

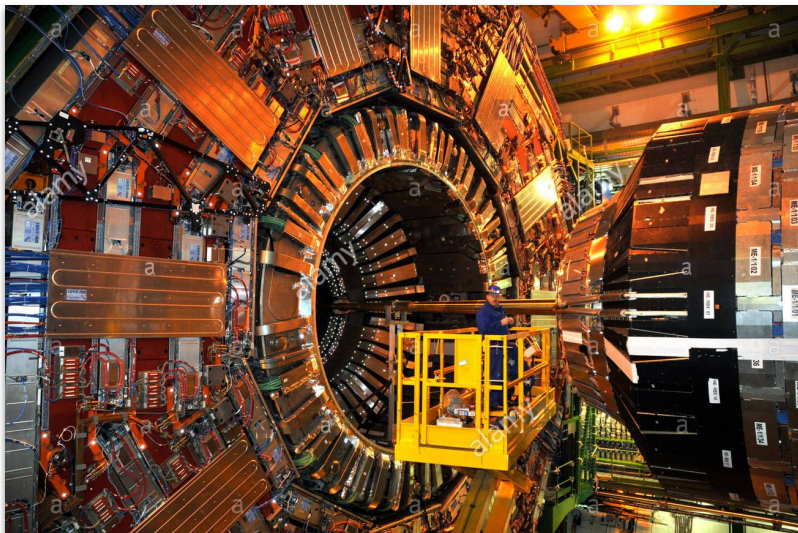


# Improving CMS CPU Efficiency through Strategic Pilot Overloading

M. Mascheroni, A. Pérez-Calero Yzquierdo, F. Von  
Cube, V. Zokaite, H. Kim, F. Khan  
*for the CMS Collaboration*

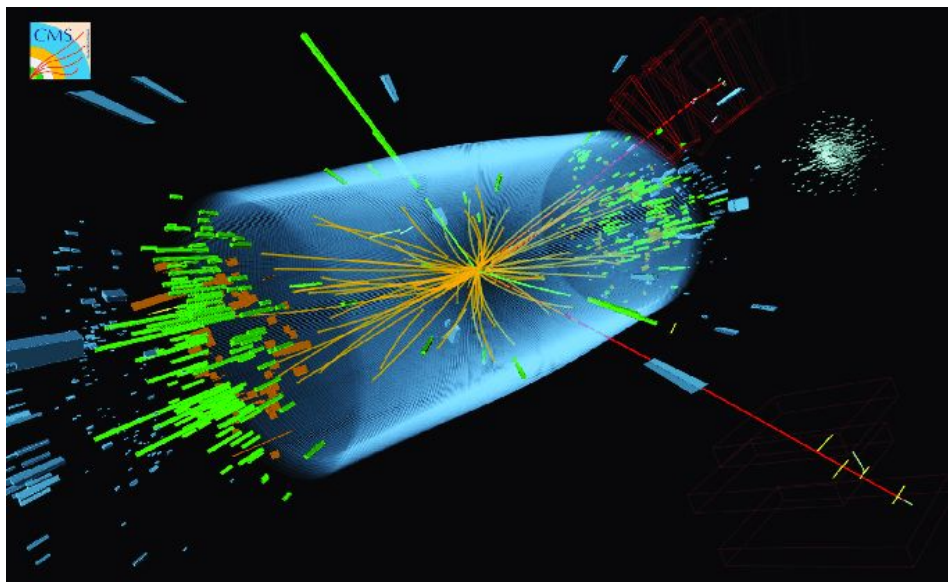


# The CMS experiment at CERN



- **High Energy Physics general-purpose experiment recording proton-proton collisions at the LHC at CERN**

- Experimental data is stored, distributed, reconstructed, and analyzed, comparing to simulated data (Monte-Carlo)
  - **Hundreds of PBs per year**



# The computing landscape - the WLCG

- Data traditionally analyzed using **Worldwide LHC Computing Grid (WLCG) resources**
  - Global collaboration of around 170 computing centers
  - Access based on dedicated resources (**pledges**)
  - **Over 1M CPU cores and 2 EB in data storage**

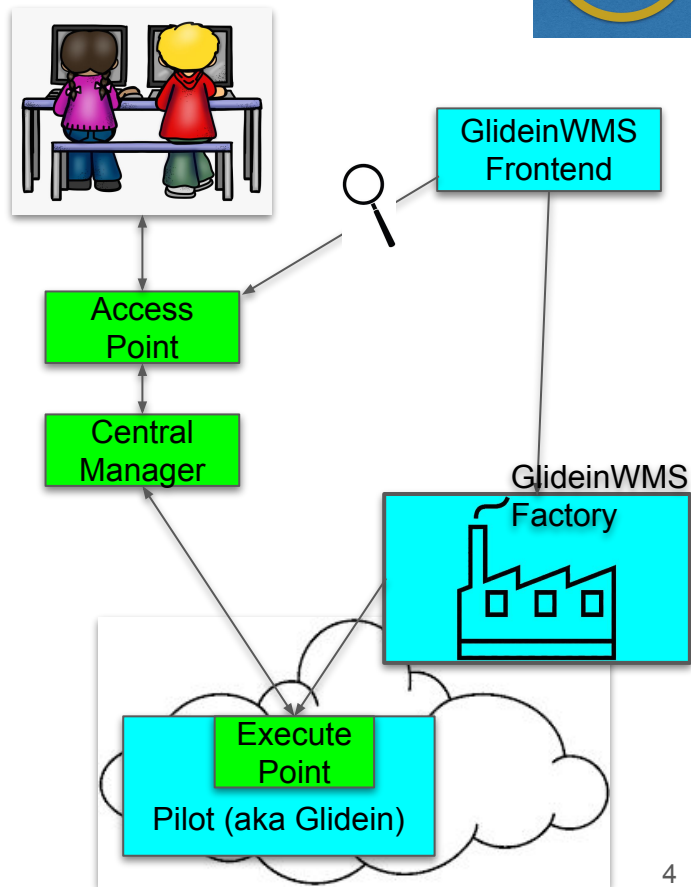




# The CMS Submission Infrastructure Group



- Part of CMS Offline and Computing in **charge** of:
  - Organizing **HTCSS** and **GlideinWMS** operations in CMS, in particular of the **Global Pool**, an infrastructure where reconstruction, simulation, and analysis of physics data takes place
  - Communicate CMS **priorities** to the **development teams** of **GlideinWMS** and **Condor**
- In practice:
  - We operate a set of federated pool of resources **distributed over 70 Grid sites, plus non-Grid resources**
  - Join them into a Global Pool of resources managed by HTCondor

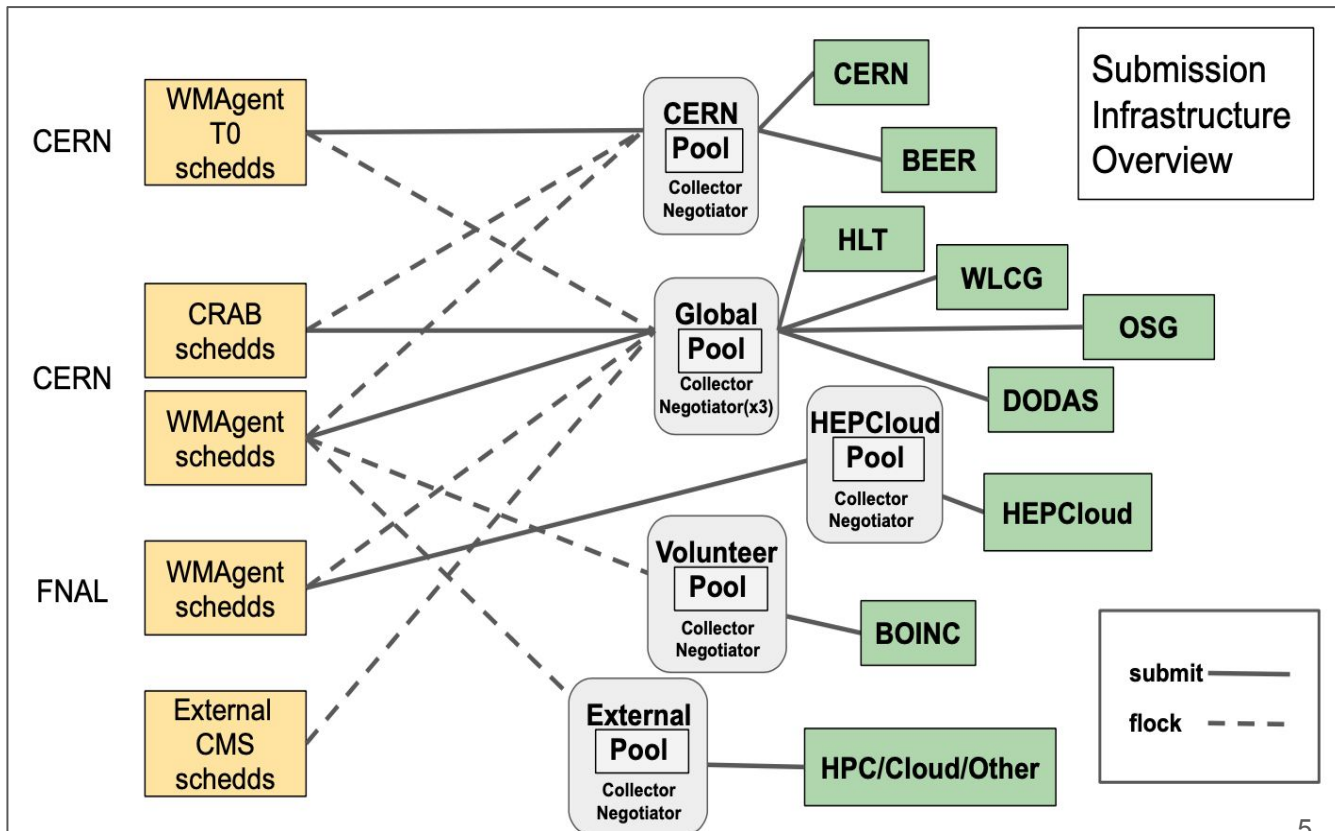




# The CMS SI: federated HTCondor pools



- Types of access point
- Types of execution point





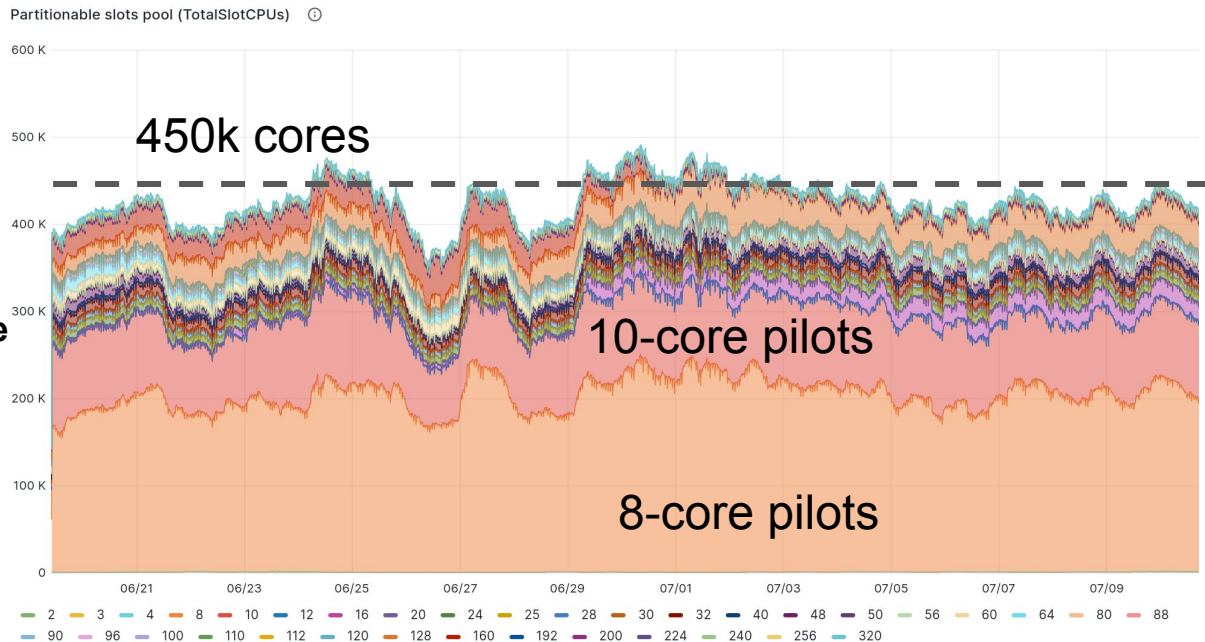
# The CMS SI: multicore pilot model



CMS Operates in a **late-binding** model

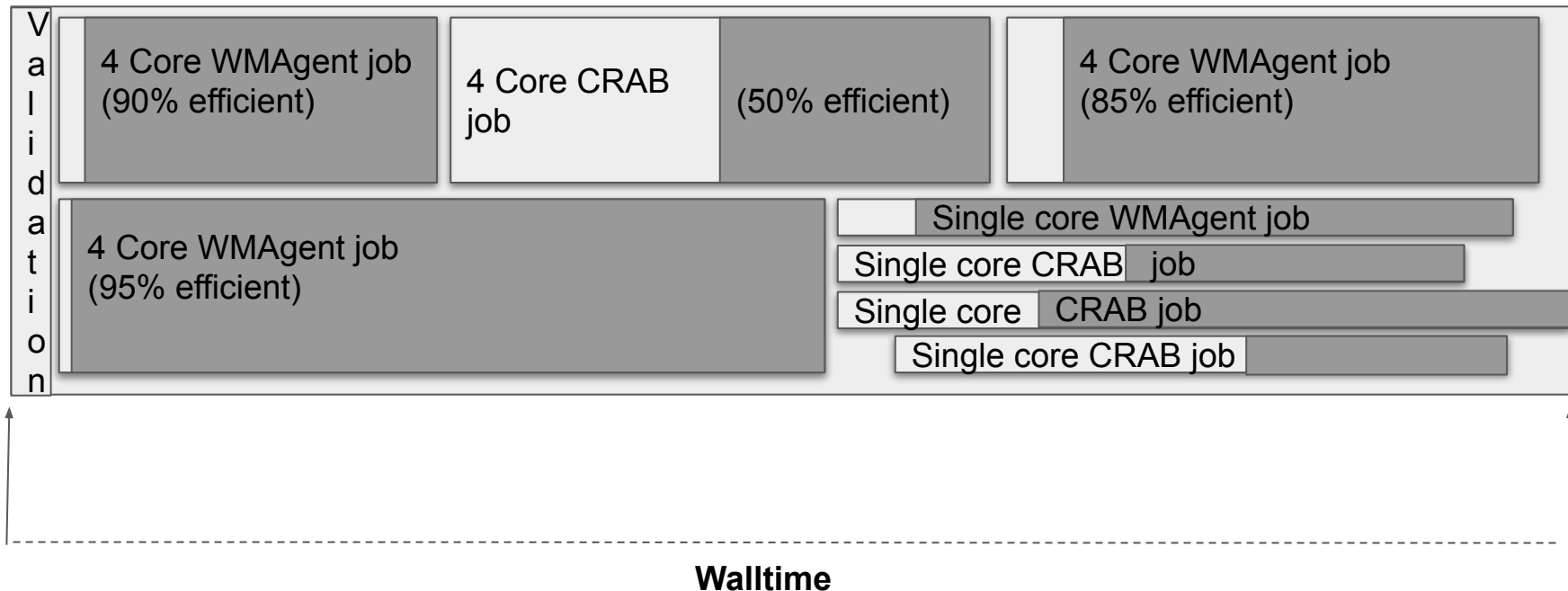
Acquiring resources for the CMS SI:

- Resources mainly acquired via **8-core pilot jobs** submitted to WLCG sites' CEs
- **Flexibility to use non-standard slots**, e.g.: high-mem, whole nodes, etc





# A typical CMS “**pilot job**”: 8-core 48h pilot job executing multiple **payloads**

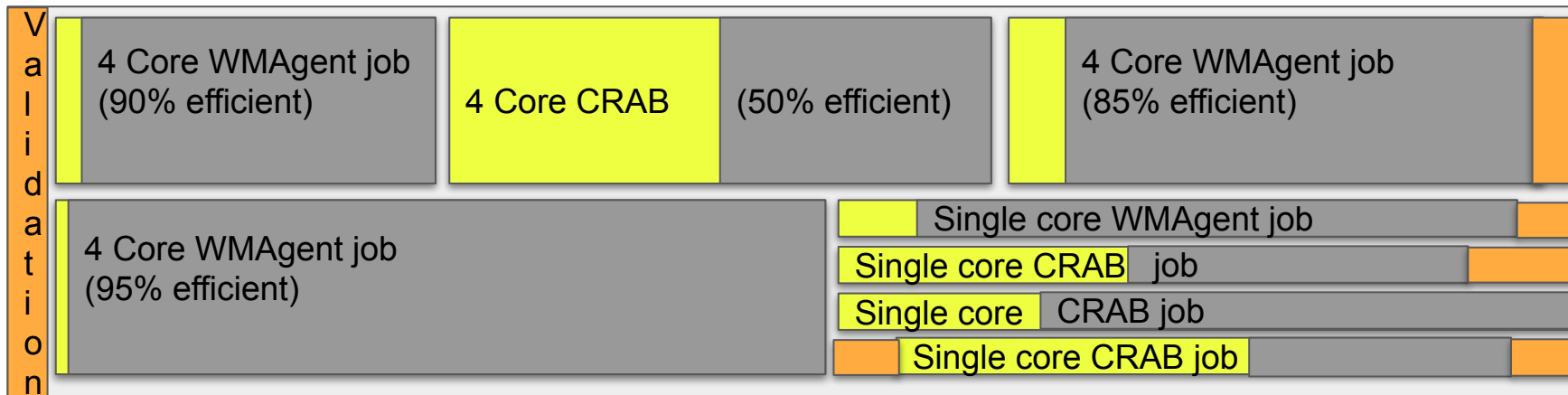


WLCG Efficiency: “**CPU Time / Walltime**”

Legend  
■ CPU Time



# A typical CMS “job”: 8-core 48h pilot job executing multiple payloads



- **Efficiency results** observed and reported by our sites to the **EGI accounting portal** include **scheduling AND payload** Inefficiencies
- **They can be factored and measured independently**
- **Scheduling efficiency typically >95% level for stable sites (T1s and big T2s)**





# Sources of Payload Inefficiencies



- Bootstrapping and staging
- I/O-bound jobs
  - Either **heavy I/O** jobs or jobs that use **remote reads**
- User code (CRAB jobs)
- StepChain (vs TaskChain): Multiple executables linked together as a single payload job
  - Pro: less jobs to manage, reduce intermediate data storage and transfers. 10x **faster turnaround**.
  - Con: diverse resource needs leading to **inefficiencies**
- Non-standard resources or jobs
  - System optimized for **2GB per core of RAM** and **8 hours** of walltime

**Valid reasons for inefficiencies, hard to reduce often.**

**Can we recover CPU cycles in some other way?**



# Strategy to recover unused CPU cycles: overloading pilots

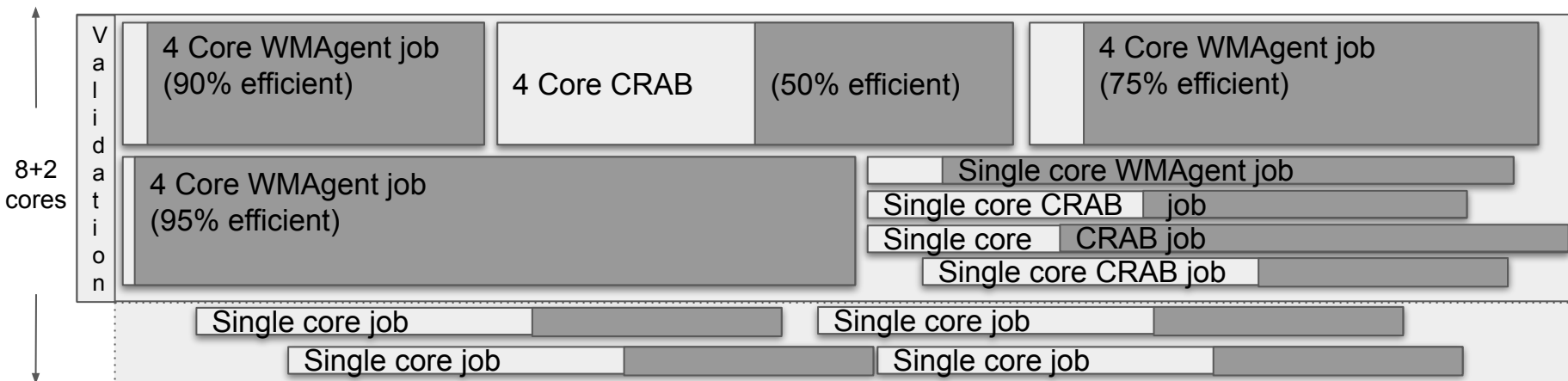


**Idea: Re-definition of the efficiency problem:**

- **Improve CPU utilization efficiency by pushing more workload into the same pilot envelope**
  - Modify pilots so that **they accept more payload jobs into the same resources**
  - **Trivial to implement and test from CMS Submission Infrastructure side**

**Principle:** we want to recover unused CPU, **not gain opportunistic cycles!**

- **Moderate overloading:** add 25% extra CPU cores and memory to the nominal values of our standard 8-core pilot. Provides 2 extra cores, e.g. available to run additional CRAB or production payload





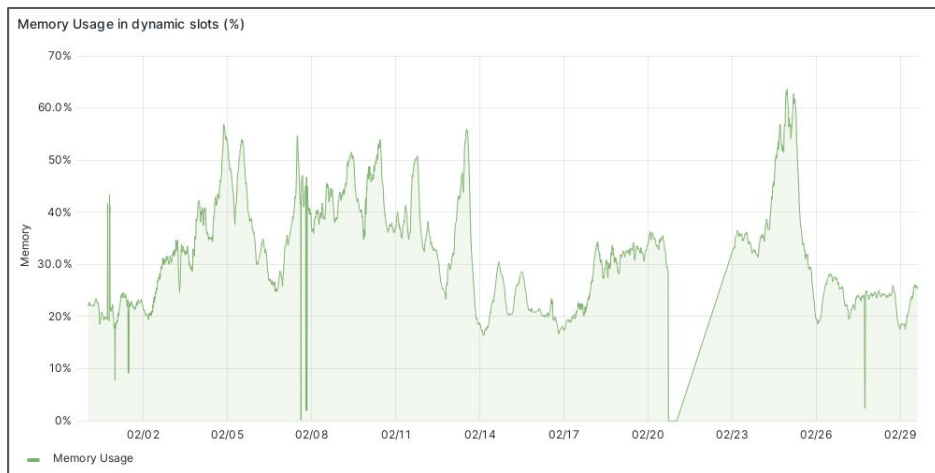
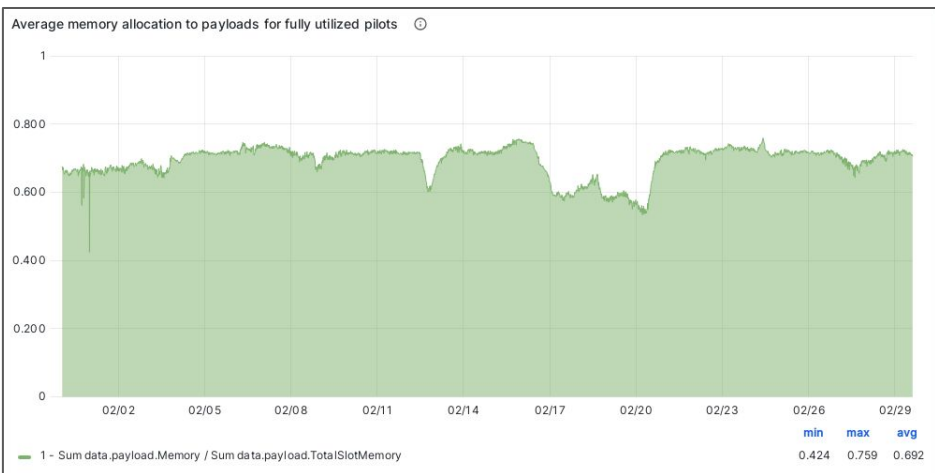
# Available memory for overloading pilots



Do we have enough memory available in the pilots to make moderate overloading work? Analyse memory use for fully used pilots at Tier-1s (e.g. 30 day plots):

- Typically, at least 20% of the partitionable slot memory remains unscheduled for fully occupied pilots
- Then, for dynamic slots running the payload jobs, the average memory utilization is typically below 50%

**There is no memory constraint for a moderate overloading strategy (e.g. +25%)**



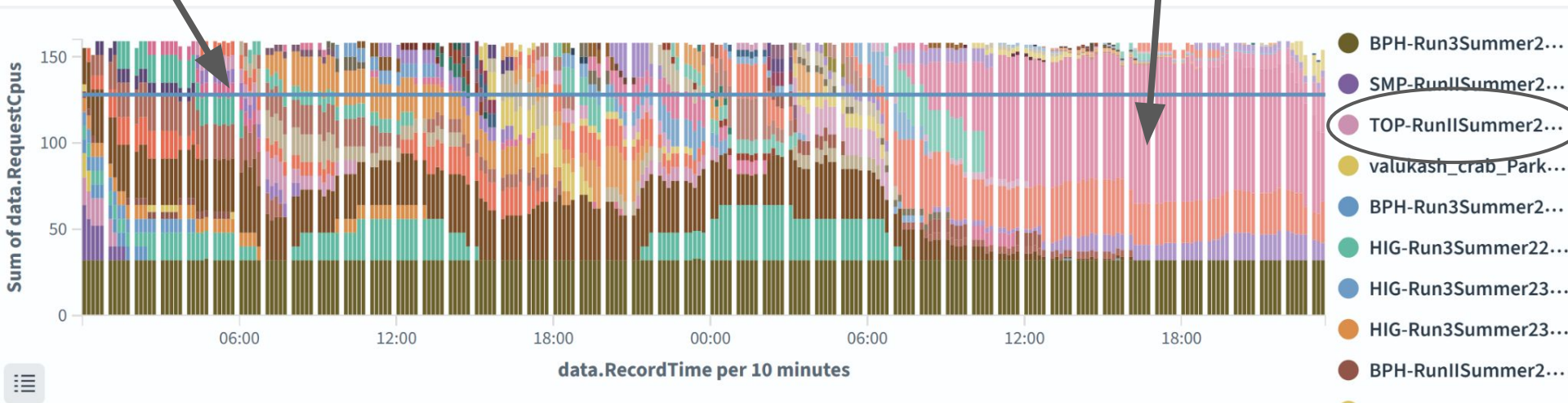


# Overloading: whole node slot real example



Links	Type	Workflow	WMAgent Request Name	Cpu Efficiency Outlier	Cpu Eff	Non Eviction Eff	Eviction Aware Eff Diff	Schedule Eff
<a href="#">McM - PMon - Unified</a>	production	TOP-RunIIISummer20UL16wmLHEGENAPV-00671	cmsunified_task_TOP-RunIIISummer20UL16wmLHEGENAPV-00671_v1_T_240126_100411_2642	0	75.05%	76.41%	1.36%	98.19%

128 cores



128 cores pilot at FNAL overloaded to 160 cores



# Some results



# Preliminary results in 2023: promising!



### Resource Centre PIC — CPU Efficiency (%) by Submit Host and Month (Custom VOs)

Submit Host	Jan 2023	Feb 2023	Mar 2023	Apr 2023	May 2023	Jun 2023	Jul 2023	Aug 2023	Sep 2023	Oct 2023	Nov 2023	Dec 2023	Jan 2024	Total
cc13.pic.es:9619/cc13.pic.es-coadur	71.58%	66.12%	71.66%	90.47%	86.59%	106.37%	95.59%	91.83%	71.93%	91.76%	113.82%	71.65%	88.38%	<b>87.86%</b>
cc14.pic.es:9619/cc14.pic.es-coadur	71.58%	65.44%	75.97%	80.14%	74.64%	84.7%	80.64%	78.43%	58.3%	78.96%	91.25%	65.71%	73.25%	<b>75.4%</b>

[Link](#)

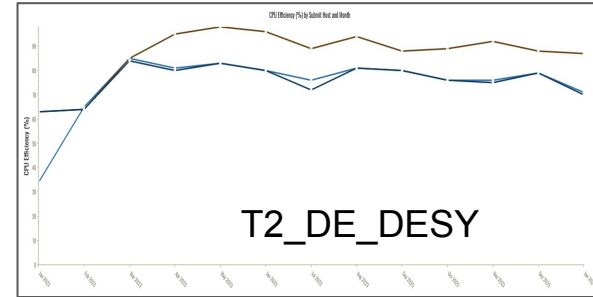
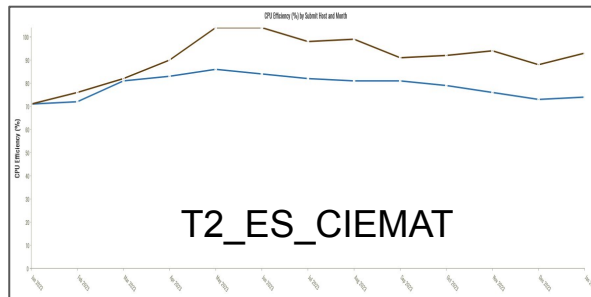
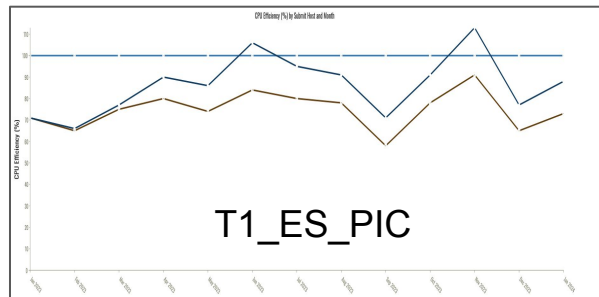
### Resource Centre CIEMAT-LCG2 — CPU Efficiency (%) by Submit Host and Month (Custom VOs)

Submit Host	Jan 2023	Feb 2023	Mar 2023	Apr 2023	May 2023	Jun 2023	Jul 2023	Aug 2023	Sep 2023	Oct 2023	Nov 2023	Dec 2023	Jan 2024	Total
condorce1.ciemat.es:9619/condorce1.ciemat.es-coadur	71.06%	76.15%	82.41%	90.45%	104.08%	104.05%	98.85%	99.25%	91.56%	92.58%	94.57%	88.52%	93.74%	<b>92.94%</b>
condorce2.ciemat.es:9619/condorce2.ciemat.es-coadur	71.01%	72.86%	81.36%	83.56%	86.01%	84.36%	82.8%	81.33%	81.61%	79.01%	76.46%	73.13%	74.97%	<b>79.72%</b>

[Link](#)

### Resource Centre DESY-HH — CPU Efficiency (%) by Submit Host and Month (Custom VOs)

Submit Host	Jan 2023	Feb 2023	Mar 2023	Apr 2023	May 2023	Jun 2023	Jul 2023	Aug 2023	Sep 2023	Oct 2023	Nov 2023	Dec 2023	Jan 2024	Total
grid-htcondorc0.desy.de:9619/grid-htcondorc0.desy.de-coadur	63.75%	64.6%	84.83%	80.32%	83.63%	80.76%	72.91%	81.03%	80.66%	76.58%	75.03%	79.45%	70.95%	<b>76.37%</b>
grid-htcondorc1.desy.de:9619/grid-htcondorc1.desy.de-coadur	63.01%	64.44%	85.31%	95.24%	98.57%	96.76%	89.58%	94.53%	88.8%	89.18%	92.39%	88.34%	87.73%	<b>87.87%</b>
grid-htcondorc2.desy.de:9619/grid-htcondorc2.desy.de-coadur	34.25%	65%	85.19%	81.33%	83.52%	80.72%	76.68%	81.9%	80.86%	76.24%	76.28%	79.21%	71.67%	<b>78.37%</b>







# Overloading pilots expansion



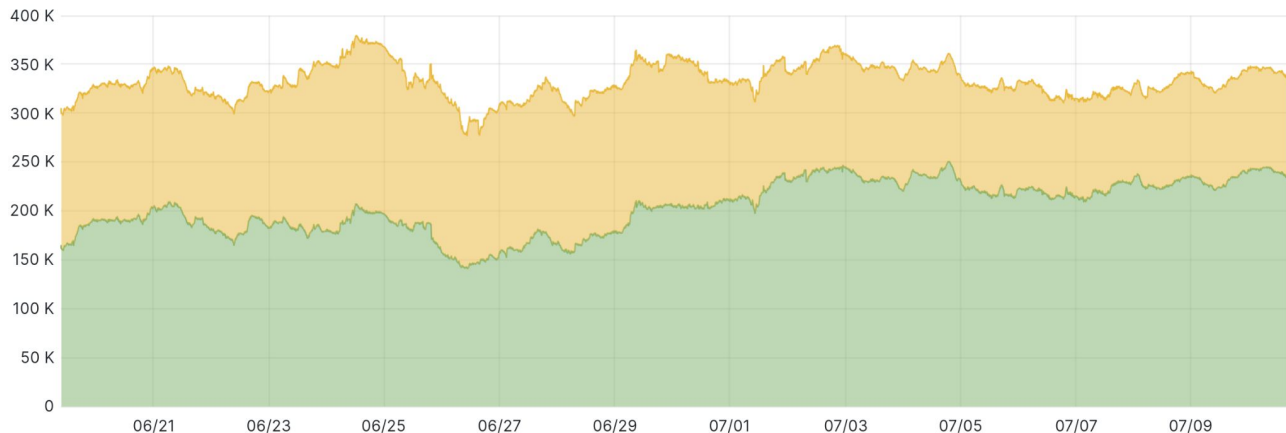
- After promising results at a few sites in 2023, CMS decided to enable overloading at more resource providers starting in January 2024:
  - All **Tier-1 sites** (\*)
  - **A set of good Tier-2s** (average scheduling efficiency already at 95%)

Still keep ~50% unchanged for each site in order to compare results

CPU cores for jobs running on overloaded pilots for last month. Global pool only.

 Overloaded cores  
 Normal cores

Slot Overloading status (CPUs) ⓘ





# Efficiency Improvements



- From the pilot logs, total walltime and used CPU time can be extracted. We can thus calculate the CPU efficiency as measured by the resource providers and reported to EGI
- Efficiencies based on pilot logs executed over last 3 months can be used to compare overloaded vs non-overloaded pilots
- **Significant improvement** of CPU utilization efficiency when allowing overloaded pilots

Site Name	Normal Efficiency	Overloaded Efficiency	Efficiency increase
T1_FR_CCIN2P3	84.57%	95.78%	11.21%
T1_IT_CNAF	79.81%	84.62%	4.81%
T1_UK_RAL	72.41%	85.00%	12.60%
T2_BE_IHHE	77.21%	89.17%	11.96%
T2_DE_RWTH	55.68%	78.01%	22.33%
T2_EE_Estonia	73.81%	85.47%	11.66%
T2_ES_CIEMAT	72.65%	88.40%	15.75%
T2_IT_Bari	74.48%	78.87%	4.39%
T2_IT_Legnaro	75.77%	85.88%	10.11%
T2_IT_Rome	60.25%	73.38%	13.13%
T2_UK_London_IC	66.14%	72.94%	6.80%
T2_US_MIT	63.74%	70.61%	6.87%
T2_US_UCSD	52.91%	67.64%	14.73%
T2_US_Vanderbilt	74.95%	77.04%	2.09%
T2_US_Vanderbilt	56.05%	65.64%	9.58%
T2_US_Wisconsin	78.03%	89.66%	11.62%

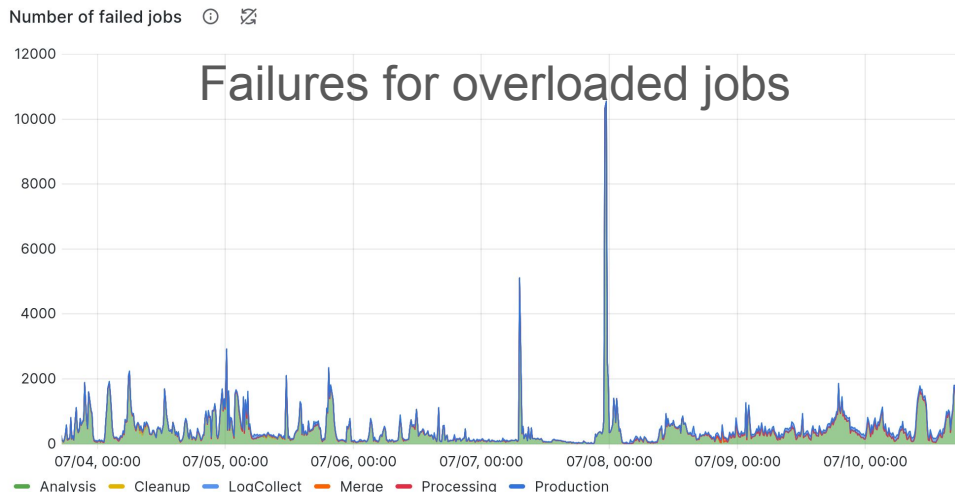
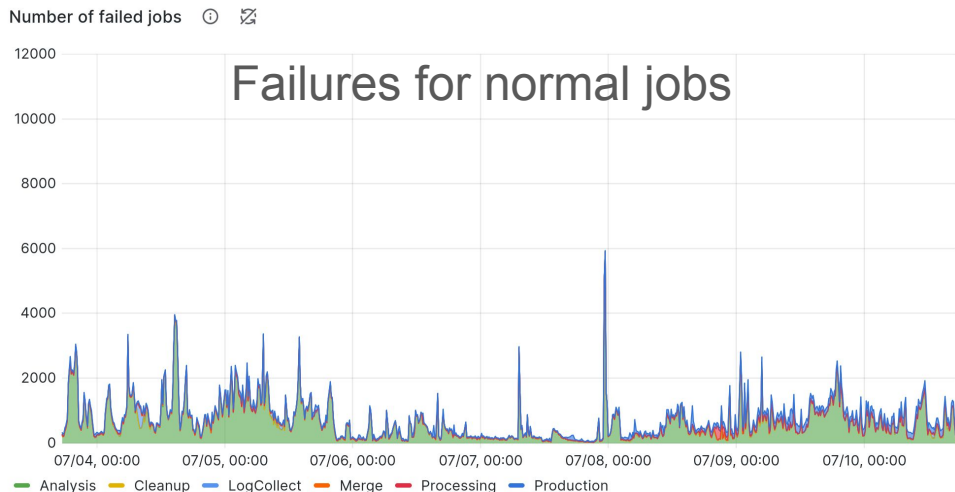




# Job Failures Comparison



Absolute number of job failures in the last week grouped by job type



No impact in terms of job failures



# Impact on event rates



# Event throughput study



While it seems clear we are recovering CPU cycles, need to evaluate the impact on event processing rate. We have analysed this for diverse CMS workload types, for diverse sites, then compared evts/s for jobs running on overloaded vs. non-overloaded pilots.

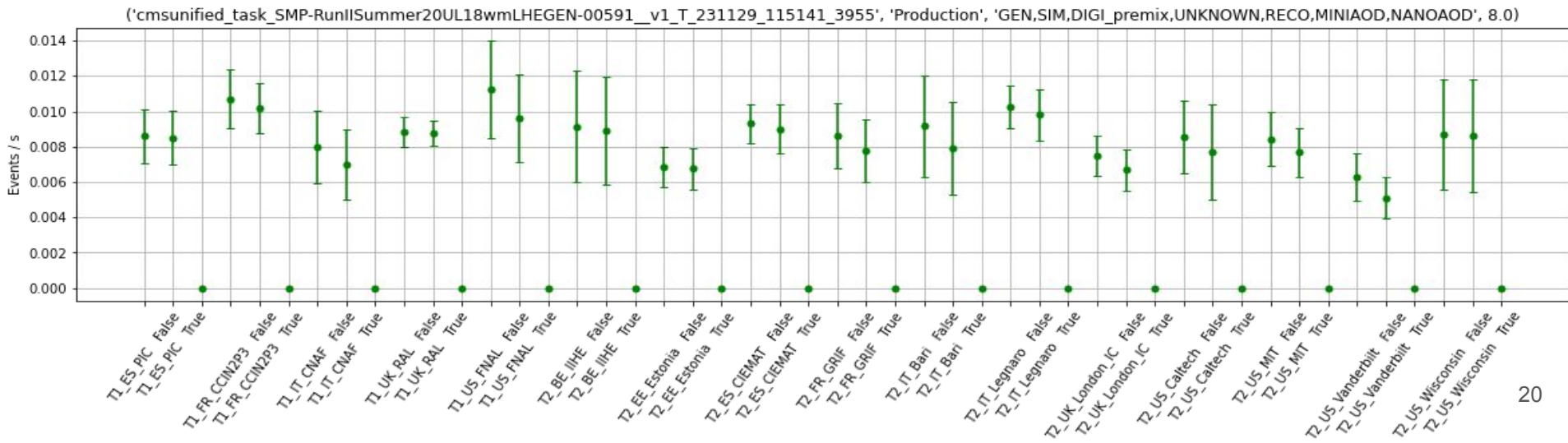
Results: while average event rate appears to be lower for the overloaded pilots at a number of sites, in fact we observe the overloading effect to be smaller compared to the variability between jobs of the same workflow and between sites



# Event Rates comparison (I)



- Compared event rates for all workflows in April, May and June, classifying jobs by execution site, workflow type, etc.
- As a first example, notice this full StepChain simulation workflow with the highest number of jobs (~450k jobs in total)
  - Results: event processing rates present high variability, ranging from 0,005 to 0,014 evts/s
  - Overloading effect on event throughput smaller than dispersion between jobs at the same site and across sites

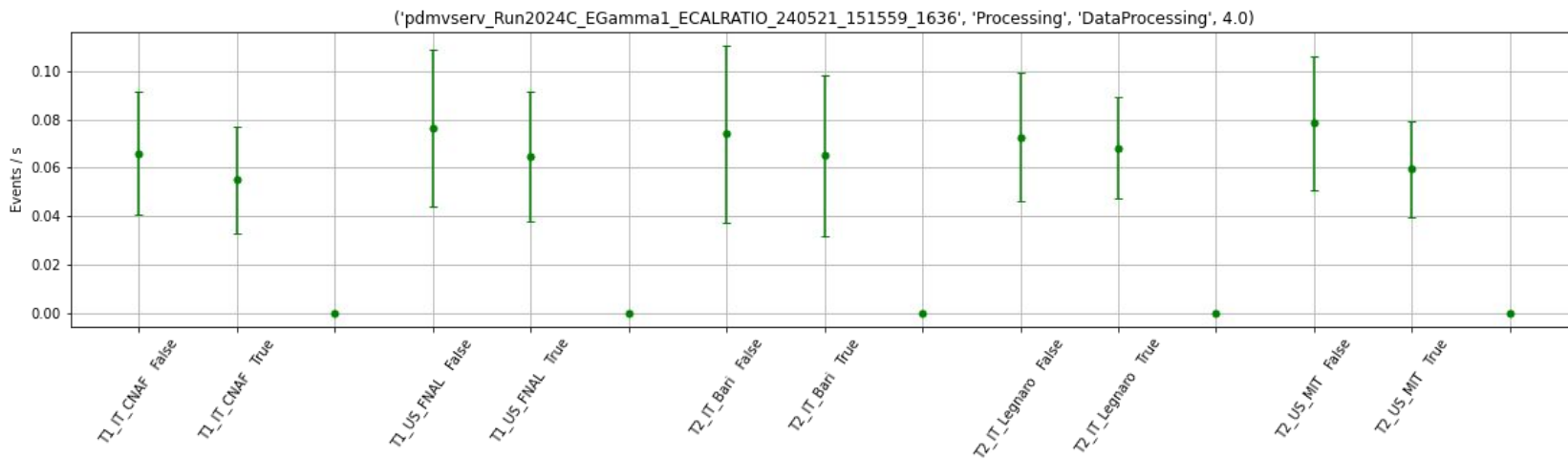




# Event Rates comparison (II)



- Data processing workflow with most jobs in the last three months
  - ~20k jobs in total
  - Event rates range from 0,04 to 0,1





# Conclusions



- CMS has been operating the **biggest condor pool in the world** since ten years
  - Average of 350k cores in the Global Pool and 450k in total.
  - From 1 to 10 millions of jobs daily
- Moderately **overloading of our pilots** allows CMS to **recover** between 5% to 20% of idle **CPU cycles**
  - Extra 30k cores (re)gained using this strategy
- **No impact** observed from the site perspective on **job error rates**, CPU or memory (ab)use, etc.
- No significant impact on **event processing rate** has been observed
  - Higher variability between jobs of the same workflow and between sites than an overloading true/false effect

Many thanks for the HTCSS team for all the help and the fruitful collaborations over the years!

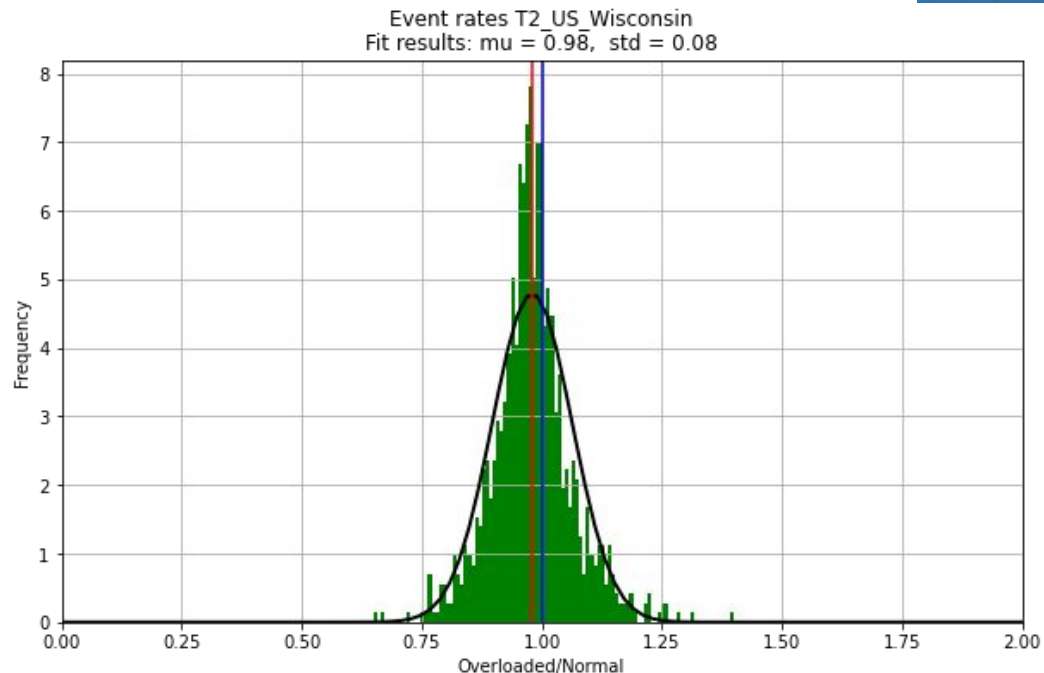


# Backup Slides



# Event rate distribution for Wisconsin

- 958 tasks run at T2\_US\_Wisconsin in the past three months
- Taking event rate average of: overloaded jobs over normal jobs, and plotting distribution
- 100 bins between 0 and 2



Event rate penalty introduced by overloaded jobs is around 2%





# CPU efficiency and event penalty. Last 3 months

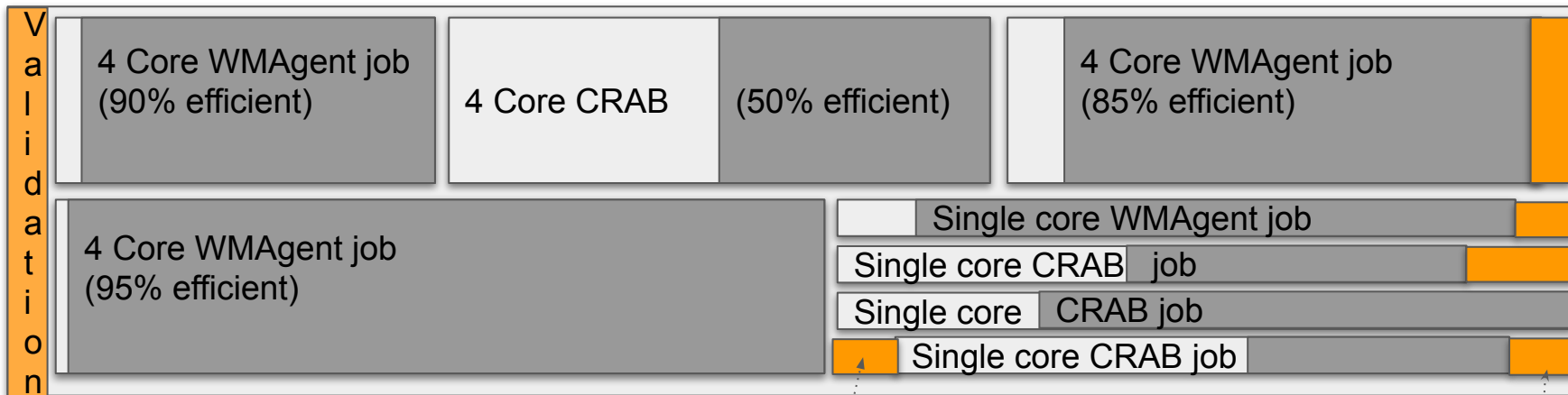


- In most cases efficiency increase benefit is higher than event rate penalty.
- Deviating results for some sites, need to be investigated

	Efficiency increase	Event rate penalty
T1_ES_PIC	6%	1%
T1_FR_CCIN2P3	11%	3%
T1_IT_CNAF	5%	11%
T1_UK_RAL	13%	1%
T2_BE_IIHE	12%	1%
T2_DE_RWTH	22%	5%
T2_EE_Estonia	12%	2%
T2_ES_CIEMAT	16%	7%
T2_FR_GRIF	11%	4%
T2_IT_Bari	4%	6%
T2_IT_Legnaro	10%	3%
T2_IT_Rome	13%	4%
T2_US_MIT	7%	7%
T2_UK_London_IC	7%	12%
T2_US_UCSD	15%	6%
T2_US_Wisconsin	12%	2%



# A typical CMS “job”: 8-core 48h pilot job executing multiple payloads



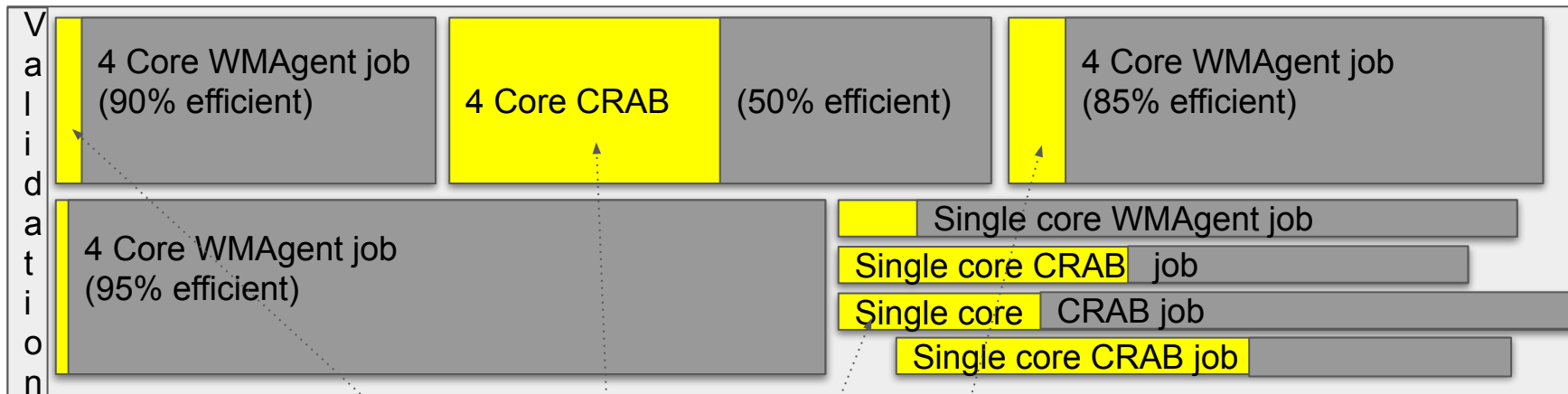
Jobs can be negotiated

Draining starts

Scheduling Inefficiencies



# A typical CMS “job”: 8-core 48h pilot job executing multiple payloads

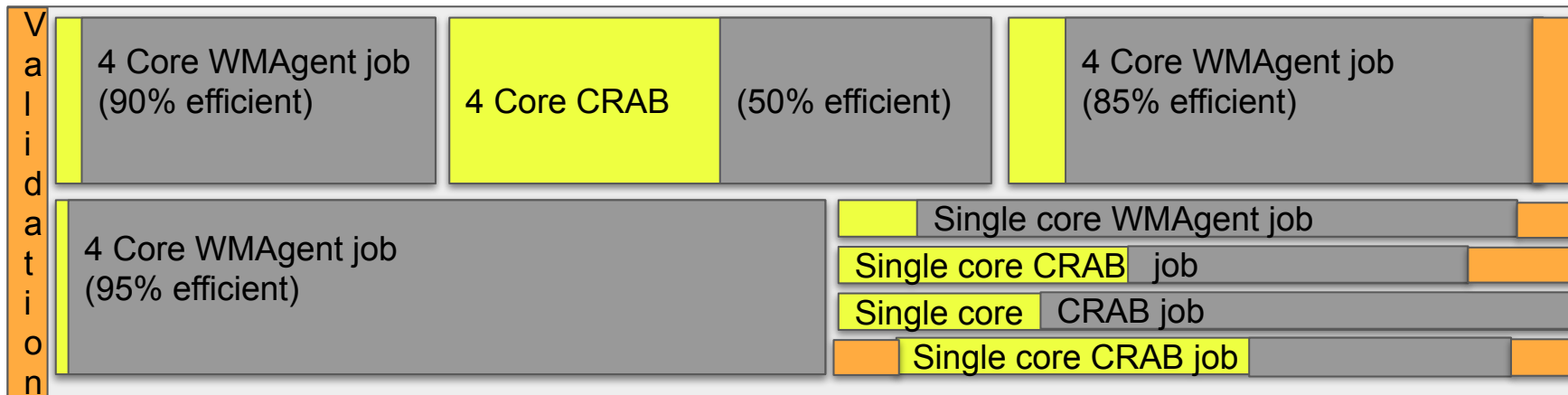


## Payload Inefficiencies

(Uses **payload walltime** as denominator)



# A typical CMS “job”: 8-core 48h pilot job executing multiple payloads



- **Efficiency results** observed and reported by our sites to the **EGI accounting portal** include **scheduling AND payload** Inefficiencies
- **They can be factored and measured independently**