Understanding Aprun Use Patterns

Hwa-Chun Wendy Lin
National Energy Research Scientific Computing Center
(NERSC/LBL)

CUG 2009, Atlanta, GA
NERSC: a DOE site providing computing resources to researchers from various disciplines.

Franklin: the newest addition -- Cray XT4 system with almost 10 thousand compute nodes

NERSC policy: give discounts to large jobs to encourage scaling up programs

Large jobs: jobs submitted to a routing queue then get dispatched to the large queue when high number of nodes (>=1024) requested

Do users take advantage of this policy? Do they ask for a large number of nodes, enough to get assigned to the large queue, but use them in independent applications that are launched in parallel?
The Players

- **ALPS (Application Level Placement Scheduler)**
  - Was described in detail at CUG 2006 by Michael Karo of Cray
  - Manages resources (nodes) via apsched
  - Uses resources via aprun

- **Torque/Moab**
  - Is the batch system choice of NERSC
  - Manages designated MOM (job scripts invocation) nodes
  - Enforces scheduling policy
  - Delegates resource management responsibility to ALPS

- **Job life cycle**
  - Next slide (borrowed from Karo) shows how ALPS and Torque/Moab work together
Data Gathering: Sources

- Apsched logs (sdb:/var/log/alps/apsched\(mmdd\))
  - Confirmed: one per job script invocation
  - Bound: one per job script invocation
    - a source for job ID in XT 2.1
  - Placed: one per aprun
  - Released: one per aprun
  - Canceled: one per job script invocation

- Syslog (sdb:/syslog/var/log/messages)
  - Set\_job: one per job script invocation
    - a source for job ID in both XT 2.0 and 2.1
Data Gathering: *aprundat*

- A Perl script
- Runs daily to process the previous day’s apsched log and syslog, as well as the overflow file
- Generates one entry for each aprun with information gathered from the source records.
- Creates four files for each run
  - `<date>_aprundat`: contains aprun records for completed jobs; used by the reporting programs
  - `<date>_overflow`: contains overflow records to be processed the following day
  - `<date>_expired`: contains old overflow records
  - `<date>_incomplete`: contains old arpun records without a job ID
Data Consumption: *aprunrpt*

- A Perl script
- Processes the \(<date>\)_aprundat files whenever desired
- Usage: aprunrpt -m -A \(<date>\)_aprundat
  - -m multiple flag; report only for jobs with multiple apruns
  - -A \(<data>\)_aprundat input data file
- Easy to add more options, such as
  - -u <uid>
  - -s <start time>
  - -e <end time>
  - -n <node name>
Data Consumption: Web Page

### Job details

<table>
<thead>
<tr>
<th>Step ID</th>
<th>Job Name</th>
<th>Owner</th>
<th>Execution queue</th>
<th>Nodes</th>
<th>Available cores per node</th>
<th>MPP secs</th>
<th>Submit</th>
<th>Completion</th>
<th>Nodelist</th>
<th>STDIN</th>
<th>Status</th>
<th>Wall secs</th>
<th>Wall hrs</th>
<th>Requested secs</th>
<th>Requested hrs</th>
<th>MPP hrs</th>
<th>Wait</th>
<th>Raw Secs</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>504759.nid00003</td>
<td></td>
<td></td>
<td>interactive</td>
<td>64</td>
<td>2</td>
<td>1,604,096</td>
<td>May-09-08 06:26:41</td>
<td>May-09-08 06:59:02</td>
<td>12704-12735, 12800-12831</td>
<td></td>
<td>265</td>
<td>1,928</td>
<td>0.54</td>
<td>1,800</td>
<td>0.50</td>
<td>445.58</td>
<td></td>
<td>0.00:13</td>
<td>0.00:00:13</td>
</tr>
</tbody>
</table>

*Indicates dispatch time

### List of aprun commands executed in this job

Number of aprun commands: 4

<table>
<thead>
<tr>
<th>Command</th>
<th>Nodes Used</th>
<th>Run Time (secs)</th>
<th>Start</th>
<th>Complete</th>
<th>Nodelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>RadHyd3D</td>
<td>64</td>
<td>4</td>
<td>May-09-08 06:28:24</td>
<td>May-09-08 06:28:28</td>
<td>12704-12735, 12800-12831</td>
</tr>
<tr>
<td>RadHyd3D</td>
<td>64</td>
<td>44</td>
<td>May-09-08 06:31:32</td>
<td>May-09-08 06:32:16</td>
<td>12704-12735, 12800-12831</td>
</tr>
<tr>
<td>RadHyd3D</td>
<td>64</td>
<td>25</td>
<td>May-09-08 06:32:43</td>
<td>May-09-08 06:33:08</td>
<td>12704-12735, 12800-12831</td>
</tr>
<tr>
<td>RadHyd3D_check</td>
<td>64</td>
<td>135</td>
<td>May-09-08 06:54:18</td>
<td>May-09-08 06:56:33</td>
<td>12704-12735, 12800-12831</td>
</tr>
</tbody>
</table>
Data Gathering Example: Single Aprun

#PBS -q debug
#PBS -l mppwidth=64
cd $PBS_O_WORKDIR
aprun -n 64 ./ping_pong

17:37:35: Confirmed apid 411088 resId 349 pagg 0 nids: 12622-12627,12632-12641
17:37:36: Bound Batch System ID 5820466 pagg 73126 to resId 349
17:37:37: Placed apid 411089 resId 349 pagg 73126 uid 40877 cmd ping_pong
   nids: 12622-12627,12632-12641
17:37:57: Released apid 411089 resId 349 pagg 73126 claim
17:38:15: Canceled apid 411088 resId 349 pagg 73126

Apr 7 17:37:36 nid00576 pbs_mom: set_job, /opt/moab/default/tools/partition.create.xt4.pl
   --confirm -p 349 -j 5820466.nid00003 -a 73126

5820466;12622-12627,12632-12641;1239151057;1239151077;hclin;ping_pong;12622-12627,12632-12641
Data Gathering Example: Sequential Apruns

#PBS -q debug
#PBS -l mppwidth=64
cd $PBS_O_WORKDIR
aprun -n 64 ./ping_pong
aprun -n 32 ./ping_pong
aprun -n 48 ./ping_pong

17:42:12: Confirmed apid 411111 resId 356 pagg 0 nids: 12800-12815
17:42:13: Bound Batch System ID 5820474 pagg 852 to resId 356
17:42:13: Placed apid 411112 resId 356 pagg 852 uid 40877 cmd ping_pong nids: 12800-12815
17:42:34: Released apid 411112 resId 356 pagg 852 claim
17:42:34: Placed apid 411113 resId 356 pagg 852 uid 40877 cmd ping_pong nids: 12800-12807
17:42:45: Released apid 411113 resId 356 pagg 852 claim
17:42:45: Placed apid 411115 resId 356 pagg 852 uid 40877 cmd ping_pong nids: 12800-12811
17:43:00: Released apid 411115 resId 356 pagg 852 claim
17:43:11: Canceled apid 411111 resId 356 pagg 852
Data Gathering Example: Sequential Apruns (cont.)

Apr 7 17:42:13 nid04096 pbs_mom: set_job,
   /opt/moab/default/tools/partition.create.xt4.pl
     --confirm -p 356 -j 5820474.nid00003 -a 852

5820474;12800-12815; 1239151333;1239151354; hclin;ping_pong; 12800-12815
5820474;12800-12815; 1239151354;1239151365; hclin;ping_pong; 12800-12807
5820474;12800-12815; 1239151365;1239151380; hclin;ping_pong; 12800-12811
Data Gathering Example: Parallel Apruns

#PBS -q debug
#PBS -l mppwidth=64
cd $PBS_O_WORKDIR
aprun -n 8 ./ping_pong &
aprun -n 32 ./ping_pong &
aprun -n 16 ./ping_pong
wait

17:43:14: Confirmed apid 411117 resId 357 pagg 0 nids: 12800-12815
17:43:14: Bound Batch System ID 5820475 pagg 1162 to resId 357
17:43:15: Placed apid 411119 resId 357 pagg 1162 uid 40877 cmd ping_pong nids: 12800-12803
17:43:15: Placed apid 411120 resId 357 pagg 1162 uid 40877 cmd ping_pong nids: 12804-12805
17:43:15: Placed apid 411121 resId 357 pagg 1162 uid 40877 cmd ping_pong nids: 12806-12813
17:43:18: Released apid 411120 resId 357 pagg 1162 claim
17:43:20: Released apid 411119 resId 357 pagg 1162 claim
17:43:25: Released apid 411121 resId 357 pagg 1162 claim
17:44:14: Canceled apid 411117 resId 357 pagg 1162
Apr 7 17:43:14 nid04096 pbs_mom: set_job,
   /opt/moab/default/tools/partition.create.xt4.pl
   --confirm -p 357 -j 5820475.nid00003 -a 1162

5820475;12800-12815; 1239151395;1239151398; hclin;ping_pong; 12804-12805
5820475;12800-12815; 1239151395;1239151400; hclin;ping_pong; 12800-12803
5820475;12800-12815; 1239151395;1239151405; hclin;ping_pong; 12806-12813
Data Gathering Example: MPMD Application

#PBS -q debug
#PBS -l mppwidth=64
cd $PBS_O_WORKDIR
aprun -n 8 ./ping_pong : -n 32 ./ping_pong : -n 16 ./ping_pong

17:54:29: Confirmed apid 411173 resId 370 pagg 0 nids: 5787-5789,6586-6598
17:54:30: Bound Batch System ID 5820529 pagg 4171 to resId 370
17:54:31: Placed apid 411174 resId 370 pagg 4171 uid 40877 MPMD cmd ping_pong
    nids: 5787-5789,6586-6596
17:54:51: Released apid 411174 resId 370 pagg 4171 claim
17:55:10: Canceled apid 411173 resId 370 pagg 4171

Apr  7 17:54:30 nid04096 pbs_mom: set_job, /opt/moab/default/tools/partition.create.xt4.pl
    --confirm -p 370 -j 5820529.nid00003 -a 4171

5820529;5787-5789,6586-6598;1239152071;1239152091;hclin;ping_pong;5787-5789,6586-6596
## Data Consumption Example: *Aprunrpt* Output

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Reserved</th>
<th>Used</th>
<th>Start</th>
<th>End</th>
<th>User</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>5820466</td>
<td>16</td>
<td>16</td>
<td>09/04/07 17:37:37</td>
<td>09/04/07 17:37:57</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td>5820474</td>
<td>16</td>
<td>16</td>
<td>09/04/07 17:42:13</td>
<td>09/04/07 17:42:34</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>09/04/07 17:42:34</td>
<td>09/04/07 17:42:45</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>09/04/07 17:42:45</td>
<td>09/04/07 17:43:00</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td>5820475</td>
<td>16</td>
<td>2</td>
<td>09/04/07 17:43:15</td>
<td>09/04/07 17:43:18</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>09/04/07 17:43:15</td>
<td>09/04/07 17:43:20</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>09/04/07 17:43:15</td>
<td>09/04/07 17:43:25</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
<tr>
<td>5820529</td>
<td>16</td>
<td>14</td>
<td>09/04/07 17:54:31</td>
<td>09/04/07 17:54:51</td>
<td>hclin</td>
<td>ping_pong</td>
</tr>
</tbody>
</table>

- Job 5820475 ran multiple apruns in parallel, but was not gaming the system
Challenges

• Constructing timestamps
  – Different format in source files
  – Timestamps for apsched log entries no date
    • month/day: from the file name
    • year: current year
    • -y <year> for processing 12/31 apsched log on 1/1
• Finding job ID in syslog
  – Syslog switches at boot time every so often
  – Syslog contains multiple days’ worth of entries
  – First attempt: use reservation ID as the hash key
    • Not unique due to rapid recycling of reservation ID
  – Second attempt: use reservation ID AND session ID as the key
    • Not unique when syslog spanned many days
  – Finally: save set_job record time for breaking a tie
Future Enhancements

• Data gathering
  – Would like to include aprun command line options
    • syslog
      Apr 11 20:26:40 nid00576 aprun[63195]: apid=437384, Starting, user=32407,\
      cmd_line="aprun -n 32 -d 1 cpl : -n 32 –d 1 csim : -n 16 -d 1 clm : \n      -n 96 -d 1 pop : -n 64 -d 1 cam",num_nodes=60, node_list=6454-6513
  – Would like to include aprun exit status
    • console log
        apid: 453270
      [2009-04-14 13:16:42][c10-3c0s2n3] nwchem[30104]: segfault at 00000003204b1dd0 \n        rip 00000000000ff5e35 rsp 00007fffffff930 error 4

• Data consumption
  – Would like to add more flags for records selection
No, we did not see patterns to indicate users gaming the system

- Were surprised to see a job containing 41,007 apruns ran sequentially and in parallel
- Found unexpected uses of data
  - Frequencies of software package use
  - Association between applications and node failures
- Proved two-step approach wise
  - Collect more info about applications from other system logs
  - Expect more uses for the data
Acknowledgments

• DOE for supporting NERSC
• Michael Karo of Cray for using his slide and providing additional information

• Follow-up e-mail: send to hclin@lbl.gov