Workload management at KEK/CRC
-- status and plan

KEK/CRC

Hiroyuki Matsunaga

Most of the slides are prepared by Koichi Murakami and Go Iwai

CPU in KEKCC

- Work server & Batch server
  - Xeon 5670 (2.93 GHz / 3.33 GHz TB, 6core)
  - 282 nodes : 4GB /core
  - 58 nodes : 8GB /core
  - 2 CPU (12 cores)/node : 4080 cores in total
- Interconnect
  - InfiniBand 4xQDR (4GB/s), RDMA
  - Connection to storage system
- Job scheduler
  - LSF (ver. 8 -> ver.9 @last Aug.)
  - Scalability up to 1M jobs
- Grid deployment
  - EMI
  - Work server also acts as UI, Batch server as WN

IBM System x iDataPlex

Platform LSF

- Job scheduler
  - IBM Platform LSF
  - Scalabe up to 1M jobs
  - Work server as Grid-UI, Batch server as Grid-WN
    - Both local and Grid jobs are managed by LSF (1 instance)
- Platform LSF
  - Updated to ver.9 (from ver.8) in last Aug.
  - New components introduced
    - RTM (real time monitoring) (Cacti)
    - Session Scheduler (short duration job management)

Queue setting

- Setting queue parameters is vital in system operation.
  - Maximize system usage
  - Fair-share between users, but give priority if needed

- Kinds of queue
  - Single thread jobs (normal/huge memory usage)
  - MPI jobs
  - Workload types of Belle data processing (analysis framework, data skimming, ...)
  - Grid jobs go through separate queues

- Job slots per host
  - In principle, 1 process/core : 12 processes/node (12cores)
  - In reality, some inefficiency (resource vacancy) -> “Over-subscription” 13 or14 processes/node

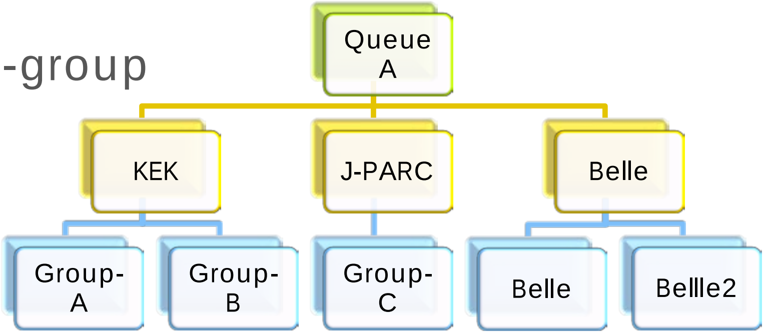
- Turn-off job interruption (suspend/resume by higher priority job)
  - Job slots are always full, suspended jobs never resumed.

- Cross-queue setting
  - Fair-share value is shared between similar queues
  - Users always try to run as many jobs as possible. (submit same-kind jobs into different queues)

- Queue parameters are continuously tuned.

Fair-share Scheduling and Resource Management

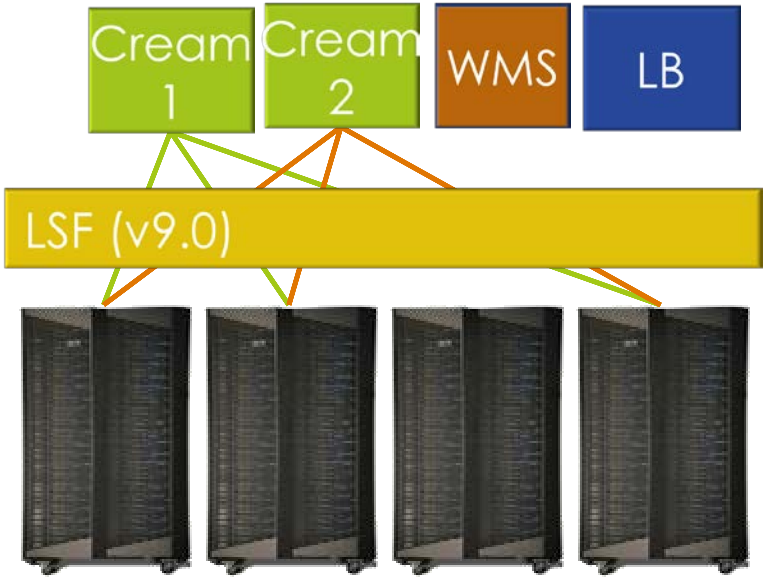
- Dynamic prioritization for job dispatch between users and groups
  - Dispatch according to a fair-share rule
  - A rule can be shared between (similar) queues (cross-queue)
  - Fair-share grouping
    - control job throughputs by group/sub-group



- Resource management
  - Prepare special queues with higher priority when a group needs CPU resource for a limited period
  - More "active (agile)" resource management will be needed in the near future.
    - Currently CPU usage is high (~90%). Every group/user has the same priority.
    - e.g. Belle II production jobs (MC, raw data processing) should be given a higher priority when the experiment is running.

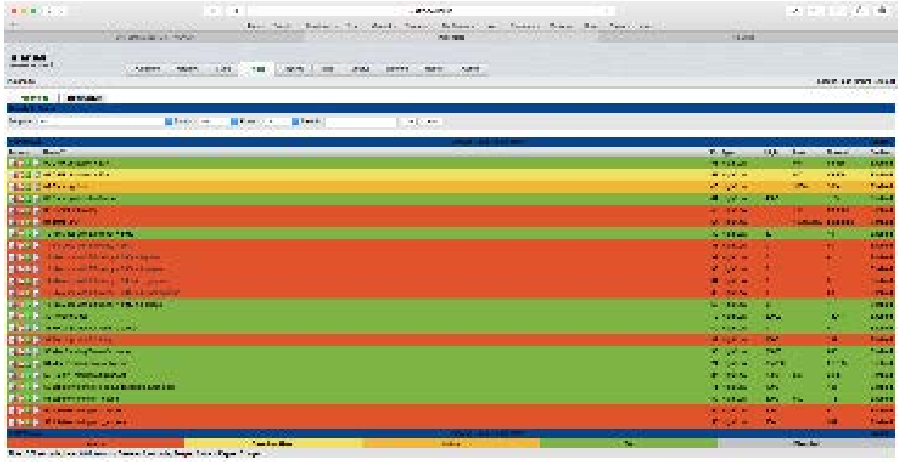
Grid Setup

- 2 Cream CE + 1 WMS/LB
- >20 UI (work server)
- Different queues from local job
- Grid user is mapped to a pool account
  - Fair-share does not work if both local and Grid jobs run by the same user…

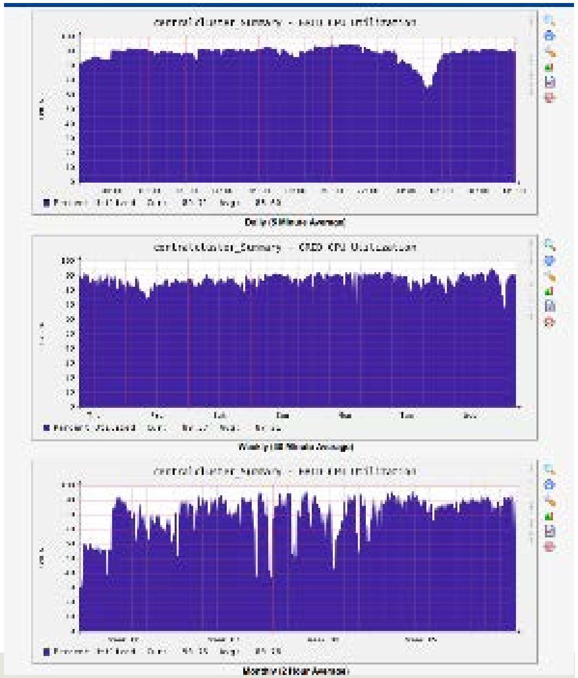


RTM, monitoring

- RTM component was introduced last May.
  - Web monitoring based on Cacti

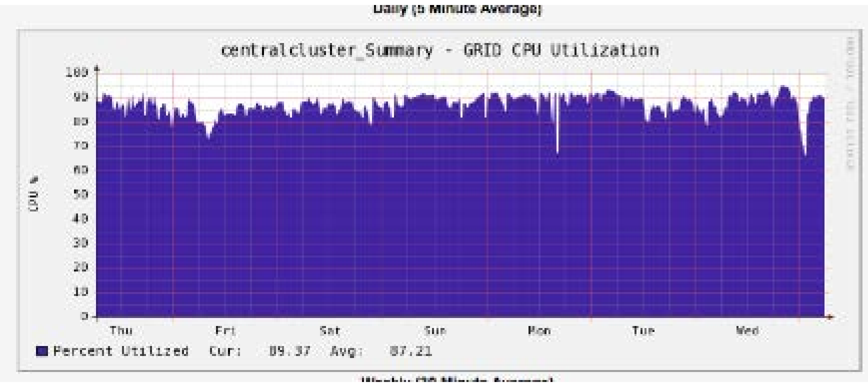


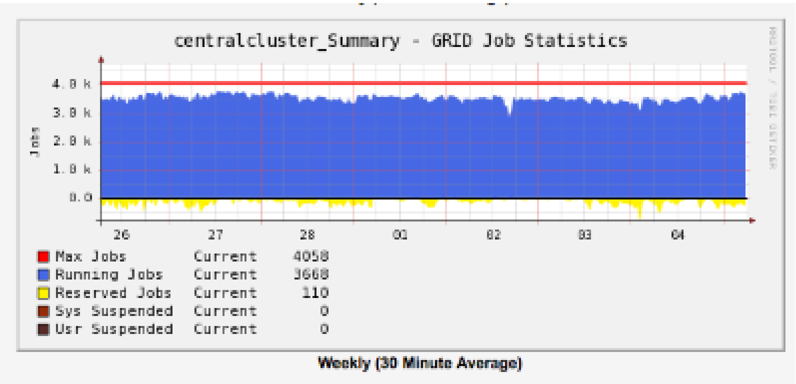
- Monitoring
  - CPU utilization, Running jobs (per queue), Job throughput, Pending jobs, Queue status, Inefficient jobs, …
  - Monitor/Alert system efficiency
  - What/who triggers an inefficiency?
    - By massive short-jobs submission
    - Inefficient jobs (low CPU utilization) caused by user’s code

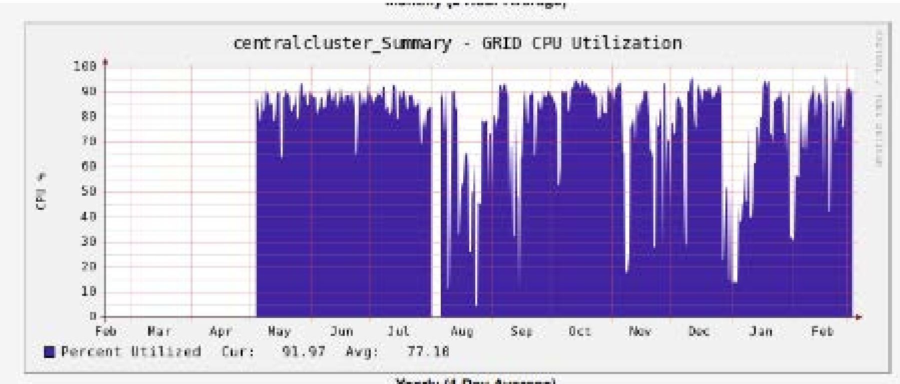


Statistics

- CPU utilization is nearly full (~90%).
  - 10% of the inefficiency might be from:
    - overhead of the job scheduler
    - job slot allocation for multi-thread jobs (yellow)

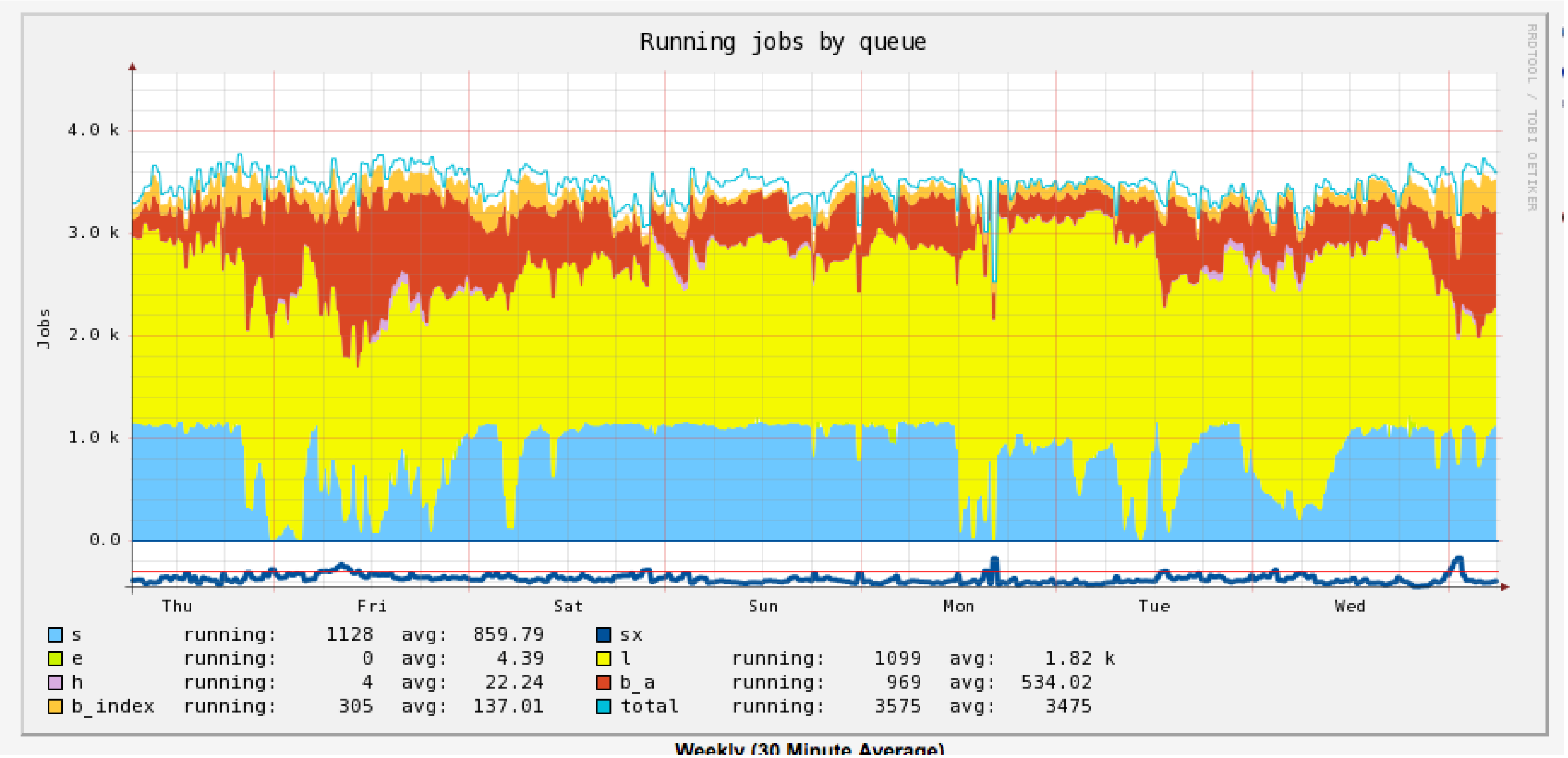






weekly CPU/Jobs (top)
yearly CPU (bottom)

Jobs per Queue -Weekly

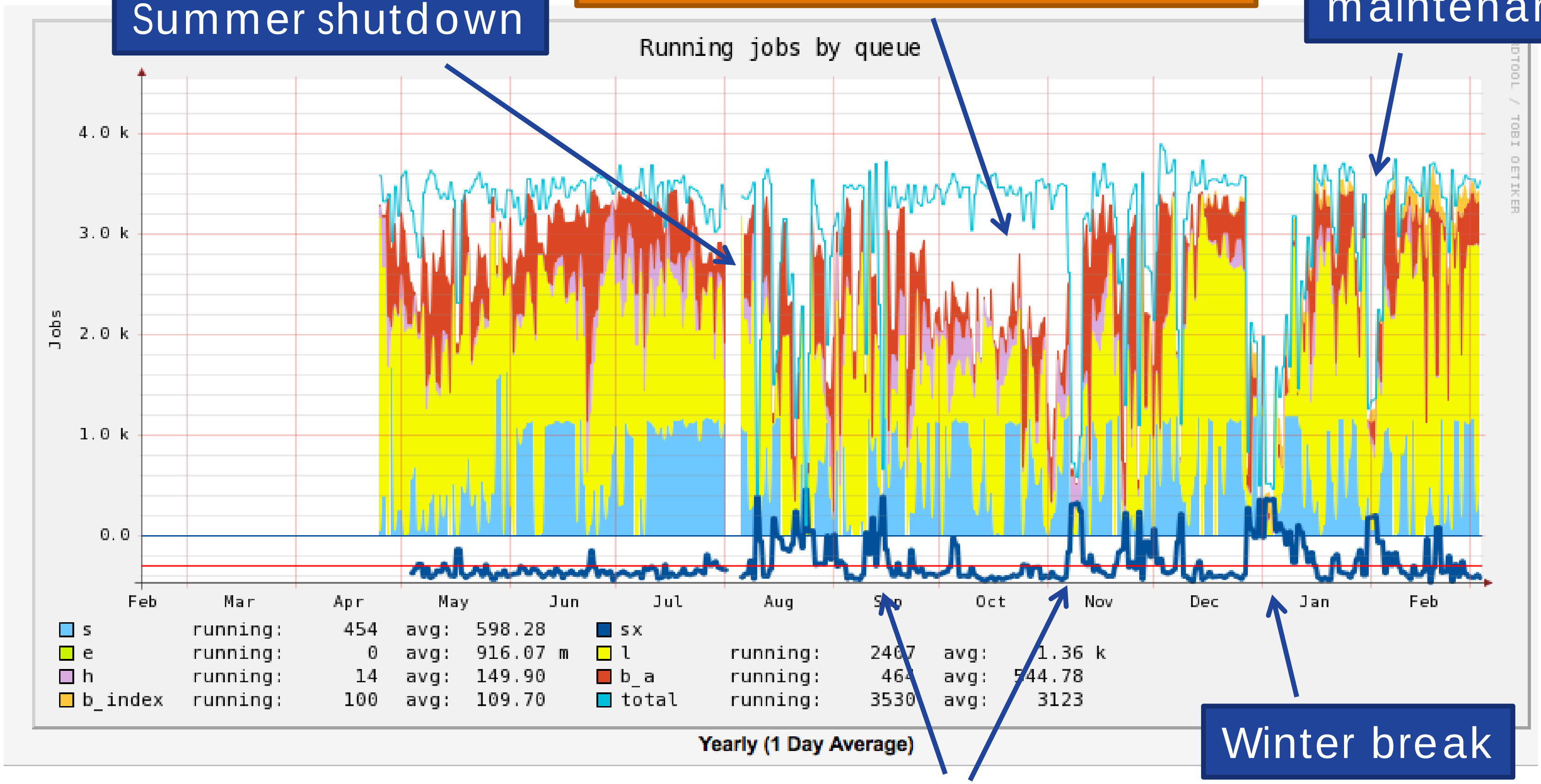


```
🟦 s        running:   1128  avg:  859.79   🟦 sx
🟩 e        running:      0  avg:    4.39   🟨 l        running:   1099  avg:    1.82 k
🟪 h        running:      4  avg:   22.24   🟥 b_a      running:    969  avg:  534.02
🟧 b_index  running:    305  avg:  137.01   🟦 total    running:   3575  avg:    3475
```

Weekly (30 Minute Average)

Light blue line : Total # of jobs
Red (Belle), Orange (Belle I/O)
Yellow (24h), Cyan (short)
White (gap below LBL) : grid jobs and others
Blue line : CPU utilization (bottom:100%, red 80%)

Jobs per Queue - Yearly



Summer shutdown

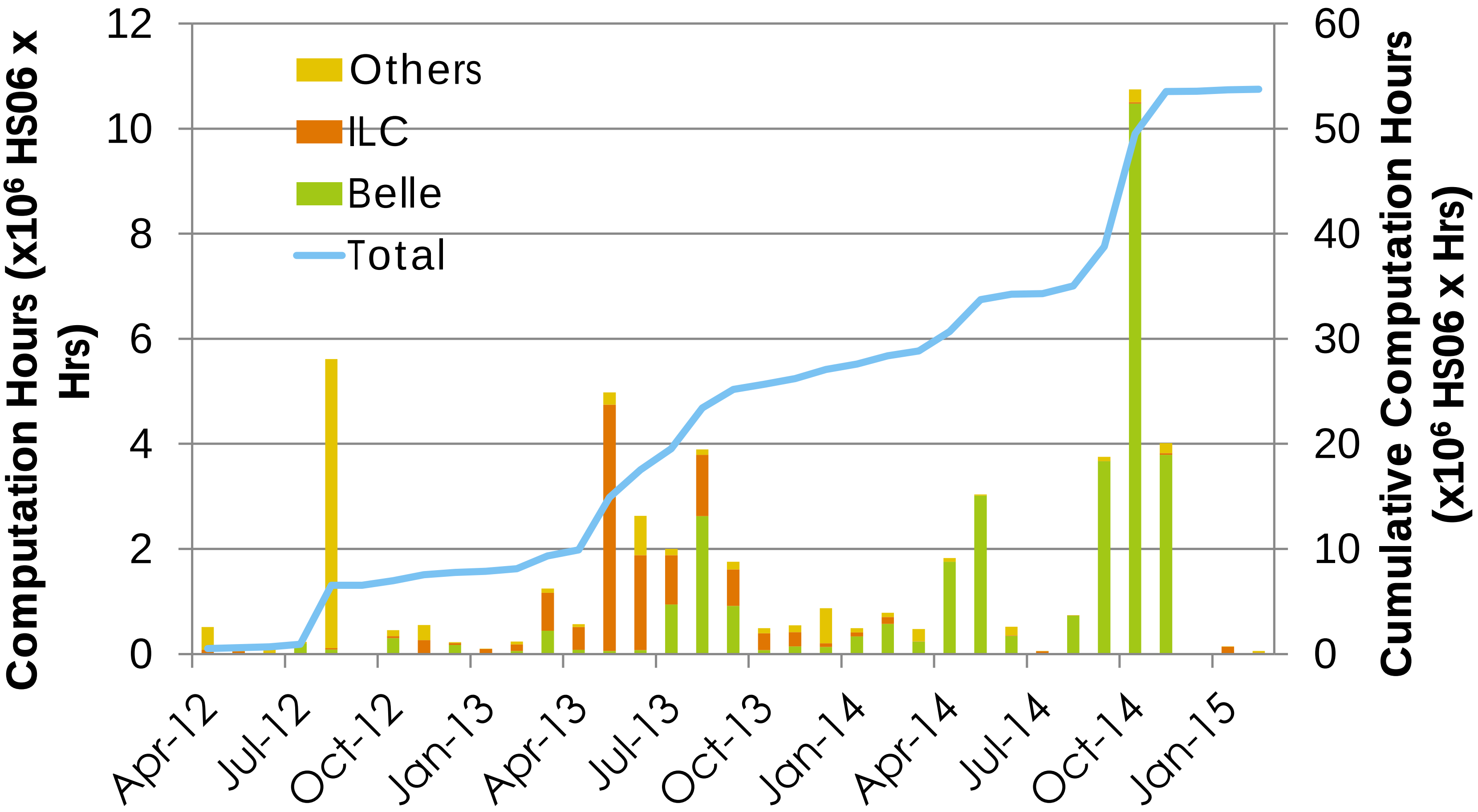
Belle II MC production by Grid

glibc/security maintenance

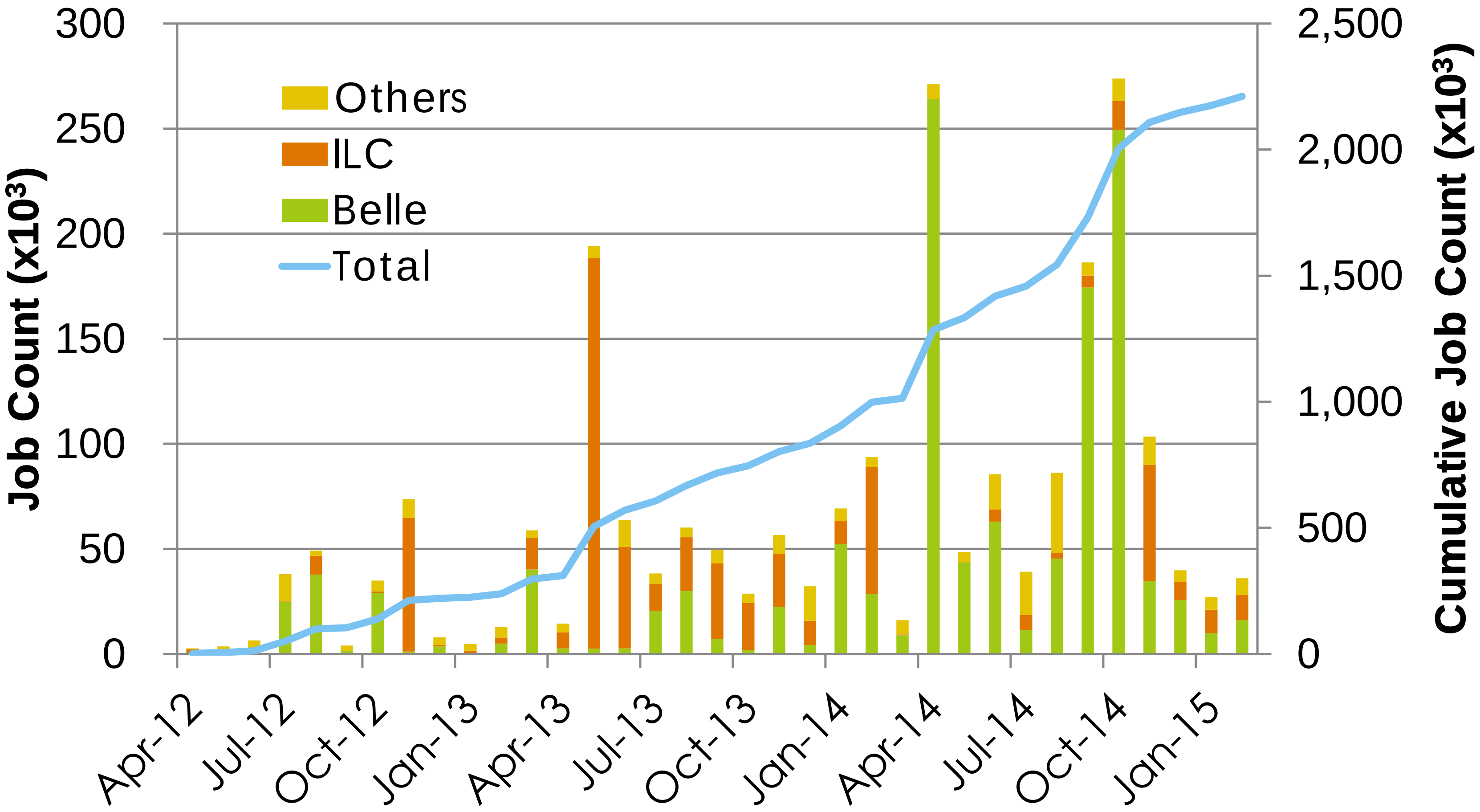
Winter break

System trouble

CPU Usage for Grid Jobs



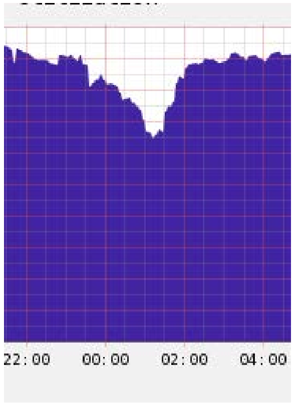
Grid Submission Jobs



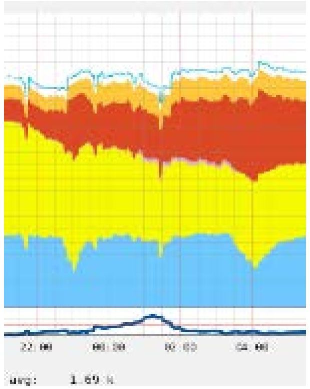
Non-negligible overhead of short jobs

- Job dispatch of massive short duration jobs leads to system inefficiency.
  - E.g. tons of < 1min jobs
- We ask users to use scripts to merge short jobs, but this kind of situation sometimes happens.
- Session Scheduler could help by describing a job list file
  - Reduce a dispatch overhead of scheduler for short duration jobs

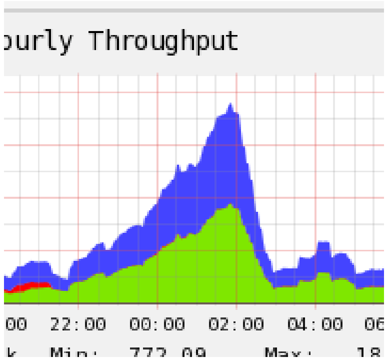
CPU



Jobs



Throughput



Workload management and Cloud Service

- More flexible resource provision is desirable for better utilization
  - Different groups want different environments
    - E.g. Grid services need to migrate to SL6, but Belle I wants to stick to SL5.
  - Efficient resource management (servers on demand)
  - Development and test (OS, software)

- IaaS/PaaS-type of service (internal cloud)
  - Middleware choice
    - PCMAE + Platform Dynamic Cluster : coherence with LSF
    - OpenStack (open solution)
  - Provisioning tools
    - KVM (VM), xCAT (baremetal), Docker (future)?
  - Virtualization technology needs to be improved.
    - CPU virtualization (MC) is usable, but I/O virtualization's performance is not yet enough.

- External cloud service
  - Amazon EC2 is tested with Dirac for Belle II MC campaign.

- See Wataru's talk tomorrow

Concerns about IBM Platform

- Platform was acquired by IBM 2 years ago.
- Purchase cost of Platform product would be much more expensive than before.
  - We are trying to find a solution to purchase with lower cost
  - Otherwise, need to migrate to another scheduler (Univa grid-engine, PBS, ...) + open cloud middleware solution (OpenStack)?
- IBM is shifting cloud service more
  - Commercial cloud is promising.
    - IBM makes good benefits in HPC cloud?

Summary

- We have been using LSF for long time
  - Experience in the previous systems
- LSF works well for both local and Grid jobs
- Queue settings is vital, and needs to be optimized continuously
- Try to improve the efficiency by new tools
  - Education for users is still important
- For the next systems, we should carefully choose the best solution for us within the limits of the budget.
  - Batch scheduler, Cloud middleware,…