How High Throughput was my cluster?

Greg Thain
Center for High Throughput Computing

High Throughput Defined

$$\frac{\sum Job\ Runtimes}{Wall\ Time}$$

More Correctly

$$\frac{\sum Completed\ Job\ Runtime}{Wall\ Time}$$

Even more Correctly

$$\left(\frac{\sum Completed\ Job\ Runtime}{Wall\ Time}\right)^{*}$$

* Subject to some notion of fairness

There's always fine print

- Optimize goodput subject to following
- "Subject to some notion of fairness"
  - Recent usage
  - Machine ownership
  - Real world urgency
    - Temporary or otherwise
  - Group membership
  - Etc, etc.

What’s your policy?

- Are you sure you know?
- We’d like to know.

- We’ve got lots of mechanisms
- We would really like to know if sufficient

- Please talk to me!

Example policy

- Global limit on job from each group
- Also limit on sum of sub-groups
- One Free-for-all group, can use whole pool
  - Maybe not such a good idea

- If any job runs longer than two days:
  - It's drunk, send it home

Policy for CHTC pools

- Big question:
  - Longest allowable job runtime
- Currently 72 hours. Good? Bad?
- Policy note: set with negotiator, not startd

Why do we care?

```
condor_status -tot
                   Total Owner Claimed Unclaimed Matched
         INTEL/LINUX     1     0       1         0       0
        X86_64/LINUX  6639    63    6141       435       0
               Total  6640    63    6142       435       0
```

$$\frac{72\,\frac{hours}{job} * \frac{3600\,seconds}{hour}}{6000\,machines} = 43\text{ secs}$$

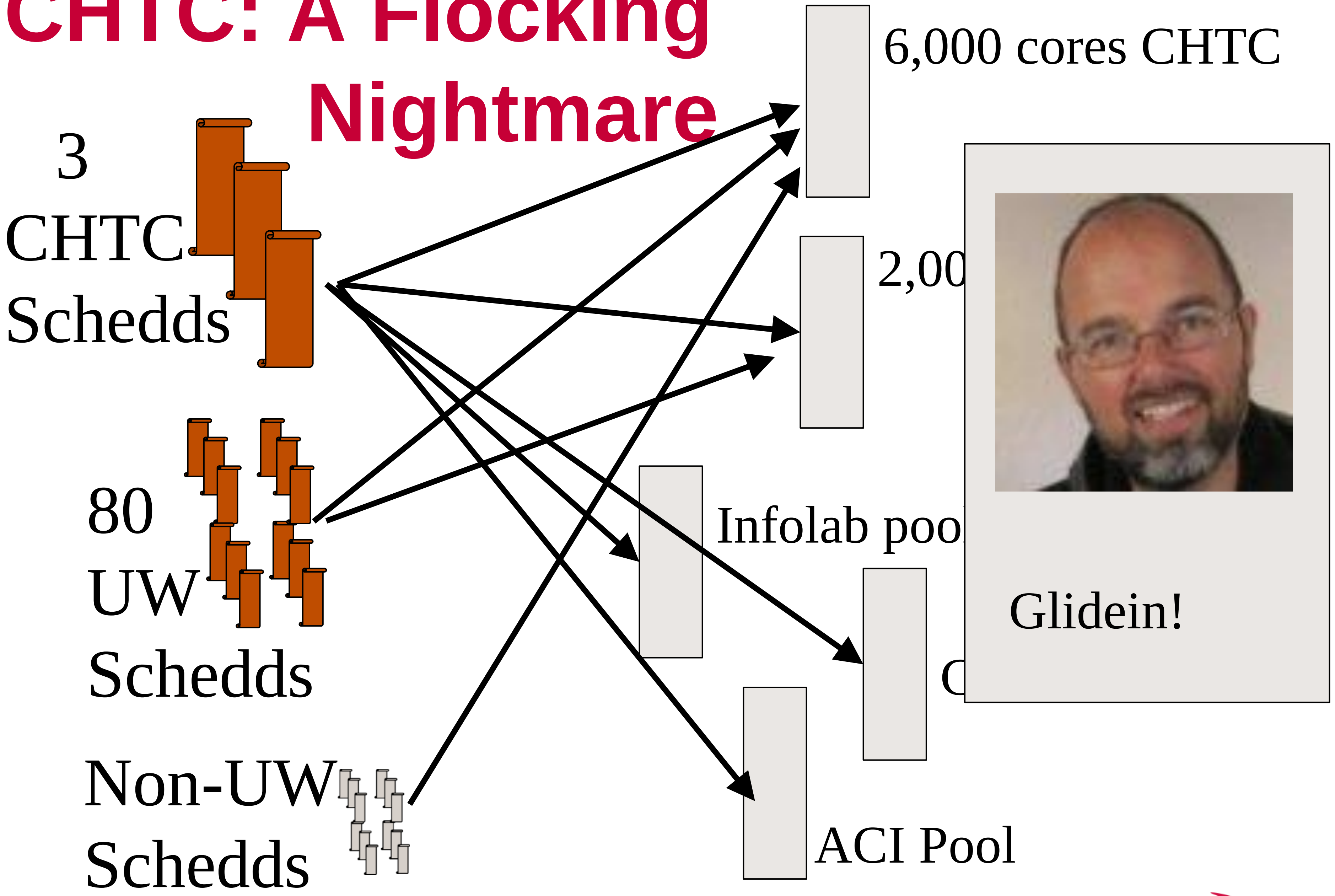
Problem: draining

- With homogenous slots, wait time a function of pool size, which is big
- Assuming no checkpointing

- If draining needed, job wait time a function of longest job. ☹

- More demand for HTPC jobs.

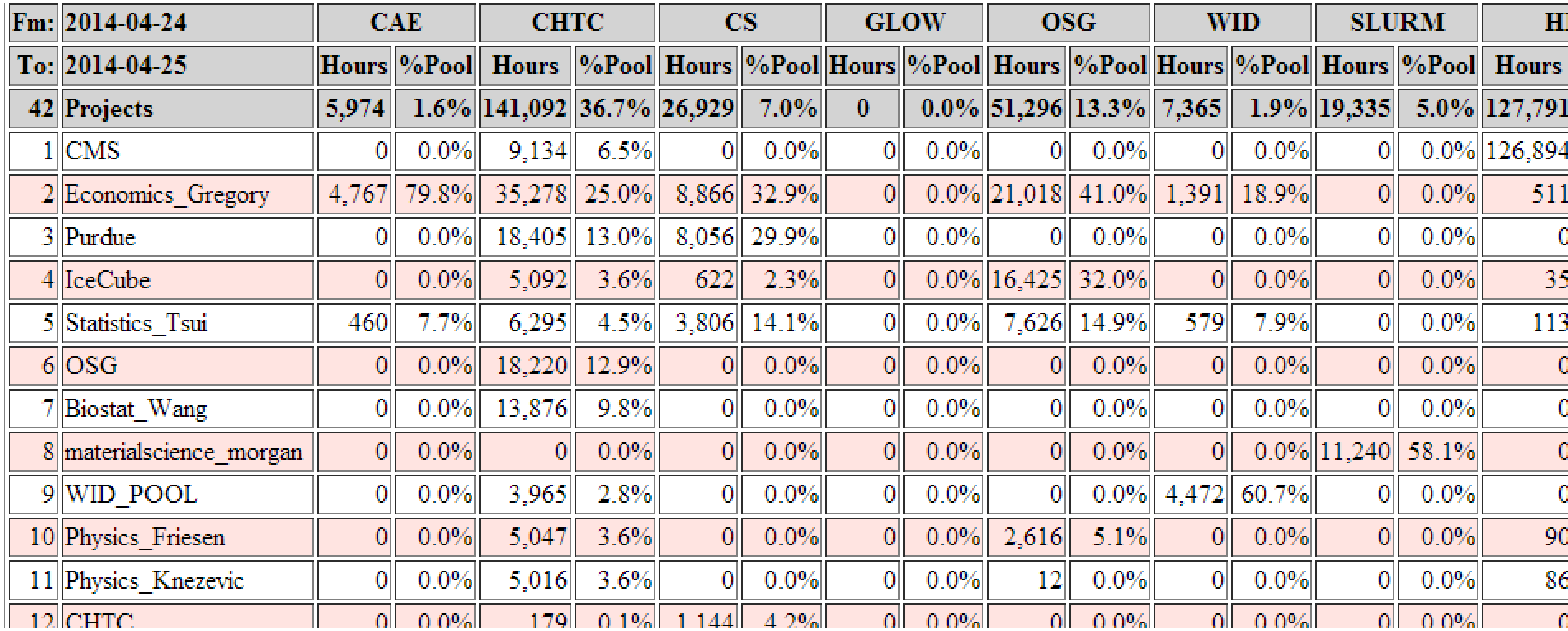
CHTC: A Flocking Nightmare



Negotiator Records

- “The Accountant”
- Access via
  - condor_userprio
- Records matches,
- Not jobs – e.g. glidein problem

Negotiator Reporting



Fm:	2014-04-24	CAE		CHTC		CS		GLOW		OSG		WID		SLURM		HI
To:	**2014-04-25**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**	**%Pool**	**Hours**
42	**Projects**	**5,974**	**1.6%**	**141,092**	**36.7%**	**26,929**	**7.0%**	**0**	**0.0%**	**51,296**	**13.3%**	**7,365**	**1.9%**	**19,335**	**5.0%**	**127,791**
1	CMS	0	0.0%	9,134	6.5%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	126,894
2	Economics_Gregory	4,767	79.8%	35,278	25.0%	8,866	32.9%	0	0.0%	21,018	41.0%	1,391	18.9%	0	0.0%	511
3	Purdue	0	0.0%	18,405	13.0%	8,056	29.9%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0
4	IceCube	0	0.0%	5,092	3.6%	622	2.3%	0	0.0%	16,425	32.0%	0	0.0%	0	0.0%	35
5	Statistics_Tsui	460	7.7%	6,295	4.5%	3,806	14.1%	0	0.0%	7,626	14.9%	579	7.9%	0	0.0%	113
6	OSG	0	0.0%	18,220	12.9%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0
7	Biostat_Wang	0	0.0%	13,876	9.8%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0
8	materialscience_morgan	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	11,240	58.1%	0
9	WID_POOL	0	0.0%	3,965	2.8%	0	0.0%	0	0.0%	0	0.0%	4,472	60.7%	0	0.0%	0
10	Physics_Friesen	0	0.0%	5,047	3.6%	0	0.0%	0	0.0%	2,616	5.1%	0	0.0%	0	0.0%	90
11	Physics_Knezevic	0	0.0%	5,016	3.6%	0	0.0%	0	0.0%	12	0.0%	0	0.0%	0	0.0%	86
12	CHTC	0	0.0%	179	0.1%	1,144	4.2%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0

Schedd Records

- “Event Log”: enable in config file
- “History file”: condor_history
- We don’t control them all

Startd also keeps history

- This is the one we use
  - condor_history –f startd_history
  - Enable by setting
  - STARTD_HISTORY = /path/to/file

condor_pool_job_report

```
The following users have run vanilla jobs that have hit the
MaxJobRetirementTime (72) hour limit in CHTC yesterday.
# of      User
Jobs
----      ----
  3 jchen48@submit.chtc.wisc.edu
 79 dschultz@skua.icecube.wisc.edu
 81 yqzhao@submit.chtc.wisc.edu
353 jsebald@skua.icecube.wisc.edu
```

= 31 K hours badput!

What is/isn’t a job “completion”?

- Strict definition: job exits of own accord
  - Two problems:
    - Very, very short jobs
    - Self checkpointable jobs
      - How to ID?
      - When_to_transfer_output = on_exit_or_evict
      - Adding explicit flag – requires a carrot
      - +is_resumable = true
- All this requires understanding users

Then, on to runtimes.

Averages can be deceiving

User	Starts	Total Hours	Mean
gthain	8442	8427	00:59

What about quartiles?

1st quartile	00:01 (One Minute)
2nd quartile	00:12
3rd quartile	00:42
4th quartile	68:41

“Jobs” vs “Execution attempts”

- If 25% of runs less than one minute
- Is that just one bad job?
- Or all of the jobs are bad?

Added new columns to report

- "Restarted jobs"
- Quartiles
- Short jobs (less than minute)
- Removed hours
- Mean, Median, SD

- Requires a lot of user facilitating

Problem: Zoo of a pool

Order of magnitude different speeds in pool

Naïve Solution:

  Create scaled performance numbers

Actual solution

  Remove very slow machines from pool

  Require users to ask for fast machines

Results of looking at data

- Can lower 72 hour limit to 24
- Probably need “escape hatch” for some
- Can drastically improve draining response

Future Work

- Support for slot-based scheduling?
- Support for mixed HPC / HTC submissions/

Thank you!

- Please talk to me about pool policy
  - We'd love to hear from you!
- Important to know the shape of jobs
- Pure hours consumed not important metric
- Preempt-Resume right the first time!