Comprehensive Resource Use Monitoring for HPC Systems with TACC Stats

Texas Advanced Computing Center
R. Todd Evans, Bill Barth, James Browne

University at Buffalo, SUNY
Robert Deleon, Thomas Furlani, Steven Gallo, Matthew Jones, Abani Patra

January 12, 2015

- Introduction
- Operation
- Data & Tools
- Case Studies

TACC Stats is NOT a Line-level Performance Analysis Tool

TACC Stats ...

- Is NOT like: TAU, IPM, gprof, Vampir, Perfexpert, Intel VTune etc.
- Does NOT provide profiling at the level of function calls, control structures, or statements
- Does NOT require instrumentation
- Does NOT need to be invoked by the user or system managers

TACC Stats IS a Transparent Resource Usage Monitoring Tool

TACC Stats . . .

- Knows which users, jobs, executables etc. are running at what times
- Is always running on every node for every job
- Collects hardware counters (oncore/offcore) and a comprehensive list of Linux/Lustre stats
- Updates automatically daily and provides a Web Interface
- Tests jobs automatically for inefficient resource utilization or software/hardware failures
- Is Open Source Software: `https://github.com/TACC/tacc_stats`

TACC Stats Operation

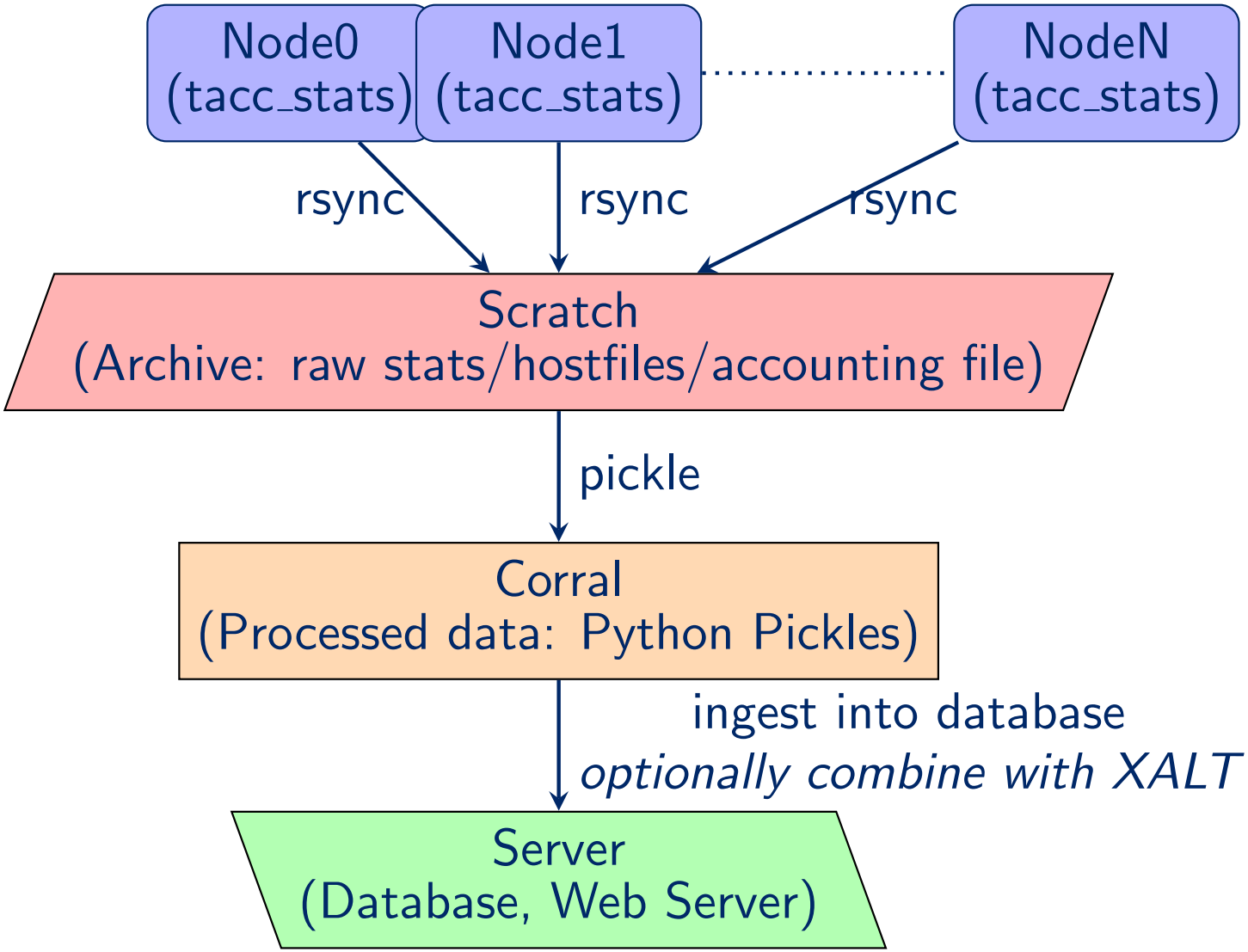
Deployment

- Distributed as a Python package
- Light-weight version (raw C) run on compute nodes
- Can be installed with pip install or rpm

Automated Operation

- Measurements for all jobs & nodes in prolog/epilog + every 10 minutes $\rightarrow$ at least two measurements per job
- `rsync`'d once a day at random times between 2 and 4 am.
- pickled once per day
- ingested into database once per day
- tests run once per day
- Every Production Job $>$ 10 minutes is subjected to Daily Tests

TACC Stats Operation



Data Collected

Summary

- Block device I/O
- CPU utilization
- IB stats
- Chip stats - "on-core" and "off-core" - instructions, flops (SSE scalar/SSE vector/256), memory reads/writes ...
- Lustre I/O
- Network (scif0, eth0)
- Memory, Virtual Memory
- 150KB per host per day - Stampede 0.8GB/day

Daily Tests

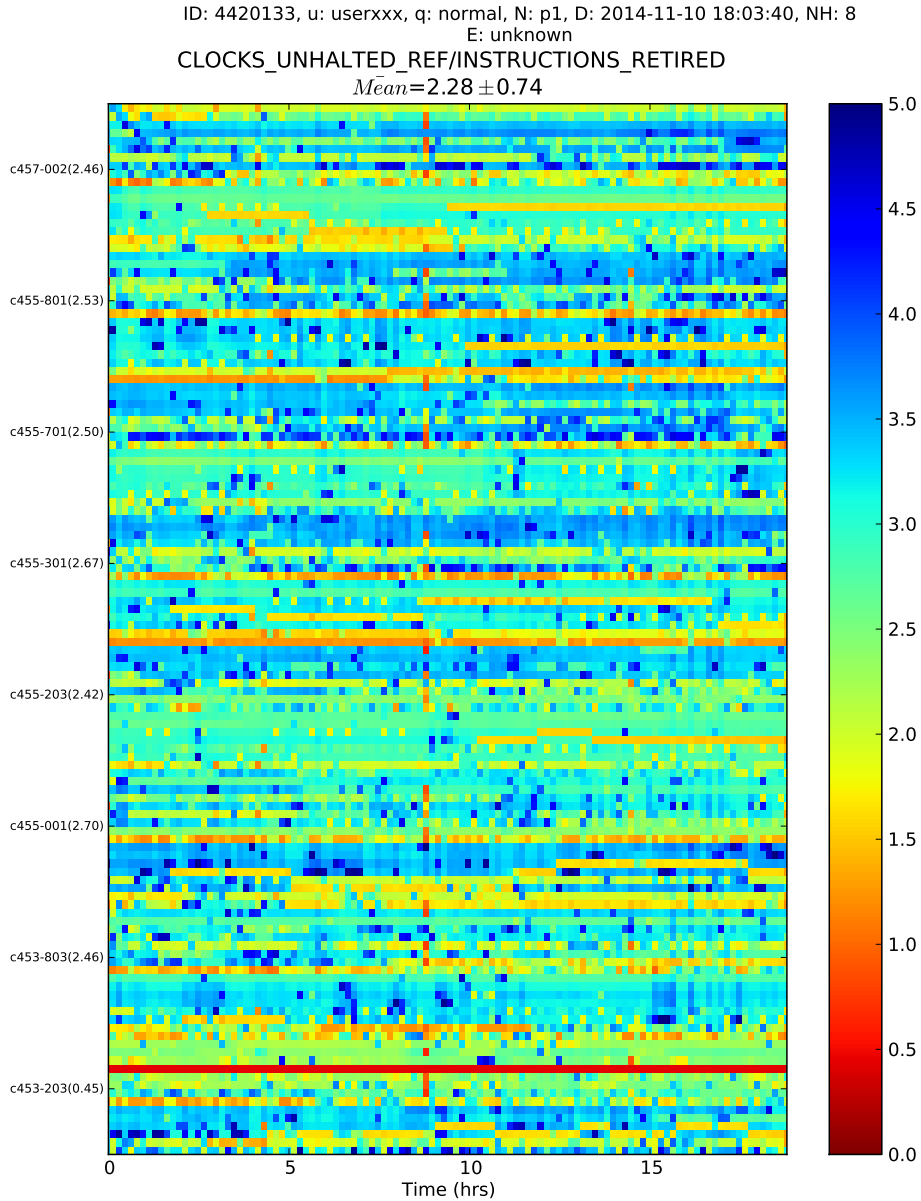
Metrics computed - averaged over time and hosts/cores

- Cycles per Instruction (CPI)
- Cycles per L1D replacement (CPLD)
- Load L1Hits
- Load L2 Hits
- Load LLC Hits
- Memory Bandwidth
- Memory Usage High Water Mark
- Infiniband Packet Rate
- Infiniband Packet Size
- FLOPS
- Vectorization Fraction
- Ethernet Bandwidth

Daily Tests

Also test for:

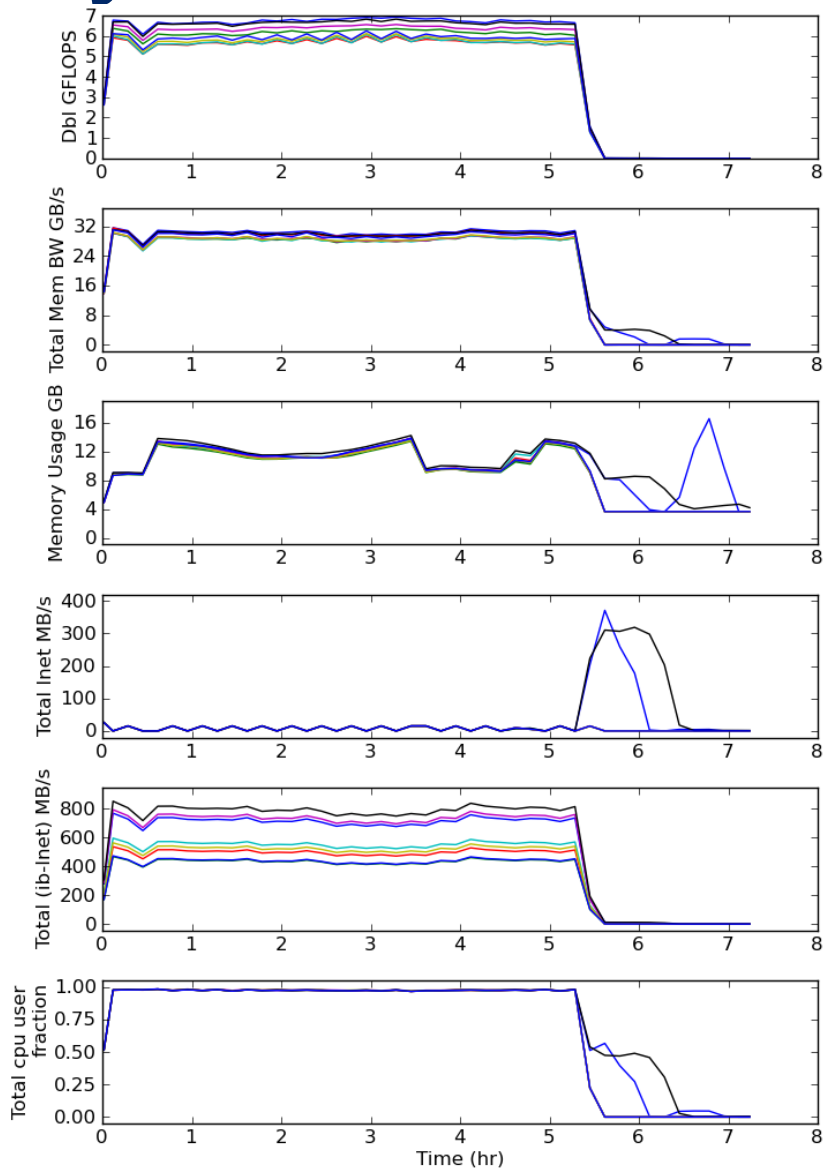
- Idle Nodes - only some reserved nodes running for job.
- Roughly $\sim$ 1% of SUs spend on jobs w/ idle hosts



Daily Tests

Also test for:

- Catastrophes
- Alerts us to Hardware & Software issues



Web Interface

Backed by SQL Database

- Combined queries are supported in search fields
- Search by jobid, date, user, executable, project, nodes, run time, metrics ...
- Data is ingested from accounting file, XALT, processed stats data
- Raw time-series data is served and plots are generated on the fly
- Linked into Splunk logs - link takes user to logs filtered by host and time range for a given job.
- Used for day to day consultant activities typically for diagnosing issues experienced by our users

Web Interface

11/18/2014 TACC Stats Stampede

TACC Stats Stampede

Search Using JOBID:

Search

Search Using Combined Fields:

Start Date mm/dd/yyyy End Date mm/dd/yyyy

UID
User Name
Project
Executable
Search Field Value

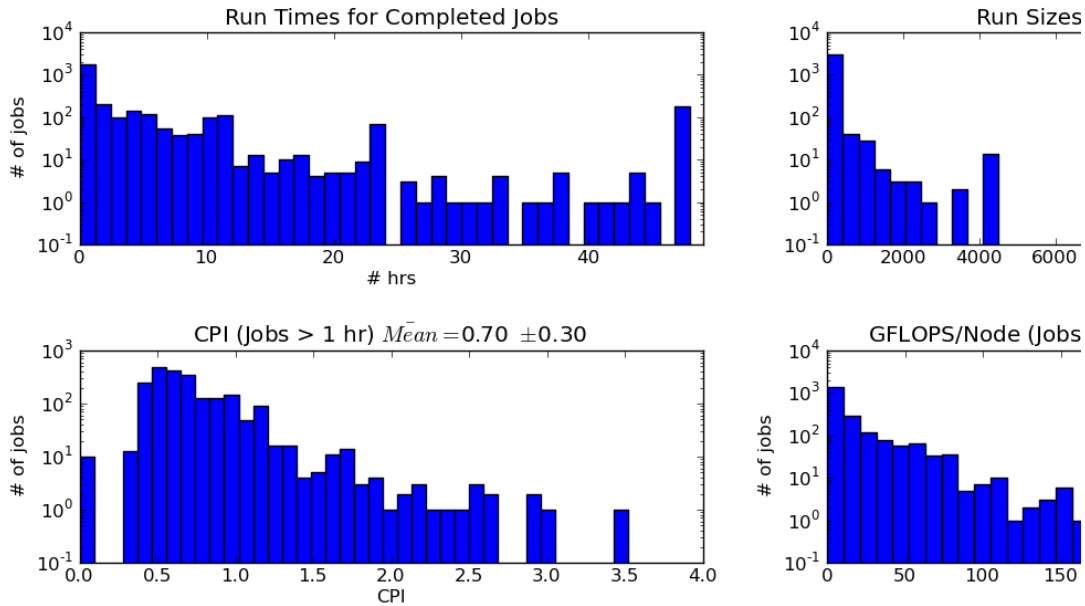
Search

2013 / 09	17	18	19	20	21	22	23	24	25	26	27	28	29	30																
2013 / 10	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2013 / 11	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
2013 / 12	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2014	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Web Interface

11/18/2014 TACC Stats

[date=2014-11-17]-search



Jobs over 1 mn in run time = 3626

Job Listing

Jobs flagged for:

- GigE BW > 1.049e+06 B

lagarder	4457699(1.612e+06)			
iplant	4457846(5.406e+06)	4456558(2.384e+06)		
unknown	4456785(2.247e+06)	4453596(1.806e+06)	4453592(1.768e+06)	4453591(1.612e+06)

Case Studies

MILC code

- One project using MILC found to be running higher than expected CPI (1.1 vs 0.7)
- Members were not using available vectorized intrinsics
- 11% reduction in runtime

DNS Turbulence Code

- CPI of 1.1 w/ lots of SUs
- Line-level profiling revealed MPI/OpenMP hot spots
- Converted OpenMP workshare block to parallel do block
- Improvement: 7% overall, 10% in main loop, 76% in code block

Case Studies

GR application - Singularity

- CPI of 1.15 w/ lots of SUs: 239,631 for 1st quarter 2014
- Code was making many extraneous calls to `cat` and `rm`
- Code was not using any optimization flags (-O3 or -xhost)
- 26% decrease in run time after simple changes made

A few other observations

- Very few codes using large memory nodes use the 1 TB available (mean is ∼120 GB)
- Many codes are not using AVX (256b) instructions. Identified via VecPercent metric and verified via inspection of AVX FLOPS data.
-

Work in Progress

- Build a basis of metrics on which applications can be mapped.
- User/project performance can be compared to expected performance
- Validating/interpreting raw data
- Real-time, continual stats updates via the network: syslog or RabbitMQ (just needs scaling testing)
- Deployment at additional Centers
- Adding support for additional architectures and accelerators: Haswell, MIC, GPU
- Data is being incorporated into XDMoD (XD Metrics on Demand Portal, a comprehensive HPC management tool)

Acknowledgements

- NSF-funded STCI project ACI-1203604
- Robert McLay (TACC), Mark Fahey (NICS) for XALT