Production Condor in the Cloud

Ian D. Alderman

CycleComputing

http://twitter.com/cyclecomputing

CYCLECOMPUTING

CYCLECLOUD
THE EASE & POWER OF CLOUD HPC

Case Studies

- This is a presentation of some work we've recently completed with customers.
- Focus on two particular customer workflows:
  - Part 1: 10,000 core cluster "Tanuki."
  - Part 2: Easily running the same workflow "locally" and in the cloud including data synchronization.
- We will discuss the technical challenges.

Case 1: 10,000 Cores "Tanuki"

- Run time = 8 hours
- 1.14 compute-years of computing executed every hour
- Cluster Time = 80,000 hours = 9.1 compute years.
- Total run time cost = ~$8,500

- 1250 c1.xlarge ec2 instances (8 cores / 7-GB RAM)
- 10,000 cores, 8.75 TB RAM, 2 PB of disk space
- Weighs in at number 75 of Top 500 SuperComputing list
- Cost to run = ~ $1,060 / hour

Customer Goals

- Genentech: “Examine how proteins bind to each other in research that may lead to medical treatments.”
  - www.networkworld.com
- Customer wants to test the scalability of CycleCloud: “Can we run 10,000 jobs at once?”
- Same workflow would take weeks or months on existing internal infrastructure.

Customer Goals (cont)

- They can’t get answers to their scientific questions quickly enough on their internal cluster.
  - Need to know “What do we do next?”
- Genentech wanted to find out what Cycle can do:
  - How much compute time can we get simultaneously?
  - How much will it cost?

Run Timeline

- 12:35 – 10,000 Jobs submitted and requests for batches cores are initiated
- 12:45 – 2,000 cores acquired
- 1:18 – 10,000 cores acquired
- 9:15 – Cluster shut down

System Components

- Condor (& Workflow)
- Chef
- CycleCloud custom CentOS AMIs
- CycleCloud.com
- AWS

Technical Challenges: AWS

- Rare problems:
  - If a particular problem occurs .4% of the time, if you run 1254 instances it will happen 5 times on average.
  - Rare problem examples:
    - DOA instances (unreachable).
    - Disk problems: can’t mount “ephemeral” disk.
- Need chunking: “Thundering Herd” both for us and AWS.
- Almost certainly will fill up availability zones.

Technical Challenges: Chef

- Chef works by periodic client pulls.
  - If all the pulls occur at the same time, run out of resources.
  - That's exactly what happens in our infrastructure when you start 1250 machines at once.
  - Machines do 'fast converge' initially.
  - So we need to use beefy servers, configure appropriately, and then stagger launches at a rate the server can handle.
- This was the bottleneck in the system
  - Future work: pre-stage/cache chef results locally to reduce initial impact on the server.

Technical Challenges: CycleCloud

- Implemented monitoring and detection of classes of rare problems.
- Batching of requests with delays between successive requests.
- Testing: better to request 256 every 5 minutes or 128 every 2.5 minutes? What's the best rate to make requests?
- Set chunk size to 256 ec2 instances at a time
- Did not overwhelm AWS/CycleCloud/Chef infrastructure
- 2048 cores got job work stream running immediately
- 1250 ec2 requests launched in 25 minutes (12:35am – 12:56 am Eastern Time)

Technical Challenges: Images + FS

- Images not affected by scale of this run.
- Filesystem: large NFS server.
- We didn’t have problems with this run (we were worried!)
- Future work: parallel filesystem in the cloud.

Technical Challenges: Condor

- …

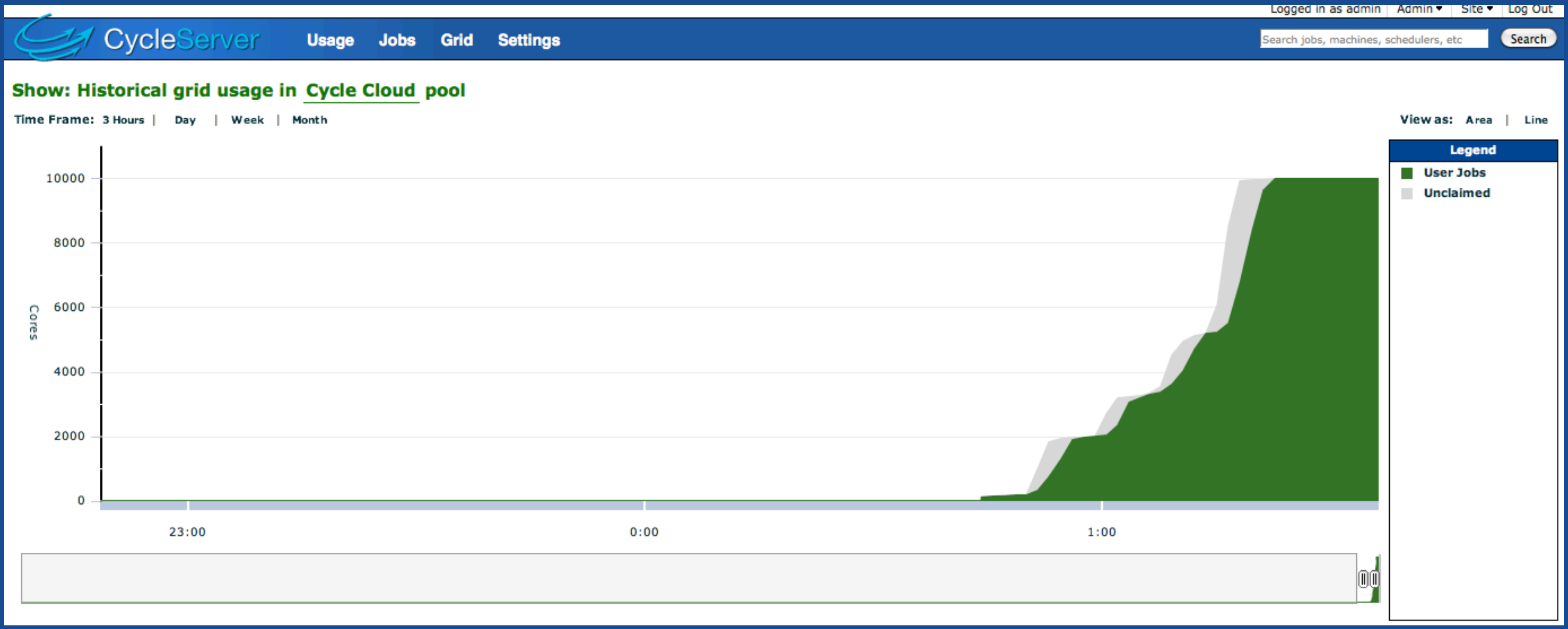
Technical Challenges: Condor

- Basic configuration changes.
  - See condor-wiki.cs.wisc.edu
- Make sure user job log isn't on NFS
  - Really a workflow problem…
- Used 3 scheduler instances (could have used one)
  - 1 main scheduler
  - 2 auxilary schedulers
  - Worked very well and handled the queue of 10,000 jobs just fine
    - Scheduler instance 1: 3333 jobs
    - Scheduler instance 2: 3333 jobs
    - Scheduler instance 3: 3334 jobs
  - Very nice, steady and even stream of condor job distribution (see graph)

So how did it go?

- Set up cluster using CycleCloud.com web interface.
- All the customer had to do was submit jobs, go to bed, and check results in the morning.

Cores used in Condor



Logged in as admin | Admin ▾ | Site ▾ | Log Out
CycleServer
Usage
Jobs
Grid
Settings
Search jobs, machines, schedulers, etc
Search

Show: Historical grid usage in Cycle Cloud pool

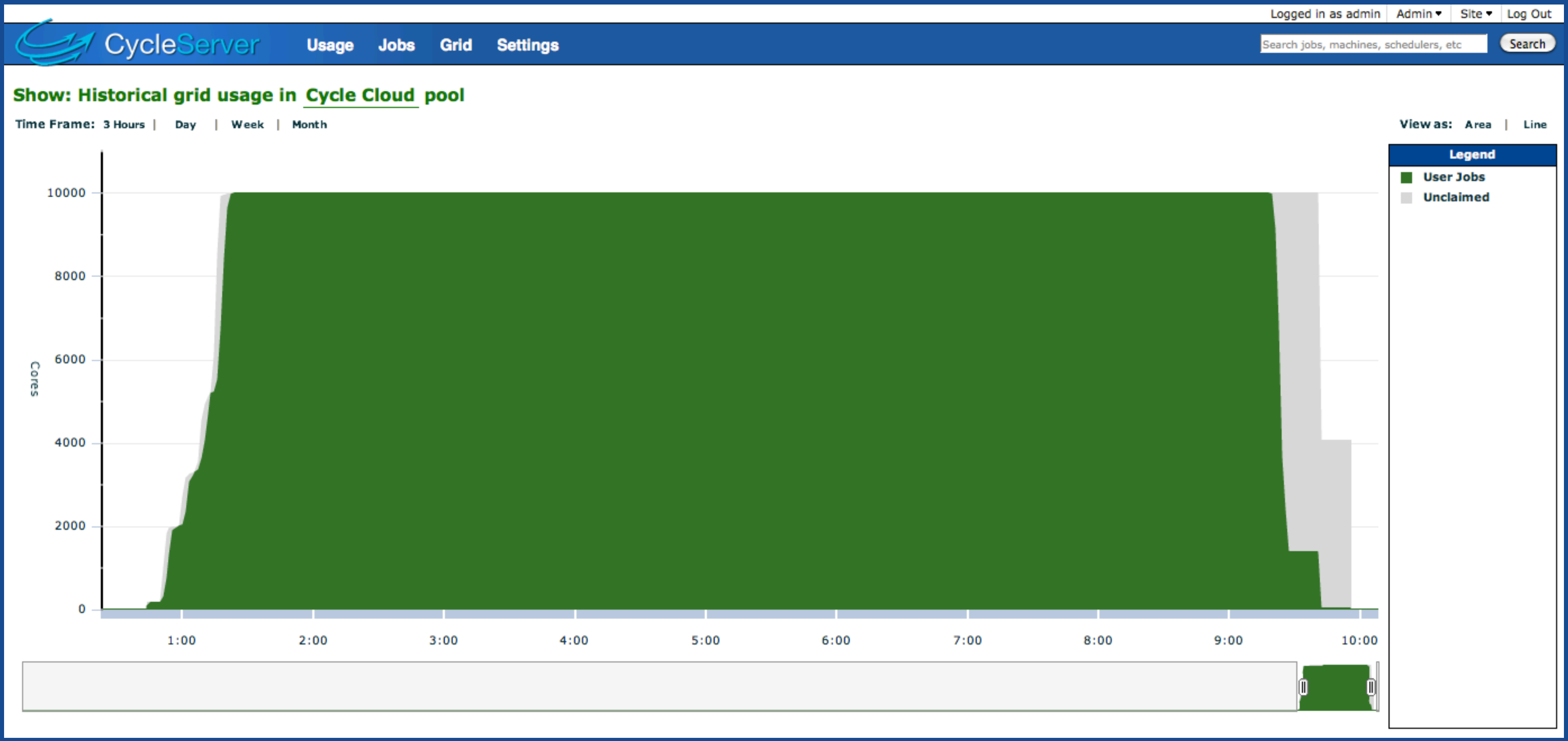
Time Frame: 3 Hours | Day | Week | Month

View as: Area | Line

CYCLECOMPUTING

CYCLECLOUD
THE EASE & POWER OF CLOUD HPC

Cores used Cluster Life-time



Show: Historical grid usage in Cycle Cloud pool

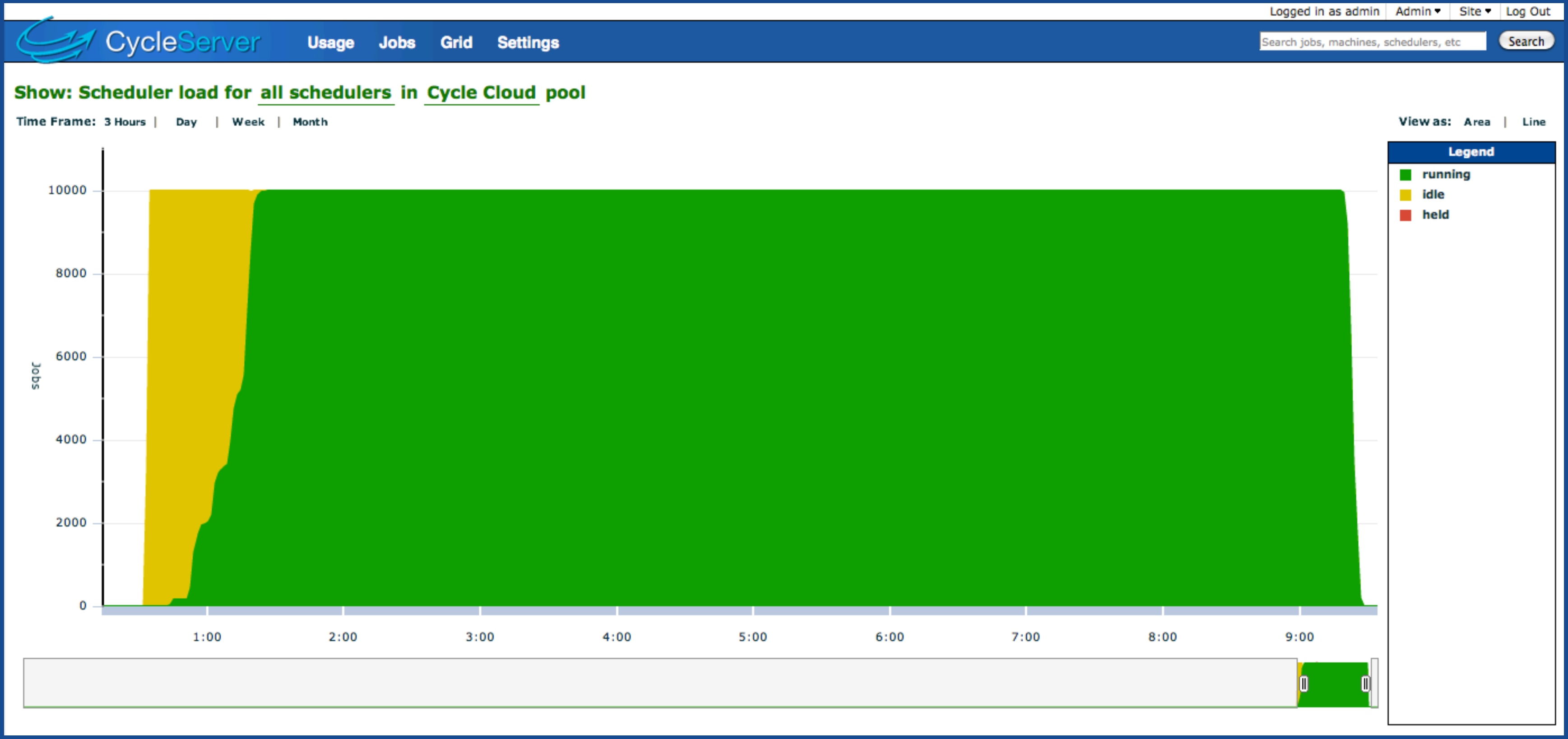
Time Frame: 3 Hours | Day | Week | Month

View as: Area | Line

CYCLECOMPUTING

CYCLECLOUD
THE EASE & POWER OF CLOUD HPC

Condor Queue

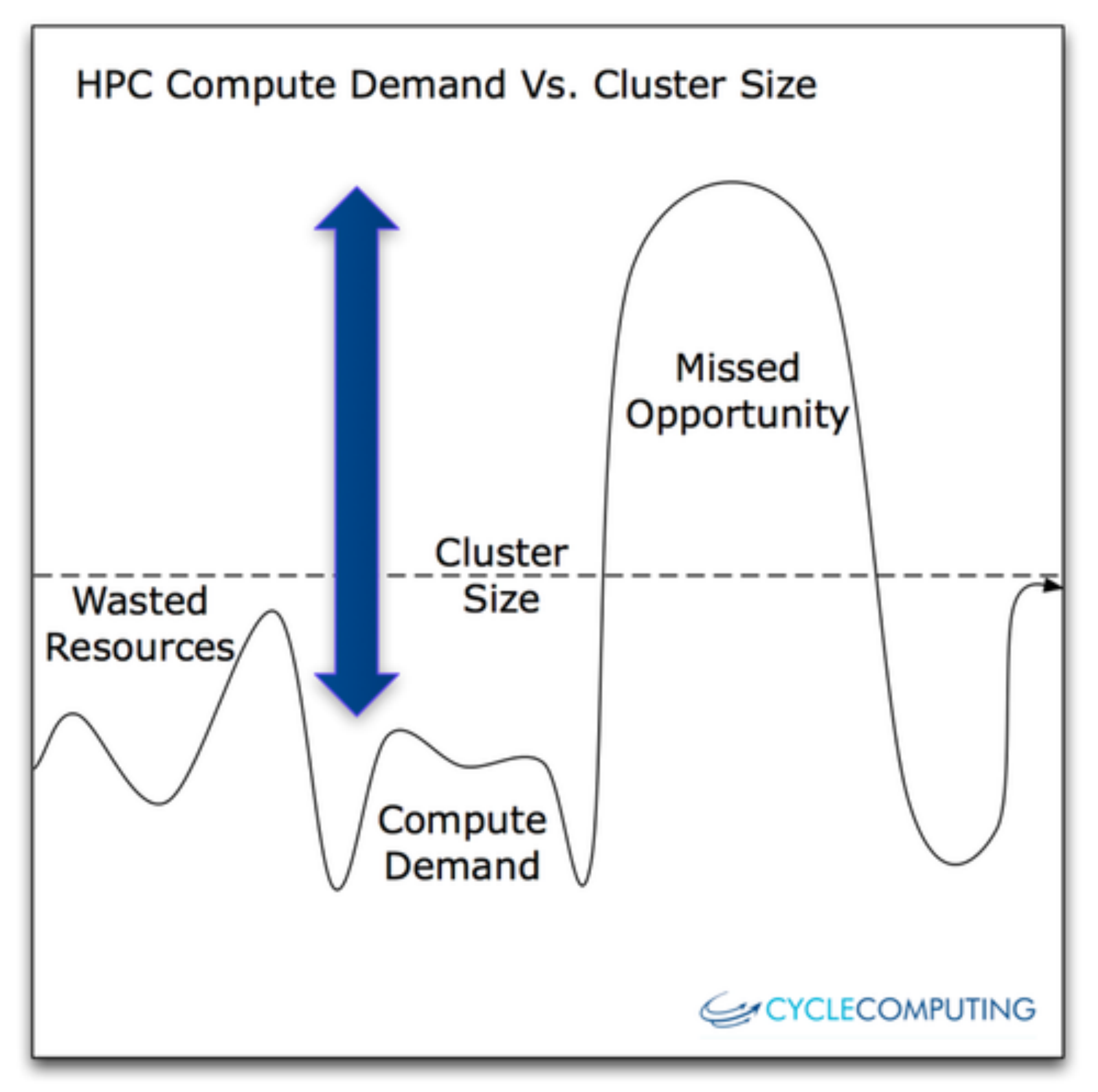


Show: Scheduler load for all schedulers in Cycle Cloud pool

Case 2: Synchronized Cloud Overflow

- Insurance companies often have periodic (quarterly) compute demand spikes. Want results ASAP but also don't want to pay for hardware to sit idle the other 11+ weeks of the quarter.
- Replicate internal filesystem to cloud.
- Run jobs (on Windows).
- Replicate results.

/Users/ianalderman/Old_Desktop/oldDesktop/top/CondorWeek/CondorWeek2010-2.ppt



HPC Compute Demand Vs. Cluster Size

Customer Goals

- Step outside the fixed cluster size/speed trade off.
- Replicate internal Condor/Windows pipeline using shared filesystem.
- Make it look the same to the user – minor UI change to run jobs in Cloud vs. local.
- Security policy constraints – only outgoing connections to cloud.
- Continue to monitor job progress using CycleServer.

System components

- Condor (& Workflow)
- CycleServer
- Chef
- CycleCloud custom Windows AMIs / Filesystem

Technical Challenges: Images and File System

- Customized Windows images (Condor).
- Init scripts a particular problem.
- Machines reboot to get their hostname.
- Configure SAMBA on Linux Condor scheduler.

Technical Challenges: Chef

- Implemented support for a number of features for Chef on Windows that were missing.
- Lots of special case recipes because of the differences.
- First restart caused problems.

Technical Challenges: Condor

- DAGMan submission has three stages.
  - Ensure input gets transferred.
  - Submit actual work (1000-5000 compute-hours)
  - Transfer results
- Cross platform submission.

Technical Challenges: CycleServer

- CycleServer is used:
  - As a job submission point for cloud jobs.
  - To monitor the progress of jobs.
  - To monitor other components (Collect-L/Ganglia).
  - To coordinate file synchronization on both ends.
    - Custom plugins.
    - DAG jobs initiating file synchronization – wait until it completes.
    - Plugins do push/pull from local to remote due to firewall.

Cloud vs. On-demand Clusters

Cloud Cluster

Actions taken to provision:
Button pushed on website

Duration to start: 45 minutes

Copying software – minutes

Copying Data – minutes to hours

Ready for Jobs

Physical Cluster

Actions Taken to provision

- Budgeting
- Eval Vendors
- Picking hardware options
- Procurement process (4-5 mo)
- Waiting for Servers to ship
- Getting Datacenter space
- Upgrading Power/Cooling
- Unpacking, Stacking, Racking the servers
- Plugging networking equipment and storage
- Installing images/software
- Testing Burn-in and networking addrs
- Ready for jobs