HTCondor from the user point of view

by Igor Sfiligoi (UCSD), Jeff Dost (UCSD)

Acknowledgement

- These slides are heavily based on the presentation Todd Tannenbaum gave at CERN in Feb 2011
  https://indico.cern.ch/event/124982/timetable/#20110214.detailed

What is HTCondor

- HTCondor is a Workload Management System
  - i.e. a batch system
- Strong points
  - Fault tolerant
  - Robust feature set
  - Flexible
- Development team dedicated to working closely w/ scientific community as priority #1

How can HTCondor be used

- Managing local processes (local)
- Managing local cluster (~vanilla)
- Connecting clusters (flocking)

We treat these two in this talk as one

- Handling resource overlays (glideins)
- Swiss army knife for accessing other WMS (Condor-G)
  - e.g. Grid, Cloud, pbs, etc.

Submitting Jobs to HTCondor

- Get access to submit host
- Choose a "Universe" for your job
- Make your job "batch-ready"
  - Includes making your data available to your job
- Create a submit description file
- Run condor_submit to put your job(s) in the queue
- Relax while HTCondor manages and watches over your job(s)

Choose the job “Universe”

- Controls how HTCondor handles jobs
- HTCondor universes include:
  - Vanilla (aka regular single node job)
  - Parallel
  - Grid
  - Java
  - VM
  - Standard

Hello World Submit File

```
# Simple condor_submit input file
# (Lines beginning with # are comments)
# NOTE: the words on the left side are not
#       case sensitive, but filenames are!
Universe   = vanilla
Executable = cosmos       # Job's executable
Arguments  = -k 1543.3    # Job's args
Output     = cosmos.out   # Job's STDOUT
Input      = cosmos.in    # Job's STDIN
Log        = cosmos.log   # Log the job's activities
Queue 1                   # Put the job in the queue!
```

condor_submit & condor_q

```
% condor_submit sim.submit
Submitting job(s).
1 job(s) submitted to cluster 1.

% condor_q

-- Submitter: perdita.cs.wisc.edu : <128.105.165.34:1027> :
 ID      OWNER          SUBMITTED     RUN_TIME ST PRI SIZE CMD
1.0      frieda         6/16 06:52    0+00:00:00 I  0   0.0 sim.exe

1 jobs; 1 idle, 0 running, 0 held
```

View the full ClassAd

```
% condor_q -long

-- Submitter: perdita.cs.wisc.edu : <128.105.165.34:1027> :
MyType = “Job”
TargetType = “Machine”
ClusterId = 1
QDate = 1150921369
CompletionDate = 0
Owner = “frieda”
RemoteWallClockTime = 0.000000
LocalUserCpu = 0.000000
LocalSysCpu = 0.000000
RemoteUserCpu = 0.000000
RemoteSysCpu = 0.000000
ExitStatus = 0
...
```

Monitor progress through logs

- The log file you specified is updated every time something happens to the job
  - e.g. submit, start, termination
- Example log file

```
000 (001.000.000) 05/25 19:10:03 Job submitted from host: <128.105.146.14:1816>
...
001 (001.000.000) 05/25 19:12:17 Job executing on host: <128.105.146.14:1026>
...
005 (001.000.000) 05/25 19:13:06 Job terminated. 005 (001.000.000) 05/25 19:13:06 Job
terminated.
(1) Normal termination (return value 0)
...
```

condor_status

- Gives information about the pool:

```
% condor_status
Name           OpSys   Arch   State      Activ LoadAv Mem   ActvtyTime
perdita.cs.wi LINUX   INTEL   Owner      Idle  0.020   511  0+02:28:42
coral.cs.wisc LINUX   INTEL   Claimed    Busy  0.990   511  0+01:27:21
doc.cs.wisc.e LINUX   INTEL   Unclaimed Idle  0.260   511  0+00:20:04
dsonokwa.cs.w LINUX   INTEL   Claimed    Busy  0.810   511  0+00:01:45
ferdinand.cs. LINUX   INTEL   Claimed    Suspe 1.130   511  0+00:00:55
```

- To inspect full ClassAds:
  condor_status -long

General User Commands

- condor_submit # Submit new Jobs
- condor_run # Submit and block
- condor_q # View Job Queue
- condor_status # View Pool Status
- condor_rm # Remove Jobs
- condor_history # Completed Job Info
- condor_hold # Put a job on hold
- condor_release #Release a job from hold

http://research.cs.wisc.edu/htcondor/manual/current/11_Command_Reference.html

HTCondor File Transfer

- HTCondor will transfer files between submit and execute nodes (eliminating the need for NFS) if desired:
  - **ShouldTransferFiles**
    - **YES**: Always transfer files to execution site
    - **NO**: Always rely on a shared filesystem
    - **IF_NEEDED**: HTCondor will automatically transfer the files if the submit and execute machine are not in the same FileSystemDomain (Use shared file system if available)
  - **When_To_Transfer_Output**
    - **ON_EXIT**: Transfer the job's output files back to the submitting machine only when the job completes
    - **ON_EXIT_OR_EVICT**: Like above, but also when the job is evicted

HTCondor File Transfer, cont

- **Transfer_Input_Files**
  - List of files that you want HTCondor to transfer to the execute machine
- **Transfer_Output_Files**
  - List of files that you want HTCondor to transfer from the execute machine
    - If specified, the files must exist when job terminates, or the job will fail (put on hold)
    - If not specified, HTCondor will transfer back all new or modified files in the execute directory (but not subdirectories)

Simple File Transfer Example

```
# Example submit file using file transfer
Universe                = vanilla
Executable              = cosmos
Log                     = cosmos.log
ShouldTransferFiles     = YES
Transfer_input_files    = cosmos.dat
Transfer_output_files   = results.dat
When_To_Transfer_Output = ON_EXIT
Queue
```

These annoying emails

- HTCondor will send an email every time a job finishes
  - Which is potentially nice if you have 10 jobs
  - But it is annoying when you have O(10k)!
- To disable this behavior, add **Notification = Never** to the submit file

Referencing job id

- Each job has a unique ID
  - Composed of (Cluster,Process) pair
- Can reference them in the submit file with
  - $(Cluster)
  - $(Process)

Adding arbitrary attributes

- Arbitrary custom attributes can be added to the submit file in addition to HTCondor built-ins
  - Useful as mnemonic e.g. +AnaType=”Higgs”
  - or to target resources in distributed HTC
    e.g. +DESIRED_Sites=”FNAL,UCSD,Nebraska”
- HTCondor syntax expects a “+” sign in front of custom attrs
  - Otherwise it will error out during submission on attributes it doesn’t recognize

Example submit file

```
# Full submit file example

Universe                = vanilla
Executable              = cosmos
Arguments = cosmos.dat $(Cluster)
Log                     = cosmos.log
ShouldTransferFiles     = YES
Transfer_input_files    = cosmos.dat
Transfer_output_files   = results.dat
When_To_Transfer_Output = ON_EXIT
Notification            = Never
+MyAna                  = “Higgs”
+DESIRED_Sites = “FNAL,UCSD,Nebraska”
Queue
```

We just scratched the surface

- Many more options available
- See HTCondor Manual
  http://research.cs.wisc.edu/htcondor/manual/current/condor_submit.html
- See HTCondorWeek User tutorial
  http://research.cs.wisc.edu/htcondor/CondorWeek2011/tuesday_condor.html

Priorities

Priorities

- Priorities between users set by the pool administrator
  - User cannot alter it
- A user can set priorities relative to his own jobs
  - FIFO by default
  - User can designate some jobs as higher priority (or even lower priority)
    - Again FIFO within the same priority level
      http://research.cs.wisc.edu/htcondor/manual/current/condor_prio.html

Managing priorities

- Check with condor_q
- Modify with condor_prio

```
% condor_q 366701.193

-- Submitter: santa1.claus : <192.168.130.11:9615?sock=9763_cd4c_2> : santa1.claus
ID      OWNER            SUBMITTED     RUN_TIME   ST PRI SIZE CMD
366701.193 frida         12/25 12:01   0+00:00:00 I  0   0.0  cosmis -k 3.4

% condor_prio -10 366701.193
% condor_q 366701.193

-- Submitter: santa1.claus : <192.168.130.11:9615?sock=9763_cd4c_2> : santa1.claus
ID OWNER                 SUBMITTED     RUN_TIME   ST PRI SIZE CMD
366701.193 frida         12/25 12:01   0+00:00:00 I  -10 0.0  cosmos -k 3.4
```

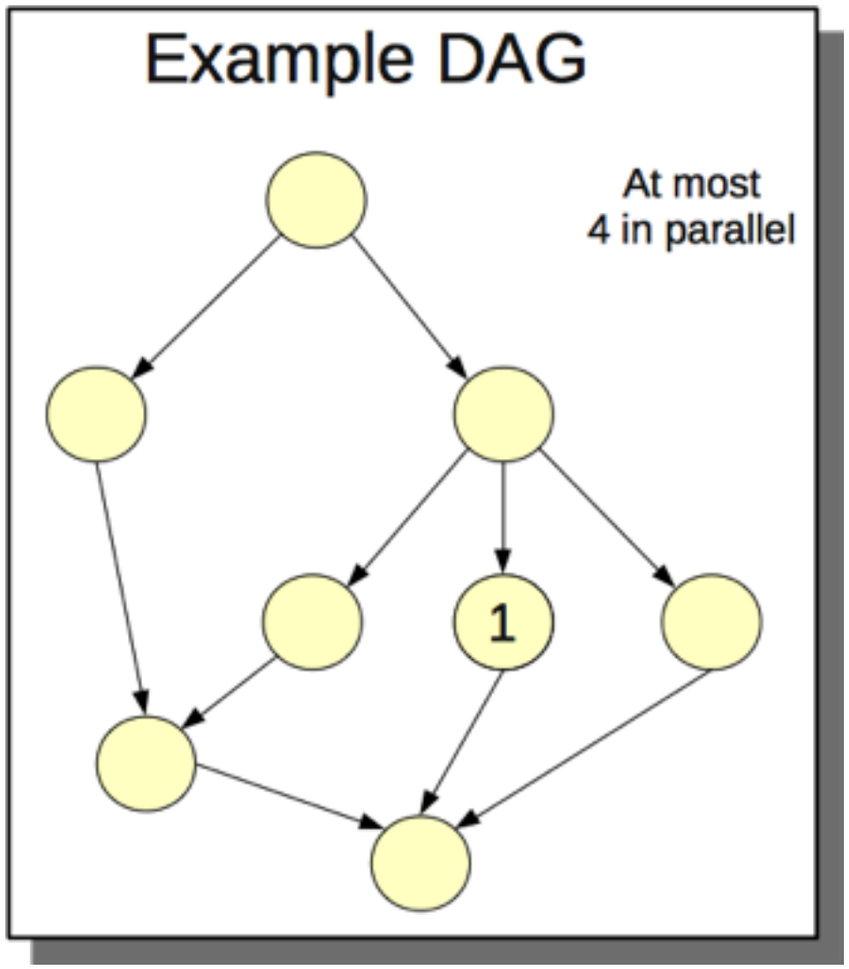
Workflows

What is workflow?

- HTC all about using many CPUs
  - Workflow = collection of jobs solving one task
- Often users have workflows that are more than “a bunch of jobs”
  - May have dependencies
- HTCondor provides a tool to handle dependencies
  - HTCondor DAGMan

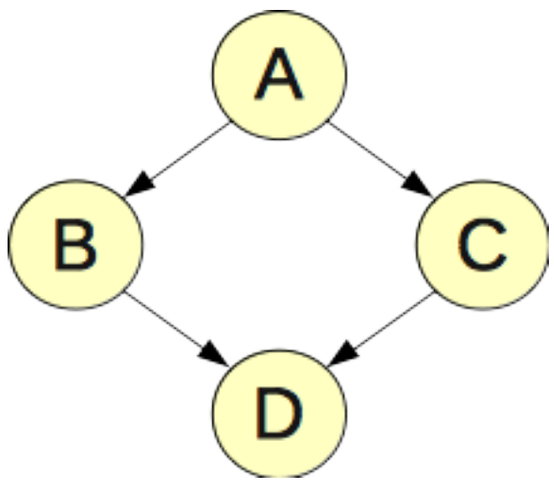
condor_dagman

- condor_dagman is a program to manage a DAG
  - DAG = Direct Acyclic Graph
- Effective way to handle a large workflow
  - HTCondor will make sure the workflow is completed
  - Or tell you which node failed
- Will have one running per submitted workflow



How to submit a workflow

- In a nutshell
  - Create submit files for nodes
  - Create a dagman submit file
  - Submit the DAG using condor_submit_dag
- HTCondor takes care of the rest




```
# Example dagman file
JOB  A  A.condor
JOB  B  B.condor
JOB  C  C.condor
JOB  D  D.condor
PARENT A CHILD B C
PARENT B C CHILD D
```

http://research.cs.wisc.edu/htcondor/manual/current/2_10DAGMan_Applications.html

The End

The HTCondor Project (Established ‘85)

- Research and Development in the Distributed High Throughput Computing field
- Team of ~35 faculty, full time staff and students
  - Face software engineering challenges in a distributed UNIX/ Linux/NT environment
  - Are involved in national and international grid collaborations
  - Actively interact with academic and commercial entities and users
  - Maintain and support large distributed production environments
  - Educate and train students

Pointers

- HTCondor Home Page
  http://research.cs.wisc.edu/htcondor/
- HTCondor Manual
  http://research.cs.wisc.edu/htcondor/manual/current/
- Support
  htcondor-users@cs.wisc.edu
  htcondor-admin@cs.wisc.edu