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

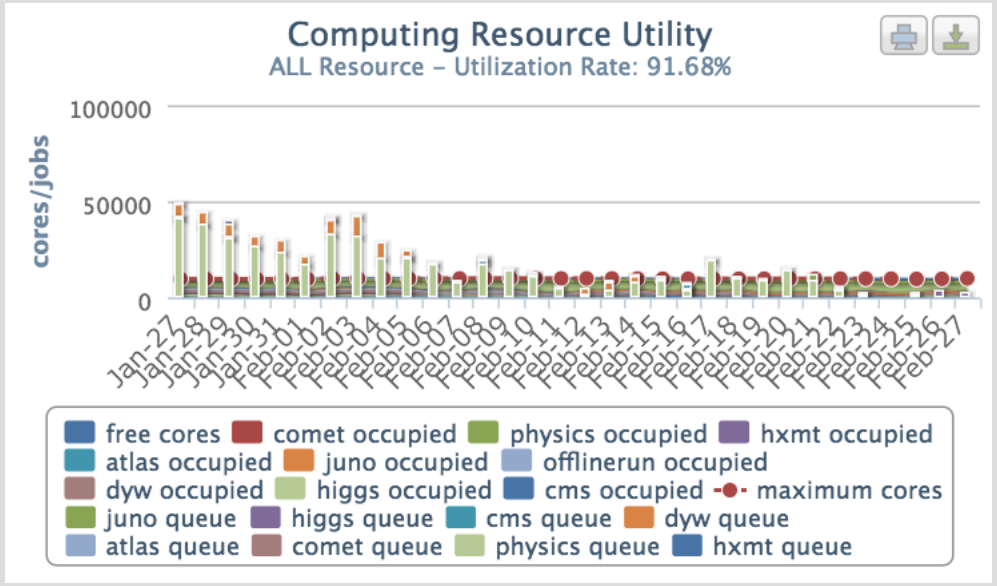
2018-07-12, CHEP2018

Outline

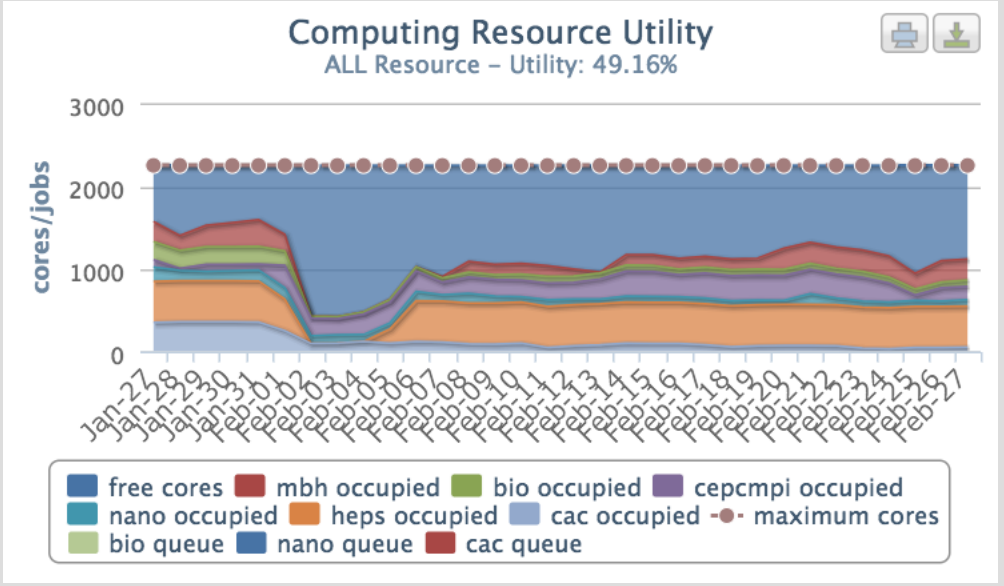
- Motivation
- Job migration between HTCondor and Slurm clusters
- Problems about job scheduling
- Possible solutions
- Summary

Motivation

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



(a) HTCondor Cluster



(b) Slurm Cluster

Aims

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

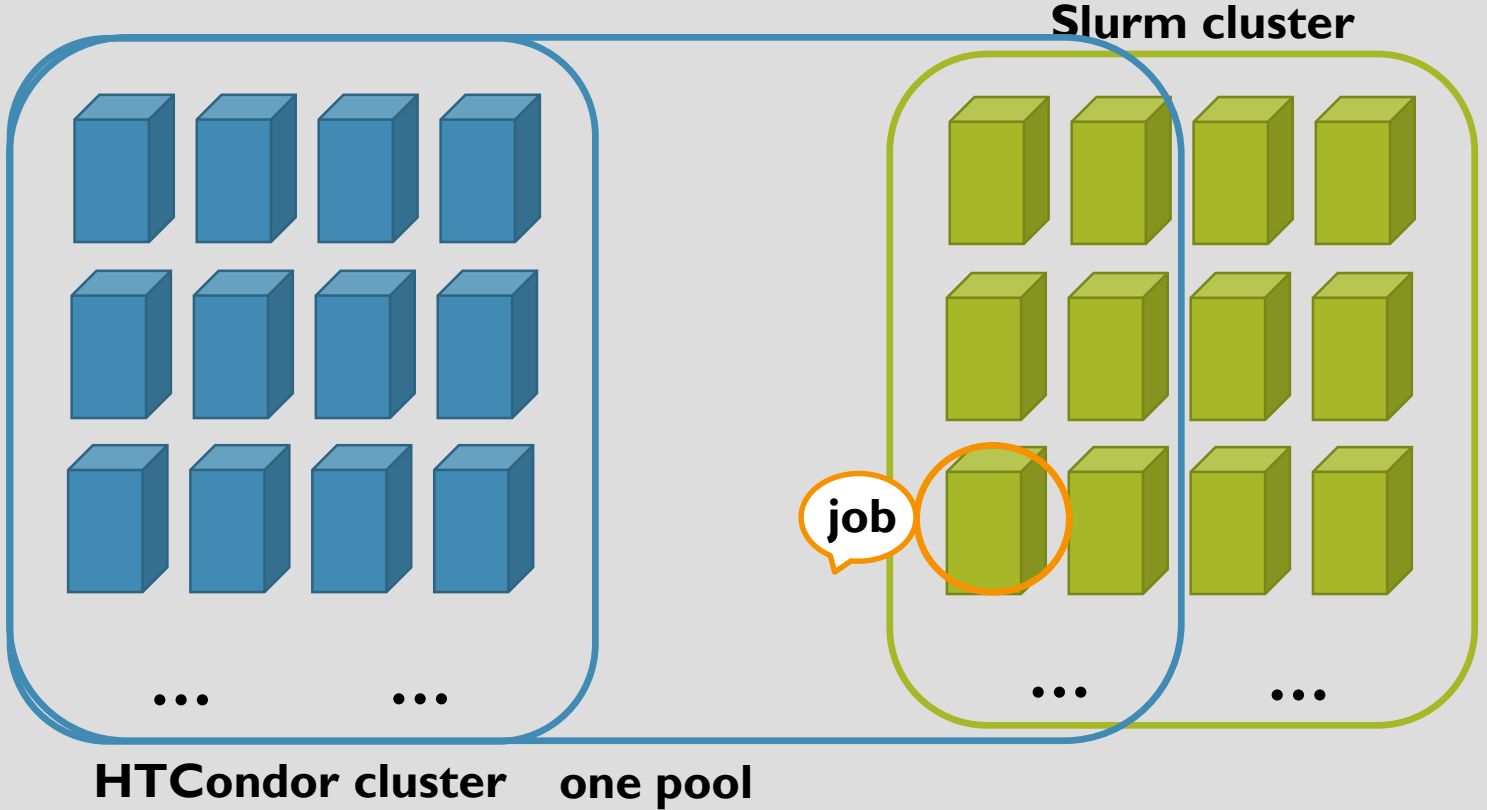
- To improve resource utilization of the SLURM cluster.
- To provide more opportunistic resources for HTC jobs.

How to migrate jobs from HTCondor Cluster to Slurm Cluster ?

- Three ways are investigated
  - Overlap
  - Flocking
  - HTCondor-C

Overlap

- **Overlap** : worker nodes in Slurm cluster are added into HTCondor cluster as a single pool (**one pool**).



Overlap testbed

Testbed	HTCondor Cluster	Slurm Cluster
Node	slurm04.ihep.ac.cn	slurm03.ihep.ac.cn
Daemons	condor_master condor_collector condor_negotiator condor_schedd condor_startd	condor_master condor_startd slurmctld slurmd slurmdbd

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 Wisconson University providing the Prolog and Epilog scripts.

Overlap jobs

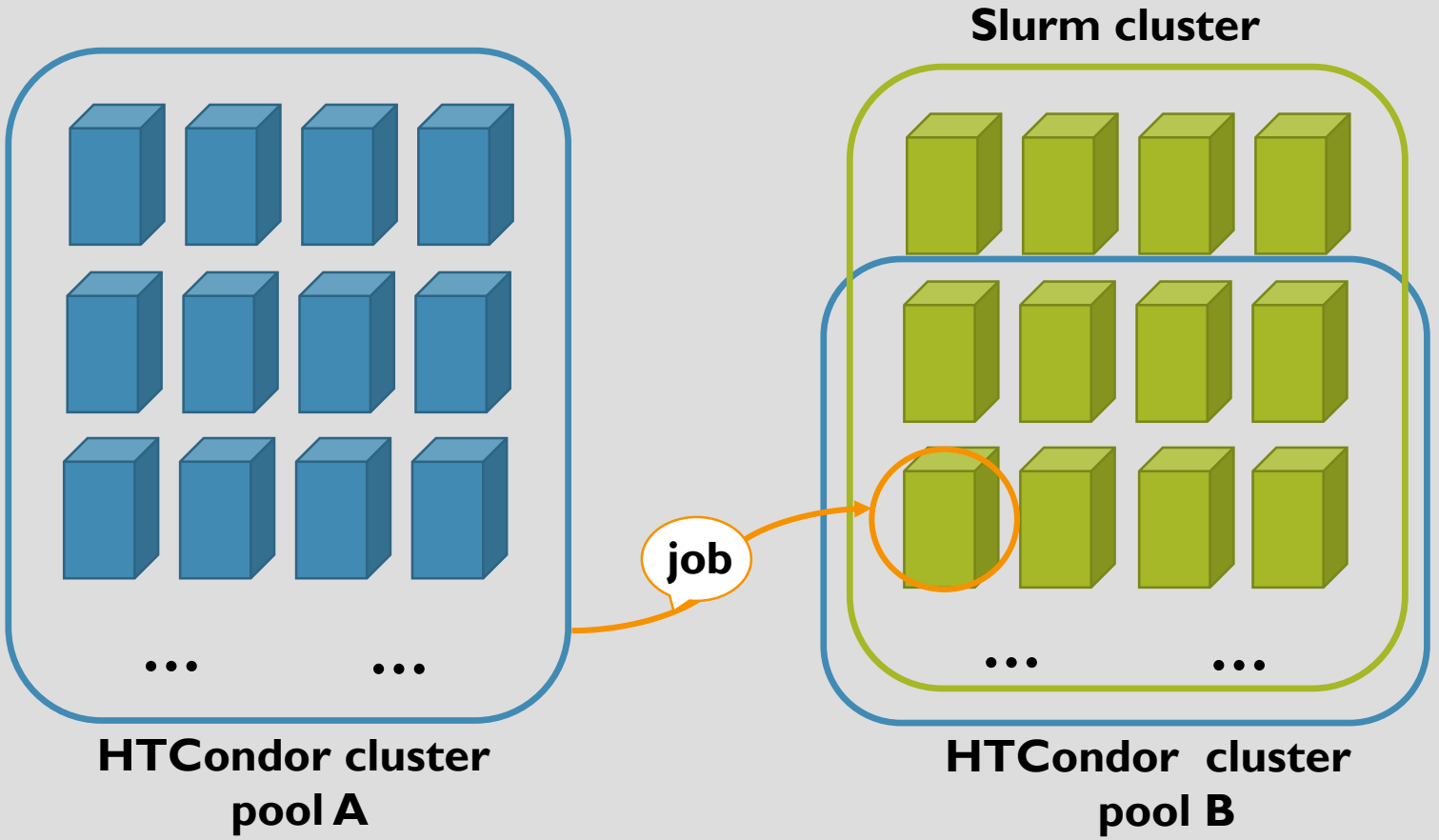
HTCondor cluster : slurm04.ihep.ac.cn

```
# 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

- **Flocking** : worker nodes from Slurm cluster are treated as the second pool of the HTCondor cluster (**two pools**).



Flocking testbed

Testbed	HTCondor Cluster	Slurm Cluster
Node	slurm04.ihep.ac.cn	slurm03.ihep.ac.cn
Daemons	condor_master condor_collector condor_negotiator condor_schedd condor_startd	condor_master condor_collector condor_nogotiator condor_startd slurmctld slurmd slurmdbd

Flocking configuration

Configuration on the HTCondor Cluster

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

```
# flocking_to conf
CONDOR_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 Wisconson University providing the Prolog and Epilog scripts.

Flocking jobs

 # HTCondor Cluster : slurm04.ihep.ac.cn

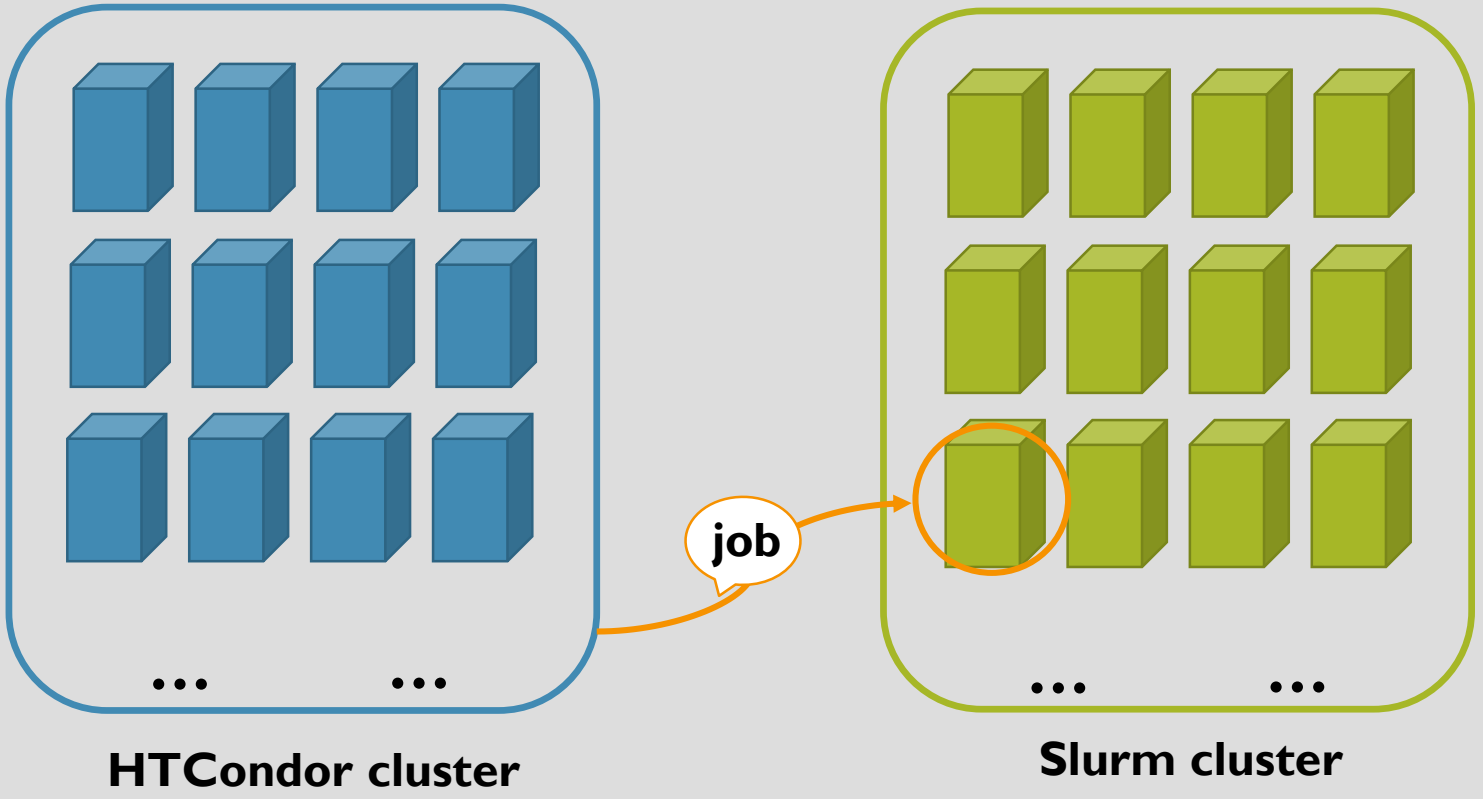
```
#  check slots status
> condor_status
Name                          OpSys      Arch      State      Activity LoadAv Mem    ActvtyTime
slot1@slurm04.ihep.ac.cn LINUX      X86_64 Claimed   Busy       0.060 32204  0+00:00:03
slot2@slurm04.ihep.ac.cn LINUX      X86_64 Claimed   Busy       0.000 32204  0+00:00:03
slot3@slurm04.ihep.ac.cn LINUX      X86_64 Claimed   Busy       0.000 32204  0+00:00:03
                     Machines Owner Claimed Unclaimed Matched Preempting  Drain
  X86_64/LINUX              3        0         3         0         0          0      0
         Total              3        0         3         0         0          0      0

# 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 **seperated clusters**.



HTCondor-C testbed

Testbed	HTCondor Cluster	Slurm Cluster
Node	slurm04.ihep.ac.cn	slurm03.ihep.ac.cn
Daemons	condor_master condor_collector condor_negotiator condor_schedd condor_startd **condor_job_router**	condor_master condor_collector condor_schedd slurmctld slurmd slurmdbd

HTCondor-C configuration

Configuration on the HTCondor Cluster

/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)
```

/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

/etc/slurm/slurm.conf

```
# Partitions conf
PartitionName=comp Nodes=slurm03 Default=YES
MaxTime=INFINITE State=UP
```

/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
```

HTCondor-C Jobs 2/2

 # 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
OWNER BATCH_NAME      SUBMITTED   DONE   RUN    IDLE  TOTAL JOB_IDS
duran      CMD: sleep.sh          7/2  18:01        _      I        _          1 7.0

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

```
# check Slurm job queue
> squeue
             JOBID PARTITION     NAME     USER    ST       TIME  NODES   NODELIST(REASON)
                27      comp   bl_a7e50    duran     R       1:23          1   slurm03
```

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**.

Metrics	Overlap	Flocking	HTCondor-C
Configuration	Scale up with worker nodes number.	Scale up with pools number.	Scale up with cluster number.
Customized job scheduling	**Almost none**, depends on HTCondor configuration.	**Almost none**, depends on the job queue status.	**Rich** : Job Router, Slurm spank plugins, blahpd

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 plungin : 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.
- Resoure 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 sytem 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