CMS Multicore Scheduling

tools and strategy

Outline

- Intro:
  - CMS Multicore application
  - Scheduling goals
- Getting resources
- Using resources

Foreword: CMS Multicore application

- Forked processes multicore MC production developed and tested by CHEP12:
  - Memory reduction: up to 40%
  - Small CPU penalty (output merging)
- Not finally needed for LHC run1: not used for production
- However, needed for run2, CMS decided to go for **multithreaded software**
- CMS multithreaded application **not yet ready**

CMS Multicore job scheduling

Objectives:

- **Avoid splitting resources at sites,** such as dedicated whole node slots, separated queues, etc: complexity and inefficiency

- Integrate scheduling of both **multicore and single-core jobs.**

- Maximize CPU usage:
  - No idle CPUs while jobs are in queue
  - Minimize CPU inefficiency derived from scheduling

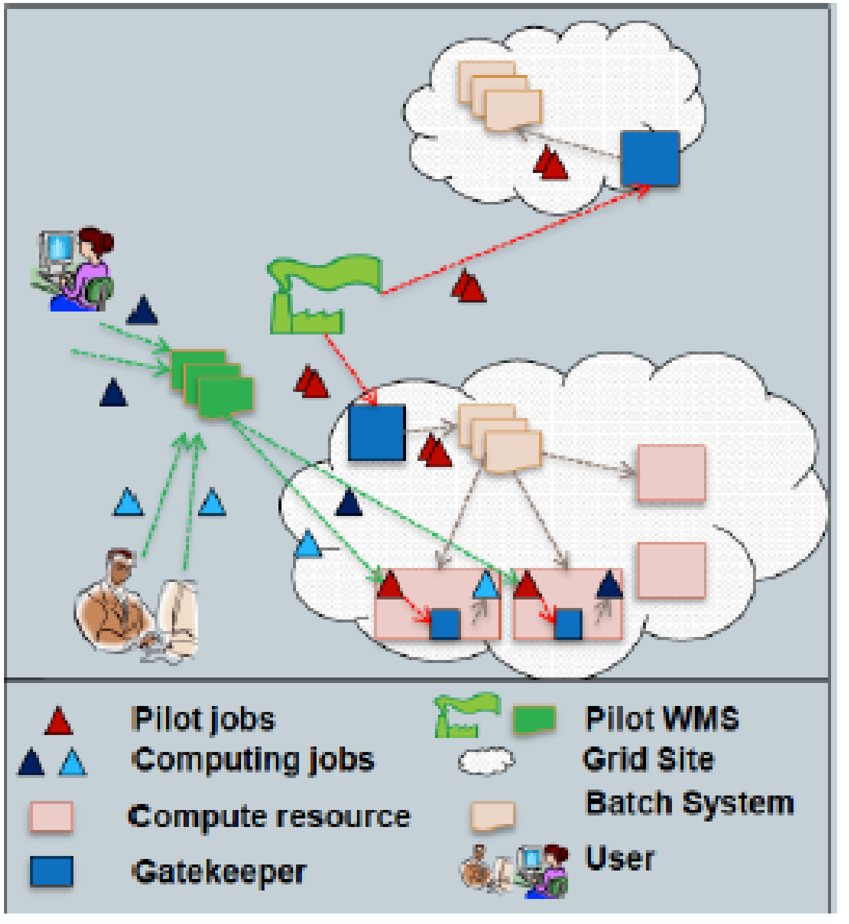
- Getting resources

CMS workflow management

- CMS WMS infrastructure is currently built on **glideinWMS**, a grid-wide batch system, derived from **HTCondor** WMS.

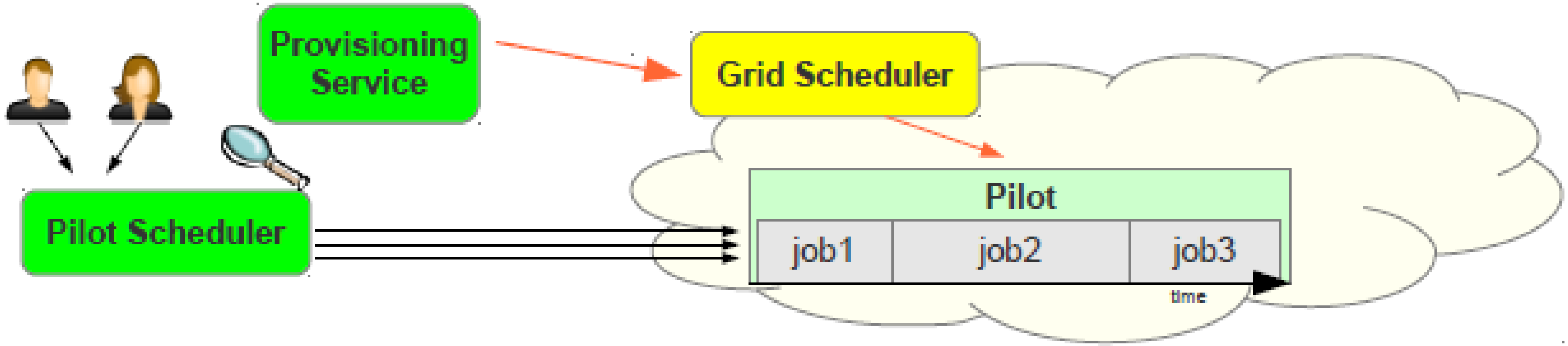
Key concept: **pilot jobs pulling user jobs**

- pilots are sent to all different grid sites matching job resources request
- pilots enter local batch systems queues
- if resources are allocated, running pilots at one or several sites define a virtual pool of computing resources to be used by the grid-wide WMS
- User job is assigned to the first pilot that makes it run



Where are we coming from?

- Grid users have embraced the Pilot model
  - Separates resource provisioning (via pilots) from user job scheduling
  - Pilot resources are temporary, but can execute several user jobs



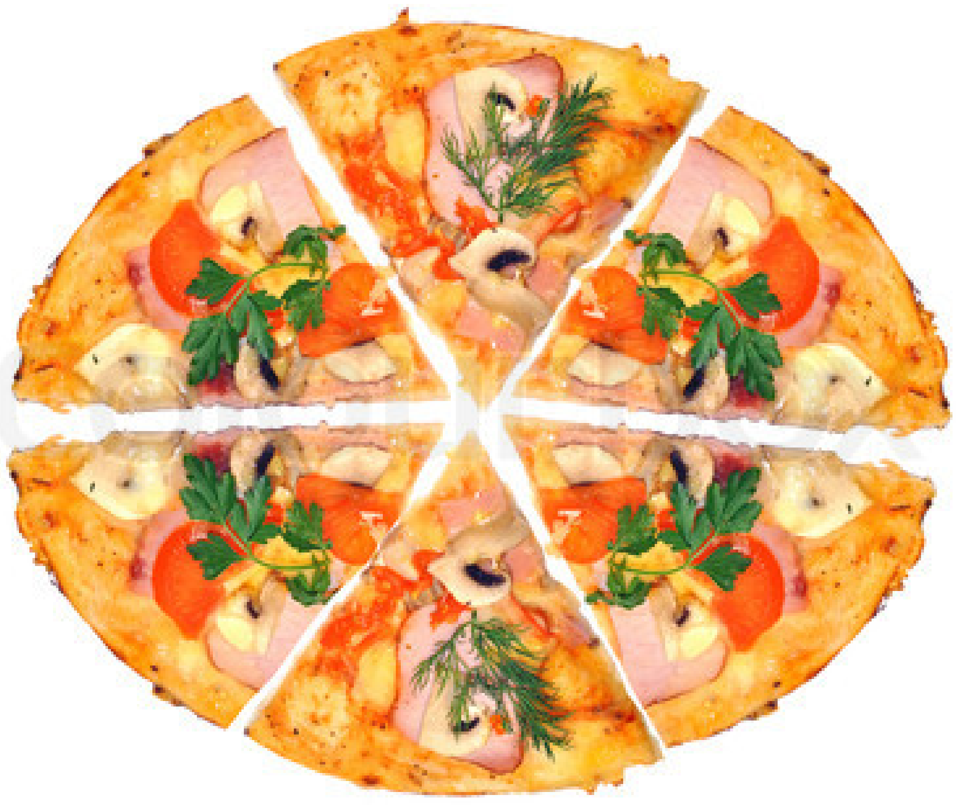
- Pilot overheads have by-and-large been small
  - At most minutes wasted for job fetching and cleanup

What is changing?

- A pilot has traditionally managed a single CPU
  - Which was assigned to a single user job at a time
- Several scientific communities now want more flexibility
  - A single job may need more than one CPU
  - But single-CPU jobs should not be forbidden
- As a consequence, pilots will be expected to **grab multiple-CPUs at once, and then partition** them among user jobs

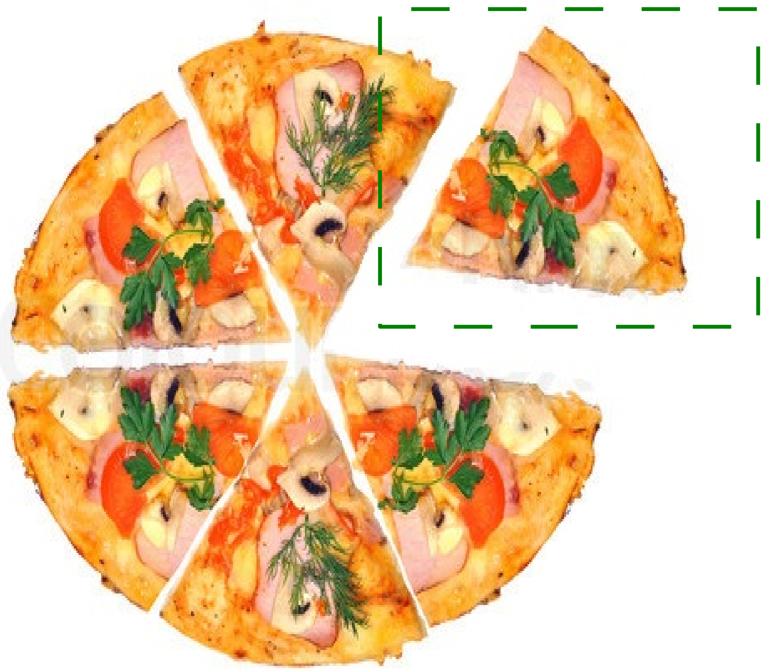
Getting resources

- Fully partitioned WNs: N cores = N slots



Getting resources

- Single core pilots:
  - 1 pilot per core
  - 1 job per pilot at a time



Getting resources

Multicore pilots with dynamic partitioning of allocated resources:

- Take N slots, make M internal slots of variable size.

```
rsl="WholeNodes = False; HostNumber = 1
CPUNumber = 4"
```

Take N slots...

Example for 4-core pilots run at PIC:

- PIC batch system = torque/maui
- 1 core = 1 slot

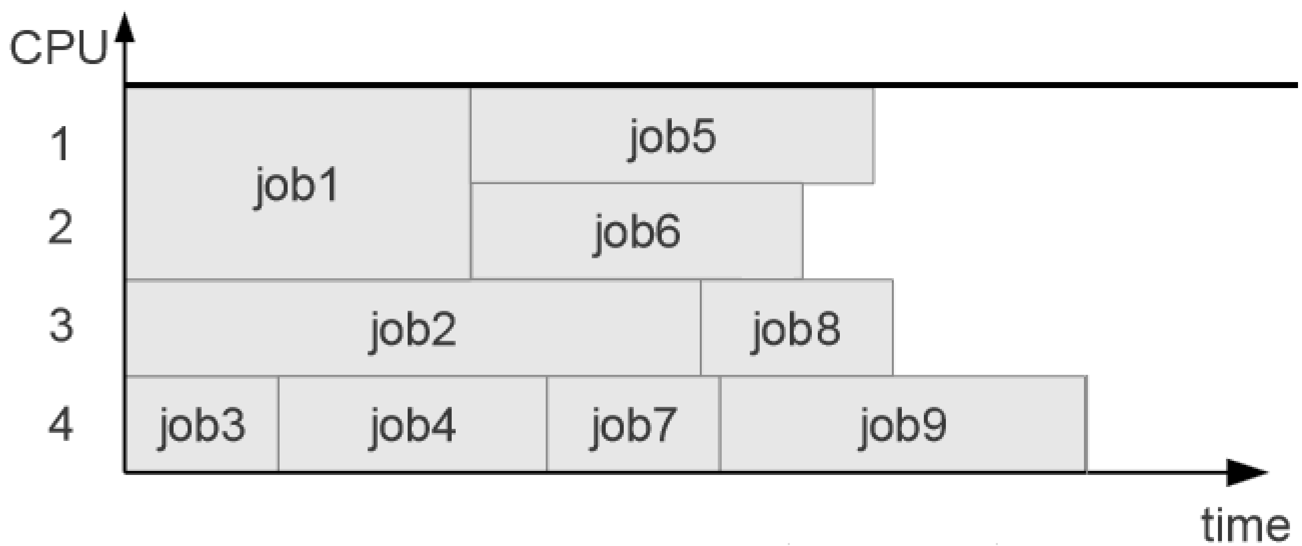
- Pilots arrive at local batch system and request resources: jobs asking for N slots!

```
Job ID               Queue    NDS   TSK    Memory Time  S Time
-------------------- -------- ----- ------ ------ ----- - -----
23744131.pbs02.p     cms_mco      1      4     --  100:0 R 63:38    td457+td457+td457+td457
23744132.pbs02.p     cms_mco      1      4     --  100:0 R 32:21    td458+td458+td458+td458
23744133.pbs02.p     cms_mco      1      4     --  100:0 R 31:45    td457+td457+td457+td457
23744134.pbs02.p     cms_mco      1      4     --  100:0 R 31:12    td458+td458+td458+td458
23744135.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744136.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744137.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744138.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744139.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744140.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744141.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744142.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744143.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
23744144.pbs02.p     cms_mco      1      4     --  100:0 Q    --      --
```

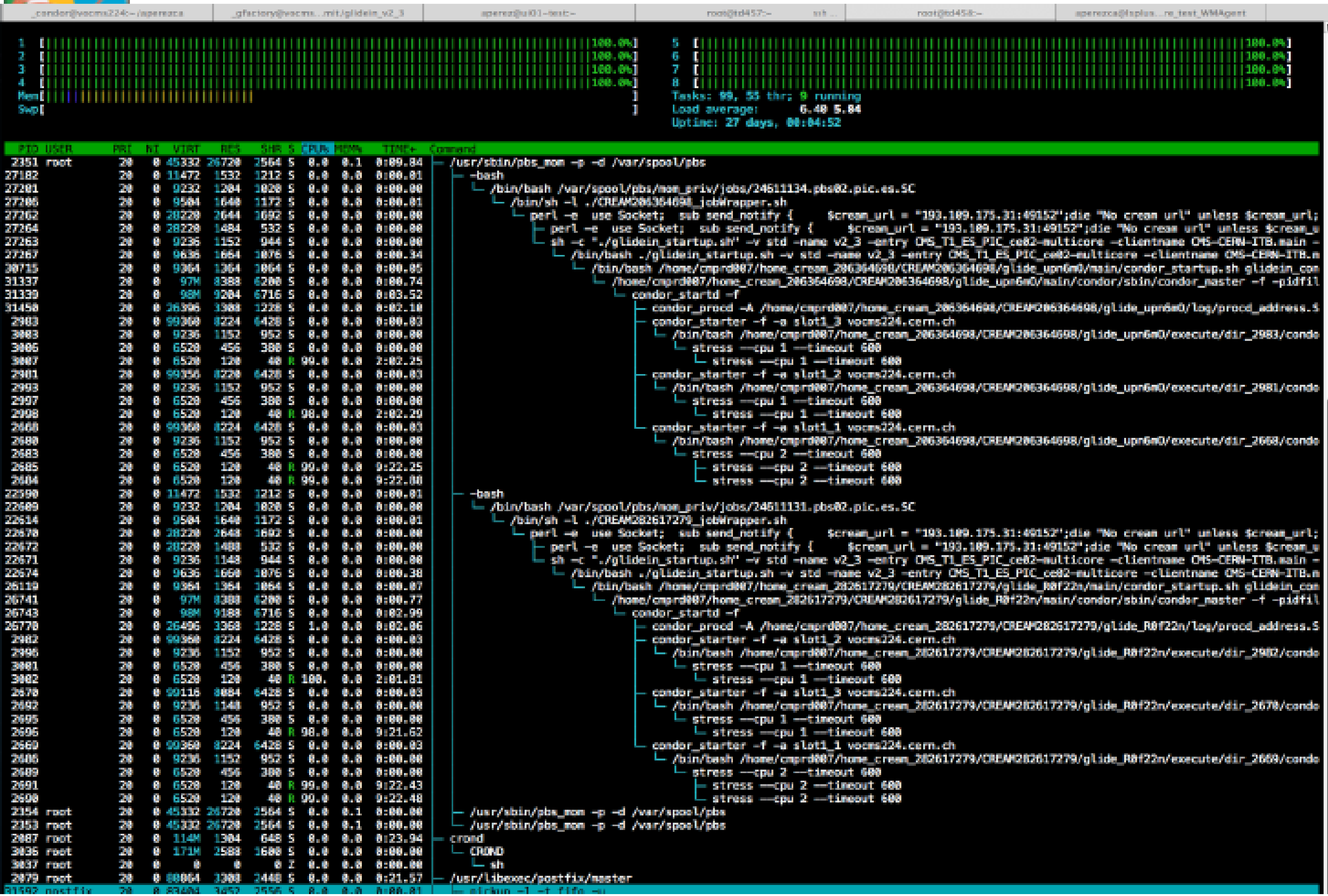
- Using allocated resources

Multicore pilots

- Multicore pilots with internal partitioning or resouces: dynamical internal slots
  - Essential for multicore jobs
  - Advantageous for single core jobs
  - Can handle both types simultaneously



...make M internal slots



Pilot resources allocation

- Pilots can pull different types of stress test jobs and run them simultaneously

```
├─ /home/cmprd007/home_cream_876670542/CREAM876670542/glic
│   └─ condor_startd -f
│       ├─ condor_starter -f -a slot1_4 vocms224.cern.ch
│       │   └─ /bin/bash /home/cmprd007/home_cream_876670542/
│       │       └─ stress --cpu 1 --timeout 200
│       │           └─ stress --cpu 1 --timeout 200
│       ├─ condor_starter -f -a slot1_3 vocms224.cern.ch
│       │   └─ /bin/bash /home/cmprd007/home_cream_876670542/
│       │       └─ stress --cpu 1 --timeout 200
│       │           └─ stress --cpu 1 --timeout 200
│       ├─ condor_starter -f -a slot1_2 vocms224.cern.ch
│       │   └─ /bin/bash /home/cmprd007/home_cream_876670542/
│       │       └─ stress --cpu 1 --timeout 200
│       │           └─ stress --cpu 1 --timeout 200
│       ├─ condor_starter -f -a slot1_1 vocms224.cern.ch
│       │   └─ /bin/bash /home/cmprd007/home_cream_876670542/
│       │       └─ stress --cpu 1 --timeout 200
│       │           └─ stress --cpu 1 --timeout 200
│       └─ condor_procd -A /home/cmprd007/home_cream_8766705
└─ /home/cmprd007/home_cream_876670542/CREAM876670542/glic
    └─ condor_startd -f
```

Pilot resources allocation

- Pilots can pull different types of stress test jobs and run them simultaneously

```
'bin/bash ./glidein_startup.sh -v std -cluster 26592 -name
— /bin/bash /home/cmprd007/home_cream_412750610/CREAM41275
  ├─ /home/cmprd007/home_cream_412750610/CREAM412750610/gl
  │   └─ condor_startd -f
  │       ├─ condor_starter -f -a slot1_1 vocms224.cern.ch
  │       │   └─ /bin/bash /home/cmprd007/home_cream_41275061
  │       │       └─ stress --cpu 4 --timeout 200
  │       │           ├─ stress --cpu 4 --timeout 200
  │       │           ├─ stress --cpu 4 --timeout 200
  │       │           ├─ stress --cpu 4 --timeout 200
  │       │           └─ stress --cpu 4 --timeout 200
  │       └─ condor_procd -A /home/cmprd007/home_cream_41275
  └─ /home/cmprd007/home_cream_412750610/CREAM412750610/gl
      └─ condor_startd -f
```

Pilot resources allocation

- Pilots can pull different types of stress test jobs and run them simultaneously

```
- /bin/bash ./glidein_startup.sh -v std -cluster 26592 -name v2_3 -entr
   └─ /bin/bash /home/cmprd007/home_cream_114180671/CREAM114180671/glide
       ├─ /home/cmprd007/home_cream_114180671/CREAM114180671/glide_i1tE5Y
       │   └─ condor_startd -f
       │       ├─ condor_starter -f -a slot1_3 vocms224.cern.ch
       │       │   └─ /bin/bash /home/cmprd007/home_cream_114180671/CREAM114
       │       │       └─ stress --cpu 2 --timeout 200
       │       │           ├─ stress --cpu 2 --timeout 200
       │       │           └─ stress --cpu 2 --timeout 200
       │       ├─ condor_starter -f -a slot1_2 vocms224.cern.ch
       │       │   └─ /bin/bash /home/cmprd007/home_cream_114180671/CREAM114
       │       │       └─ stress --cpu 1 --timeout 200
       │       │           └─ stress --cpu 1 --timeout 200
       │       ├─ condor_starter -f -a slot1_1 vocms224.cern.ch
       │       │   └─ /bin/bash /home/cmprd007/home_cream_114180671/CREAM114
       │       │       └─ stress --cpu 1 --timeout 200
       │       │           └─ stress --cpu 1 --timeout 200
       │       └─ condor_procd -A /home/cmprd007/home_cream_114180671/CREAM
       └─ /home/cmprd007/home_cream_114180671/CREAM114180671/glide_i1tE5Y
           └─ condor_startd -f
```

Pilot resources allocation

- Pilots can pull different types of stress test jobs and run them simultaneously

```
└─ /bin/bash ./glidein_startup.sh -v std -cluster 27583 -name
     └─ /bin/bash /home/cmprd007/home_cream_626441369/CREAM62644
          ├─ /home/cmprd007/home_cream_626441369/CREAM626441369/gl
          │    └─ condor_startd -f
          │         ├─ condor_starter -f -a slot1_4 vocms224.cern.ch
          │         │    └─ /bin/bash /home/cmprd007/home_cream_62644136
          │         │         └─ stress --cpu 2 --timeout 300
          │         │              ├─ stress --cpu 2 --timeout 300
          │         │              └─ stress --cpu 2 --timeout 300
          │         ├─ condor_starter -f -a slot1_3 vocms224.cern.ch
          │         │    └─ /bin/bash /home/cmprd007/home_cream_62644136
          │         │         └─ stress --cpu 2 --timeout 300
          │         │              ├─ stress --cpu 2 --timeout 300
          │         │              └─ stress --cpu 2 --timeout 300
          │         ├─ condor_starter -f -a slot1_2 vocms224.cern.ch
          │         │    └─ /bin/bash /home/cmprd007/home_cream_62644136
          │         │         └─ stress --cpu 2 --timeout 300
          │         │              ├─ stress --cpu 2 --timeout 300
          │         │              └─ stress --cpu 2 --timeout 300
          │         ├─ condor_starter -f -a slot1_1 vocms224.cern.ch
          │         │    └─ /bin/bash /home/cmprd007/home_cream_62644136
          │         │         └─ stress --cpu 2 --timeout 300
          │         │              ├─ stress --cpu 2 --timeout 300
          │         │              └─ stress --cpu 2 --timeout 300
          │         └─ condor_procd -A /home/cmprd007/home_cream_62644
          └─ /home/cmprd007/home_cream_626441369/CREAM626441369/gl
               └─ condor_startd -f
```

Pilot resources allocation

- Pilots can pull different types of stress test jobs and run them simultaneously

```
─ /bin/bash ./glidein_startup.sh -v std -cluster 27583 -r
  └─ /bin/bash /home/cmprd007/home_cream_050643276/CREAM0
      ├─ /home/cmprd007/home_cream_050643276/CREAM05064327
      │   └─ condor_startd -f
      │       ├─ condor_procd -A /home/cmprd007/home_cream_0
      │       ├─ condor_starter -f -a slot1_1 vocms224.cern.
      │       │   └─ /bin/bash /home/cmprd007/home_cream_0506
      │       │       └─ stress --cpu 4 --timeout 300
      │       │           ├─ stress --cpu 4 --timeout 300
      │       │           ├─ stress --cpu 4 --timeout 300
      │       │           ├─ stress --cpu 4 --timeout 300
      │       │           └─ stress --cpu 4 --timeout 300
      │       └─ condor_starter -f -a slot1_3 vocms224.cern.
      │           └─ /bin/bash /home/cmprd007/home_cream_0506
      │               └─ stress --cpu 4 --timeout 300
      │                   ├─ stress --cpu 4 --timeout 300
      │                   ├─ stress --cpu 4 --timeout 300
      │                   ├─ stress --cpu 4 --timeout 300
      │                   └─ stress --cpu 4 --timeout 300
      └─ /home/cmprd007/home_cream_050643276/CREAM05064327
          └─ condor_startd -f
```

Pilot resources allocation

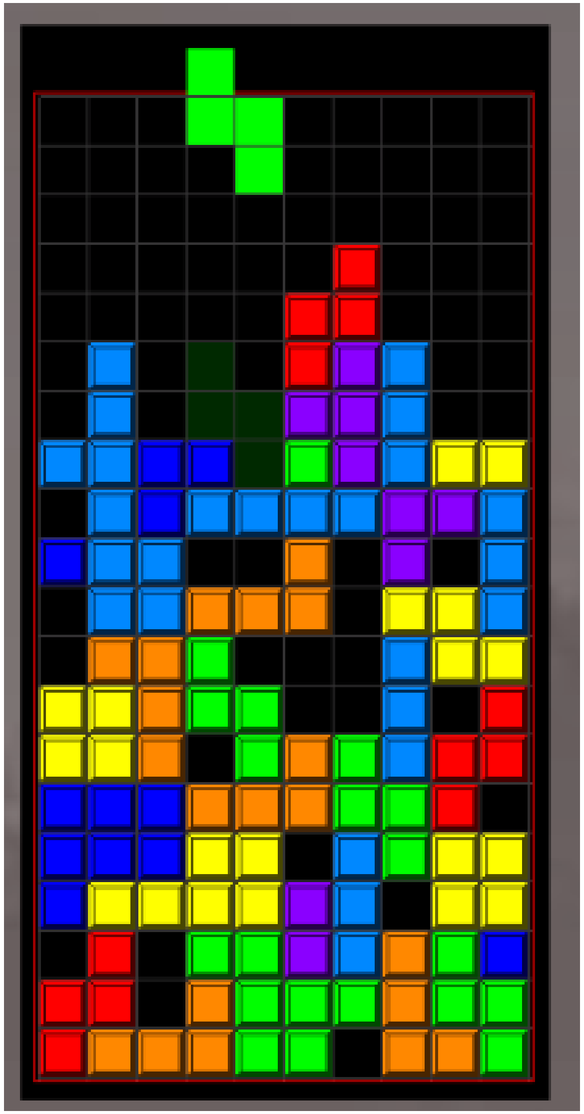
- Pilots can pull different types of stress test jobs and run them simultaneously

```
└─ /bin/bash ./glidein_startup.sh -v std -cluster 27583 -name
    └─ /bin/bash /home/cmprd007/home_cream_050643276/CREAM05064
        ├─ /home/cmprd007/home_cream_050643276/CREAM050643276/gl
        │   └─ condor_startd -f
        │       ├─ condor_procd -A /home/cmprd007/home_cream_05064
        │       ├─ condor_starter -f -a slot1_3 vocms224.cern.ch
        │       │   └─ /bin/bash /home/cmprd007/home_cream_05064327
        │       │       └─ stress --cpu 4 --timeout 300
        │       │           ├─ stress --cpu 4 --timeout 300
        │       │           ├─ stress --cpu 4 --timeout 300
        │       │           ├─ stress --cpu 4 --timeout 300
        │       │           └─ stress --cpu 4 --timeout 300
        │       ├─ condor_starter -f -a slot1_2 vocms224.cern.ch
        │       │   └─ /bin/bash /home/cmprd007/home_cream_05064327
        │       │       └─ stress --cpu 2 --timeout 300
        │       │           ├─ stress --cpu 2 --timeout 300
        │       │           └─ stress --cpu 2 --timeout 300
        │       └─ condor_starter -f -a slot1_1 vocms224.cern.ch
        │           └─ /bin/bash /home/cmprd007/home_cream_05064327
        │               └─ stress --cpu 2 --timeout 300
        │                   ├─ stress --cpu 2 --timeout 300
        │                   └─ stress --cpu 2 --timeout 300
        └─ /home/cmprd007/home_cream_050643276/CREAM050643276/gl
            └─ condor_startd -f
```

Job Scheduling

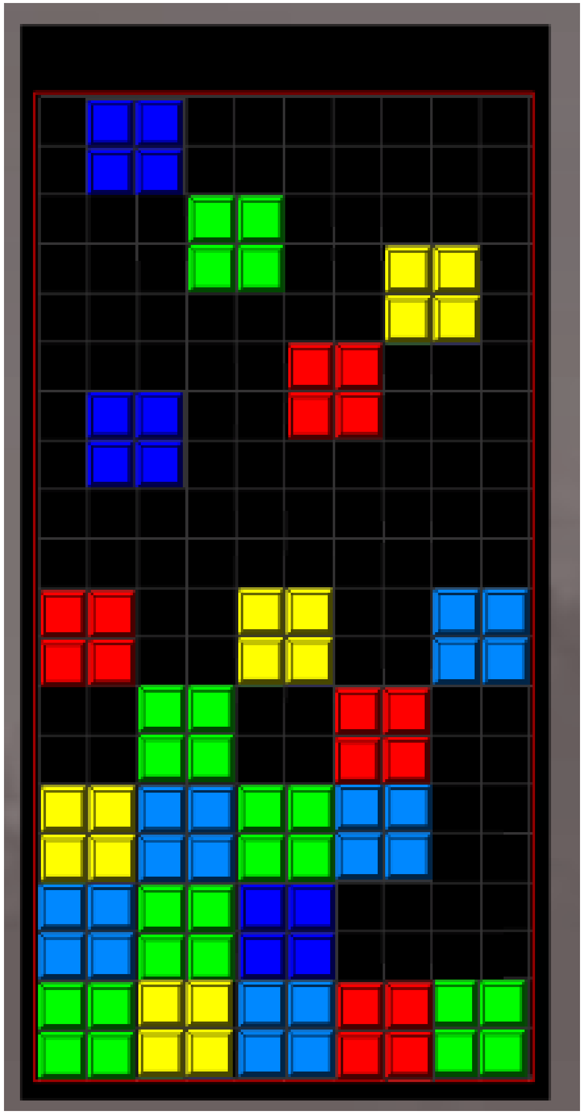
Play Tetris:

- Jobs with different resources requirements
- FIFO: Idle CPUs while enough resources are being released (draining)
- Scheduler with backfilling: needs job lifetime estimation, a complex problem in itself



Job Scheduling

How about playig tetris like this?



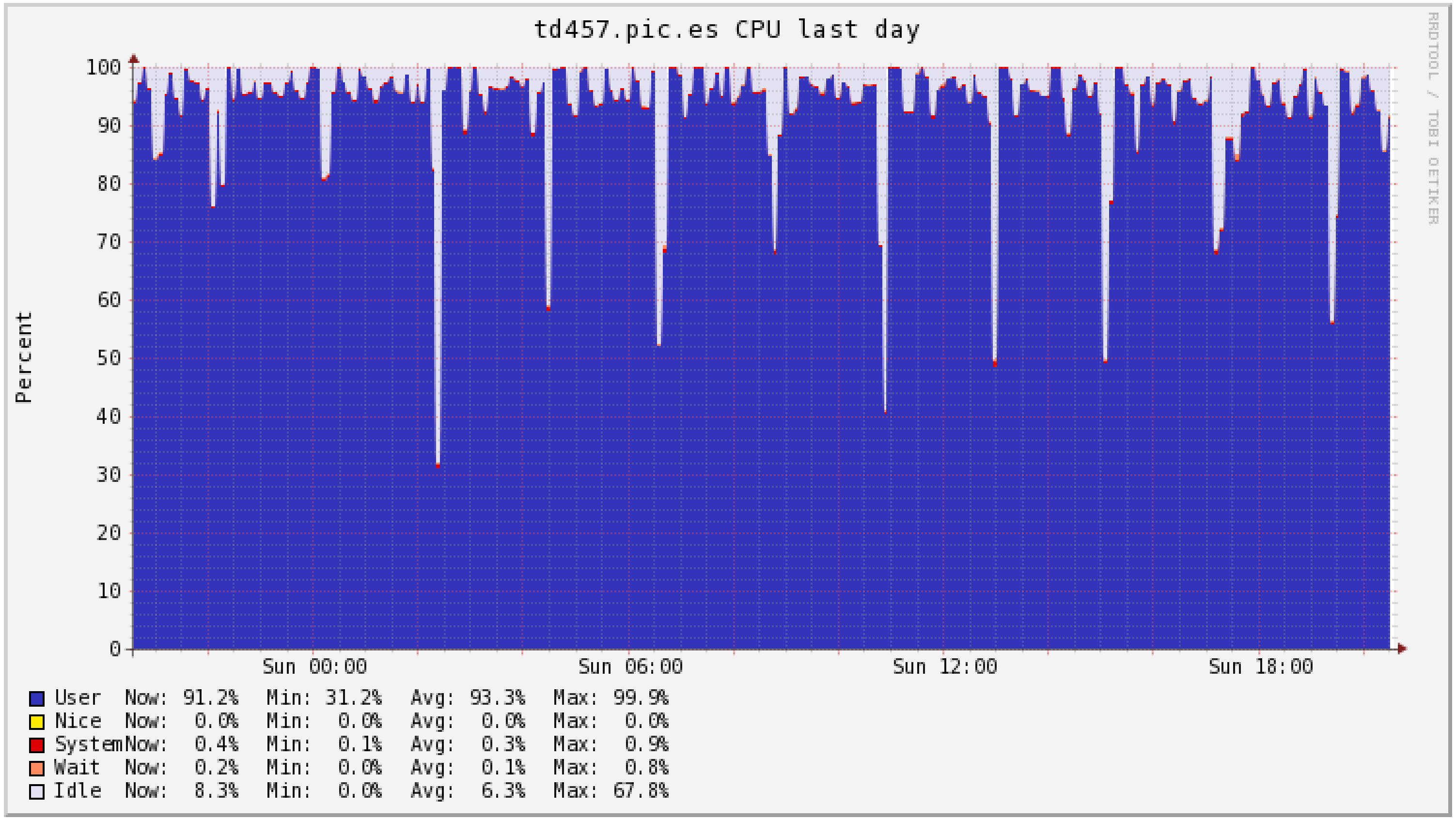
- Multicore pilots hide the different jobs resources requirements from the local batch system/ scheduler: no distinction between single-core and multicore jobs
- Fixed pilot lifetime: no need to estimate job duration (pilot>>job)
- The internal machinery takes care of good CPU usage

Single core MC jobs running inside 4 core pilots

```
  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
17957 cmprd007   20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_17949/condo
18022            20   0  174M 16984  1464 S  0.0  0.1  0:00.37 │  │                    │     └─ python2.6 Startup.py
18308            20   0  9384  1408  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_17949
18404            20   0  581M  238M 25008 R 100.  1.0  1:04.51 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
18051            20   0  174M 16984  1464 S  0.0  0.1  0:00.01 │  │                    │        └─ python2.6 Startup.py
17720            20   0 99116  8080  6428 S  0.0  0.0  0:00.04 │  │                    ├─ condor_starter -f -a slot1_2 vocms231.cern.ch
17724            20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_17720/condo
17756            20   0  174M 16988  1468 S  0.0  0.1  0:00.36 │  │                    │     └─ python2.6 Startup.py
17883            20   0  9384  1408  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_17720
17947            20   0  580M  224M  9648 R 100.  0.9  1:07.87 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
17761            20   0  174M 16988  1468 S  0.0  0.1  0:00.02 │  │                    │        └─ python2.6 Startup.py
16973            20   0 99116  8080  6428 S  0.0  0.0  0:00.05 │  │                    ├─ condor_starter -f -a slot1_1 vocms231.cern.ch
16977            20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_16973/condo
17009            20   0  174M 16984  1468 S  0.0  0.1  0:00.46 │  │                    │     └─ python2.6 Startup.py
17136            20   0  9384  1416  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/execute/dir_16973
17200            20   0  796M  460M 17704 R 99.0  1.9  2:38.11 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
17014            20   0  174M 16984  1468 S  0.0  0.1  0:00.07 │  │                    │        └─ python2.6 Startup.py
13325            20   0 26504  3256  1256 S  0.0  0.0  0:00.63 │  │                    └─ condor_procd -A /home/cmprd007/home_cream_462798618/CREAM462798618/glide_ICNdjD/log/procd_address.ST
 4956            20   0 11472  1532  1212 S  0.0  0.0  0:00.00 │  ├─ -bash
 4981            20   0  9232  1204  1020 S  0.0  0.0  0:00.00 │  │  └─ /bin/bash /var/spool/pbs/mom_priv/jobs/24611431.pbs02.pic.es.SC
 4987            20   0  9504  1640  1172 S  0.0  0.0  0:00.01 │  │     └─ /bin/sh -l ./CREAM624905422_jobWrapper.sh
 5112            20   0 28220  2644  1692 S  0.0  0.0  0:00.00 │  │        └─ perl -e  use Socket;  sub send_notify {     $cream_url = "193.109.175.31:49152";die "No cream url" unless $cream_url;
 5116            20   0 28220  1484   532 S  0.0  0.0  0:00.00 │  │           ├─ perl -e  use Socket;  sub send_notify {     $cream_url = "193.109.175.31:49152";die "No cream url" unless $cream_ur
 5115            20   0  9236  1148   944 S  0.0  0.0  0:00.00 │  │           └─ sh -c "./glidein_startup.sh" -v std -name v2_3 -entry CMS_T1_ES_PIC_ce02-multicore -clientname CMS-CERN-ITB.main -s
 5120            20   0  9636  1656  1076 S  0.0  0.0  0:00.52 │  │              └─ /bin/bash ./glidein_startup.sh -v std -name v2_3 -entry CMS_T1_ES_PIC_ce02-multicore -clientname CMS-CERN-ITB.ma
11967            20   0  9364  1364  1064 S  0.0  0.0  0:00.08 │  │                 └─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/main/condor_startup.sh glidein_conf
13233            20   0   97M  8420  6200 S  0.0  0.0  0:00.14 │  │                    └─ /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/main/condor/sbin/condor_master -f -pidfile
13290            20   0   98M  9260  6708 S  0.0  0.0  0:01.14 │  │                       └─ condor_startd -f
17948            20   0 99104  8072  6428 S  0.0  0.0  0:00.05 │  │                    ├─ condor_starter -f -a slot1_3 vocms231.cern.ch
17956            20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_17948/condo
18012            20   0  174M 16984  1468 S  0.0  0.1  0:00.38 │  │                    │     └─ python2.6 Startup.py
18274            20   0  9384  1412  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_17948
18397            20   0  581M  242M 26564 R 100.  1.0  1:04.48 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
18030            20   0  174M 16984  1468 S  0.0  0.1  0:00.02 │  │                    │        └─ python2.6 Startup.py
17405            20   0 99372  8236  6428 S  0.0  0.0  0:00.05 │  │                    ├─ condor_starter -f -a slot1_2 vocms231.cern.ch
17409            20   0  9236  1252  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_17405/condo
17441            20   0  174M 16988  1468 S  0.0  0.1  0:00.39 │  │                    │     └─ python2.6 Startup.py
17568            20   0  9384  1408  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_17405
17632            20   0  607M  258M 18188 R 100.  1.1  1:21.01 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
17446            20   0  174M 16988  1468 S  0.0  0.1  0:00.03 │  │                    │        └─ python2.6 Startup.py
16542            20   0 99116  8080  6428 S  0.0  0.0  0:00.04 │  │                    ├─ condor_starter -f -a slot1_1 vocms231.cern.ch
16546            20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_16542/condo
16578            20   0  174M 16984  1464 S  0.0  0.1  0:00.50 │  │                    │     └─ python2.6 Startup.py
16887            20   0  9384  1412  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_16542
16951            20   0  796M  459M 16612 R 99.0  1.9  2:52.40 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
16583            20   0  174M 16984  1464 S  0.0  0.1  0:00.10 │  │                    │        └─ python2.6 Startup.py
13349            20   0 99368  8228  6428 S  0.0  0.0  0:00.07 │  │                    ├─ condor_starter -f -a slot1_4 vocms231.cern.ch
13819            20   0  9236  1256  1036 S  0.0  0.0  0:00.00 │  │                    │  └─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_13349/condo
14420            20   0  174M 17004  1464 S  0.0  0.1  0:00.70 │  │                    │     └─ python2.6 Startup.py
15216            20   0  9384  1412  1020 S  0.0  0.0  0:00.00 │  │                    │        ├─ /bin/bash /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/execute/dir_13349
15329            20   0  829M  484M  9468 R 99.0  2.0  6:00.50 │  │                    │        │  └─ cmsRun -j FrameworkJobReport.xml PSet.py
14594            20   0  174M 17004  1468 S  0.0  0.1  0:00.18 │  │                    │        └─ python2.6 Startup.py
13327            20   0 26464  3264  1228 S  0.0  0.0  0:00.61 │  │                    └─ condor_procd -A /home/cmprd007/home_cream_624905422/CREAM624905422/glide_wra8i1/log/procd_address.ST
 2354 root       20   0 45324 26712  2564 S  0.0  0.1  0:00.01 │  └─ /usr/sbin/pbs_mom -p -d /var/spool/pbs
```

CPU Usage

- Test MC production workflow managed by 4-core pilots.
  - Job lifetime: ~15min
  - Pilot lifetime: 2h



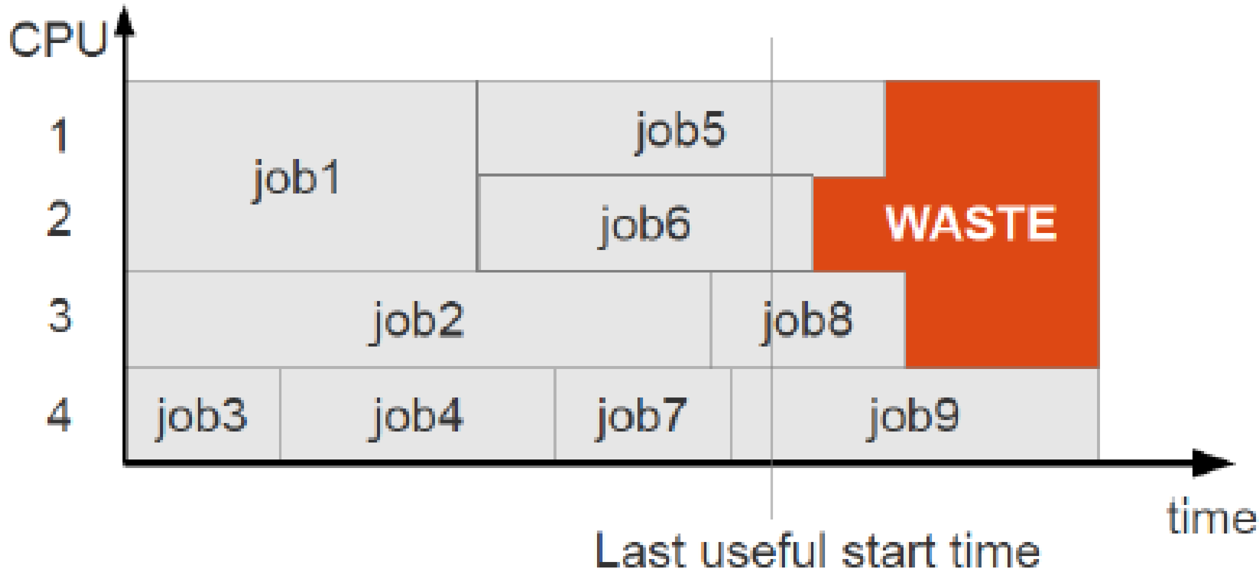
CPU inefficiency from scheduling

- Not specific to multicore pilots: e.g. since pilot starts until first job is pulled, after job completion, etc.

- Exclusive to multicore pilots

  - **Internal slot reconfiguration for dynamic partitioning:** negligible for long jobs

  - **Draining inefficiency** while finishing long jobs using only a fraction of the cores



Minimize scheduling inefficiency

Reducing CPU inefficiency in multicore pilots:

- **Increase pilot lifetime,** to reduce the impact of "draining waste"
- Tune relation between **job duration and pilot lifetime** to minimize inefficiencies at job completion, draining, etc
- **Improved communication** between pilots, jobs and local batch systems. Ideas under development, see:
    - Machine/Job Features WLCG TF:

      https://twiki.cern.ch/twiki/bin/view/LCG/MachineJobFeatures
    - I. Sfiligoi talk at CHEP13:

      http://indico.cern.ch/getFile.py/access?contribId=47&sessionId=5&resId=5&materialId=slides&confId=214784

Implications for sites

OK, so CMS infrastructure is only doing internally what batch systems+schedulers can do at the sites...

- Yes, but:
  - We are providing part of the dynamic feature already included into our pilots
  - Dynamic provisioning of resources may not be an option for some sites:
    - batch system technology
    - local expertise and manpower
  - Separated resources is not the only option.
- By presenting our jobs in a uniform way, we share the responsibility of optimal scheduling with the sites:
  - Uniform resource requests
  - Well defined pilot lifetime
  - Potential for improvement from new tools (MJF TF)

Implications for sites

CMS is proposing a model which potentially helps in solving the scheduling problem:

- CMS does not impose sites to either solve dynamic allocation themselves or separate resources
- Providing resources by 1core=1slot, just as they are doing now, could also be sufficient
  - just allow to take N slots at a time
  - accounting implications to be solved
- If sites do have some advanced scheduling algorithms, that's ok for us too, our pilots will just take resources, then internally use then in a dynamic way

Tests

We propose to continue the development of our tools and do some tests to find out:

- How helpful CMS proposal really is for scheduling at sites
- Could ATLAS potentially use multicore pilots for single core jobs: unify the way resources are requested from the two main players.
- What is the most useful N value: 4, 8, 16...? Who should define this value?
  - The developers of the multicore applications?
  - Just decide on a small number to ease job scheduling?
  - What if we then just redefine the “CPU quantum” to be this number? The min. CPU you get is N cores, then to be used by multicore pilots
- Optimize the relation between job/pilot lifetimes
- ...