A Feasibility Study about Workload Integration between HTCondor Cluster and Slurm Cluster

Ran Du
On behalf of the scheduling group: J. Shi, J. Zou, X. Jiang, Z. Sun
Computing Center of IHEP, CAS
2018-07-12, CHEP2018
Outline

- Motivation
- Job migration between HTCondor and Slurm clusters
- Problems about job scheduling
- Possible solutions
- Summary
Motivation

- HTCondor cluster resource utilization ratio: > 90%
- Slurm cluster resource utilization ratio: ~50%
Aims

Aims to migrate jobs from the HTCondor cluster to the Slurm cluster in some way under control.

• To improve resource utilization of the SLURM cluster.
• To provide more opportunistic resources for HTC jobs.
How to migrate jobs from HTCondor Cluster to Slurm Cluster?

• Three ways are investigated
  • Overlap
  • Flocking
  • HTCondor-C
• **Overlap**: worker nodes in Slurm cluster are added into HTCondor cluster as a single pool (**one pool**).
## Overlap testbed

<table>
<thead>
<tr>
<th>Testbed</th>
<th>HTCondor Cluster</th>
<th>Slurm Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>slurm04.ihep.ac.cn</td>
<td>slurm03.ihep.ac.cn</td>
</tr>
<tr>
<td>Daemons</td>
<td>condor_master</td>
<td>condor_master</td>
</tr>
<tr>
<td></td>
<td>condor_collector</td>
<td>condor_startd</td>
</tr>
<tr>
<td></td>
<td>condor_negotiator</td>
<td>slurmcctld</td>
</tr>
<tr>
<td></td>
<td>condor_schedd</td>
<td>slurmd</td>
</tr>
<tr>
<td></td>
<td>condor_startd</td>
<td>slurmdb</td>
</tr>
</tbody>
</table>
Overlap configuration

HTCondor Cluster

slurm04.ihep.ac.cn : /etc/condor/config.d/conf

DAEMON_LIST = MASTER, SCHEDD, COLLECTOR, NEGOTIATOR, STARTD
CONDOR_HOST = slurm04.ihep.ac.cn

Slurm Cluster

slurm03.ihep.ac.cn : /etc/condor/config.d/worker.conf

DAEMON_LIST = MASTER, STARTD
CONDOR_HOST = slurm04.ihep.ac.cn

slurm03.ihep.ac.cn : /etc/slurm/slurm.conf

# stop condor_startd when slurm jobs coming
Prolog=/usr/local/slurm/condor_off.prolog
# start condor_startd when slurm jobs finished
Epilog=/usr/local/slurm/condor_on.epilog

Thanks to Greg Thain gthain@cs.wisc.edu from Wisconsin University providing the Prolog and Epilog scripts.
Overlap jobs

HTCondor cluster: slurm04.ihep.ac.cn

# submit new jobs
> condor_submit submit_basic_queue_multi.jdf
Submitting job(s)...
3 job(s) submitted to cluster 46

# check job status
> condor_q 46 -af remotehost
slot1@slurm03.ihep.ac.cn
slot2@slurm03.ihep.ac.cn
slot3@slurm03.ihep.ac.cn
• **Flocking**: worker nodes from Slurm cluster are treated as the second pool of the HTCondor cluster (two pools).
# Flocking testbed

<table>
<thead>
<tr>
<th>Testbed</th>
<th>HTCondor Cluster</th>
<th>Slurm Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>slurm04.ihep.ac.cn</td>
<td>slurm03.ihep.ac.cn</td>
</tr>
<tr>
<td>Daemons</td>
<td><code>condor_master</code> <code>condor_collector</code> <code>condor_negotiator</code> <code>condor_schedd</code> <code>condor_startd</code></td>
<td><code>condor_master</code> <code>condor_collector</code> <code>condor_negotiator</code> <code>condor_startd</code> <code>slurmctld</code> <code>slurmd</code> <code>slurmdbd</code></td>
</tr>
</tbody>
</table>
Flocking configuration

Configuration on the HTCondor Cluster

```
slurm04.ihep.ac.cn : /etc/condor/config.d/conf

# flocking_to conf
CONДOR_HOST = slurm04.ihep.ac.cn
IP_ADDRESS = 192.168.51.47

FLOCK_TO = slurm03.ihep.ac.cn
FLOCK_COLLECTOR_HOSTS = $(FLOCK_TO)
FLOCK_NEGOTIATOR_HOSTS = $(FLOCK_TO)

ALLOW_NEGOTIATOR_SCHEDD = $(CONDOR_HOST), $(FLOCK_NEGOTIATOR_HOSTS), $(IP_ADDRESS)
```

Configuration on the Slurm Cluster

```
slurm03.ihep.ac.cn : /etc/slurm/slurm.conf

# stop condor_startd when slurm jobs coming
Prolog=/usr/local/slurm/condor_off.prolog
# start condor_stard when slurm jobs finished
Epilog=/usr/local/slurm/condor_on.epilog

slurm03.ihep.ac.cn: /etc/condor/config.d/conf

# flocking_from conf
FLOCK_FROM = slurm04.ihep.ac.cn
ALLOW_WRITE_COLLECTOR = $(ALLOW_WRITE), $(FLOCK_FROM)
ALLOW_WRITE_STARTD = $(ALLOW_WRITE), $(FLOCK_FROM)
ALLOW_READ_COLLECTOR = $(ALLOW_READ), $(FLOCK_FROM)
$ALLOW_READ_STARTD = $(ALLOW_READ), $(FLOCK_FROM)
```

Thanks to Greg Thain gthain@cs.wisc.edu from Wisconsin University providing the Prolog and Epilog scripts.
# HTCondor Cluster : slurm04.ihep.ac.cn

# check slots status
> condor_status

<table>
<thead>
<tr>
<th>Name</th>
<th>OpSys</th>
<th>Arch</th>
<th>State</th>
<th>Activity</th>
<th>LoadAv</th>
<th>Mem</th>
<th>ActvtyTime</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:slot1@slurm04.ihep.ac.cn">slot1@slurm04.ihep.ac.cn</a></td>
<td>LINUX</td>
<td>X86_64</td>
<td>Claimed</td>
<td>Busy</td>
<td>0.060</td>
<td>32204</td>
<td>0+00:00:03</td>
</tr>
<tr>
<td><a href="mailto:slot2@slurm04.ihep.ac.cn">slot2@slurm04.ihep.ac.cn</a></td>
<td>LINUX</td>
<td>X86_64</td>
<td>Claimed</td>
<td>Busy</td>
<td>0.000</td>
<td>32204</td>
<td>0+00:00:03</td>
</tr>
<tr>
<td><a href="mailto:slot3@slurm04.ihep.ac.cn">slot3@slurm04.ihep.ac.cn</a></td>
<td>LINUX</td>
<td>X86_64</td>
<td>Claimed</td>
<td>Busy</td>
<td>0.000</td>
<td>32204</td>
<td>0+00:00:03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machines Owner Claimed Unclaimed Matched Preempting Drain</th>
</tr>
</thead>
<tbody>
<tr>
<td>X86_64/LINUX</td>
</tr>
<tr>
<td>3 0 3 0 0 0 0</td>
</tr>
<tr>
<td>Total</td>
</tr>
<tr>
<td>3 0 3 0 0 0 0</td>
</tr>
</tbody>
</table>

# submit a new job which will be flocked to slurm03.ihep.ac.cn
> condor_submit submit_basic.jdf
Submitting job(s).
1 job(s) submitted to cluster 43.

# The new job is running on the Slurm cluster
> condor_q 43 -af remotehost
slot1@slurm03.ihep.ac.cn
**HTCondor-C**

- **HTCondor-C**: Slurm cluster and HTCondor cluster are separated clusters.
## HTCondor-C testbed

<table>
<thead>
<tr>
<th>Testbed</th>
<th>HTCondor Cluster</th>
<th>Slurm Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>slurm04.ihep.ac.cn</td>
<td>slurm03.ihep.ac.cn</td>
</tr>
<tr>
<td>Daemons</td>
<td><code>condor_master</code> <code>condor_collector</code> <code>condor_negotiator</code> <code>condor_schedd</code> <code>condor_startd</code> <code>&lt;condor_job_router&gt;</code></td>
<td><code>condor_master</code> <code>condor_collector</code> <code>condor_schedd</code> <code>slurmctld</code> <code>slurmd</code> <code>slurmdbd</code></td>
</tr>
</tbody>
</table>
**HTCondor-C configuration**

### Configuration on the HTCondor Cluster

```bash
/etc/condor/config.d/conf
```

- `SBIN = /usr/sbin`
- `CONDOR_GAHP = $(SBIN)/condor_c-gahp`
- `C_GAHP_LOG = /tmp/CGAHPLog.$(USERNAME)`
- `C_GAHP_WORKER_THREAD_LOG=/tmp/CGAHPWorkerLog.$(USERNAME)`
- `C_GAHP_WORKER_THREAD_LOCK=/tmp/CGAHPWorkerLock.$(USERNAME)`

```bash
/etc/condor/config.d/job_router
```

- `JOB_ROUTER_DEFAULTS
  [ requirements=target.WantJobRouter is True;
    MaxIdleJobs = 10;
    MaxJobs = 100;]`
- `JOB_ROUTER_ENTRIES
  [ GridResource = "condor slurm04.ihep.ac.cn slurm04.ihep.ac.cn";
    name = "slurm testbed";
    set_remote_JobUniverse = 9;
    set_remote_GridResource = "batch slurm";
    set_remote_Requirements = False;]
- `JOB_ROUTER_SCHEDD2_NAME = slurm03.ihep.ac.cn`
- `JOB_ROUTER_SCHEDD2_POOL = slurm03.ihep.ac.cn`
- `DAEMON_LIST = $(DAEMON_LIST) JOB_ROUTER`

### Configuration on the Slurm Cluster

```bash
/etc/slurm/slurm.conf
```

- `# Partitions conf
  PartitionName=comp Nodes=slurm03 Default=NO
  MaxTime=INFINITE State=UP`

```bash
/etc/condor/config.d/conf
```

- `# htcondor-c conf
  SEC_DEFAULT_NEGOTIATION = OPTIONAL
  SEC_DEFAULT_AUTHENTICATION_METHODS = CLAIMTOBE`
HTCondor-C Jobs 1/2

# HTCondor Cluster : slurm04.ihep.ac.cn

# submit an grid universe job
> condor_submit submit_basic_condor_c.jdf
Submitting job(s).
1 job(s) submitted to cluster 41.

# check job status on slurm04.ihep.ac.cn
> condor_q
-- Schedd: slurm04.ihep.ac.cn : <192.168.51.47:6397> @ 07/02/18 18:04:18
OWNER BATCH_NAME SUBMITTED DONE RUN IDLE TOTAL JOB_IDS
duran CMD: sleep.sh 7/2 18:01 _ _ 1 _ _ 1 41.0

1 jobs; 0 completed, 0 removed, 0 idle, 1 running, 0 held, 0 suspended
# Slurm Cluster : slurm03.ihep.ac.cn

# check HTCondor job queue
> condor_q

-- Schedd: slurm03.ihep.ac.cn : <192.168.51.46:26921> @ 07/02/18 18:05:09

<table>
<thead>
<tr>
<th>OWNER</th>
<th>SUBMITTED</th>
<th>DONE</th>
<th>RUN</th>
<th>IDLE</th>
<th>TOTAL</th>
<th>JOB_IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>duran</td>
<td>7/2</td>
<td>18:01</td>
<td>_</td>
<td>_</td>
<td></td>
<td>7.0</td>
</tr>
</tbody>
</table>

1 jobs; 0 completed, 0 removed, 0 idle, 1 running, 0 held, 0 suspended

# check Slurm job queue
> squeue

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>comp</td>
<td>bl_a7e50</td>
<td>duran</td>
<td>R</td>
<td>1:23</td>
<td>1</td>
<td>slurm03</td>
</tr>
</tbody>
</table>
Overlap VS. Flocking VS. HTCondor-C

- Overlap and Flocking are easy to configure, but lack of customized methods to control job scheduling.
- HTCondor-C provides rich customized methods to schedule jobs.
- **HTCondor-C is preferred.**

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Overlap</th>
<th>Flocking</th>
<th>HTCondor-C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>Scale up with worker nodes number.</td>
<td>Scale up with pools number.</td>
<td>Scale up with cluster number.</td>
</tr>
<tr>
<td>Customized job scheduling</td>
<td><strong>Almost none</strong>, depends on HTCondor configuration.</td>
<td><strong>Almost none</strong>, depends on the job queue status.</td>
<td><strong>Rich</strong>: Job Router, Slurm spank plugins, blahpd</td>
</tr>
</tbody>
</table>
Problems with Job Scheduling

- **Job quantity problem**: lots of single-core jobs (> 2000) will generate scheduling burden to Slurm.
- **Resource allocation problem**: resources are from different experiments.
- **System environment problem**: different from Slurm node environment, e.g., OS, mounted filesystem repository, software packages.
Possible Solution to the job quantity problem

• Set Job quantity threshold with scheduling performance feedback
  • HTCondor job_router: limit the number of max routed jobs
  • Slurm spank plugin: set accepted job limit and select proper jobs to run
• Package multiple single-core jobs into one multi-core job
  • Combine a cluster of HTCondor jobs into a parallel Slurm job.
Possible Solution to the Resource Allocation Problem

• An Information system to provide resource share information.
• Resource sharing information including: experiment_lender, group_lender, experiment_borrower, group_borrower, start_date, end_date.
• Support query, add, delete, modify operations.
• NoSql database is considered for the good query performance.
Possible Solution to the system environment problem

• Container: docker / singularity
• To provide lightweight & multiple system environment for jobs from different experiments.
• Image generation for different experiments is in progress.
Summary

• Motivation of workload integration is to improve resource utilization.
• HTCondor-C is more preferred compared with overlap and flocking because of rich customized scheduling strategies.
• Three scheduling problems are confronted and possible solutions are in progress.
Thank you !
& Questions