Condor Support for Multi-core jobs

Condor Project
Computer Sciences Department
University of Wisconsin-Madison

HTPC

Allow jobs to consume an whole machine

- Today 4/8 cores, tomorrow 12 cores, …
- Package jobs with a parallel library
  - HTPC jobs as portable as any other job
  - MPI, OpenMP, pthreads, your own scripts, …
  - Parallel libraries can be optimized for on-board memory access
- All memory is available
- Serial jobs still available

Configuring Condor for HTPC

- Mix & Match HTPC with serial
- 1 extra condor slot configured HTPC
  - Rest serial
- Two strategies:
  - Suspend/drain serial slots for HTPC
  - Hold empty cores until HTPC slot is open

Configuring Condor

```
# require that whole-machine jobs only match to Slot1
START = ($(START)) && (TARGET.RequiresWholeMachine =!= TRUE || SlotID == 1)
# have the machine advertise when it is running a whole-machine job
STARTD_JOB_EXPRS = $(STARTD_JOB_EXPRS) RequiresWholeMachine

# Export the job expr to all other slots
STARTD_SLOT_EXPRS = RequiresWholeMachine

# require that no single-cpu jobs may start when a whole-machine job is running
START = ($(START)) && (SlotID == 1 || Slot1_RequiresWholeMachine =!= True)

# suspend existing single-cpu jobs when there is a whole-machine job
SUSPEND = ($(SUSPEND)) || (SlotID != 1 && Slot1_RequiresWholeMachine =?= True)
CONTINUE = ( $(SUSPEND) =!= True )
```

Get all that?

http://condor-wiki.cs.wisc.edu

(Look for Admin How-To recipes)

How to submit

```
universe = vanilla
requirements = (CAN_RUN_WHOLE_MACHINE =?= TRUE)
+RequiresWholeMachine=true
executable = some job
arguments = arguments
should_transfer_files = yes
when_to_transfer_output = on_exit
transfer_input_files = inputs
queue
```

MPI on Whole machine jobs

Whole machine mpi submit file

```
universe = vanilla
requirements = (CAN_RUN_WHOLE_MACHINE =?= TRUE)
+RequiresWholeMachine=true
executable = mpiexec
arguments = -np 8 real_exe
should_transfer_files = yes
when_to_transfer_output = on_exit
transfer_input_files = real_exe
queue
```

Downside: Funny Looking status/monitoring

```
[gthain@submit ~]$ condor_status c015
Name                OpSys      Arch   State     Activity LoadAv Mem   ActvtyTime

slot10@c015.chtc.w LINUX      X86_64 Claimed   Busy     7.980  12017  0+10:21:19
slot1@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  4599  0+18:14:40
slot2@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+10:21:29
slot3@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+10:21:35
slot4@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+10:20:57
slot5@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+21:01:42
slot6@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+21:01:43
slot7@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+21:01:44
slot8@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.000  1024  0+21:01:37
slot9@c015.chtc.wi LINUX      X86_64 Owner     Idle     0.020   250  2+16:20:26
                     Total Owner Claimed Unclaimed Matched Preempting Backfill

        X86_64/LINUX    10     9       1         0       0          0        0

               Total    10     9       1         0       0          0        0
```

Another downside: FSS

- Condor accounting/fair share
  - Before 7.4: doesn't know about full slots
  - Still doesn't expose enough info for Gratia to do proper accounting
- Current, UW statically partitions pool
- (We're working on it)

How to submit to OSG

```
universe = grid
GridResource = some_grid_host
GlobusRSL = MagicRSL
executable = wrapper.sh
arguments = arguments
should_transfer_files = yes
when_to_transfer_output = on_exit
transfer_input_files = inputs
transfer_output_files = output
queue
```

What's the magic RSL?

Site Specific

We're working on documents/standards

PBS

(host_xcount=1)(xcount=8)(queue=?)

LSF

(queue=?)(exclusive=1)

Condor

(condorsubmit=('+WholeMachine' true))

What's with the wrapper?

Chmod executable

Create output files

```
#!/bin/sh
chmod 0755 real.ex
touch output
./mpiexec -np 8 real.ex
```

Integration of local w/OSG

- Today condor_job_router

- Tomorrow:
  - GlideInWMS // CorralWMS

GLUE schema for HTPC

Proposed HTPC Schema

The following attributes will be added to the Glue schema as a CECapability:

Attribute Name	Attribute Type	Description
GlueCECapability	string	htpc
HTPCAccessControlBaseRule	string	ACBR format to specify one or more of VO: or VOMS:
HTPCrsl	string	extra rsl needed to enable HTPC jobs

Another useful variable for HTPC is the "number of cores per machine" but this can be calculated from:

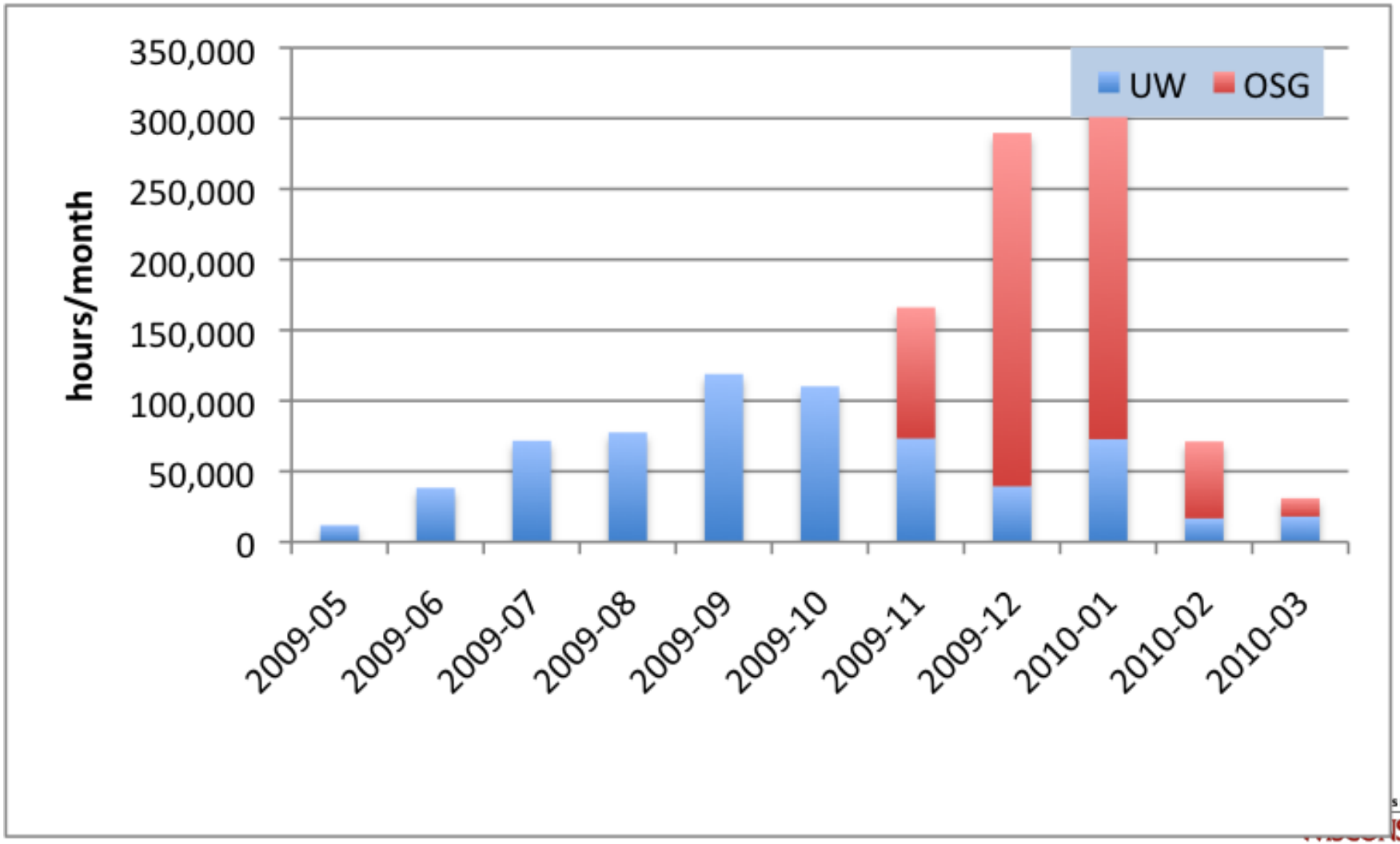
- number of cores per machine = GlueHostArchitectureSMPSize * (LogicalCPUs / PhysicalCPUs).

Who's using HTPC?

- Chemists
  - UW Chemistry group
  - Gromacs
  - Jobs take 24 hours on 8 cores
  - Steady stream of 20-40 jobs/day
  - Peak usage is 320,000 hours per month
    - Written 9 papers in 10 months based on this

This could be you!

Chemistry Usage of HTPC



Current OSG operations

- OU, > 100 machines
  - Logged over 2M HTPC hours so far
- Purdue, >100 machines
- Clemson, > 500 machines
- San Diego, CMS T2, 1 machine
- Wisconsin, 10 machines
- (Your site here…)

Questions

- Why Whole Machine (cores vs RAM?)
- What about HyperThreads?
- Need partial machine?
- What to do about 12 cores?
- How to know node #s (not cpuinfo)
- Reporting/Gratia
- What else do you need from Condor

Thank you

- Resources:
  - OSG HTPC twiki page
  - Condor admin how to recipes