ALTAIR IS THE ONLY COMPANY THAT...

...makes HPC middleware:

Altair | PBS Works

...develops HPC applications:

Altair | HyperWorks

...and uses these to do HPC every day:

Altair | ProductDesign
PBS WORKS – SOFTWARE & SOLUTIONS FOR HPC

Engineers & Researchers

ACCESS HPC resources naturally (no IT expertise): run solvers, view progress, manage data, and use 3D remote visualization

Clusters & Clouds

OPTIMIZE HPC clusters and clouds: manage policies, speed turnaround, maximize utilization, ensure availability, stay within budget, maintain security

Admins & Managers

CONTROL HPC resources and provide 360° visibility and agility: configure, deploy, monitor, troubleshoot, report, simulate, analyze, tune
### EDA PRODUCTS

<table>
<thead>
<tr>
<th>NetworkComputer™</th>
<th>WorkloadXelerator™</th>
<th>HERO™</th>
<th>FlowTracer™</th>
</tr>
</thead>
<tbody>
<tr>
<td>• High-performance job scheduler&lt;br&gt; • &gt;5x faster than competition&lt;br&gt; • Full featured: FairShare, preemption, reservations, etc.</td>
<td>• High-performance hierarchical scheduler&lt;br&gt; • ~6-10x increased throughput&lt;br&gt; • Scale-out strategy</td>
<td>• High-performance scheduler for hardware emulation&lt;br&gt; • End-to-end solution: compilation, emulator selection, &amp; regression&lt;br&gt; • Emulation specific metrics for greater visibility into hardware assets</td>
<td>• Platform for developing, executing design flows&lt;br&gt; • Reduces design risk &amp; cost&lt;br&gt; • Accelerates time to market</td>
</tr>
</tbody>
</table>
WHY PBS PRO OPEN SOURCE?
THE HPC WORLD

Public Sector
- Risk takers
- Early adopters
- Natural collaborators
- Open Source

Private Sector
- Risk avoiders
- Later adopters
- Natural competitors
- Commercial

PBS Pro®
Dual-licensing
OPEN & COMMUNITY ORIENTED

Opening the full core of PBS Pro
  ▪ not just a weak subset or older version

Adding staff to support the community

Reorganizing to behave as one of the many contributors

Using community-accepted practices
  ▪ OSI-approved license
  ▪ Github, JIRA, open roadmaps, …

Focusing on longevity
  ▪ For a viable, sustainable community
WHERE CAN I GET IT?
What is PBS Professional®?

PBS Professional software optimizes job scheduling and workload management in high-performance computing (HPC) environments – clusters, clouds, and supercomputers – improving system efficiency and people’s productivity. Built by HPC people for HPC people, PBS Pro™ is fast, scalable, secure, and resilient, and supports all modern infrastructure, middleware, and applications.
COMMUNITY FORUM
An HPC workload manager and job scheduler for desktops, clusters, and clouds.  

http://www.pbspro.org

<table>
<thead>
<tr>
<th>Branch: master</th>
<th>New pull request</th>
</tr>
</thead>
<tbody>
<tr>
<td>latha-subramanian</td>
<td>Automated tests for verifying server dynamic resources(11-14)</td>
</tr>
</tbody>
</table>

- **appveyor**: PP-1021: PBSPro Windows debug executables frequently crash  
a month ago
- **github**: Removing checksign from Travis  
2 days ago
- **.travis**: Add sanitize build support in Travis  
a month ago
- **buildutils**: PP-1174: Fix copyright headers in several files  
3 months ago
- **doc**: PP-1070: Revert if call pbs_statsched signature back to pre-mutlisch...  
14 days ago
- **docker**: PP-1016, PP-1043: choose components of pbs to launch in docker container...  
2 months ago
- **m4**: PP-1067: Utilize new topology information to accurately count devices...  
13 days ago
- **src**: don’t enable PBS_translate_mpp hook  
a day ago
- **test**: Automated tests for verifying server dynamic resources(11-14)  
18 hours ago
- **win_configure**: PP-1067: Utilize new topology information to accurately count devices...  
13 days ago
- **appveyor.yml**: PP-1132, PP-1133, PP-1134, PP-1135, PP-1136, PP-1126: Windows build s...  
3 months ago
- **.gitignore**: Fixed .gitignore to ignore make dist files  
25 days ago
PBS Pro Roadmap

Jon Shelley
Zuletzt geändert Mar 16, 2018 von Bill Nitzberg

v18.1 – target release Q2 2018

Checked-in and available (as beta) in master branch at GitHub.com/PBSPro/pbspro:

- PP-516: Direct write of the job's stdout/err files.
- PP-483: PP-484: PP-485: Enhance output of qstat to make it more admin and script friendly
- PP-617: PP-788: acl_groups and acl_resv_groups will allow secondary groups to submit jobs and reservations
- PP-765: possibility to allow all moms in acl
- PP-389: Allow the admin to suspend jobs for node maintenance
- PP-662, PP-663: Reservation Enhancements
  - PP-662: Allow an admin to modify the start/end times of a reservation
  - PP-663: A scheduling cycle should also be triggered by the end of an advance reservation
- PP-734: Ability to free limited resources when a job is suspended
- PP-339: node ramp down feature: release vnodes early from running jobs
- PP-647: node ramp down feature: release sister nodes before files are staged out
- PP-480: Job Equivalence Class Optimization
- PP-718: Add fairshare usage values to the job_sort_formula
- PP-482: Soft Walltime
- PP-481: Execute execjob_prologue hooks on all sister moms all the time
- PP-927: Update the installation procedure for PBS on Cray CLE 6.0 to use the PBS rpm
- PP-428: New periodic server hook event
WHICH VERSION SHOULD I USE?
YOU HAVE 3 CHOICES

1. Use version 14.1.2 – most current „stable“ version

   ![PBSPro 14.1.2](image)

   [ContOS_7.zip](#) (16 MB) Checksum (SHA256):
   b62bade23be86735e27c720e004ed7cad879c5b3b0d83f9ebebb07a9aa22d228

   [openSUSE_42.3.zip](#) (18 MB) Checksum (SHA256):
   1cd990c3e1d2f4c5a77d0ee711643b3b798790f99da3c31e7ae8324170b086a926a

2. Use the current Master branch – most current version, v18 Beta

   ![GitHub repository](image)

   - 489 commits
   - 2 branches
   - 3 releases
   - 52 contributors

   Branch: master

   New pull request

3. Wait for 18.1 GA – ETA May 2018
### EXAMPLES OF INNOVATION PROJECTS . . .
#### MANY Targeted for V18 (Q2 2018)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MultiSched</strong></td>
<td>(parallel schedulers)</td>
</tr>
<tr>
<td><strong>Green Provisioning™</strong></td>
<td>(power mgmt)</td>
</tr>
<tr>
<td><strong>Conditional resource requests</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Filtering resource requests</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Release limited resources on suspend</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Release nodes from running jobs</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Add fairshare to job sort formula</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Move scheduler config to qmgr</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Modify reservation start/end</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Job equivalence classes</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Docker &amp; nvidia-docker</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Backfill using estimated walltime</strong></td>
<td></td>
</tr>
<tr>
<td><strong>10k jobs/min throughput</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Hadoop/Spark integration</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ARMv8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Advanced allocation management</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Active-active resilience</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Burst buffer (NVRAM) scheduling</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DRMAA v2 support</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CYLC Workflow Engine</strong></td>
<td></td>
</tr>
</tbody>
</table>
DRMAA V2 – RESOURCE MANAGEMENT API

- Open Grid Forum standard
- Heavily used in Life Sciences domain
  - Galaxy: computational biomedical research platform
  - COSMOS: Python library for parallel workflows
  - BioUno project
  - QIAGEN’s CLC Genomics Server
  - BIONANO SOLVE
  - …
- Available now:
  
github.com/PBSPro/DRMAAv2
LET US BUILD IT!
1. Pre-requisites
   srv:~/pbspro-14.1.0 # zypper install gcc make postgresql-server python [...] libical1

2. Configure and make
   srv:~/pbspro-14.1.0 # ./configure --prefix=/opt/pbs --libexecdir=/opt/pbs/libexec --enable-cpu-set
   srv:~/pbspro-14.1.0 # make
   srv:~/pbspro-14.1.0 # make install

3. Run PBS post-install script
   srv:~/pbspro-14.1.0 # /opt/pbs/libexec/pbs_postinstall

4. Change permissions
   srv:~/pbspro-14.1.0 # chmod 4755 /opt/pbs/sbin/pbs_iff /opt/pbs/sbin/pbs_rcp
PBS PRO CONFIGURATION

1. Review pbs.conf

```
srv:~/pbspro-14.1.0 # cat /etc/pbs.conf
PBS_SERVER=srv
PBS_START_SERVER=1
PBS_START_SCHED=1
PBS_START_COMM=1
PBS_START_MOM=0
PBS_EXEC=/opt/pbs
PBS_HOME=/var/spool/pbs
PBS_CORE_LIMIT=unlimited
PBS_RCP=/usr/bin/false
PBS_SCP=/usr/bin/scp
```

2. Set PBS server attributes

```
srv:~/pbspro-14.1.0 # qmgr -c 'set server eligible_time_enable = True'
srv:~/pbspro-14.1.0 # qmgr -c 'set server job_history_enable = True'
```

3. Review PBS MOM config

```
srv:/var/spool/pbs/mom_priv # cat config
...
$jobdir_root /scratch
$usecp *:/home/ /home/
$usecp *:/gpfs/ /gpfs/
```
1. Submit single host jobs
   
   ```bash
echo "sleep 5" | qsub -l select=1:ncpus=12:host=r01n01 -l walltime=00:05:00
   ```

2. Submit multi host jobs
   
   ```bash
echo "sleep 5" | qsub -l select=12:ncpus=12:mpiprocs=2:ompthreads=6
   ```

3. Submit interactive jobs
   
   ```bash
csub -l -X -l select=1:ncpus=12
   ```
FAQ
FAQ

Is this „OpenPBS“?
No, it’s full PBS Professional. You can call it PBS Pro Open Source. OpenPBS was from 1999.

Can I get commercial support for the Open Source version?
Yes, contact pbssales@altair.com

Where do I get documentation?

What license is used?
AGPLv3 & commercial

Where should I start if I want to contribute?
https://pbspro.atlassian.net/wiki

Will new OSS features make their way into the commercial version (and vice-versa)?
Yes, we expect merges for all major versions
Inter-daemon Communication Using TPP

PBS Scheduler

TCP

pbs_comm

TCP

PBS Server

TCP

PBS Commands

PBS MoM

Job Task

MPI

PBS MoM

Job Task

MPI

PBS MoM

Job Task
BEST PRACTICE
When in doubt use Strict Ordering and Backfill plus a Job Sort Formula.

set server job_sort_formula = queue_priority+eligible_time/3600.0-walltime/3600 + adminprio

\[(1+\min(1,\text{eligible\_time}/(\text{walltime}*\text{xover\_frac}-1))/2*\log(\text{walltime})/13.67)\]

\[+ 10^*(\text{pow(ncpus,1.0/3.0)}/\text{cubic\_root\_of\_max\_job\_size})\]

\[+ 100^*(0.5+\log((\text{cur\_user}/100.0+0.005)/(\text{des\_user}/100.0+0.005))/(2*\log(0.005/1.005)))\]

\[+1000^*(\text{queue\_priority}/100.0)\]

\[+10000^*\text{adminp}\]
```
[root@pbs002 ~]# qmgr -c 'p q @default' | grep -v -e ^#
  create queue computeq
  set queue computeq queue_type = Route
  set queue computeq route_destinations = computeq1
  set queue computeq enabled = True
  set queue computeq started = True
...
  create queue computeq1
  set queue computeq1 queue_type = Execution
  set queue computeq1 Priority = 60
  set queue computeq1 from_route_only = True
  set queue computeq1 default_chunk.nodetype = cn
  set queue computeq1 queued_jobs_threshold_res.ncpus = [u:PBS_GENERIC=100]
  set queue computeq1 enabled = True
  set queue computeq1 started = True
```
```python
# gmgr -c 'create hook delay_job_end event="execjob_epilogue"'
# gmgr -c 'set hook delay_job_end alarm = 240'
# gmgr -c 'import hook delay_job_end application/x-python default delay_job_end.py'

import sys
import pbs

e = pbs.event()

try:
    import time
    import random

    if e.job.in_ms_mom():
        min_walltime = 180
        jitter = 30

        walltime = str(e.job.resources_used["walltime"])
        (h, m, s) = walltime.split(':')

        walltime_in_sec = int(h) * 3600 + int(m) * 60 + int(s)
        pbs.logmsg(pbs.LOG_DEBUG, "delay_job_end: walltime used in seconds is %s" % walltime_in_sec)

        if walltime_in_sec < min_walltime:
            delay = min_walltime - walltime_in_sec
            jitter_delay = delay + random.random() * jitter
            pbs.logmsg(pbs.LOG_DEBUG, "delay_job_end: sleep for %s seconds" % str(jitter_delay))
            time.sleep(jitter_delay)

except SystemExit:
    pass
except:
    pbs.logmsg(pbs.LOG_DEBUG, "delay_job_end: failed with %s" % str(sys.exc_info()))
    pass
```
THANK YOU!