to be just dual-threading vs single threaded core?

- Don't see all that much to choose between Intel in terms of specs:

Intel E-2650 v2
t = 10418 ± 2322 secs

CPU Specifications	
Total Cores	8
Total Threads	16
Max Turbo Frequency	3.40 GHz
Intel® Turbo Boost Technology 2.0 Frequency‡	3.40 GHz
Processor Base Frequency	2.60 GHz
Cache	20 MB Intel® Smart Cache
Bus Speed	8 GT/s
# of QPI Links	2
TDP	95 W

Intel E-2650 v3
t = 18888 ± 6384 secs

CPU Specifications	
Total Cores	10
Total Threads	20
Max Turbo Frequency	3.00 GHz
Intel® Turbo Boost Technology 2.0 Frequency‡	3.00 GHz
Processor Base Frequency	2.30 GHz
Cache	25 MB Intel® Smart Cache
Bus Speed	9.6 GT/s
# of QPI Links	2
TDP	105 W

Some Questions:

- Why do some CPUs have such large variation in terms of real time?
  - Is it all disk access? What else should I investigate there?
  - Is there any way to tell if they'll be good or bad ahead of time?

- What is the bimodal structure in the trackReco and caloReco plots? If it's not in all CPU types, suggests it's not the input data?

- What counts as a slot on the grid? Is the AMD Opteron so slow because of one of our settings or is it like that for everyone?

- I don't think we have any compiler optimization for specific CPUs. Should we be investigating this?

- If we were to veto certain nodes from our job submissions, will that make SCD unhappy?