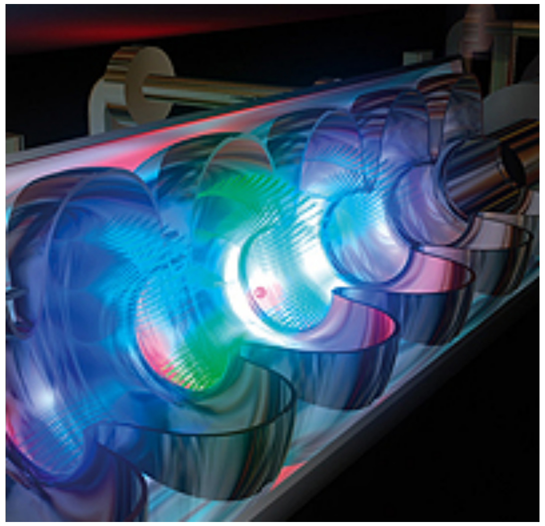
Examination of dynamic partitioning for multi-core jobs in the Tokyo Tier-2 center

Tomoe Kishimoto

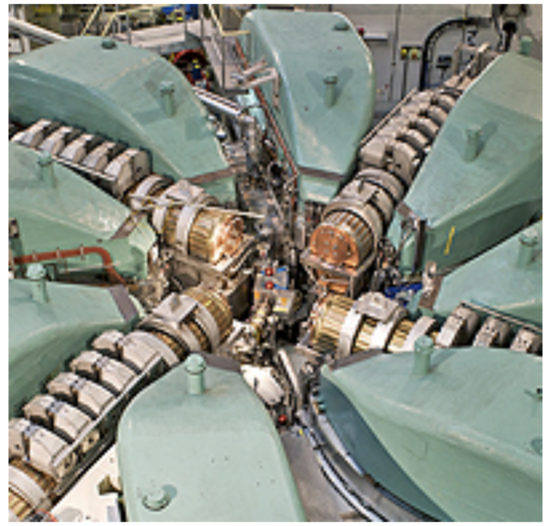
ICEPP, The University of Tokyo

Mar. 10 2017

International Center for Elementary Particle Physics



R&D for ILC

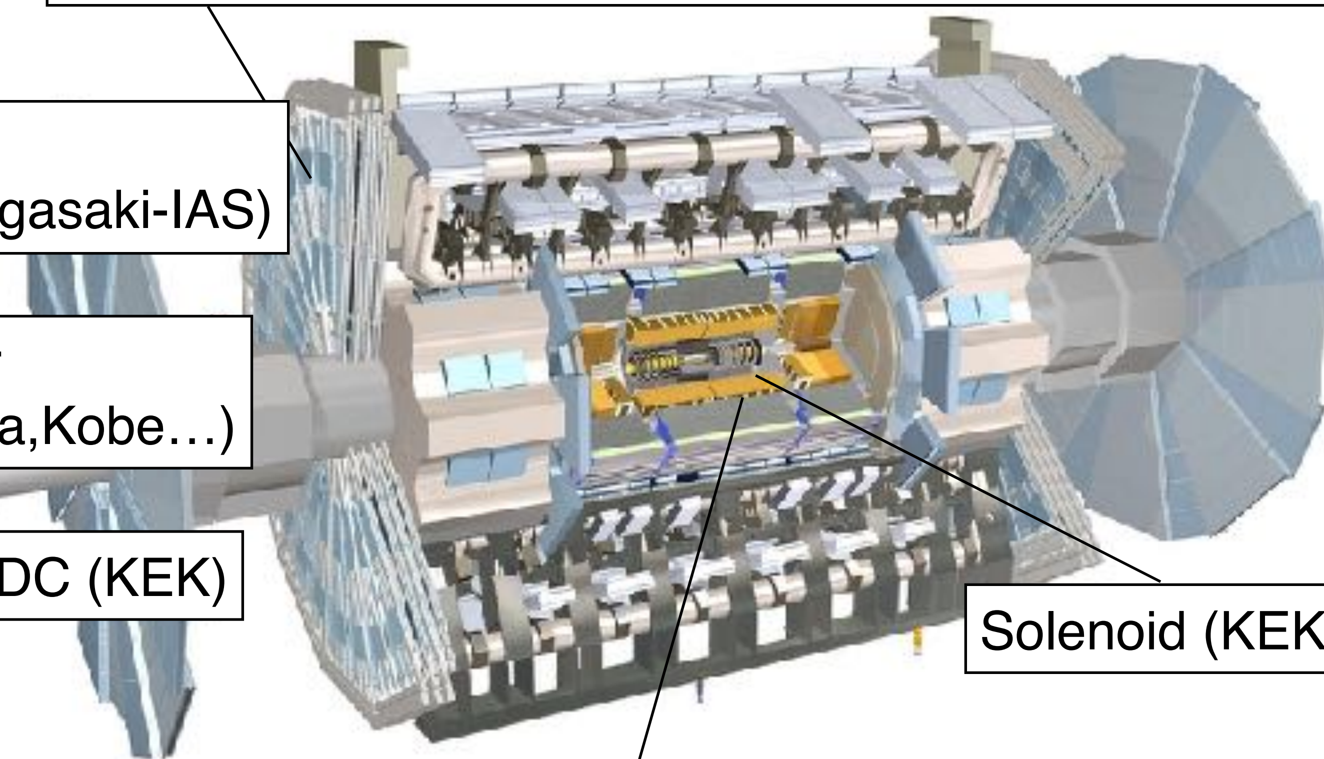


MEG at PSI

東京大学
素粒子物理国際研究センター
International Center for Elementary Particle Physics
The University of Tokyo

ATLAS Japan

ATLAS



Computing Center (Tokyo ICEPP)

✓ **Tokyo Tier-2 is the the only WLCG site in ATLAS-Japan**

ICEPP regional analysis center

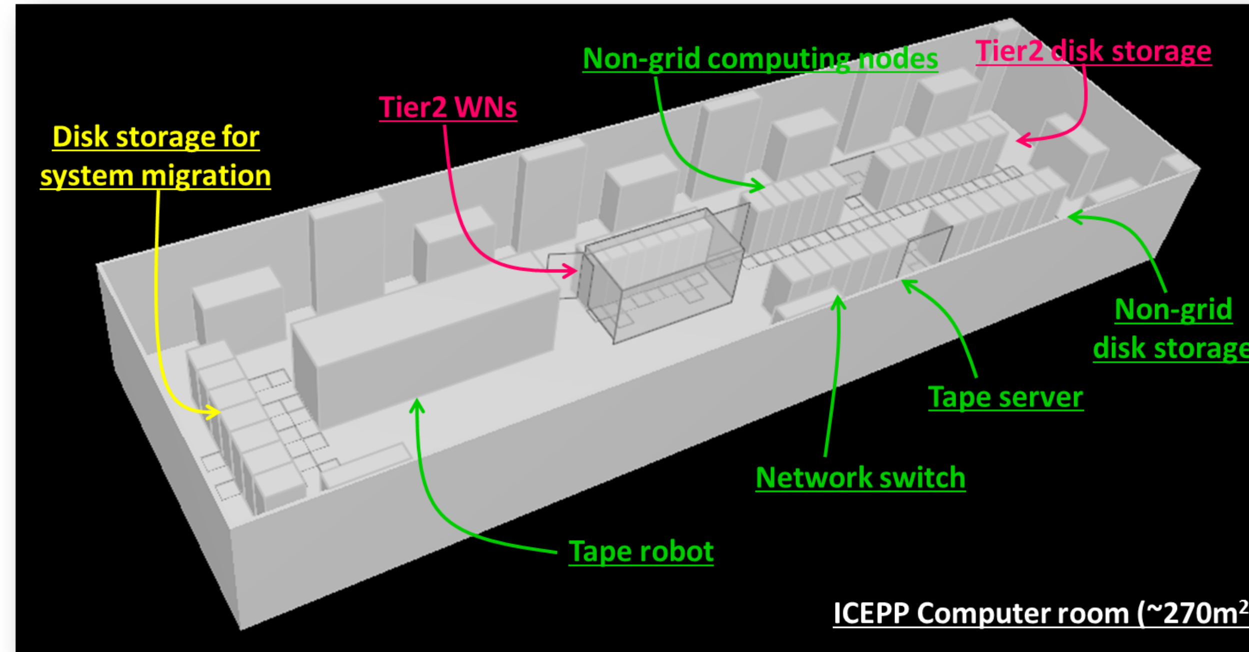
✓ Resource overview

- Support only ATLAS VO in WLCG (**Tier2**) and provide ATLAS-Japan dedicated resources (local use)
- Hardwares are leased, and are replaced in every three years
- **4th system** is running since Jan. 2016
  - ~10000 CPU cores including service instances and ~10 PB disk storage (T2 + local use)

Single VO and uniform architecture

✓ Operation team

- 5 university staffs + 2 SEs from company



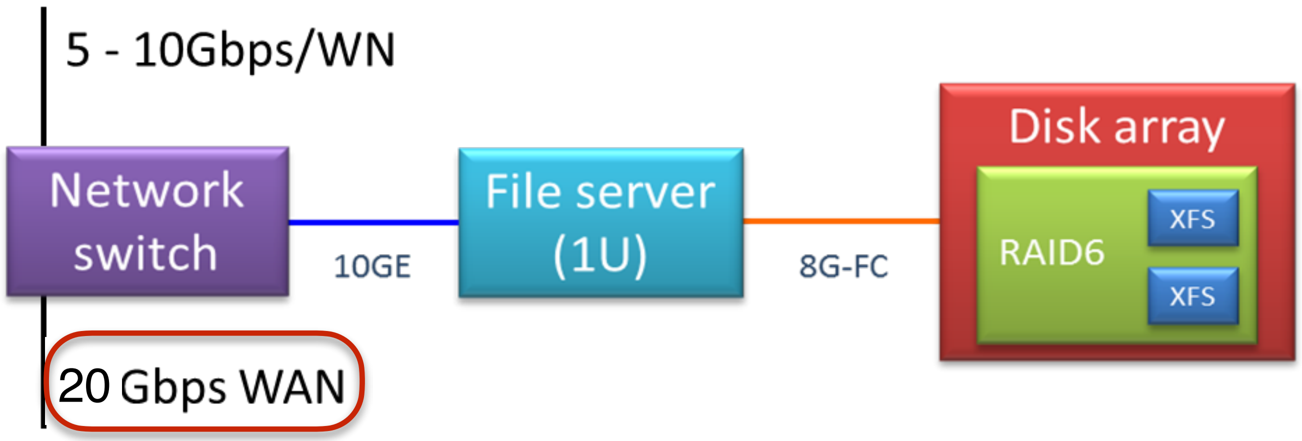
4th system

worker nodes

Disk arrays

Tier2 configuration of the 4th system

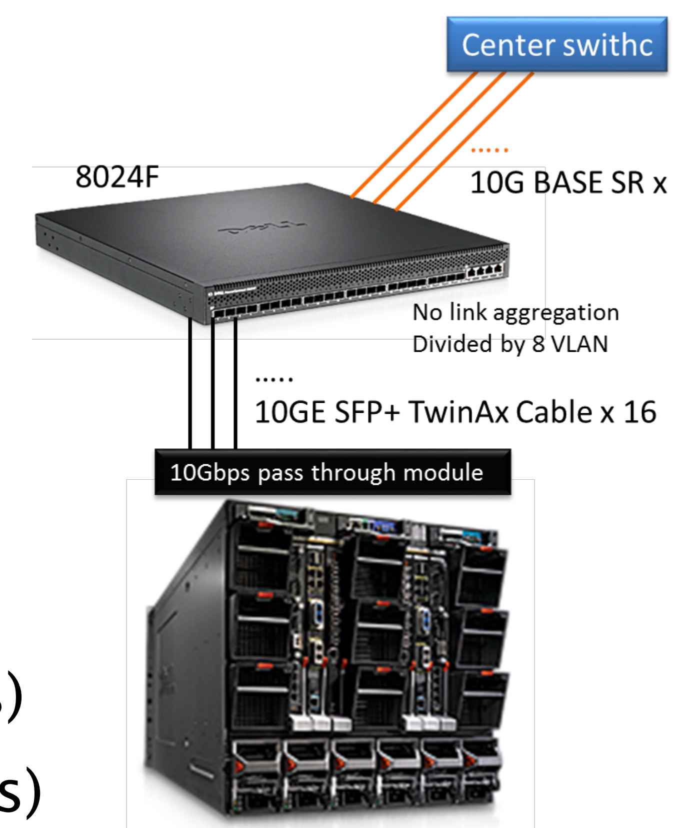
Disk server (×48)



- 132TB × 48 servers
- **Total capacity is 6.336PB** (DPM)
- 10Gbps NIC (for LAN)
- 8G-FC (for disk array)
  - 500~700MB/sec (sequential I/O)

Worker node (×256)

- 24 CPU cores/node, **total 6144 CPU cores**
- Memory: 2.66GB/core
- 10Gbps pass through module (SFP+ TwinAx cable)
- Rack mount type 10GE switch (10G BASE SR SFP+)
- Band width:
  - For 160 WNs: 10Gbps/2nodes (max 10Gbps,min 5Gbps)
  - For 96 WNs: 10Gbps/4nodes (max 10Gbps,min 2.5Gbps)



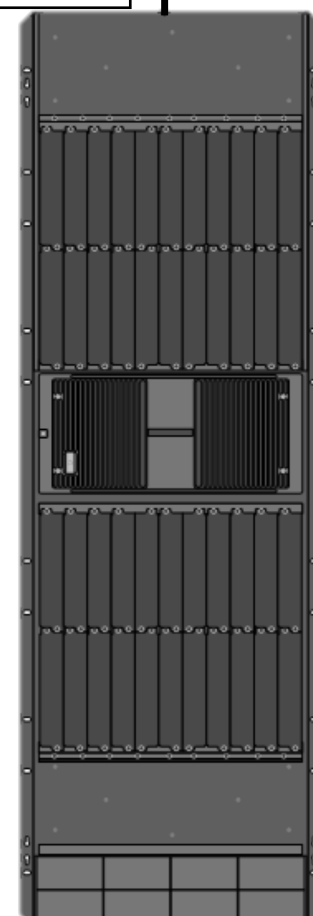
Tier2 configuration of the 4th system

Network

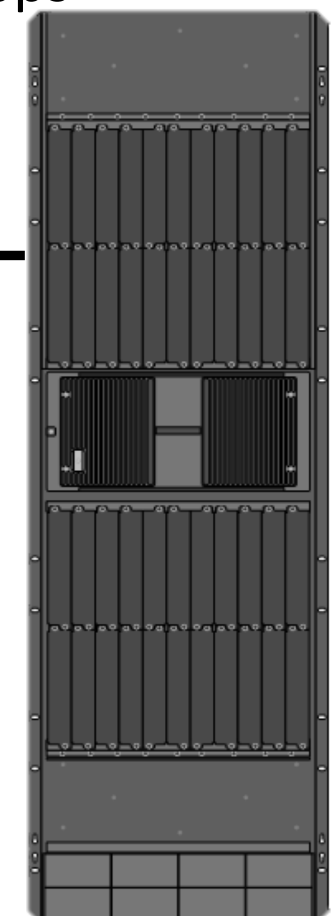
SINET5 — 20Gbps to WAN

Brocade MLXe-32 x 2
Non-blocking 10Gbps

Main switches: continued use from 3rd system



Inter link
16 x 10Gbps



10GE (SFP+)
176 ports

10GE (SFP+)
176 ports

Tier2

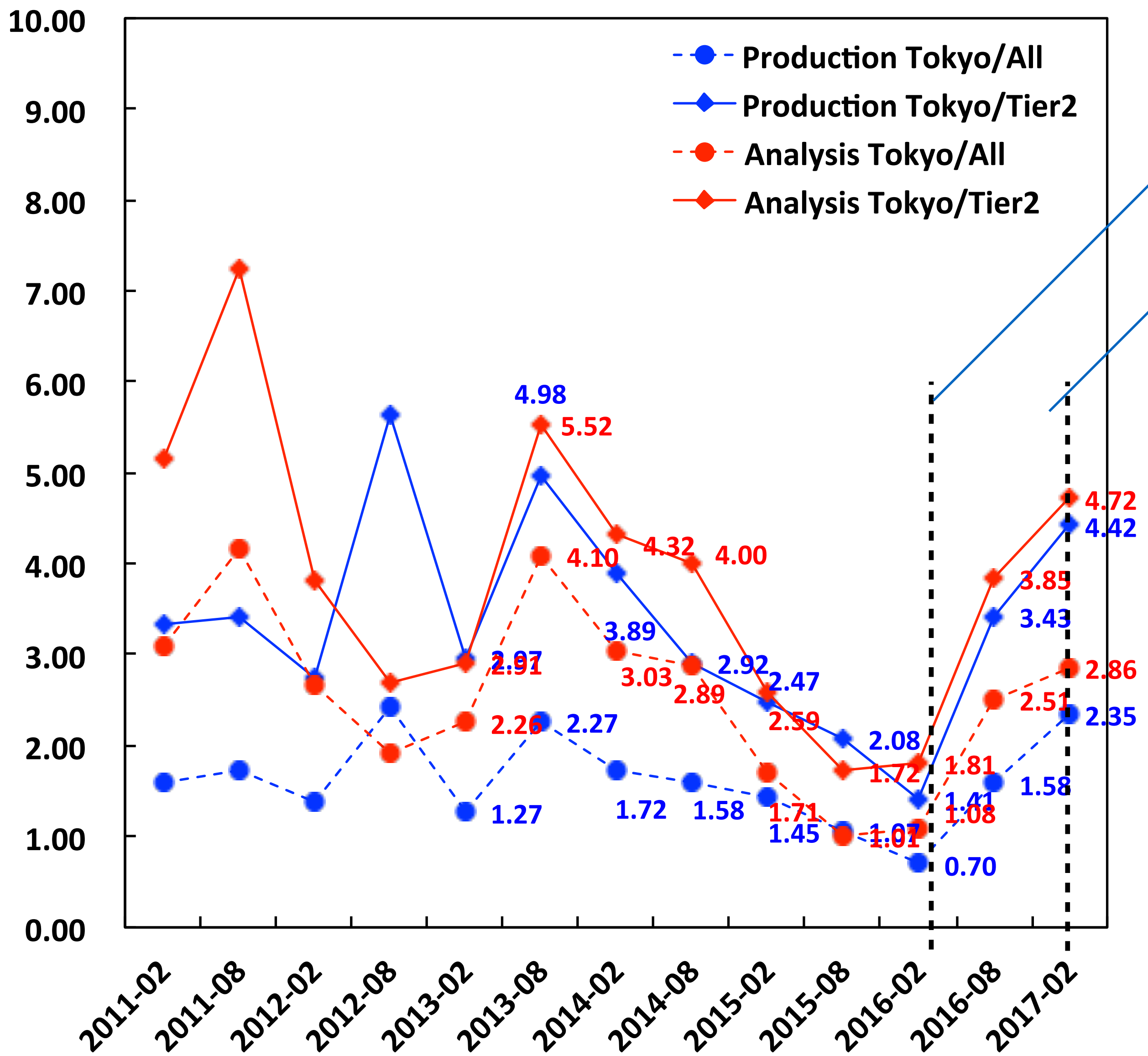
DPM file servers
LCG service nodes
LCG worker nodes

Non-grid

GPFS/NFS file servers
Tape servers
Non-grid service nodes
Non-grid computing nodes

Status in ATLAS

✓ Fraction of number of completed jobs



3840 CPU cores deployed

6144 CPU cores deployed

Contains ambiguities on the multicore jobs
Slot allocation:
analysis : score prod : 8core prod
= 20% : 20% : 60%

✓ Results in the last six months:

- Production, **4.4% (Tier2)** – 2.4% (All)
- Analysis, **4.7% (Tier2)** – 2.9% (All)

← Good contributions

# of ATLAS-J authors ∼ 100
# of ATLAS authors ∼ 3000

✓ > 99 % site availability has been achieved using the 4th system

ATLAS job types

✓ Summary of ATLAS job types:

Multi-cores jobs were developed to provide an efficient memory sharing

	Required cores	Target share	
Production jobs	1	20%	Event generation, etc
	8	60%	Simulation, Reconstruction, etc
Analysis jobs	1	20%	User analysis

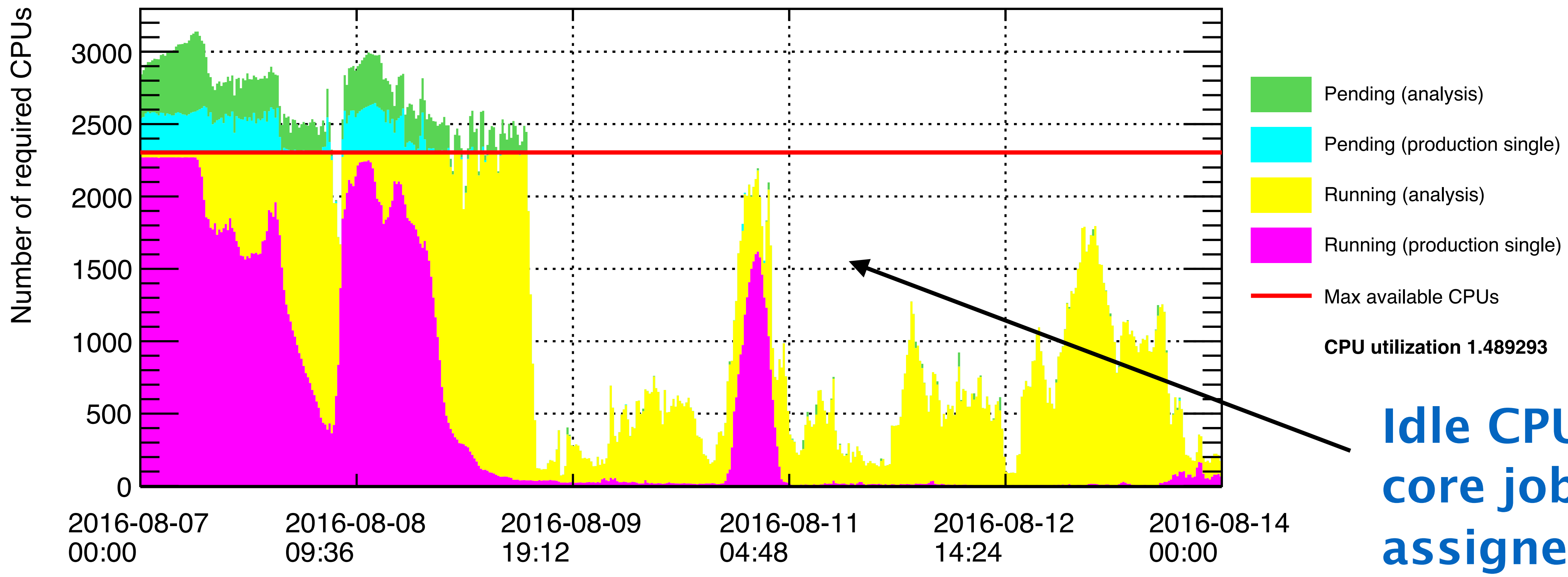
✓ Tokyo Tier2 site has been processing single-core jobs and multi-core jobs separately using dedicated WNs and CEs

- Simple configuration
- CREM-CE + Torque/Maui

✓ However, we have often observed idle CPU cores due to this **static partitioning**

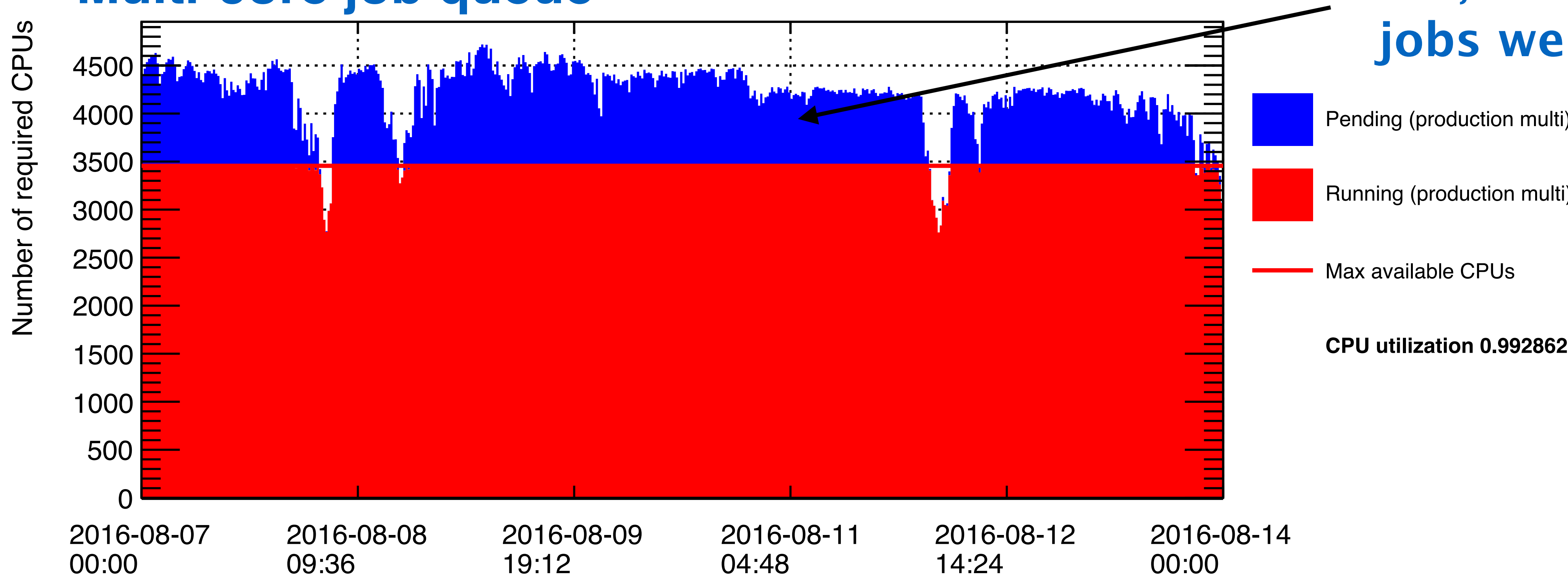
Problem in static partitioning

Single-core job queue



Idle CPUs due to single-core jobs were not assigned

Multi-core job queue



But, many multi-core jobs were waiting...

Dynamic partitioning

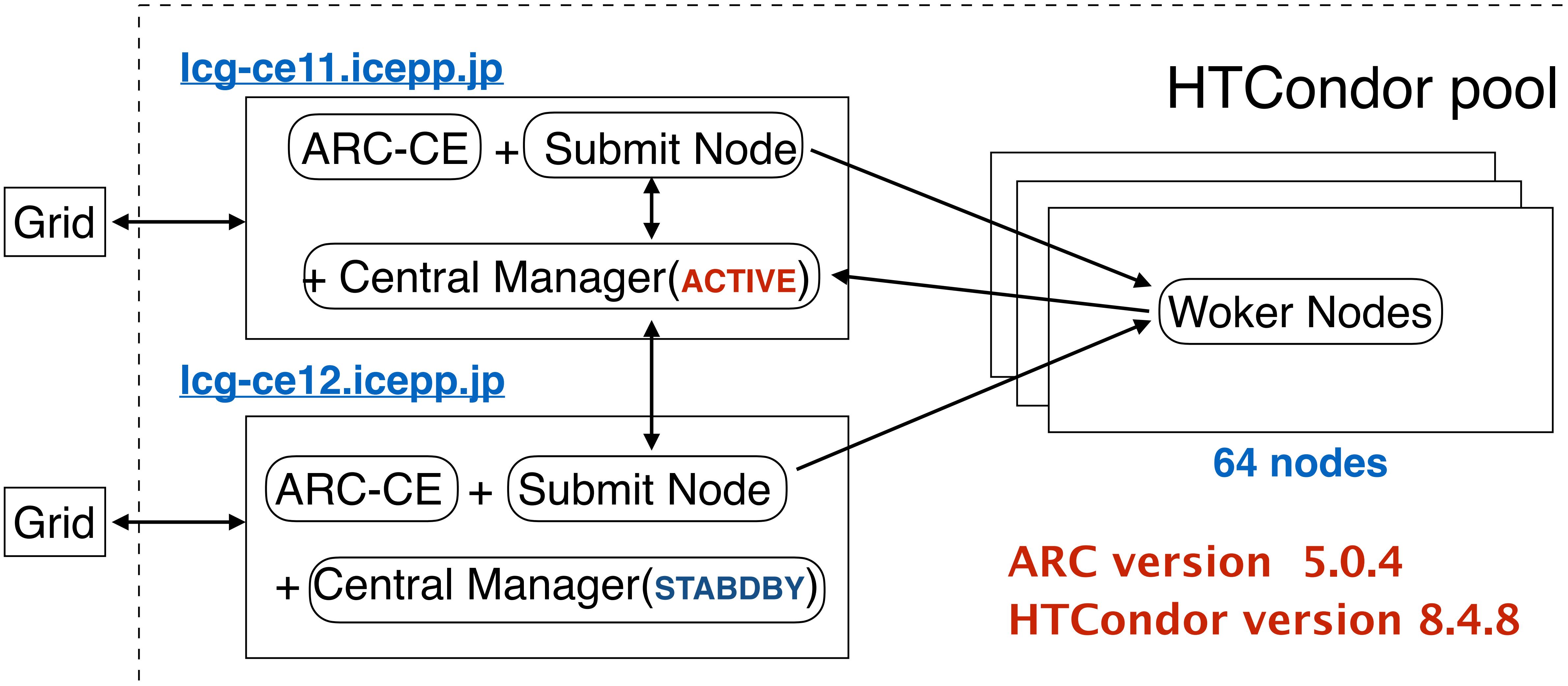
✓ We started to evaluate an implementation of the dynamic partitioning using HTCondor batch scheduler

- HTCondor is becoming standard in WLCG
- Well documented

✓ In Nov. 2016, we deployed a small cluster (1536 cores) of HTCondor (+ ARC-CE) into the production:

CE	Batch system	CPU cores	Job types
CREAM	Torque/Maui	4608 (192 WNs)	Single- and Multi-core jobs (static partitioning)
ARC	HTCondor	1536 (64 WNs)	Single- and Multi-core jobs (**dynamic partitioning**)

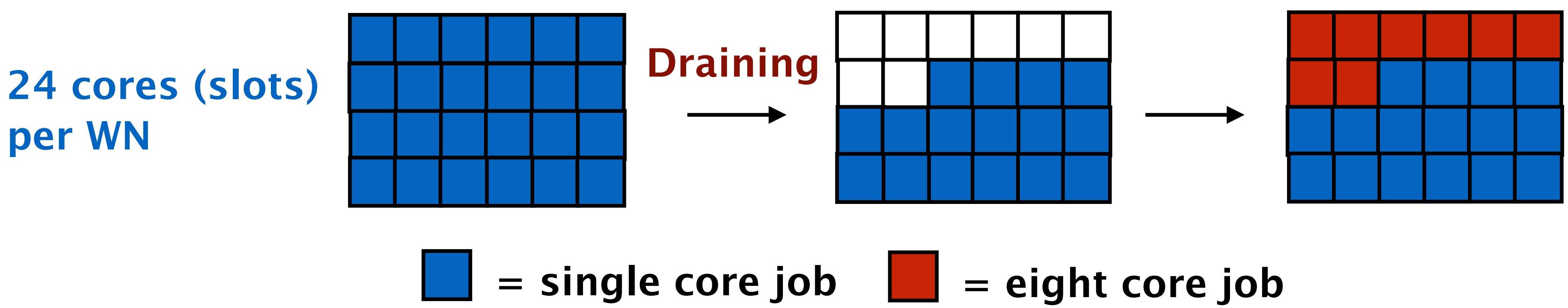
ARC–CE + HTCondor configuration



- ✓ Two ARC–CEs for redundancy
- ✓ High availability of central managers
- ✓ Draining for multi–core job is managed by **Defrag daemon**
  - Need optimizations

Optimization of draining

✓ In the dynamic partitioning, draining of single core jobs is necessary in order to dispatch new multi jobs



✓ Defrag daemon in HTCondor is used to manage this draining

✓ Need to optimize several parameters

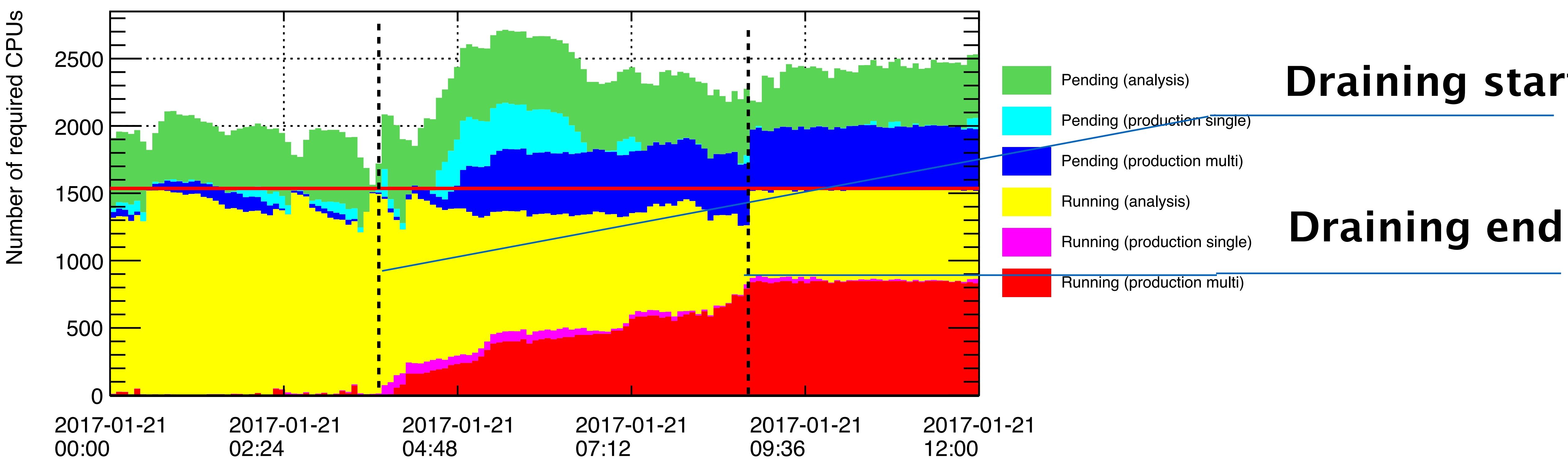
- When should draining start/end ?
- Which machines are more desirable to drain ?
- How many machines are drained at once ?

When should draining start/end?

$N^{multi-core}_{target}$: target share for multi-core jobs

- ✓ Criteria for stating the draining:
  - $(N^{multi-core}_{runnning} < N^{multi-core}_{target})$ && $(N^{multi-core}_{waiting} > 0)$
- ✓ Criteria for finishing the draining
  - $(N^{murti-core}_{runnning} >= N^{multi-core}_{target})$

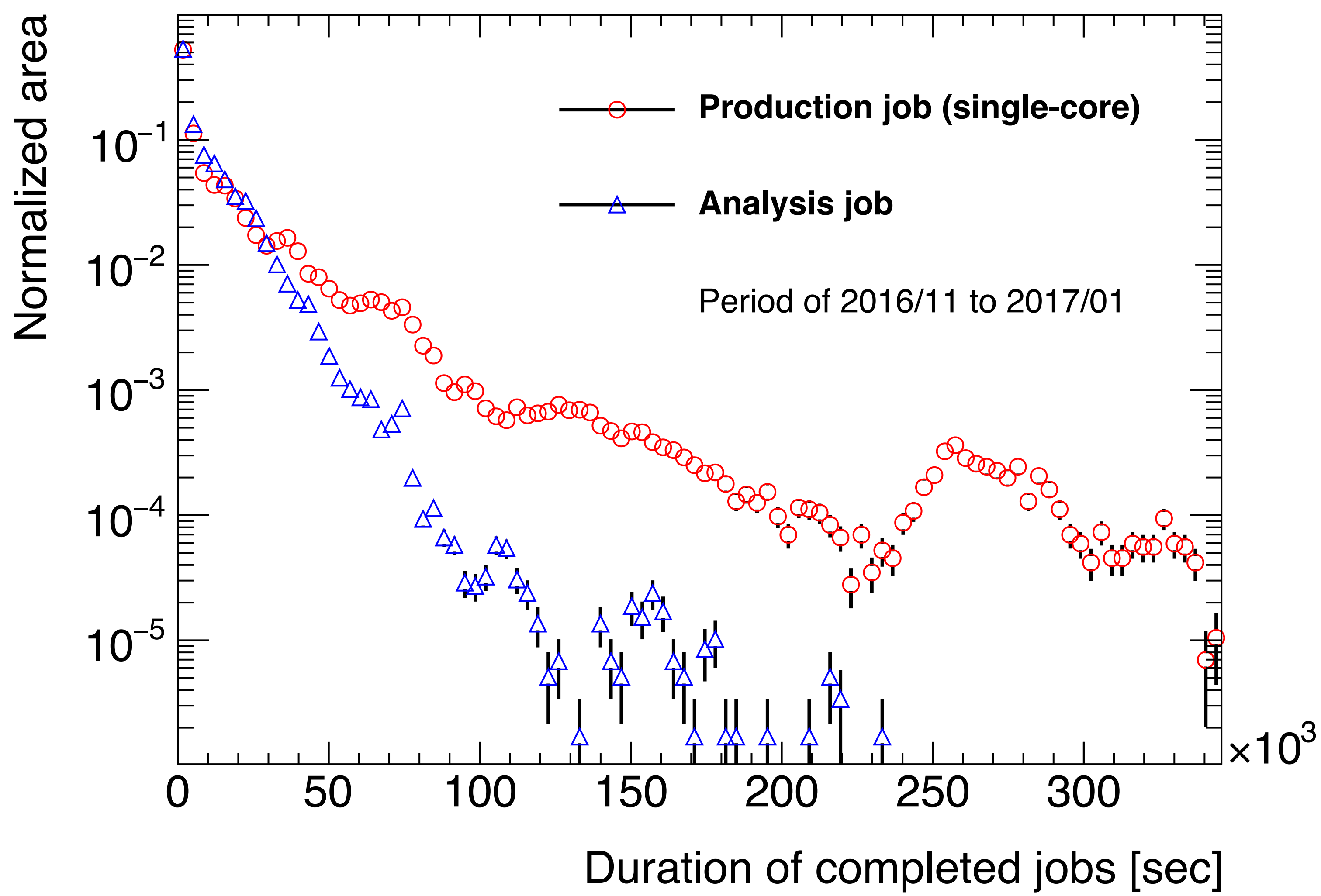
A cron script monitors $N^{multi-core}_{runnning}$ and $N^{multi-core}_{waiting}$, and sets `DEFRAG_WHOLE_MACHINE_EXPR` in Defrag daemon



→ The criteria are woking well in the production system

Which machines are desirable to drain?

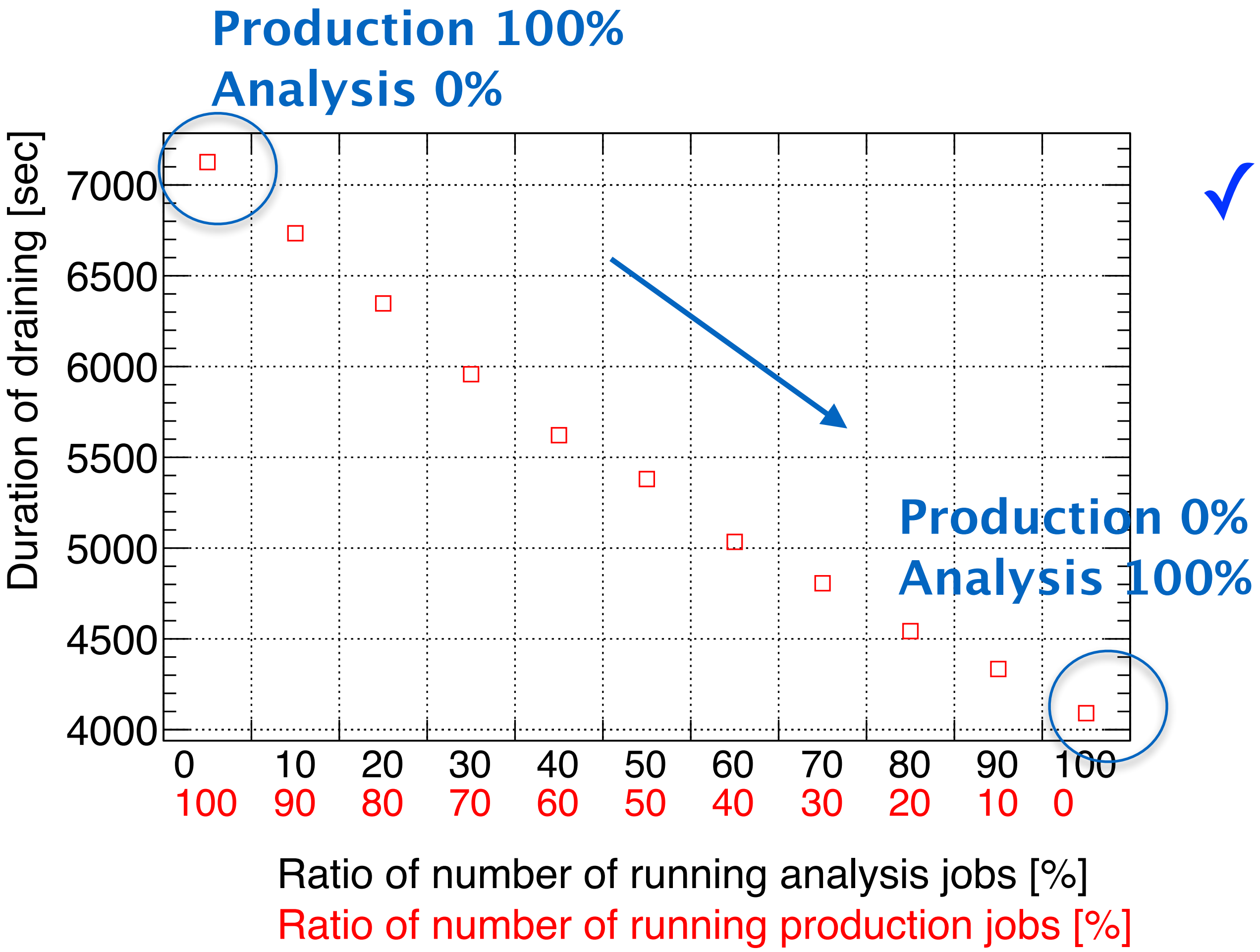
✓ Observed job duration in the HTCondor system:



	Average duration [sec]	
	Production (single-core)	Analysis
Nov.	1.2×10^4	0.7×10^4
Dec.	1.8×10^4	0.8×10^4
Jan.	1.0×10^4	0.7×10^4

✓ In the Tokyo Tier2 center, user analysis jobs tend to finish earlier than production jobs

Which machines are desirable to drain?



- ✓ Duration of draining until 8 slots are available in a WN :
  - Estimated by simulation, using the observed job duration as inputs

→ Desirable to drain WNs, which have many analysis jobs

- ✓ A cron script has been developed:
  - Monitors WNs status, and set `DEFRAG_RANK` for each WN
  - `DEFRAG_RANK` = $(N^{production}_{runnning})*1 + (N^{analysis}_{runnning})*2 +$ (# of free slots)*3
    - WNs with higher rank will be chosen for draining by Defrag daemon
    - All WNs are homogeneous in Tokyo Tier-2

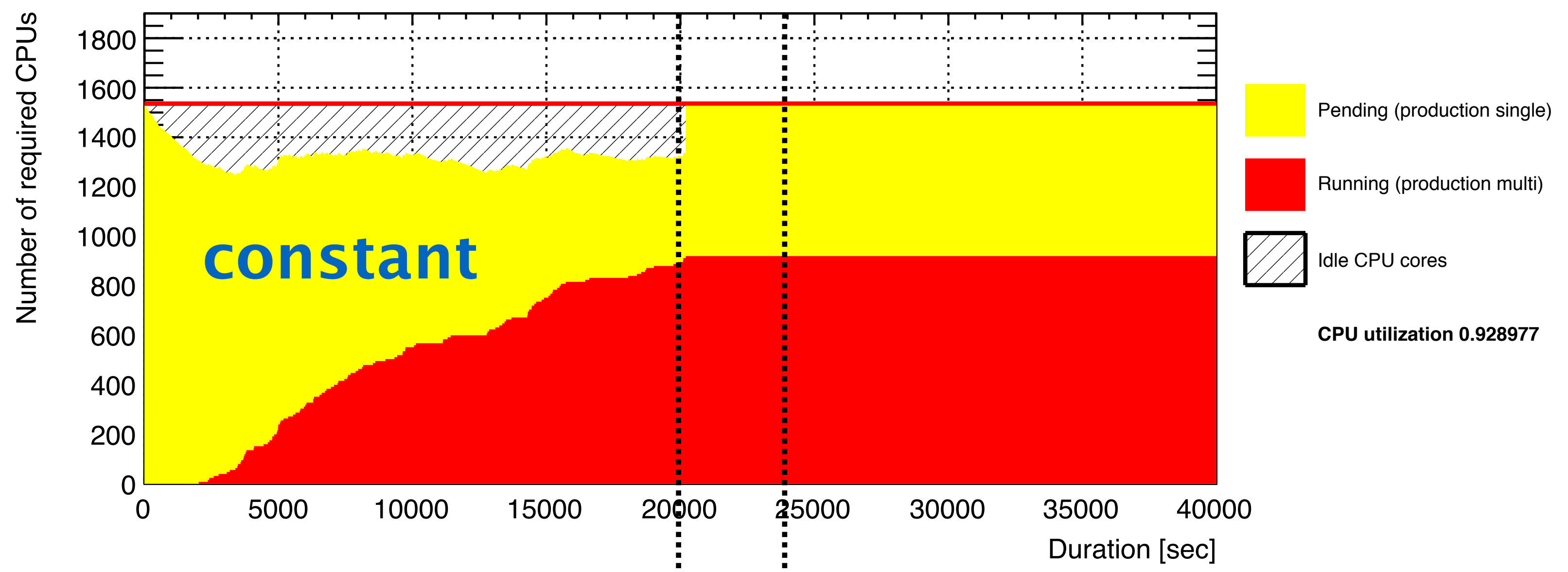
* **Implemented into the production at Feb.**

How many machines are drained at once?

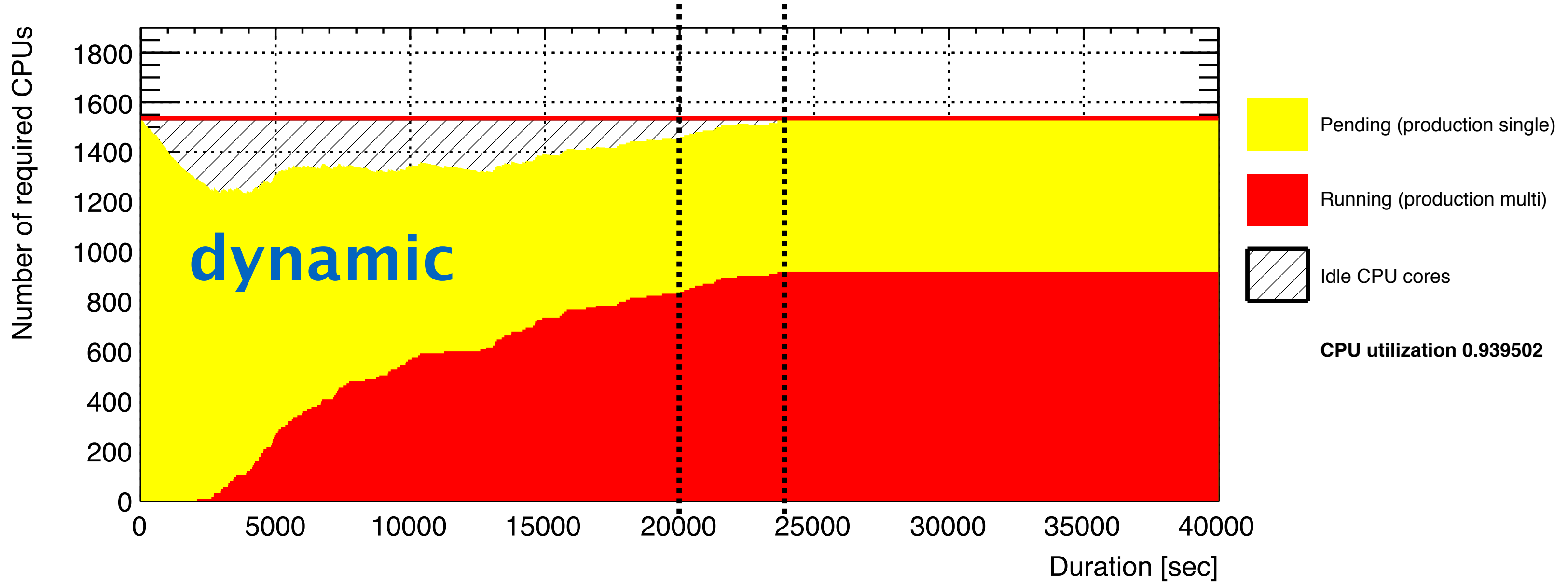
✓ Simulated queue status during the draining :

– $N^{machine}_{draining}$: number of concurrent draining machines

$N^{machine}_{draining}$ = 64



$N^{machine}_{draining}$ = XX

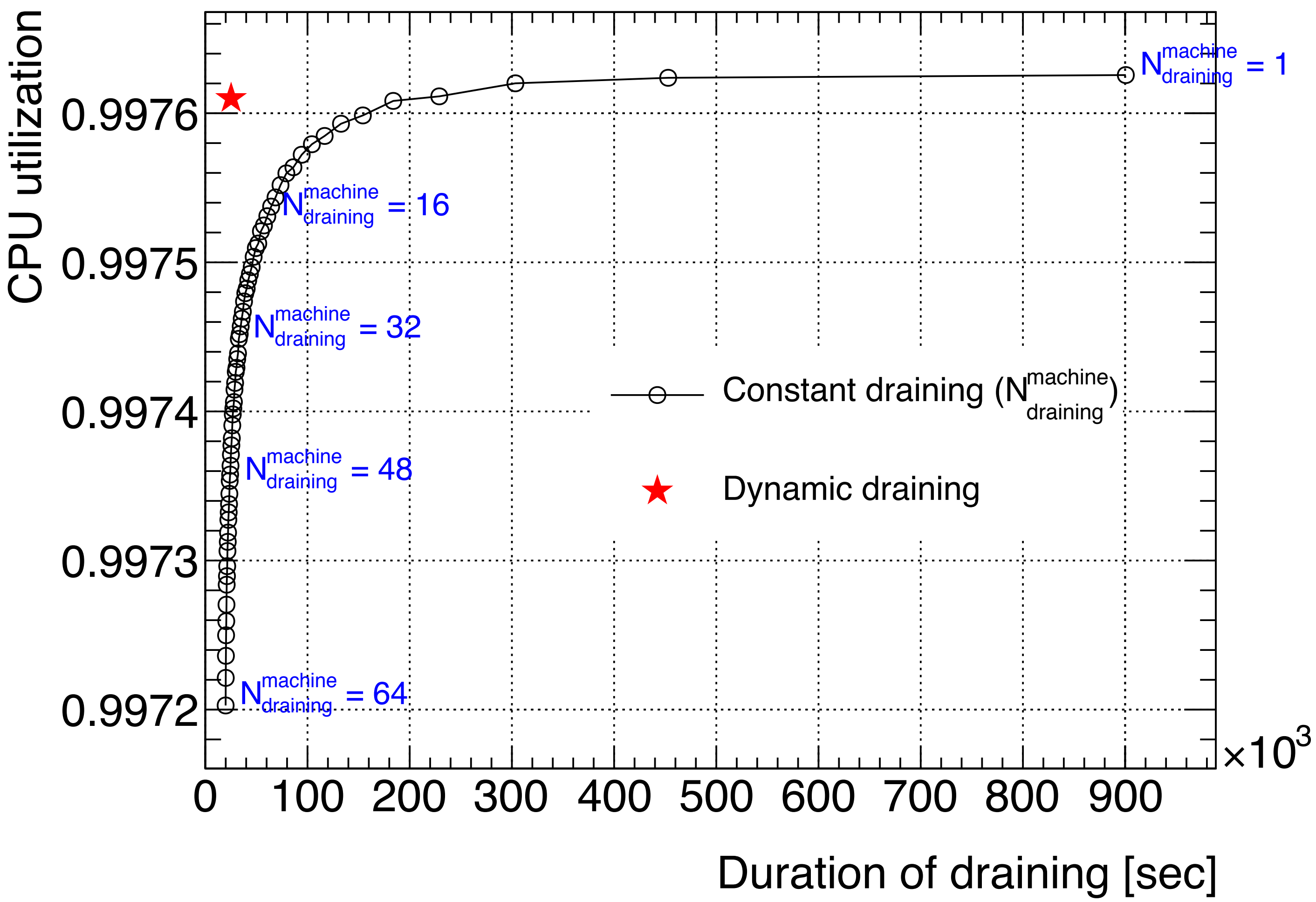


– If all machines (64 WNs) are drained constantly :
  - **Pros**: the draining finishes earliest
  - **Cons**: CPU utilization is lowest because of useless drainings

→ $N^{machine}_{draining}$ is dynamically changed based on progress of draining

$$N^{machine}_{draining} = N^{multi\text{-}core}_{target} - N^{multi\text{-}core}_{runnig}$$

How many machines are drained at once?



✓ CPU utilization VS draining duration:

- Estimated by simulation, 50 % production (single) jobs and 50 % analysis jobs are filled initially
- X axis: Duration of draining until target share is achieved
- Y axis: CPU utilization for 1.0×10^6 seconds

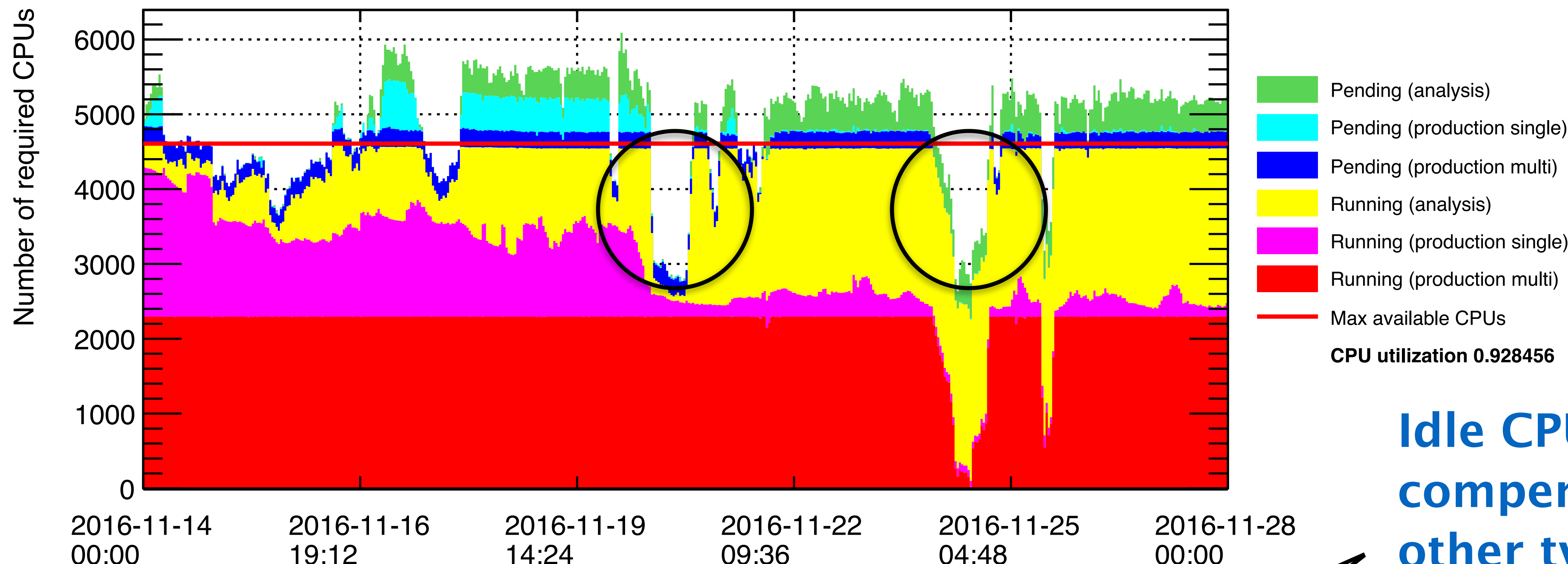
✓ Good CPU utilization and draining duration (★) are achieved by introducing the dynamic value of $N^{machine}_{draining}$

A cron script monitors $N^{multi-core}_{runnig}$ and calculates $N^{machine}_{draining}$, then sets `DEFRAG_MAX_CONCURRENT_DRAINING` in Defrag daemon

* **Implemented into the production at Feb.**

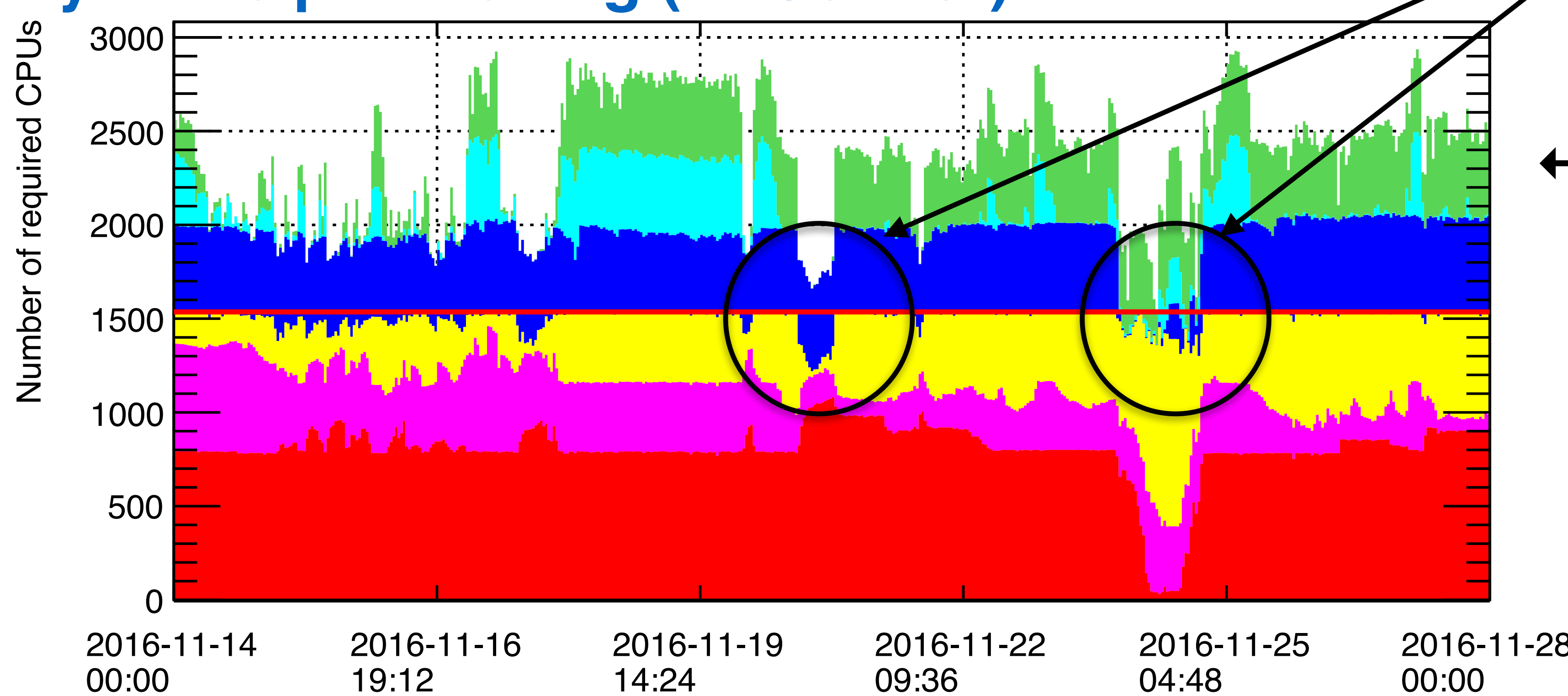
Results by introducing dynamic partitioning

Static partitioning (Torque/Maui)



Idle CPUs were compensated by the other type of jobs

Dynamic partitioning (HTCondor)



← **The Dynamic partitioning is working very well**

Observed CPU utilization

pbs_server crush

Software maintenance

	Nov. 2016		Dec. 2016		Jan. 2017	
	week1,2	week3,4	week1,2	week3,4	week1,2	week3,4
Static partitioning (Torque/Maui)	98.8%	93.9%	88.8%	94.2%	95.1%	75.0%
Dynamic partitioning (HTCondor)	**99.4%**	**98.0%**	94.8%	**98.5%**	**98.4%**	91.1%

* **Test jobs (i.e. ops job) are overcommitted in HTCondor system**

- ✓ Improvement of CPU utilization has been observed thanks to the dynamic partitioning
- ✓ HTCondor is stable so far
  - We plan to migrate all CPUs from Torque/Maui to HTCondor in the near future

Summary

- ✓ Tokyo Tier2 with the 4th system is running
  - Providing enough computing resources for ATLAS
  - > 99% site availability is achieved
- ✓ Dynamic partitioning has been introduced using HTCondor
  - Parameters of draining have been optimized based on the job features at Tokyo Tier2 center
    - Draining starts/ends based on # of running/waiting multi-core jobs
    - Analysis jobs tend to finish earlier
    - $N^{machine}_{draining}$ is dynamically changed based on progress of draining
  - Improvement of CPU utilization has been observed
    - e.g. ~5% improvement in 2016/12/15 to 2016/12/31
  - Will migrate all CPUs from Torque/Maui to HTCondor

Backup