CHTC Policy and Configuration

Greg Thain

HTCondor Week 2017

Image credit: flickr user shanelin cc

Image credit: wikipedia

CHTC Pool Mission

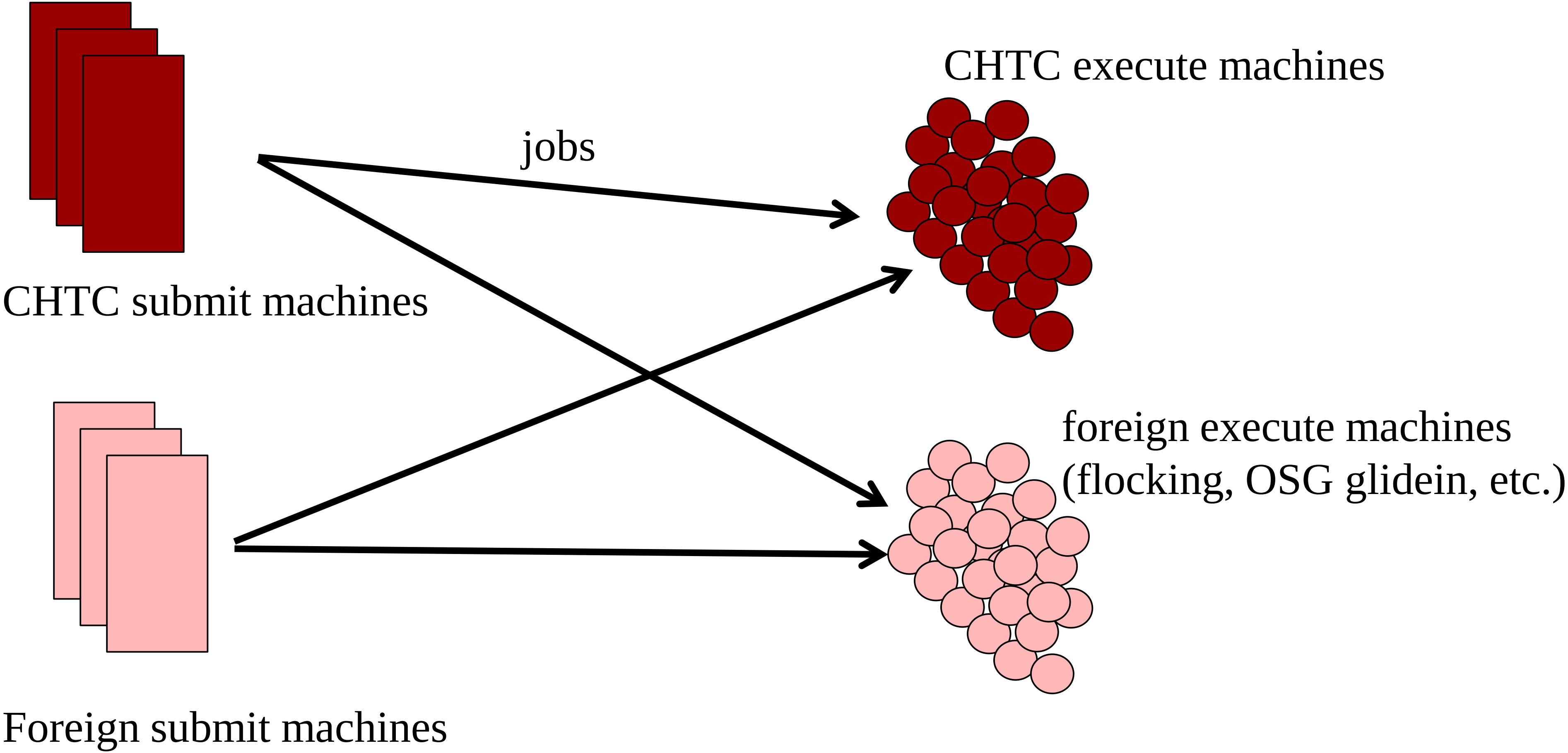
CHTC Pool Mission

“To improve computational research on campus by providing access to high throughput computing to a large number of users”

One consequence: Max Runtime

- Why?
  - Minimize badput
  - Prevent one user from monopolizing pool
  - Encourage use of shared resources like OSG
- Currently choose 72 hour max runtime

CHTC Pool



CHTC execute machines

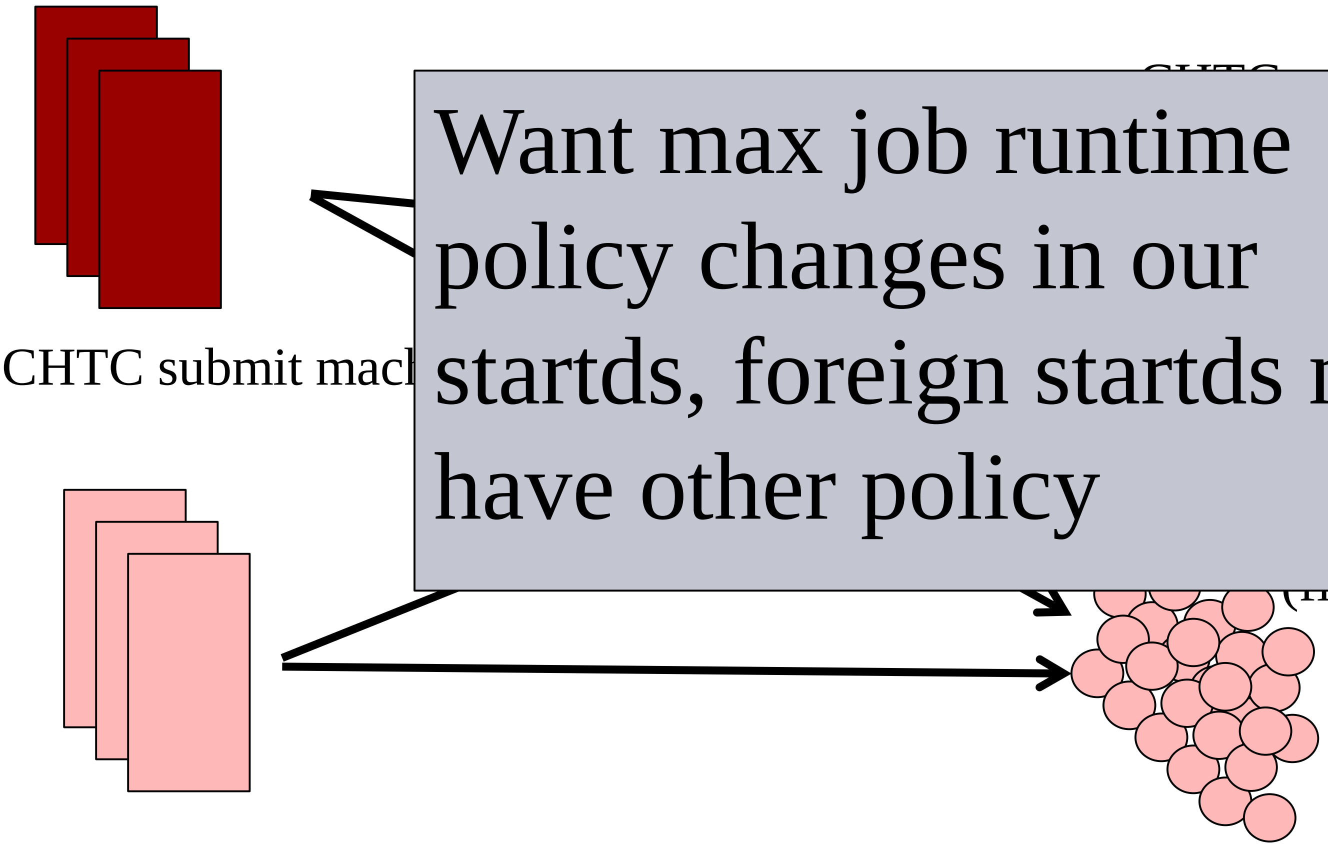
jobs

CHTC submit machines

foreign execute machines
(flocking, OSG glidein, etc.)

Foreign submit machines

CHTC Pool



CHTC submit mach

Foreign submit machines

CHTC ... nachines

...ecute machines
(flocking, OSG glidein, etc.)

Want max job runtime policy changes in our startds, foreign startds may have other policy

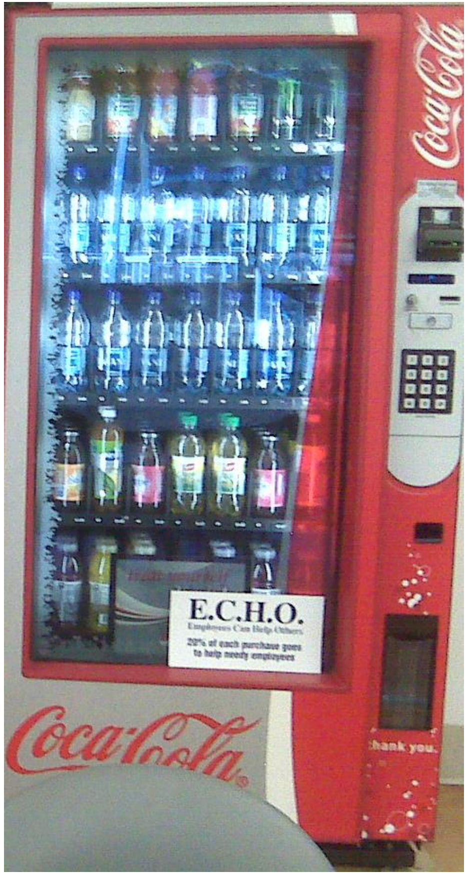
Policy Question

- Should condor preempt job at 72 hours…

  Even if no other job is runnable ???

PLAY 3 CREDITS
JACKPOTS OF 400 CREDITS OR MORE PAID BY ATTENDANT.
PAYLINE
BAR
BAR
BAR
BAR
$1
0
1
CASH CREDIT
BET ONE
SPIN
BET MAX

Coca-Cola
E.C.H.O.
Employees Can Help Others
20% of each purchase goes to help needy employees
Coca-Cola
thank you.



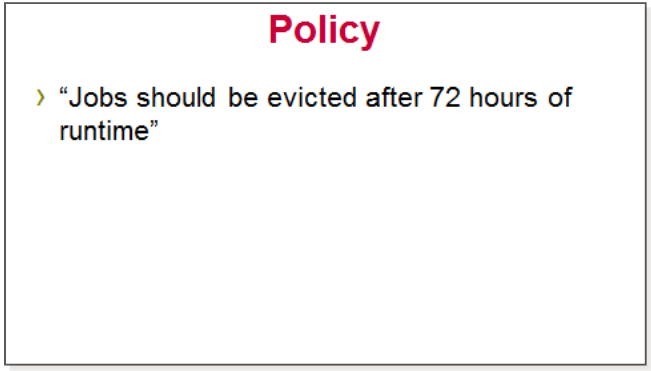
Coca-Cola
Coca-Cola
E.C.H.O.
Employees Can Help Others
20% of each purchase goes
to help needy employees
thank you.

Policy

- “Jobs should be evicted after 72 hours of runtime”

```
WANT_HOLD = \
    TotalJobRunTime >  ( 72 * 3600)
WANT_HOLD_REASON = \
    "Job failed to run in 72 hrs"
```

Not quite right…



Policy

- “Jobs should be evicted after 72 hours of runtime”

Only care about execution attempt:

What about self checkpointing job?

How to identify checkpointable jobs?

- `+Is_Resumable = true`
- Should this be 1st class in condor?

```
WANT_HOLD = Is_Resumable =?= true &&
    TotalJobRunTime >  ( 72 * 3600)

WANT_HOLD_REASON = \
    "Job failed to run in 72 hrs"
```

Is_Resumable has more uses

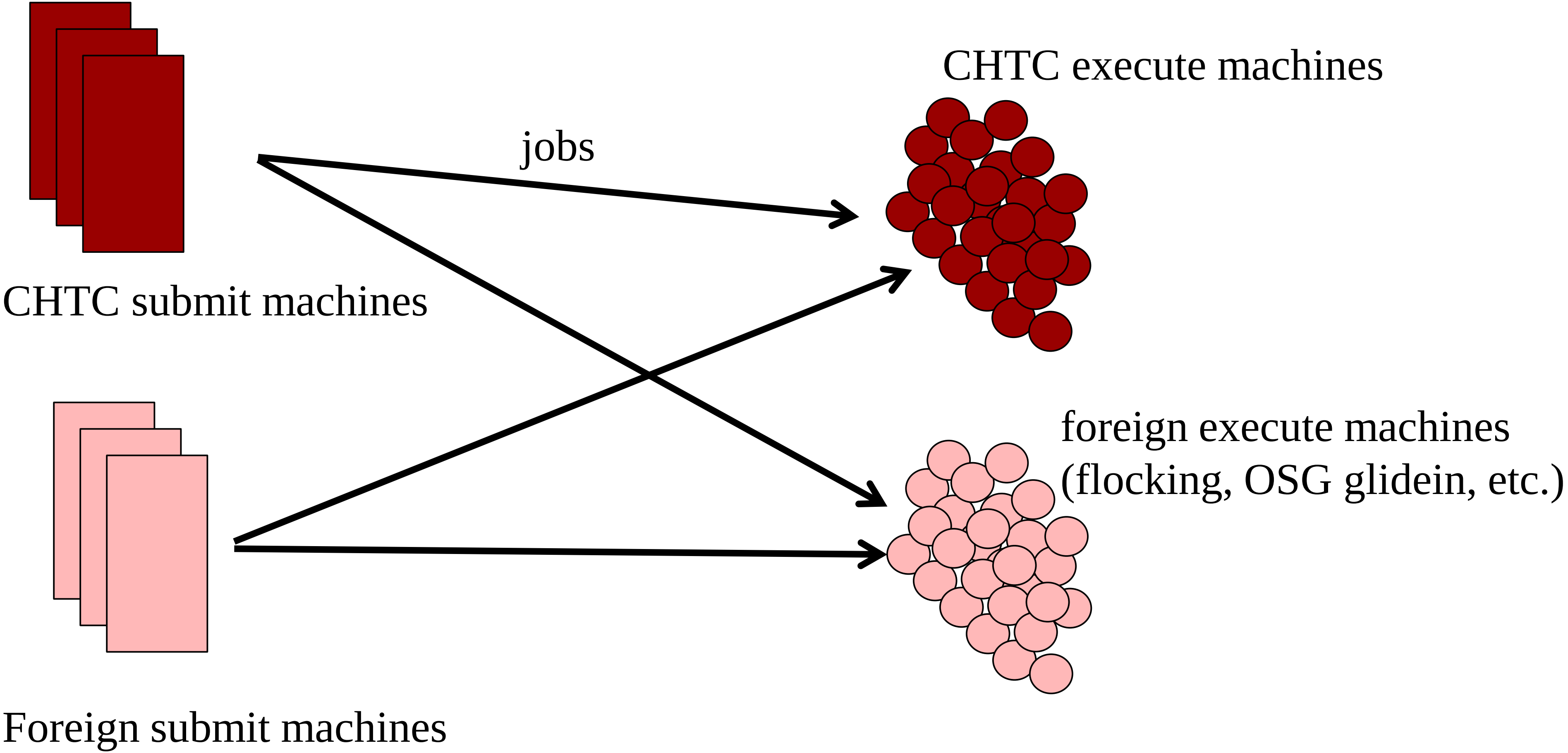
- More we know about a job, better we do
  - Even a little bit a knowledge – not full runtime guess
  - Used for backfilling clusters
  - Running in other specialized environments.

Moral of this story

- Be very careful with policy statements
- We need users to tell us more about jobs

- “Specific is Terrific”!

CHTC Pool



CHTC submit machines

jobs

CHTC execute machines

foreign execute machines
(flocking, OSG glidein, etc.)

Foreign submit machines

Consequence

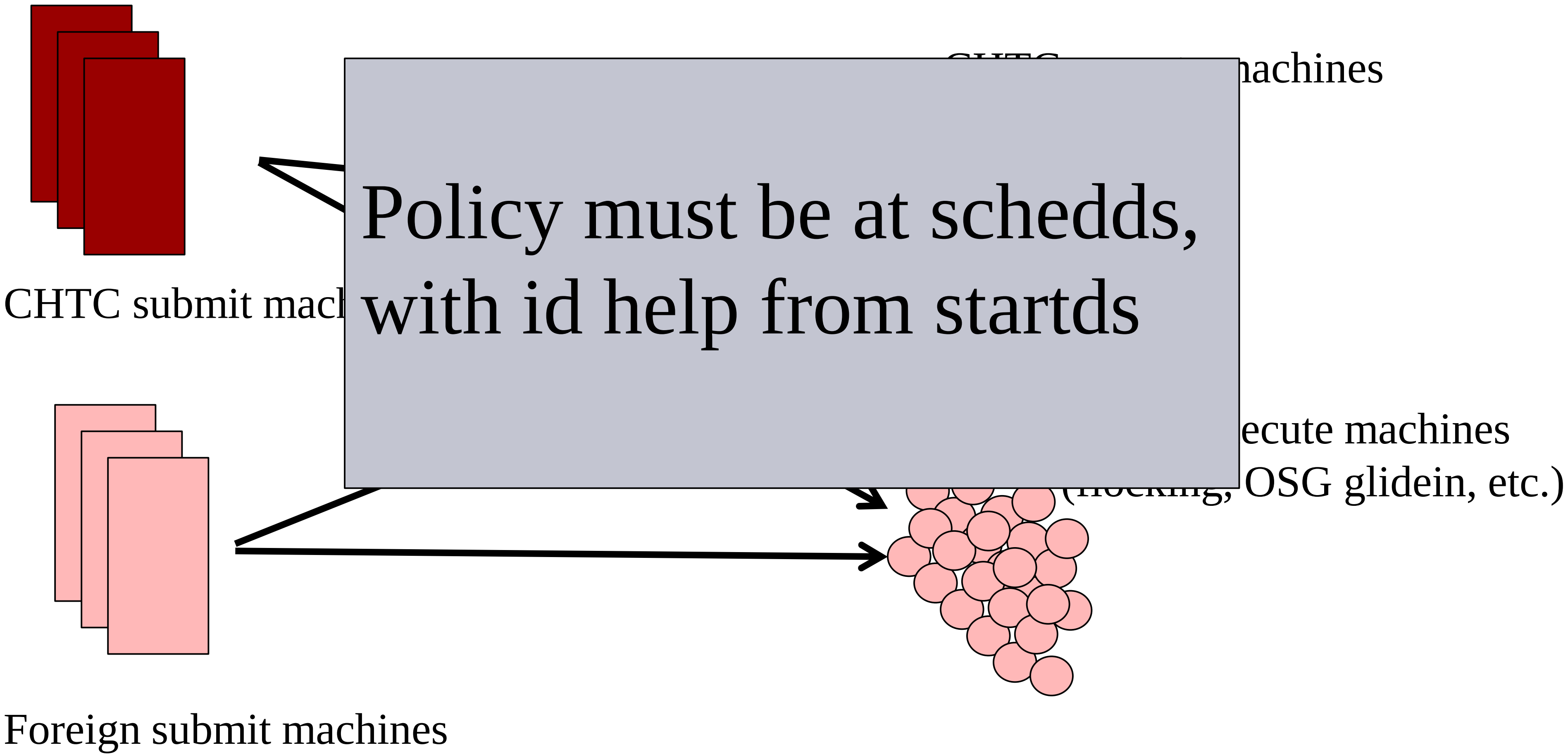
- Want to have many places to run

- But we don’t control foreign machines:
  - Operating System, policies, preinstalled software
  - Leads to user surprises

Policy

- Jobs only run “at home” by default, can opt into foreign resources
  - Two levels:
    - On campus foreign pools
    - OSG Pool (really foreign)

CHTC Pool



CHTC submit mach

Foreign submit machines

CHTC

achines

ecute machines

(flocking, OSG glidein, etc.)

Policy must be at schedds, with id help from startds

STARTD_ATTRS

```
PoolName = “CHTC”
Site = “ServerRoomNumber”
STARTD_ATTRS = PoolName, Site
```

condor_status output

```
condor_status -af Site PoolName
```

```
…
Site = “ServerRoomNumber”
Pool = “CHTC”
…
```

Digression: good condor style

- Jobs choose based on attribute of machine
- NOT on a-priori knowledge
  - Machine name, etc.
- Machines publish info about themselves

Example: CVMFS

- Our local machines have CVMFS, others don’t. If a job needs CVMS, how should it indicate this?

Wrong way

```
Requirements = PoolName == “CHTC”
```

Right way – startd cron

```
STARTD_CRON_JOBLIST    = $(STARTD_CRON_JOBLIST) CVMFS
STARTD_CRON_CVMFS_EXECUTABLE  = /path/to/check_cvmfs
STARTD_CRON_CVMFS_PERIOD                 = 10m
STARTD_CRON_CVMFS_MODE                   = periodic
```

Check_cvmfs is a script which emits
HasCVMFS = true (or not)

condor_status output

```
condor_status -af HasCVMFS
```

```
…
HasCVMFS = true
…
```

End of digression, back to flocking

- Want jobs to only travel if they opt in
- By providing default schedd requirements

APPEND_REQUIREMENTS

```
APPEND_REQ_VANILLA =
       MY.WantFlocking || TARGET.PoolName =?= "CHTC"
```

```
REQUIREMENTS = (USER STUFF) &&
       MY.WantFlocking || TARGET.PoolName =?= "CHTC"
```

My.WantFlocking comes from job ad

Trivia: What if not defined?

Policy: defragmenting partitionable slots

- With partitionable slots, and mixed size jobs, can have job starvation
- Problem: can take long time to fully defrag machine

Defrag policy

```
DEFRAG_REQUIREMENTS = PartitionableSlot \
       && State =!= "Owner" && TotalCpus > 39
DEFRAG_WHOLE_MACHINE_EXPR = PartitionableSlot && \TotalCpus
> 39 && ( State =!= "Owner" ) && \
       ( ( cpus / TotalCpus ) >= 0.8)
```

Only defrag down to 80%

Problem: high prio job goes first

- May wait a long time to defrag a machine
- But then a high prio serial job comes along…
- Negotiation hard coded to match in prio order

Solution: delayed start

```
START = $(START) && \
      RequestCpus >= IfThenElse(Cpus < 4,1,4) || \
      (time() - EnteredCurrentState) > 20 * 60)
```

After defrag, only start big jobs for a few minutes

Summary

Condor has all kinds of assembly language for policy, but we need to spend more time thinking about what the actual, english-language policy we want is.