

Day 10: More Condor

Suggested reading: Condor 7.7 Manual:

<http://www.cs.wisc.edu/condor/manual/v7.7/>

Chapter 2: Users' Manual (at most, 2.1–2.7)

Chapter 9:

`condor_q`, `condor_status`, `condor_submit`, `condor_prio`

Turn In Homework

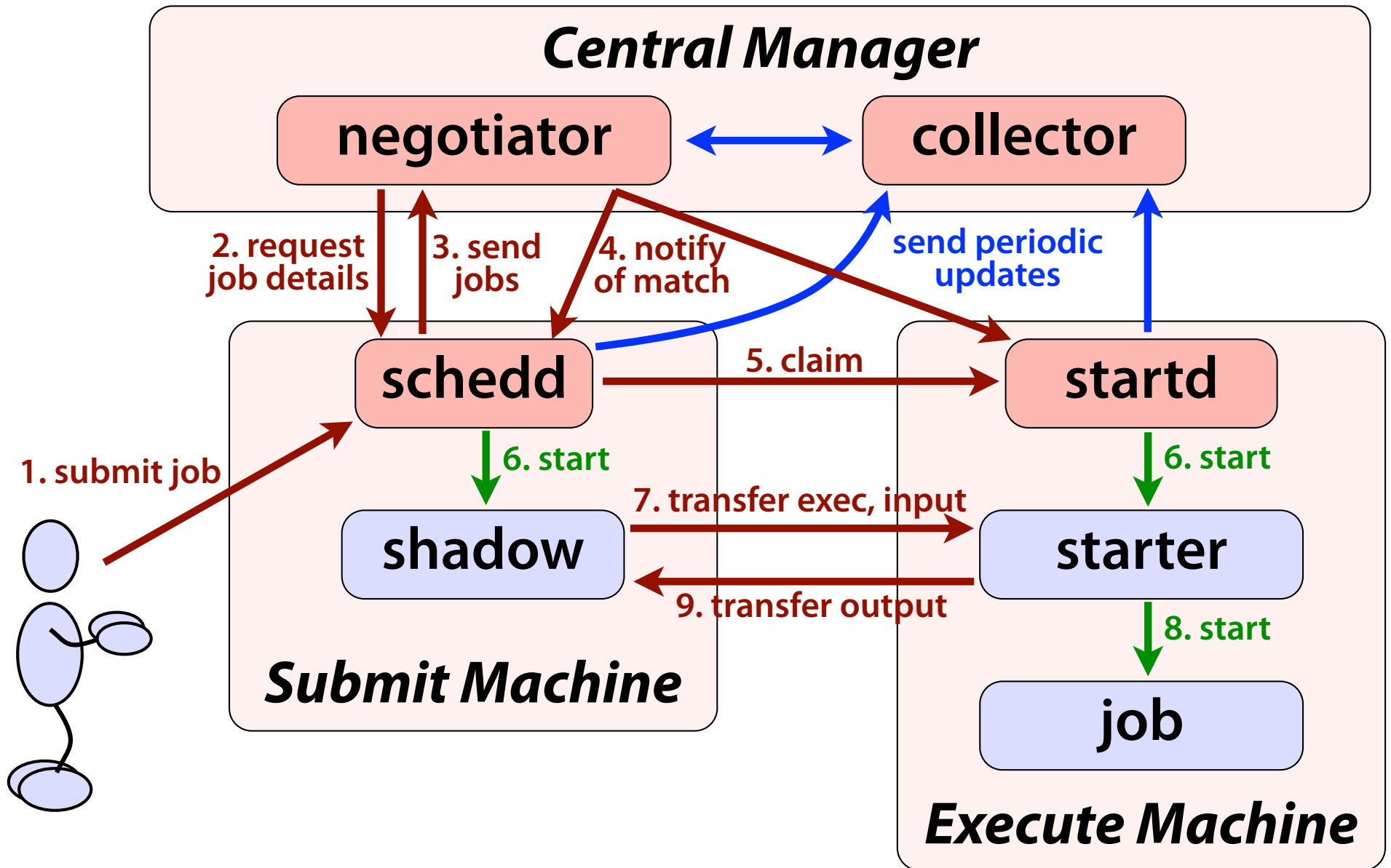
Homework Review

More Condor Background

How Does Condor Work?

| Function | Condor Name | # |
|-------------------------------|----------------------|-----------------|
| Track waiting/running jobs | schedd ("sked-dee") | 1+ |
| Track available machines | collector | 1 |
| Match jobs and machines | negotiator | 1 |
| Manage one machine | startd ("start-dee") | per machine |
| Manage one job (on submitter) | shadow | per job running |
| Manage one job (on machine) | starter | per job running |

The Life of a Job



Matchmaking Revisited

- Balances
 - Job (submitter)
 - Machine (owner)
 - Pool (administrator)
- Takes into account
 - Requirements
 - Preferences
 - Policy
- But how?



ClassAds

- For job, machine, etc.
- Loosely structured
- Few required parts
- Users can extend
- Can express:
 - Facts
 - Current state
 - Requirements
 - Preferences
 - Your shoe size
- **attribute** = expression

```

MyType = "Job"
TargetType = "Machine"
ClusterId = 14
Owner = "cat"
Cmd = ".../homework_09.py"
Requirements =
    (Arch == "X86_64") &&
    (OpSys == "LINUX") &&
    ...
Rank = 0.0
In = "/dev/null"
UserLog = ".../hw09.log"
Out = "hw09.out"
Err = "hw09.err"
NiceUser = false
  
```

string

number

operations/expressions

boolean

Priorities

- **Job priority**

- Set by user (owner)
- Is relative to *that user's* other jobs
- Higher number means run sooner

- **User priority**

- Condor calculates this priority value based on past usage
- Determines user's potential share of machines
- Lower number means run sooner (0.5 is minimum)
- Results in "fair share" access to resources

- **Preemption**

- Low priority jobs can be removed for high priority ones
- Governed by fair-share algorithm and pool policy

What Makes a Good CHTC Job?

- Single-threaded, independent batch job
- Runs for about 10 minutes to 4 hours
 - Too short: Overhead costs predominate
 - Too long: Risk getting preempted (“bad-put”)
 - CHTC removes any job after 24 hours of runtime
- Fits lots of machines — the more, the better!
 - Few requirements: low memory, low disk
 - Scripts! (few/no OS and architecture requirements)

Condor Commands

condor_q: Being More Selective

```
condor_q username [...]
```

- Lists jobs *only* owned by the user(s) (e.g., yourself)

```
condor_q cluster [...]
```

- Lists all jobs in the given cluster(s)

```
condor_q cluster.process [...]
```

- Lists only the given job(s)

```
-- Submitter: submit-368.chtc.wisc.edu : <...> : ...  
ID      OWNER      SUBMITTED      RUN_TIME  ST  PRI  SIZE  CMD  
23.2    cat         11/13 15:21     0+00:00:00  I   0    0.0  explore.py
```

condor_q: ClassAd Output

```
condor_q -long cluster.process
```

- Displays complete ClassAd for each job (80+ lines)
- Great way to explore ClassAds for jobs
- Best to limit to a single job (cluster/process combo)!

```
-- Submitter: submit-368.chtc.wisc.edu : <...> : ...  
PeriodicRemove = false  
CommittedSlotTime = 0  
Out = "explore.out.24.1"  
ImageSize_RAW = 1  
NumCkpts_RAW = 0  
EnteredCurrentStatus = 1321219554  
CommittedSuspensionTime = 0  
WhenToTransferOutput = "ON_EXIT"  
NumSystemHolds = 0  
StreamOut = false  
...
```

condor_q: Why Isn't My Job Running?

```
condor_q -analyze cluster.process
```

- Tries to figure out if your job *can* run
- Often helpful – occasionally not – good starting pt.

```
026.000: Run analysis summary. Of 2072 machines,
      2072 are rejected by your job's requirements
      0 reject your job because of their own requirements
```

```
...
```

```
No successful match recorded.
Last failed match: Sun Nov 13 15:33:29 2011
Reason for last match failure: no match found
```

```
WARNING: Be advised:
No resources matched request's constraints
```

```
The Requirements expression for your job is:
```

```
...
```

| | Condition | Machines Matched | Suggestion |
|---|------------------------------|------------------|------------------|
| | ----- | ----- | ----- |
| 1 | (target.Memory >= 9999999) | 0 | MODIFY TO 212001 |
| 2 | (TARGET.Arch == "X86_64") | 2020 | |
| 3 | (TARGET.OpSys == "LINUX") | 2020 | |

condor_status: Classes of Machines

condor_status -avail

- Lists slots that are available

condor_status -constraint *ClassAdExpr*

- Lists slots that match constraint(s)

```
% condor_status -constraint 'Memory >= 10000'
```

| Name | OpSys | Arch | State | Activity | LoadAv | Mem | ActvtyTime |
|--------------------|--------------|--------|-----------|-----------|---------|------------|------------|
| slot10@c011.chtc.w | LINUX | X86_64 | Claimed | Busy | 6.690 | 12017 | 0+14:41:56 |
| slot10@c013.chtc.w | LINUX | X86_64 | Claimed | Busy | 7.980 | 12017 | 0+14:50:57 |
| ... | | | | | | | |
| slot25@opt-a012.ch | LINUX | X86_64 | Unclaimed | Idle | 0.000 | 99111 | 0+21:01:43 |
| | Total | Owner | Claimed | Unclaimed | Matched | Preempting | Backfill |
| | X86_64/LINUX | 66 | 2 | 55 | 9 | 0 | 0 |
| | Total | 66 | 2 | 55 | 9 | 0 | 0 |

condor_status: Being More Selective

```
condor_status hostname [...]
```

- Lists slots with the given hostname(s)

```
condor_status slot@hostname [...]
```

- Lists the given slot(s)

```
% condor_status c040.chtc.wisc.edu
```

| Name | OpSys | Arch | State | Activity | LoadAv | Mem | ActvtyTime |
|--------------------|--------------|--------|---------|-----------|---------|------------|-------------|
| slot10@c040.chtc.w | LINUX | X86_64 | Claimed | Busy | 7.990 | 12017 | 0+19:36:09 |
| slot1@c040.chtc.wi | LINUX | X86_64 | Owner | Idle | 0.000 | 4599 | 0+19:36:03 |
| ... | | | | | | | |
| slot9@c040.chtc.wi | LINUX | X86_64 | Owner | Idle | 0.020 | 250 | 47+05:24:44 |
| | Total | Owner | Claimed | Unclaimed | Matched | Preempting | Backfill |
| | X86_64/LINUX | 10 | 9 | 1 | 0 | 0 | 0 |
| | Total | 10 | 9 | 1 | 0 | 0 | 0 |

condor_status: ClassAd Output

```
condor_status -long slot@hostname
```

- Displays complete ClassAd for each *slot* (120+ lines)
- Great way to understand ClassAds for *machines*
- Best to limit to a single slot!

```
Machine = "opt-a001.chtc.wisc.edu"  
DCSignalRuntime = 247.566893  
EnteredCurrentState = 1321222293  
JavaVersion = "1.6.0_20"  
DetectedMemory = 258331  
OpSysAndVer = "LINUX"  
HasMPI = true  
CpuIsBusy = false  
LastBenchmark = 1321228954  
HasVM = false  
JavaVendor = "Sun Microsystems Inc."  
...
```

condor_prio

```
condor_prio -p value cluster[.process] [...]
```

- Sets the job priority to the given value
- Identify job(s) with 1+ user(s), cluster(s), process(es)

```
condor_prio +value cluster[.process] [...]
```

```
condor_prio -value cluster[.process] [...]
```

- Raise or lower the job priority by the given amount

Submit Files

Setting Priority (Again)

`priority = integer`

- Sets job priority right in submit file
- Default is 0
- Only affects relative priority of your jobs
- Can override using **condor_prio**

Notifications by Email

```
notification = Always | Complete | Error | Never
```

- When to send email
 - **Always**: job checkpoints or completes
 - **Complete**: job completes (*default*)
 - **Error**: job completes with error
 - **Never**: do not send email

```
notify_user = email
```

- Where to send email
- Defaults to *job-owner@submit-machine*

Input Files From the Internet

```
transfer_input_files = URL[, ...]
```

- Grab input files from any available URL
- **BUT:** If the download fails, your job goes on hold
 - You don't know when your job will run
 - Maybe that will be during server maintenance, etc.
- So, great idea, but maybe wait for retries...
 - Can always pre-fetch file yourself
 - Or, job itself can download files, and do it robustly

Arbitrary Attributes

```
+AttributeName = value
```

- Adds arbitrary attribute(s) to job ClassAd
- Useful in (at least) two cases:
 - Find jobs using attribute: **condor_q -constraint**
 - Attribute has special policy meaning in pool
- As it happens, we have a special policy...

```
+WantRHEL6Job = true
```

```
rank = (IsRHEL6 == True)
```

Requirements

```
requirements = ClassAdExpression
```

- Expression must evaluate to true to run on machine
- Condor adds defaults! View with `condor_q -long`
- See Condor Manual (esp. 2.5.2 & 4.1) for details

| | |
|-----------------|----------------------|
| OpSys | operating system |
| Arch | architecture |
| Memory | memory, in MB |
| HasJava | True/False |
| IsRHEL6 | True/False |
| ShoeSize | (if defined in pool) |

```
requirements =  
  (OpSys == "LINUX") &&  
  ((Arch == "X86_64") ||  
   (Arch == "INTEL")) &&  
  (Memory >= 64) &&  
  (IsRHEL6 == true)
```


Preferences (Rank)

```
rank = ClassAdExpression
```

- Ranks *matching* machines in order by preference
- Must evaluate to a FP number, greater is preferred
 - False becomes 0.0, True becomes 1.0
 - Undefined or error values become 0.0
- Writing rank expressions is an art form

```
rank = Memory
```

```
rank = (IsRHEL6 == True)
```

```
rank = ((substr(Machine, 0, 1) == 'c') * 2) +  
        (substr(Machine, 0, 1) == 'e'))
```

One Submit, Many Jobs: I

- Can use **queue** statement many times
- Make changes between **queue** statements
 - Change **arguments, output, priority, ...**
 - Whatever you do not explicitly change stays the same

```
executable = test.py
...
log        = test.log

output     = test-1.out
arguments  = "test-input.txt 42"
queue

output     = test-2.out
arguments  = "test-input.txt 43"
queue
```

One Submit, Many Jobs: II

queue N

- Submits N copies of the job
 - One cluster number for all copies, just as before
 - Process numbers go from 0 – ($N-1$)
- What good is having N copies of the same thing?
 - Randomized processes (cf. homework #8)
 - Job fetches work description from somewhere?
 - But what about overwriting output files, etc.?
- Wouldn't it be nice to have different files and/or arguments automatically applied to each job?

Separating Files by Run

```
output = program.out.$(Cluster).$(Process)
```

- Can use either/both of these variables anywhere
 - Often used in **output**, **error**, and **log** files
- Maybe use **\$(Process)** in **arguments**?
 - No math on values; your program must handle as is

```
...  
output      = test.$(Cluster)_$(Process).out  
log         = test.$(Cluster)_$(Process).log  
  
arguments   = "test-input.txt $(Process)"  
queue 10
```

Separating Directories by Run

```
initialdir = path
```

- Use *path* (instead of submit dir.) to locate files
 - I.e., **output**, **error**, **log**, **transfer_input_files**
 - *Not executable*; always relative to submit directory
- Mix with **\$(Process)** and separate all I/O by job

```
initialdir = run-$(Process)
transfer_input_files = input-$(Process).txt
output = test.$(Cluster)-$(Process).out
log     = test.$(Cluster)-$(Process).log

arguments = "input-$(Process).txt $(Process)"
queue 10
```

Homework

Homework

- Write a little bit of Python code, lest you forget!
- Run lots of jobs from a single submit file
- Experiment with `condor_q` and `condor_status`