Open Science Grid

Intermediate Condor

Tuesday morning, 10:45am

Zach Miller <zmiller@cs.wisc.edu>
University of Wisconsin-Madison

Open Science Grid

Before we begin...

- Any questions on the lectures or exercises up to this point?

How can my jobs access their data?

Access to Data in Condor

- Use shared filesystem if available
  -
- No shared filesystem?
  -
    -
    -
    -
    -
  -
  -

Condor File Transfer

- ShouldTransferFiles = YES
  –
- ShouldTransferFiles = NO
  –
- ShouldTransferFiles = IF_NEEDED
  –

```
Universe   = vanilla
Executable = my_job
Log        = my_job.log
ShouldTransferFiles    = IF_NEEDED
Transfer_input_files   = dataset$(Process), common.data
Queue 600
```

Some of the machines in the Pool do not have enough memory or scratch disk space to run my job!

Open Science Grid

Specify Requirements

- An expression (syntax similar to C or Java)
- Must evaluate to True for a match to be made

```
Universe     = vanilla
Executable   = my_job
Log          = my_job.log
InitialDir   = run_$(Process)
Requirements = Memory >= 256 && Disk > 10000
Queue 600
```

Open Science Grid

Specify Rank

- All matches which meet the requirements can be sorted by preference with a Rank expression.
- Higher the Rank, the better the match

```
Universe    = vanilla
Executable = my_job
Log         = my_job.log
Arguments  = -arg1 -arg2
InitialDir = run_$(Process)
Requirements = Memory >= 256 && Disk > 10000
Rank = (KFLOPS*10000) + Memory
Queue 600
```

Open Science Grid

My jobs run for 20 days...

- What happens when they get pre-empted?
- How can I add fault tolerance to my jobs?

Condor's Standard Universe to the rescue!

- Condor can support various combinations of features/environments in different "Universes"
- Different Universes provide different functionality for your job:
  -
  - Standard: Support for transparent process checkpoint and restart

Open Science Grid

Process Checkpointing

- Condor’s process checkpointing mechanism saves the entire state of a process into a checkpoint file
  -
- The process can then be restarted from right where it left off
- Typically no changes to your job’s source code needed—however, your job must be relinked with Condor’s Standard Universe support library

Open Science Grid

Relinking Your Job for Standard Universe

To do this, just place “condor_compile” in front of the command you normally use to link your job:

```
% condor_compile gcc -o myjob myjob.c
```

- OR -

```
% condor_compile f77 -o myjob filea.f
fileb.f
```

Open Science Grid

Limitations of the Standard Universe

- Condor's checkpointing is not at the kernel level. Thus in the Standard Universe the job may not:
  –
  –
  –
- Many typical scientific jobs are OK
- Must be same gcc as Condor was built with

Open Science Grid

When will Condor checkpoint your job?

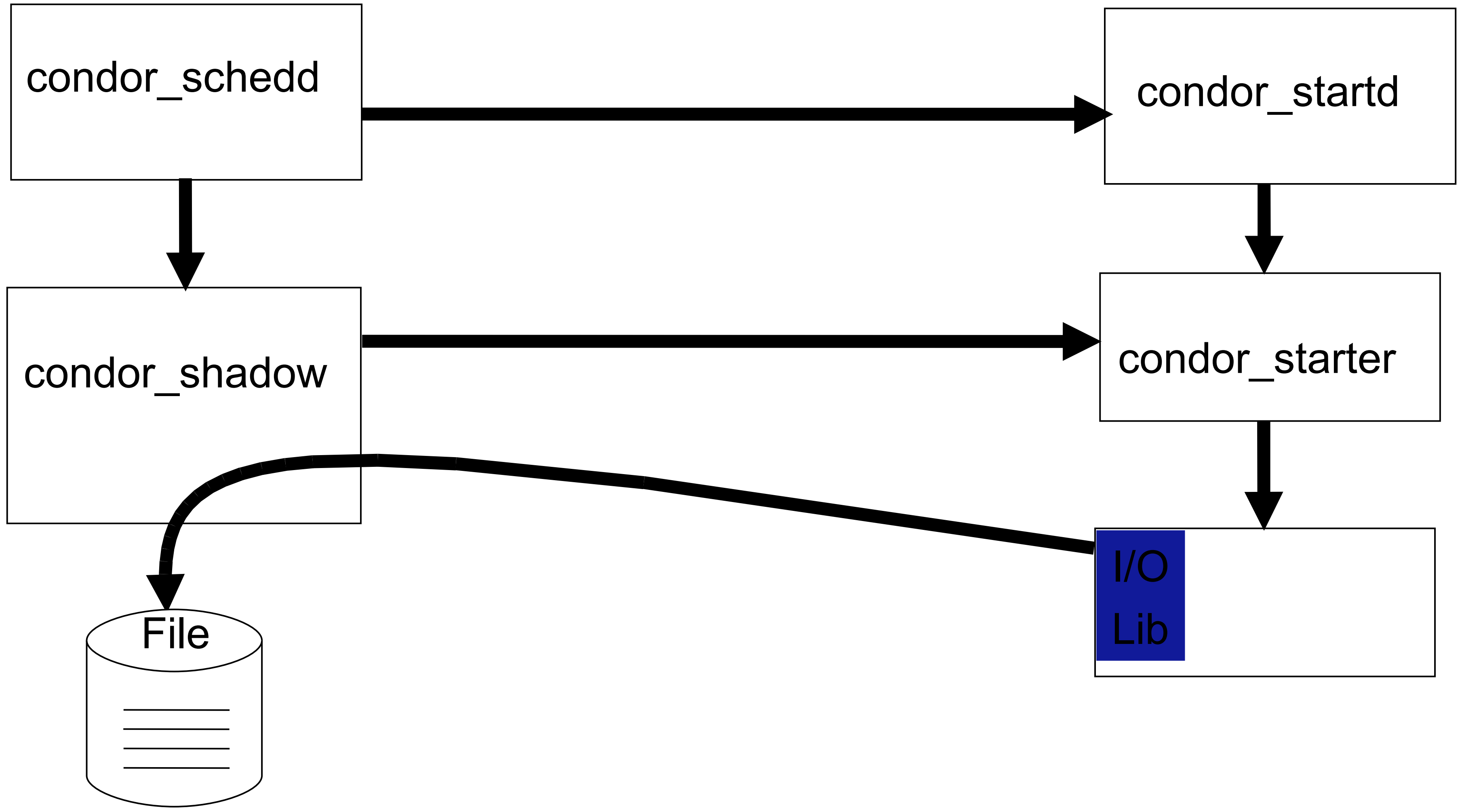
- Periodically, if desired (for fault tolerance)
- When your job is preempted by a higher priority job
- When your job is vacated because the execution machine becomes busy
- When you explicitly run:
  - –
  - –
  - –
  - –

Open Science Grid

Remote System Calls

- I/O system calls are trapped and sent back to submit machine
- Allows transparent migration across administrative domains
  –
- No source code changes required
- Language independent
- Opportunities for application steering

Remote I/O


condor_schedd
condor_startd
condor_shadow
condor_starter
I/O
Lib
File

Clusters and Processes

- If your submit file describes multiple jobs, we call this a "cluster"
- Each cluster has a unique "cluster number"
- Each job in a cluster is called a "process"
  -
- A Condor "Job ID" is the cluster number, a period, and the process number ("20.1")
- A cluster is allowed to have one or more processes.
  -

Example Submit Description File for a Cluster

```
# Example submit description file that defines a
# cluster of 2 jobs with separate working directories
Universe   = vanilla
Executable = my_job
log        = my_job.log
Arguments  = -arg1 -arg2
Input      = my_job.stdin
Output     = my_job.stdout
Error      = my_job.stderr
InitialDir = run_0
Queue
InitialDir = run_1
Queue
```

Becomes job 2.0

Becomes job 2.1

Open Science Grid

Submitting The Job

```
% condor_submit my_job.submit-file
Submitting job(s).
2 job(s) submitted to cluster 2.

% condor_q
-- Submitter: perdita.cs.wisc.edu : <128.105.165.34:1027> :
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
  2.0    frieda          4/15 06:56   0+00:00:00 I  0   0.0  my_job
  2.1    frieda          4/15 06:56   0+00:00:00 I  0   0.0  my_job
2 jobs; 2 idle, 0 running, 0 held
```

Submit Description File for a BIG Cluster of Jobs

- The initial directory for each job can be specified as run_$(Process), and instead of submitting a single job, we use “Queue 600” to submit 600 jobs at once
- The $(Process) macro will be expanded to the process number for each job in the cluster (0 - 599), so we’ll have “run_0”, “run_1”, … “run_599” directories
- All the input/output files will be in different directories!

Submit Description File for a *BIG* Cluster of Jobs

Open Science Grid

```
# Example condor_submit input file that defines
# a cluster of 600 jobs with different directories
Universe   = vanilla
Executable = my_job
Log        = my_job.log
Arguments  = -arg1 -arg2
Input      = my_job.stdin
Output     = my_job.stdout
Error      = my_job.stderr
InitialDir = run_$(Process)
Queue 600
```

- run_0 … run_599
- Creates job 3.0 … 3.599

Open Science Grid

More $(Process)

- You can use $(Process) anywhere.

```
Universe   = vanilla
Executable = my_job
Log        = my_job.$(Process).log
Arguments  = -randomseed $(Process)
Input      = my_job.stdin
Output     = my_job.stdout
Error      = my_job.stderr
InitialDir = run_$(Process)
Queue 600
```

·run_0 … run_599

·Creates job 3.0 … 3.599

Sharing a directory

- You don’t have to use separate directories.
- $(Cluster) will help distinguish runs

```
Universe   = vanilla
Executable = my_job
Arguments  = -randomseed $(Process)
Input      = my_job.input.$(Process)
Output     = my_job.stdout.$(Cluster).$(Process)
Error      = my_job.stderr.$(Cluster).$(Process)
Log        = my_job.$(Cluster).$(Process).log
Queue 600
```

Open Science Grid

Job Priorities

- Are some of the jobs in your sweep more interesting than others?
- `condor_prio` lets you set the job priority
  -
  -
- Can be set in submit file:
  -

Open Science Grid

What if you have LOTS of jobs?

- Set system limits to be high:
  - –
  - –
  - –
- Each condor_schedd limits max number of jobs running
  - –
  - –
- Consider multiple submit hosts
  - –
  - –
- We constantly strive to improve scalability

Open Science Grid

Advanced Trickery

- You submit 10 parameter sweeps
- You have five classes of parameters sweeps
  -
- How can you look at the status of jobs that are part of Type B parameter sweeps?

Advanced Trickery cont.

- In your job file:

+SweepType = “B”

- You can see this in your job ClassAd

```
condor_q -l
```

- You can show jobs of a certain type:

```
condor_q -constraint ‘SweepType == “B”’
```

- Very useful when you have a complex variety of jobs
- Try this during the exercises!
- Be careful with the quoting…

Open Science Grid

Time for more exercises!

Open Science Grid

Questions?

- Questions? Comments?
- Feel free to ask me questions later:

Zach Miller <zmiller@cs.wisc.edu>

- Upcoming sessions
  -
    -
  -
    -
  -
    -