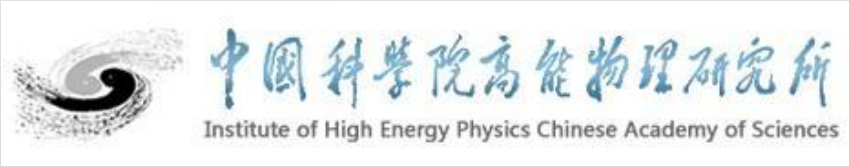
Using ATLAS@home to backfill grid sites

Wenjing Wu[1], David Cameron[2] , Rodney Walker[3]

1. Computer Center, IHEP, China
2. University of Oslo, Norway
3. Ludwig Maximilians Universitat (DE)

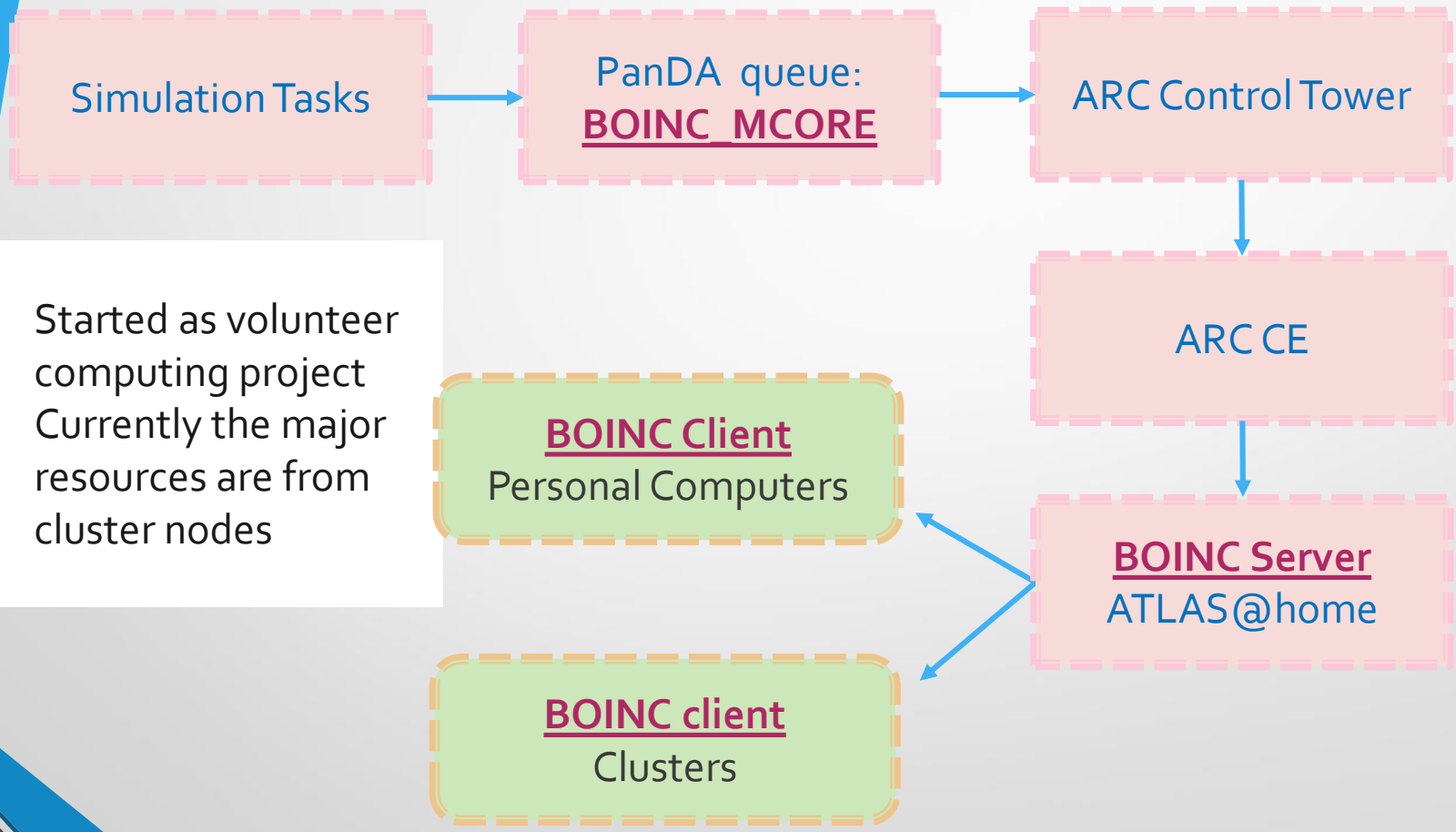
2018-03-07



Outline

- ATLAS@home
- ATLAS@home Backfilling clusters:
  - BEIJING Tier2 site
- Integration of resource contributions
- How to join as a site
- Summary

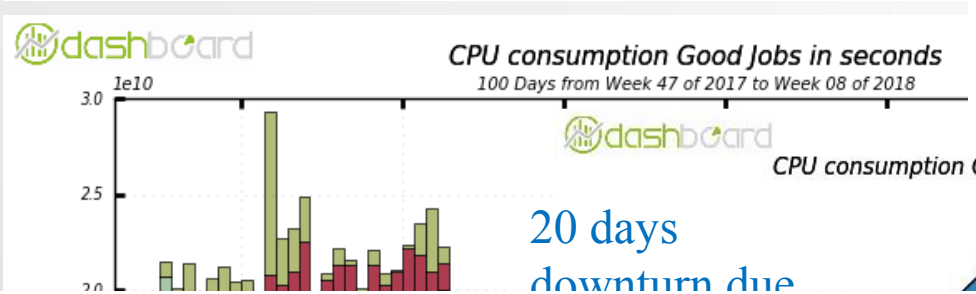
ATLAS@home



- Started as volunteer computing project
- Currently the major resources are from cluster nodes

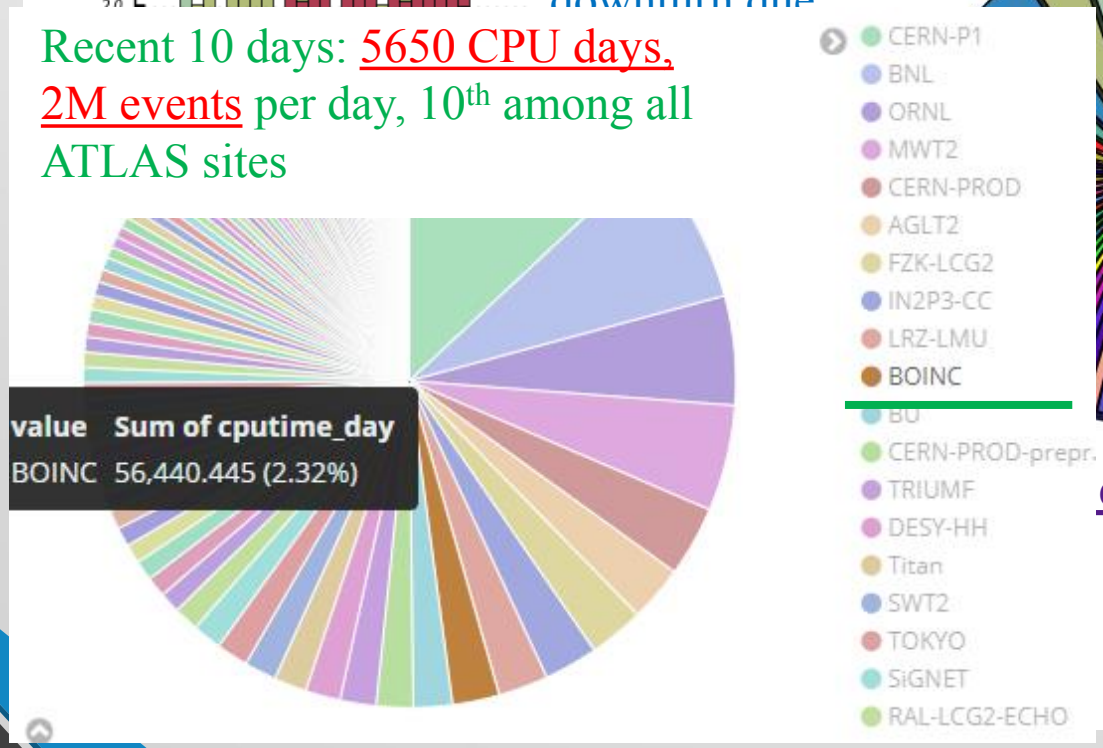
Current scale (1)

CPU time of good jobs of All ATLAS sites in the past 100 days

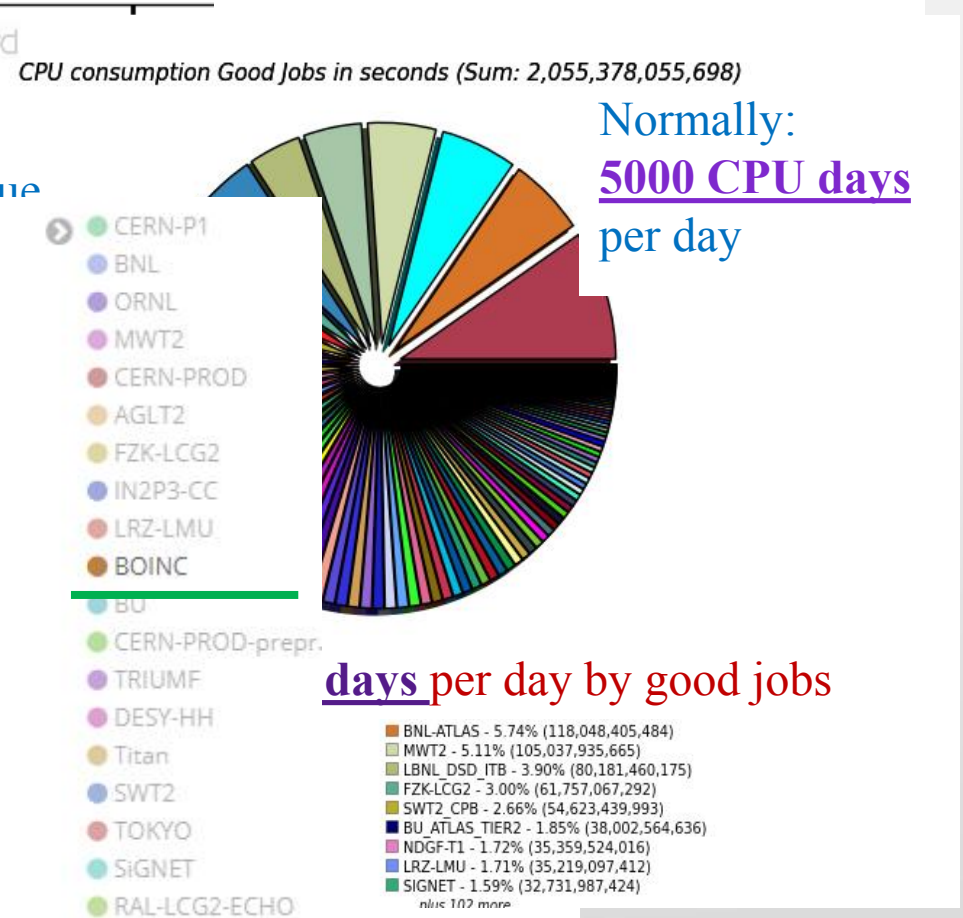


Avg. BOINC core power: 11HS06

20 days downturn due



Recent 10 days: 5650 CPU days, 2M events per day, 10th among all ATLAS sites



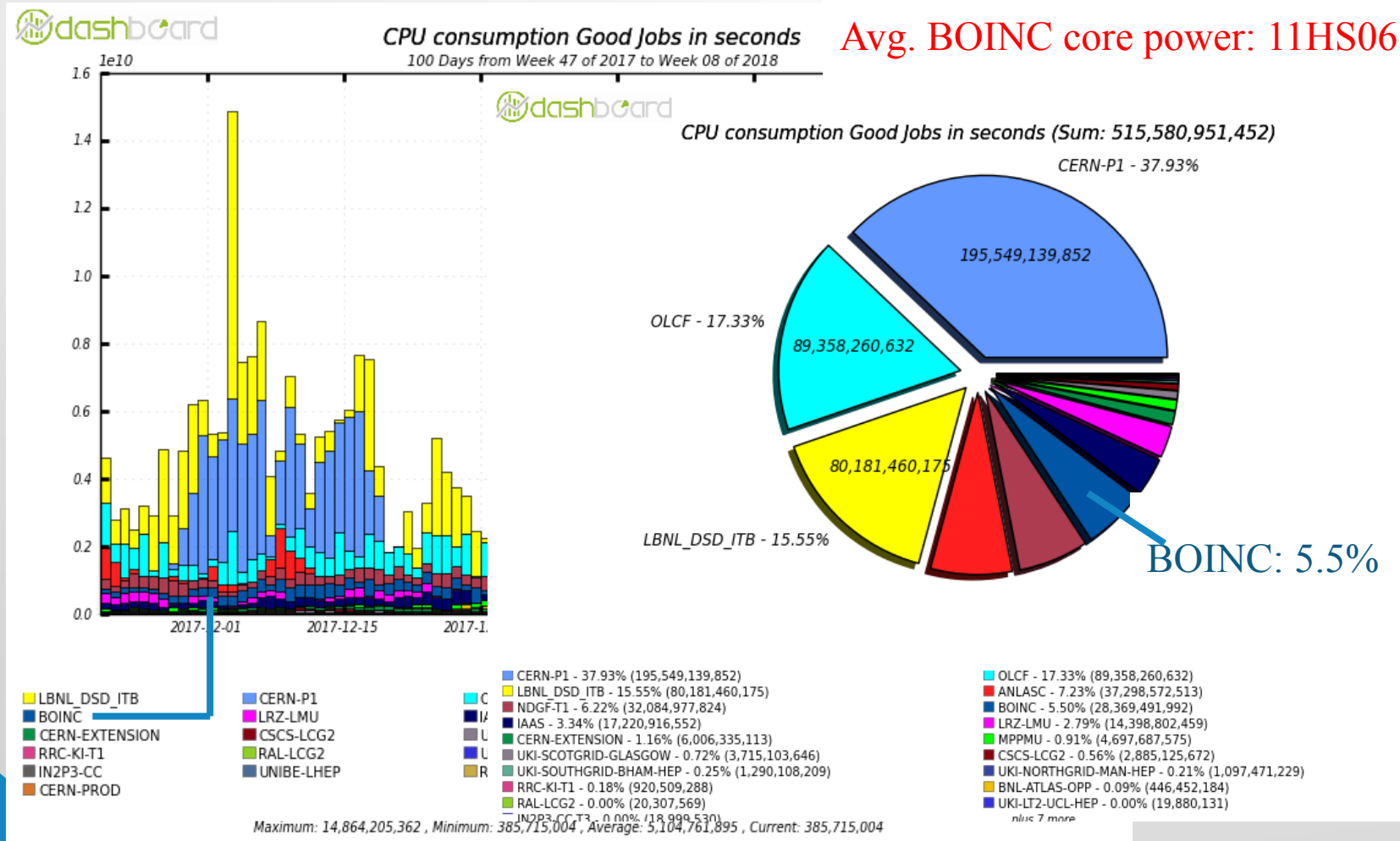
Normally: 5000 CPU days per day

days per day by good jobs

Current scale (2)

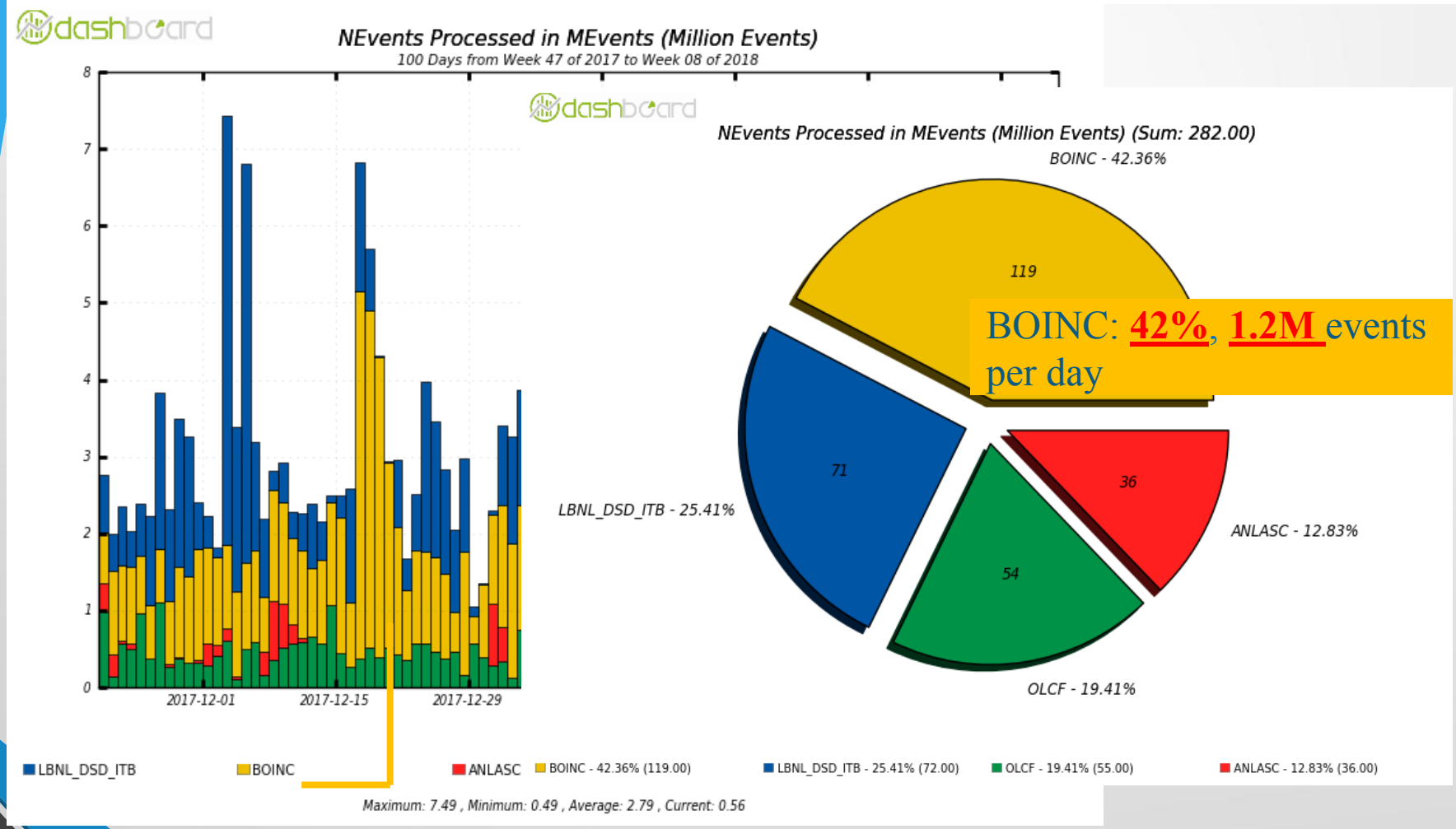
CPU time of good jobs for All ATLAS HPC and Cloud sites in the past 100 days

Avg. BOINC core power: 11HS06



Current scale (3)

BOINC compared to other non-standard resources: simulate **# events** in the past 100 days



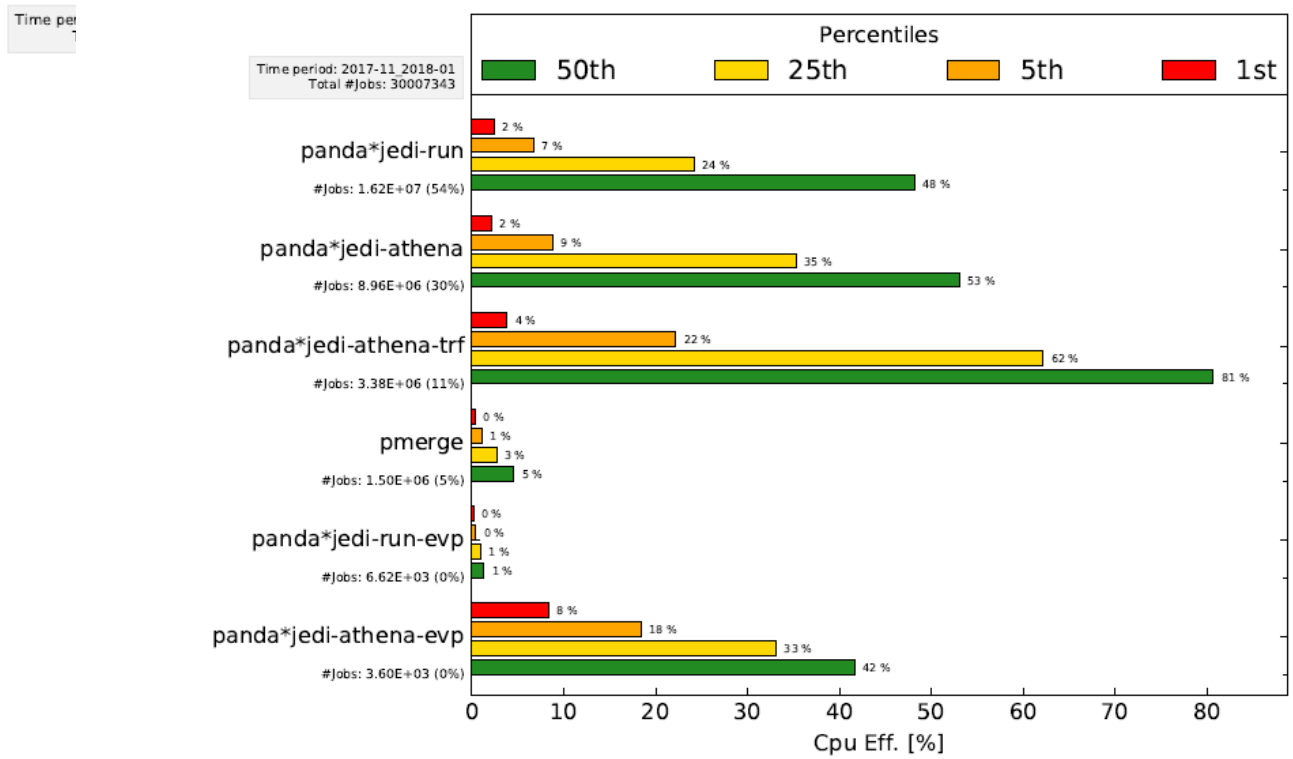
Backfilling Grid sites

- Tier 2 site utilization rate
- At BEIJING Tier2 site

Tier2 is not always fully loaded

- Site downtime
- CPU E
- CPU Eff. For production jobs are between 12% to 96%

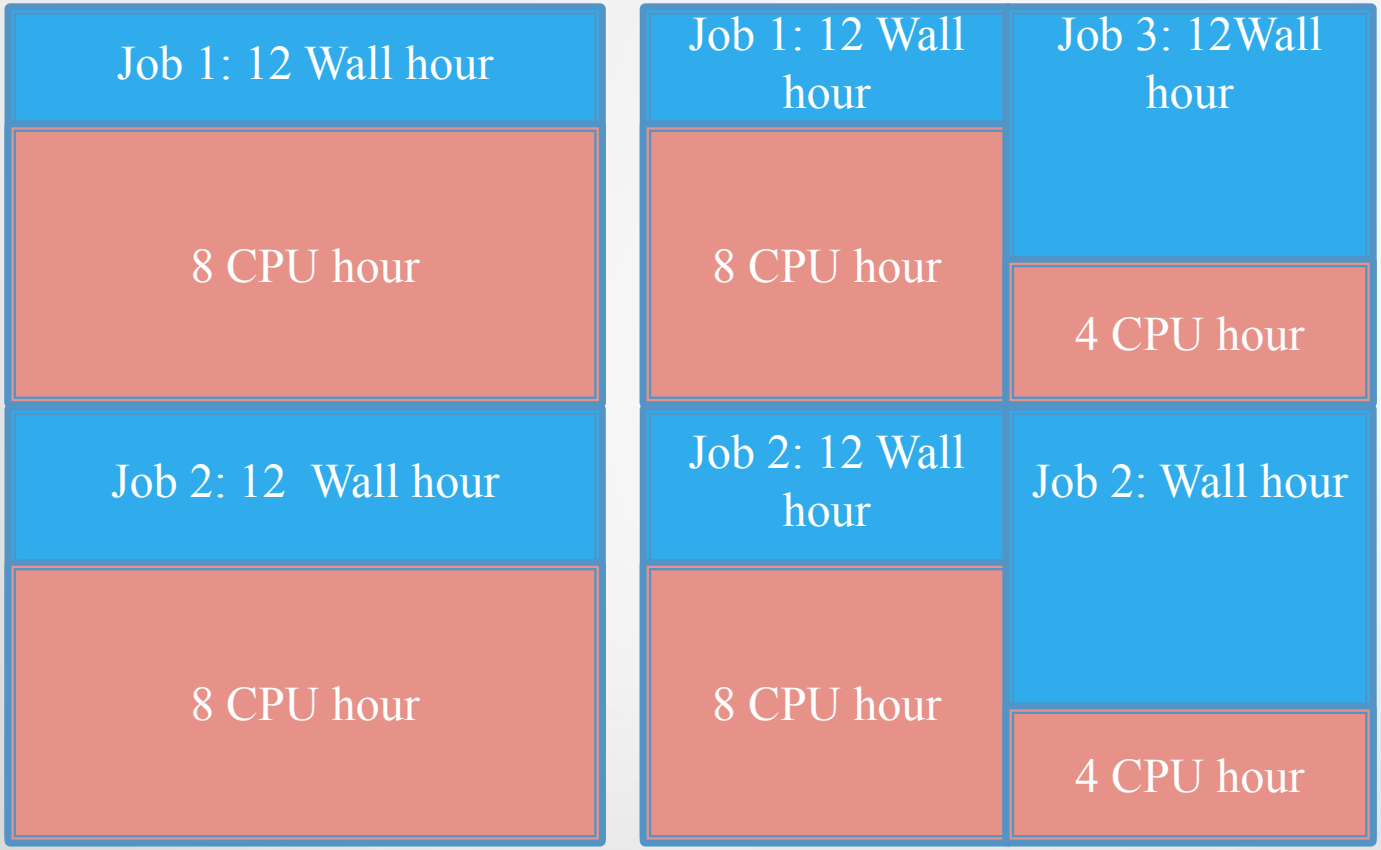
CPU Eff.



Slides from

CPU Eff. for user jobs are between 1% to 81%

100% wall != 100% CPU utilization due to job CPU Eff.



One work node
With job 1-2, 100% wall utilization, assume job CPU Eff. 75%,then 25% CPU is wasted
With job 1-4, 200% wall utilization, 100% CPU utilization, job eff 75% and 25%

Put more jobs on work nodes

- Run 2 jobs on each core
- 1 grid job with normal priority (pri=20), 1 BOINC job with the lowest priority (pri=39)
- Linux uses “non preemptive” scheduling for CPU cycles, which means high priority jobs occupies CPU until it releases the CPU voluntarily.

ATLAS Grid job

```
18781 prdat280  20   0 2630m 1.8g                    03:05.75 athena.py
18784 prdat280  20   0 2624m 1.8g  22m R 99.8  2.                02:48.01 athena.py
18786 prdat280  20   0 2624m 1.8g  21m R 99.8  2.9 202.59.28 athena.py
12445 root      39  19 2559m 1.7g  17m R 82.6  2.7  49:41.13 athena.py
12446 root      39  19 2559m 1.7g  19m R 81.9  2.7  57:28.02 athena.py
12447 root      39  19 2558m 1.7g  17m R 81.6  2.7  54:08.93 athena.py
12426 root      39  19 2557m 1.7g  15m R 80.6  2.7  48:52.96 athena.py
12449 root      39  19 2559m 1.7g  14m R 78.2  2.7  47:28.29 athena.py
12451 root      39  19 2560m 1.7g  20m R 77.                52:43.25 athena.py
```

ATLAS@home job

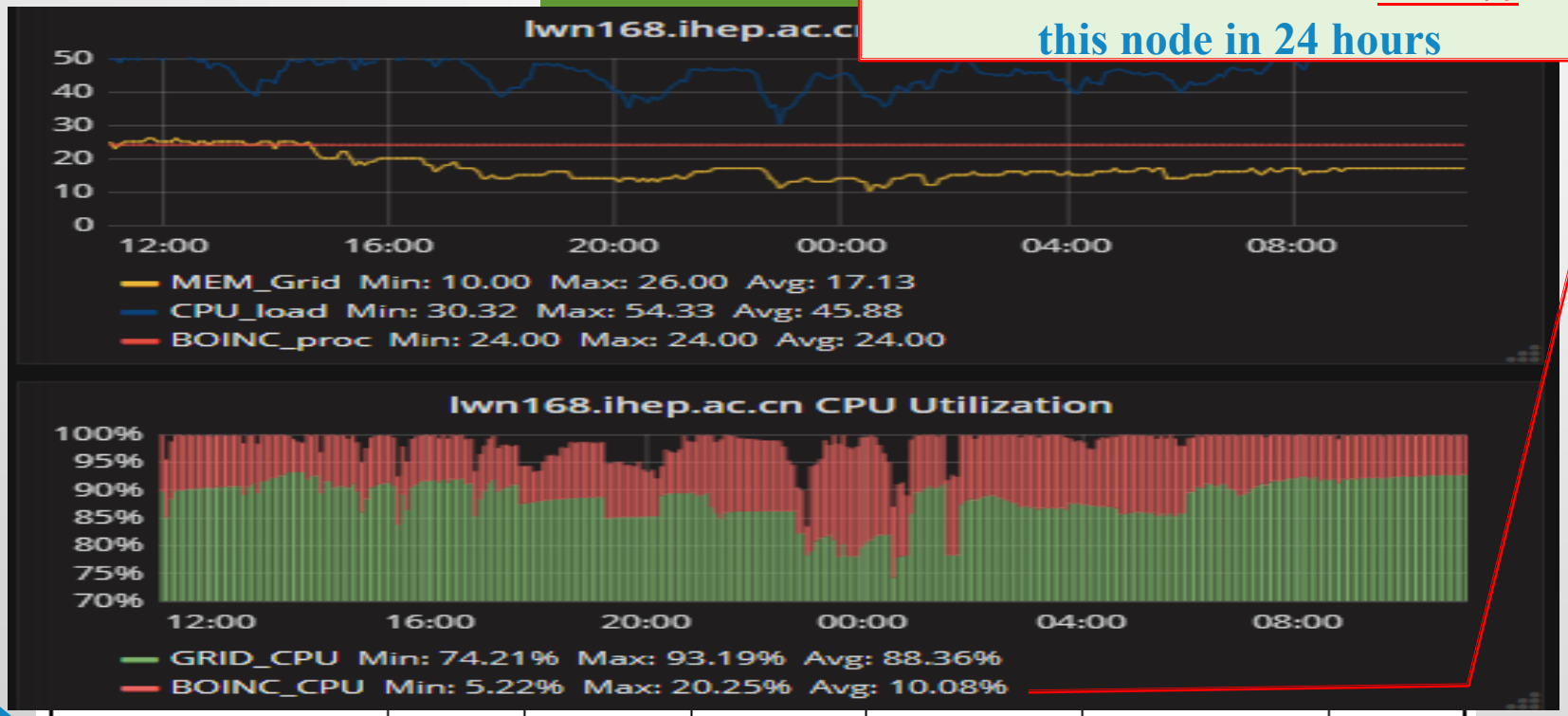
Backfilling at BEIJING site

- ATLAS Tier2 site (468 cores, managed by PBS)
- IHEP local cluster: mixed jobs (ATLAS, CMS, BESIII, BelleII, JUNO etc.), managed by HTCondor

BEIJING Tier2 site (in 100 days)

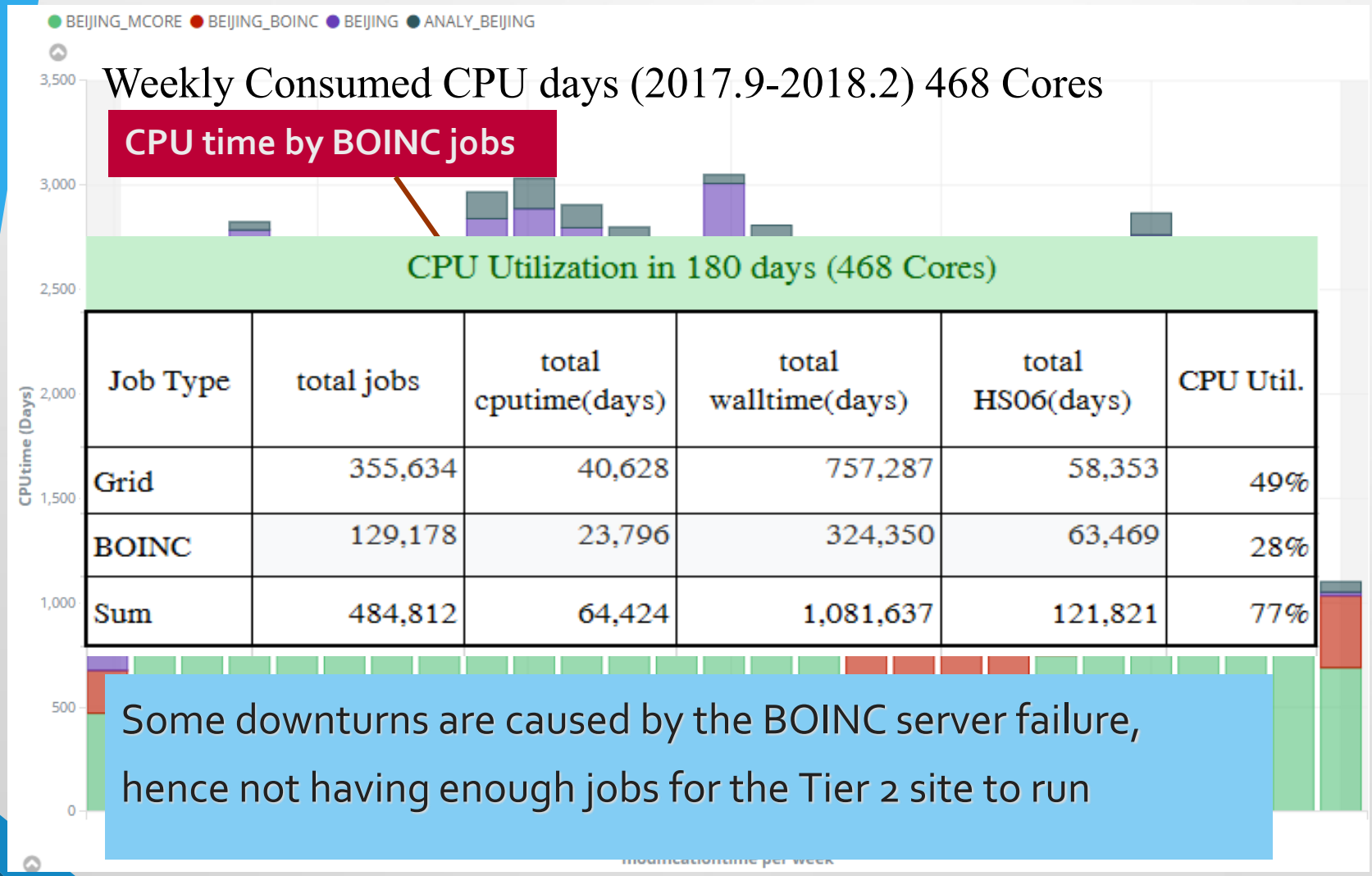
- Grid jobs *Walltime Util.* is **87.8%**, *Grid CPU Util.* is **65.6%**, BOINC exploit an **extra 23%** of CPU time, node CPU Util. reaches **89%**
- More details from here:

The overall CPU Util. is *98.44%* on this node in 24 hours



BEIJING_BOINC	468	437	110	93.38	23.50	167.7
Total	468	848	417	181.20	89.10	

In Long term



Weekly Consumed CPU days (2017.9-2018.2) 468 Cores

CPU time by BOINC jobs

CPU Utilization in 180 days (468 Cores)

Job Type	total jobs	total cputime(days)	total walltime(days)	total HS06(days)	CPU Util.
Grid	355,634	40,628	757,287	58,353	49%
BOINC	129,178	23,796	324,350	63,469	28%
Sum	484,812	64,424	1,081,637	121,821	77%

Some downturns are caused by the BOINC server failure, hence not having enough jobs for the Tier 2 site to run

DBRSC(Dynamical BOINC ReSource Configuration)

- ATLAS@home jobs : 2-12 cores per job, avg. 2-4 CPU hours per job
- Short jobs can **lose up to 27%** CPU Eff. with **big core** per job.
- Trade-off between Memory usage and CPU Eff.

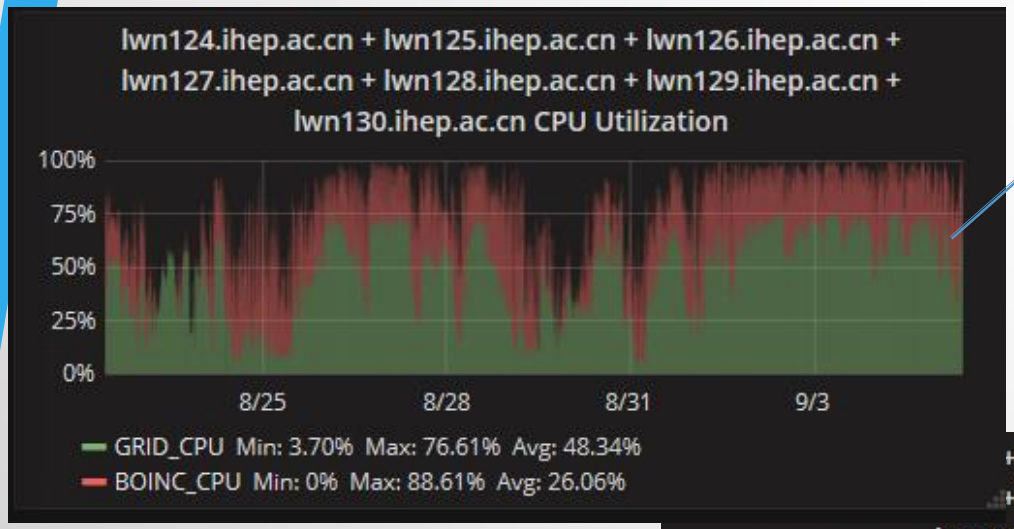
Resource Req. and CPU Eff. For ALTAS@home Jobs				
Core per Job	Max. RAM per Core (GB)	Job CPU Eff. For short jobs (2-4 CPU hours)	Job CPU Eff. For long jobs (over 12 CPU hours)	Job Eff. Difference
2	1.00	92%	96%	4%
3	0.80	86%	92%	6%
4	0.60	84%	90.50%	7%
5	0.55	77%	90.50%	14%
6	0.45	68%	90.50%	23%
7	0.40	66%	90%	24%
8	0.40	65%	90%	25%
12	0.35	63%	90%	27%

DBRSC: Dynamical BOINC ReSource Configuration

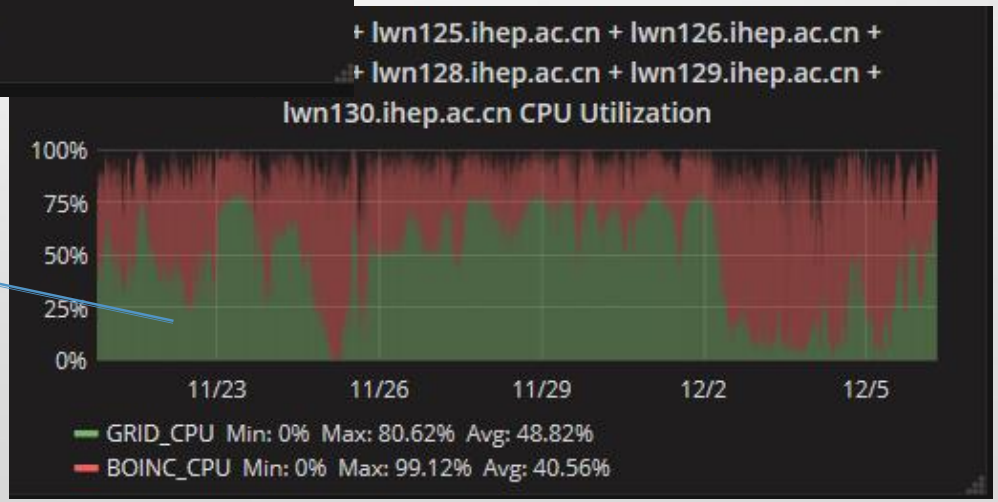
- Takes control of BOINC resource configuration, hide the BOINC details from site admins
- Do not need to know the hardware configuration and resource usage of the non-BOINC jobs on the computer, DBRSC decides how much CPU/Memory BOINC can use according to the historical resource usage of non-BOINC jobs dynamically.
- Configure core_per_job dynamically before starting a job according to available resource to BOINC, to increase job CPU Eff.
- Making sure BOINC jobs do not impact non-BOINC jobs. Kill BOINC automatically if abnormal system load appear, restart BOINC automatically when system load returns normal.

Effect of DBRSC

On the same set of nodes (ATLAS single core analysis nodes), CPU Util. is increased by **15%** (from **74.4 % to 89.4%**) while the Grid workload is the same (48%), **15% increase of BOINC CPU**



Node CPU Util. is *74.4%* before DBRSC



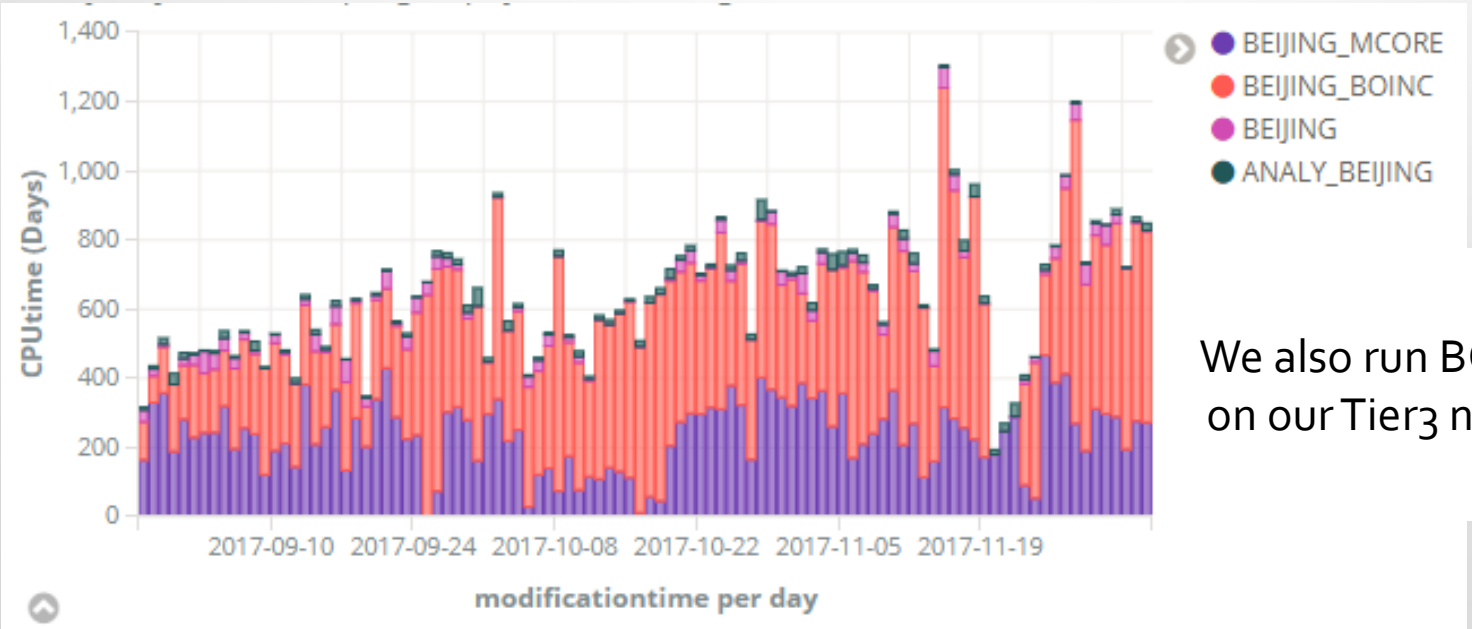
Node CPU Util. is *89.4%* After DBRSC

Resource Contribution Integration

- Goal:
  - Sites can use BOINC to manage Tier 3 clusters, or do backfilling on clusters.
  - Recognize the site's BOINC resource contribution
  - Sites' BOINC contribution can be integrated into their Grid sites.
- How
  - Site can create an ATLAS@home account, and run ATLAS@home on any computers under this account
  - The account name should be the same as their AGIS site name.
    - BEIJING PanDA queues: BEIJING, ANALY_BEIJING, BEIJING_MCORE, the AGIS site name is BEIJING_LCG2

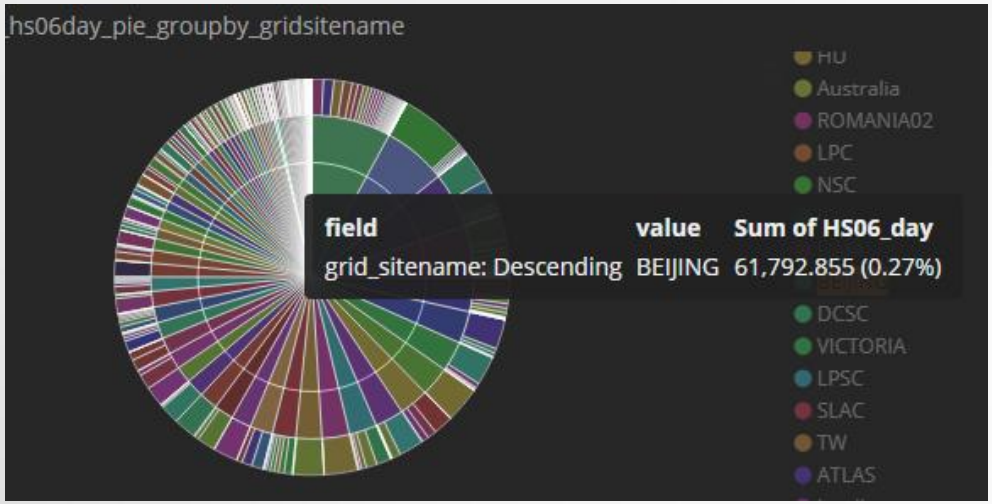
Demo visualization from Kibana

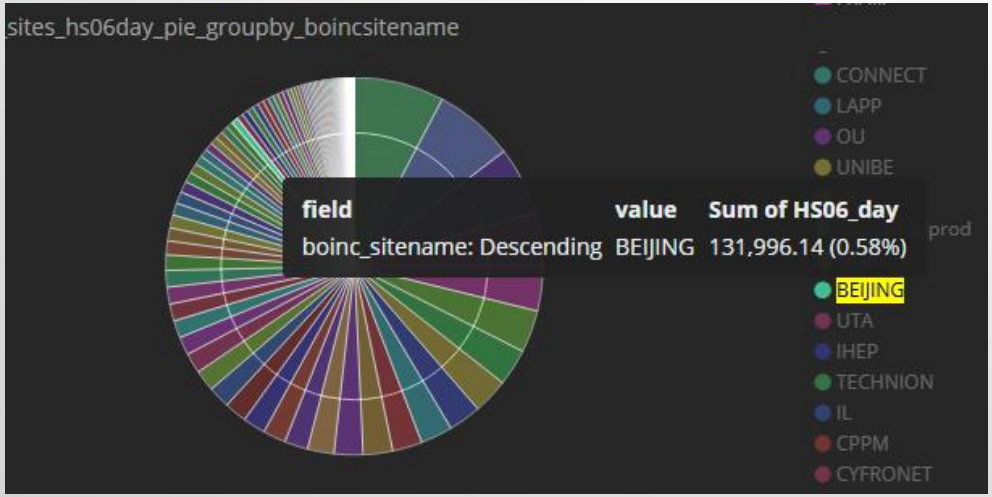
- BOINC nodes from BEIJING_LCG2 belongs to a fake PanDA queue **BEIJING_BOINC**
- Daily average: BEIJING_LCG2 site provides **640 CPU days** (1310 Wall days), and **365 CPU days** (924 Wall days) is from BOINC.



We also run BOINC on our Tier3 nodes

In sites resource contribution ranking





- Sites' BOINC contribution can also be integrated into the sites in the resource contribution ranking
- BEIJING contributes 0.27% Without adding BOINC, 0.58% with adding BOINC in whole ATLAS computing resources

These demo visualizations/dashboards are created in Kibana. Dario is pushing to include it in the official monitoring.

Quick start to run BOINC at site

- The BOINC client is installed in cvmfs
  - **/cvmfs/atlas.cern.ch/repo/sw/BOINC/BOINC**
- You only need to configure BOINC on your work node
  - cd /cvmfs/atlas.cern.ch/repo/sw/BOINC/BOINC
  - source setup.sh
  - sh config_boinc.sh [–uid your_boinc_authenticator] [–proxy hostname_proxy_server]
    - We have a default uid, and proxy is for work nodes behind firewall
- Now, BOINC is running, and this includes the DBRSC!
  - Optimize the resource allocation/configuration dynamically
  - Maintain the sanity of BOINC (kill or start according to system load)
- The command Boinc_agent is all you need

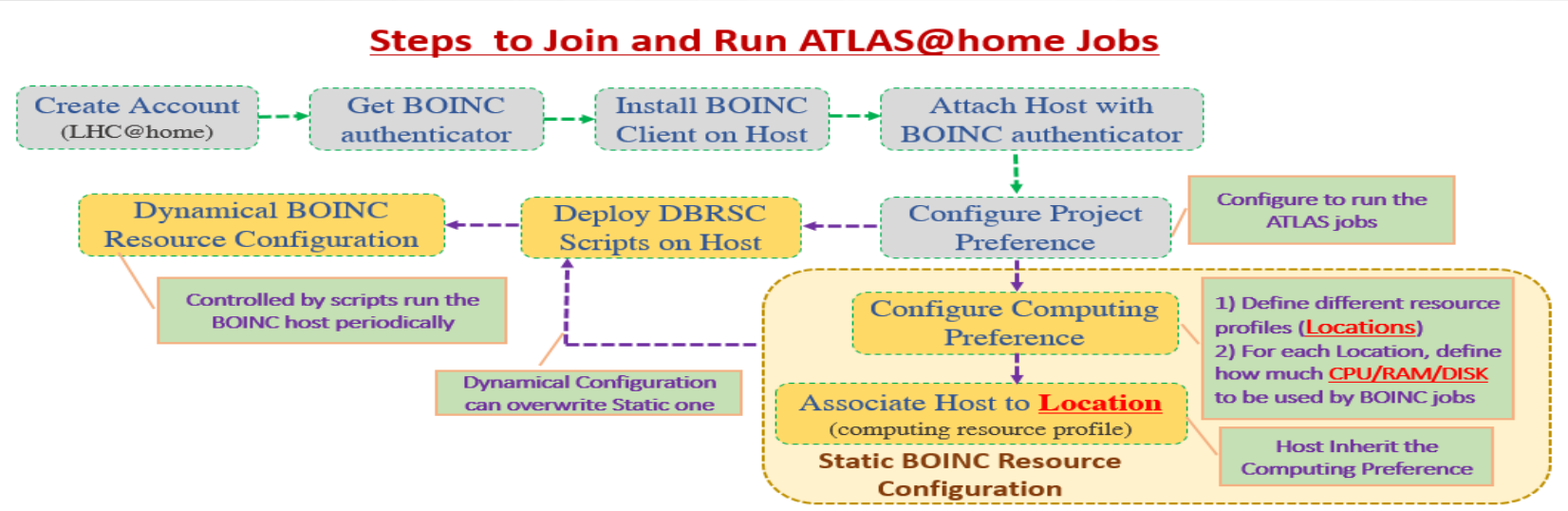
BOINC can be run as any user!

Non SLC/CC 6 nodes need singularity

```
Boinc_agent start|stop|stop_suspend|restart|status|reload|update|nmorework|allowmorework|abort|suspend|resume|show_task
```

Detailed document in Twiki

- For sites who want to join, we provide all the instruction documents here:



Steps to follow if you want to install and configure your own BOINC, otherwise, the quick start recipe works!

You can also install the BOINC client with an all in one script to deploy on cluster: installing BOINC, start running ATLAS jobs, dynamically configure BOINC resources.

https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/VolunteerComputing

从 Firefox 导入 | Bookmarks Bar | 设置 | 工作常用 | 会议 | 项目与论文 | web navigation

VolunteerComputing

This page contains links to technical details on ATLAS@Home.

BOINC Client Side

- JoinAtlasAtHome - Prerequisite before running ATLAS@home jobs, read it before getting started with ATLAS@home jobs
- RunningATLASAtHomeOnACluster - How to set up and run native ATLAS@home jobs on a cluster
- RunningATLASAtHomeOnLinux - BOINC client installation and CML instructions for Linux
- MonitoringATLASAtHomeJobs - useful for the use case of running ATLAS@home on a cluster

BOINC Server Side

- MachineSetup - Services used to run ATLAS@Home
- InstallBoincServer - How to install and setup a BOINC server (not needed any more as BOINC servers are managed by CERN IT)
- ConfigureATLASProject - Details on how the ATLAS@home project works and how to configure new app versions
- MakingVMImage - How to make or change a VM image

ARC CE Related

- InstallArcCE - Instructions on how to install an ARC CE for ATLAS@Home
- DevelopingArcCEBoincPlugin - Old page tracking development of BOINC plugin for ARC CE

Summary

- ATLAS@home is becoming a big resource contributor to ATLAS, and the resource is stable and reliable
- Backfilling on the BEIJING ATLAS grid site exploit an extra of 28% CPU(6 months), on regular cluster 46%.
- Sites are encouraged to use ATLAS@home to harness their non official ATLAS computing resources and Backfilling running it on the clusters.
- Dynamical BOINC configuration makes sure:
  - Efficiently exploit the available resource
  - Not to affect the Non BOINC jobs
  - Hide the BOINC details to site admins.

Acknowledgements

- Andrej Filipcic (Jozef Stefan Institute, Slovenia)
- Simone Campana (CERN)
- Nils Hoimyr (CERN IT, BOINC Project)
- Ilija Vukotic (University of Chicago)
- Frank Berghaus (University of Victoria)
- Douglas Gingrich (University of Alberta)
- Paul Nilsson (BNL)
- Rod Walker (LMU-München)

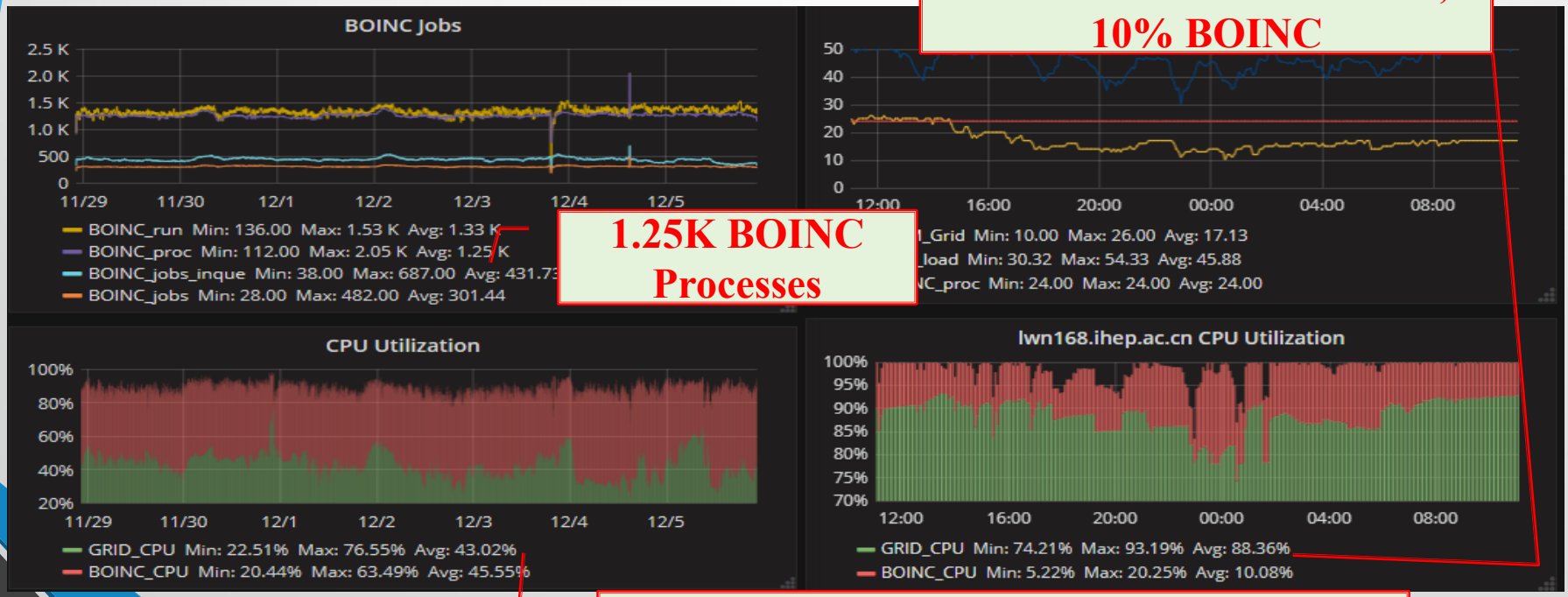
Thanks!

- Backup slides

All BEIJING Nodes

- About 1300 cores running BOINC, extra **45.55%** CPU time is exploited by BOINC, node Avg. CPU utilization reaches **88.6%**. (**98.44 %** in the best case in 24 hours)
- Translate to **595 CPU days** daily in ATLAS@home
- Implemented DBRSC (generic to BOINC)

CPU: 88.4% Non-BOINC , 10% BOINC

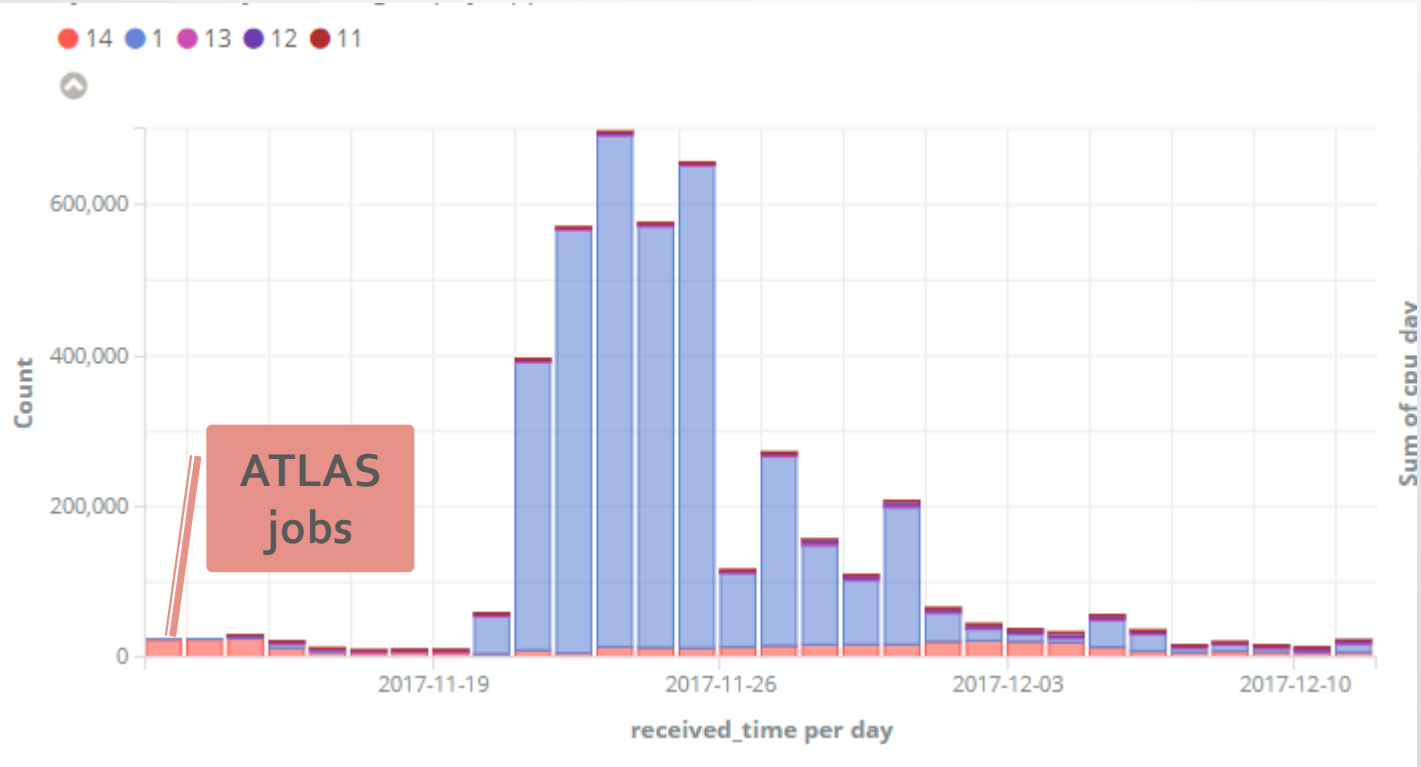


1.25K BOINC Processes

CPU: 43% Non-BOINC , 45.6% BOINC

BOINC scalability

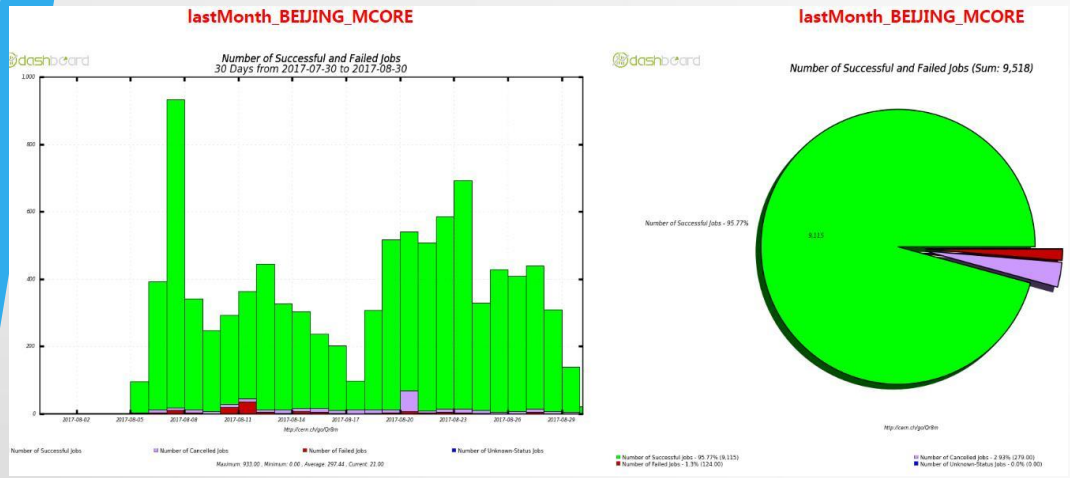
- The current ATLAS only use **2%** of LHC@home resource.
- LHC@home is just one BOINC server, handles 0.7M jobs per day in peak time, ATLAS only processes 10K jobs per day.

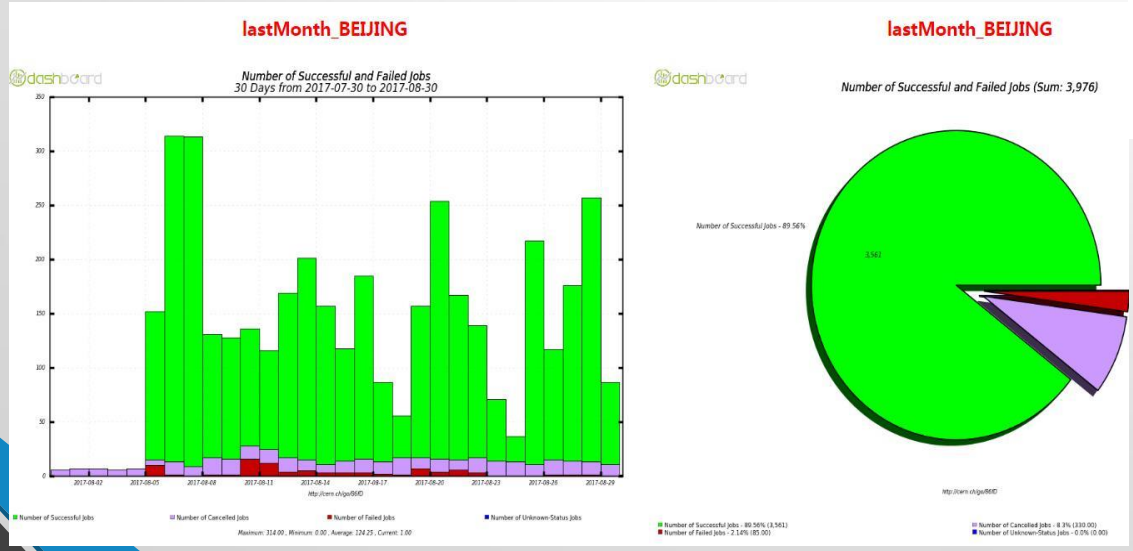


Impact on Tier2?

- Stability
  - It does not affect the production job failure rate (1-2%)
  - It does not affect the throughput of production jobs
    - with running BOINC, ATLAS grid jobs walltime utilization is around 90%
  - Work nodes have reasonable load/swap usage
- Efficiency
  - For production jobs , it does not make it less efficient
    - CPU efficient is less than 1% difference or NONE?!
  - For BOINC jobs, it does not make it less efficient
    - CPU time per event is 0.02% difference between dedicated nodes and nodes mixed with production jobs.
- Manpower to maintain
  - After the right configuration, no manual intervention is needed

Production jobs





1. Production jobs in a month: (with BOINC jobs) 1-2% failure rate

2. In recent 3 weeks, production jobs uses 90% of the walltime

Job Efficiency compare

	Average cpu_eff per job	Average cpu_eff
Production Job (without BOINC)	0.9429	0.9573
Production Job (with BOINC)	0.9384	0.9441
with VS. without	-0.48%	-1.38%

select jobs with : jobstatus=finished, processingtype=simul, nevents=1000, actualcorecount=12, avg_cpu_eff=avg_cputime_day/avg_walltime_day

	Average cputime_day per job	Average cpu_sec_perevent
BOINC Job (Mixed production)	0.2002	306.0081
BOINC Job (Dedicated)	0.1983	305.9520
Dedicated VS. Mixed	-0.94%	-0.02%

select jobs with : jobstatus=finished, processingtype=simul, avg_cpu_eff=avg_cputime_day/avg_walltime_day, nevents=(50,100), both mixed and dedicated nodes are of the same model, 5 mixed nodes, 2 dedicated nodes, mixed nodes all run single core production jobs