




# Remora: A Resource Monitoring Tool for Everyone

Carlos Rosales

carlos@tacc.utexas.edu

# Where does that odd name come from???

- It attaches to the user processes
- It travels with them in the system
- It feeds off your job (overhead) but provides some benefits (information)

# What is Remora?

- Remora monitors all user activity and provides per-node and per-job resource utilization data
- Developed by Antonio Gomez-Iglesias and Carlos Rosales at TACC
- Open source, available at github

- NOT a profiler
- NOT a debugger
- But the data collected can often be used to improve code performance or detect issues

# Common Issues

- User questions:
  - Why did I get banned from running jobs?
  - Why did my job crash?
  - Why is my performance so low in your supercomputer?
- We have some tools in place:
  - Server logs (Splunk)
  - TACC Stats (hardware counter data, 10 min period)

# Current Tools Are Insufficient

- 10 min interval in TACC Stats misses spikes of activity.
  - Fails to detect single large memory allocations
  - Fails to detect localized instances of high IO traffic.
- Splunk is tedious to parse and typically only contains catastrophic errors.
- NEITHER is visible to the user
- Many useful features, but missing some critical to our users

# How does Remora fix those issues?

- Fine-grained temporal resolution (tunable)
- Simplified output for basic user
  - Highlights possible issues without overwhelming
- Raw data available for advance users
  - Deep analysis of each run possible
  - Post-processing tools provided

# Information Collected

- Detailed timing of the application
- CPU utilization
- Memory utilization
- NUMA information
- I/O information (FS load and Lustre traffic)
- Network information (topology and IB traffic)

# Accelerator support

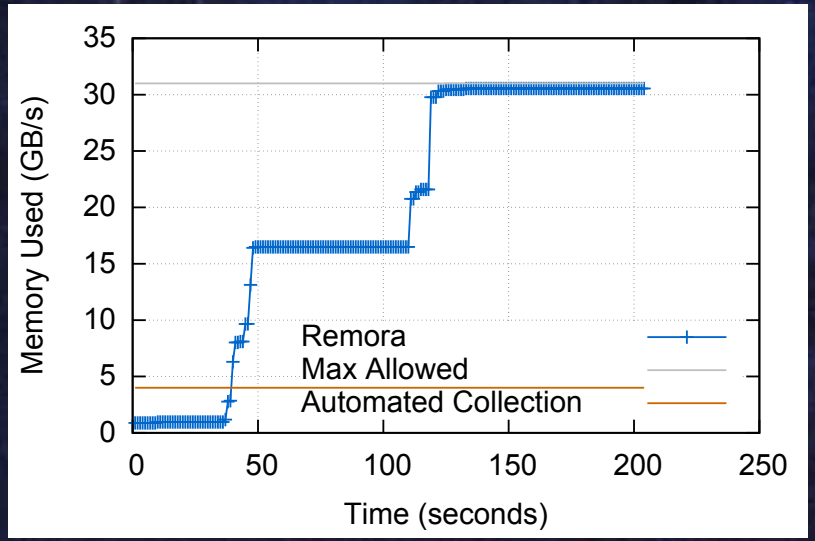
- Intel Xeon Phi
  - Treated like any other node
  - Background process is bound to core 61 to minimize overhead
- GPU
  - Collects memory information using nvidia-smi
  - Other information is much harder to get to!

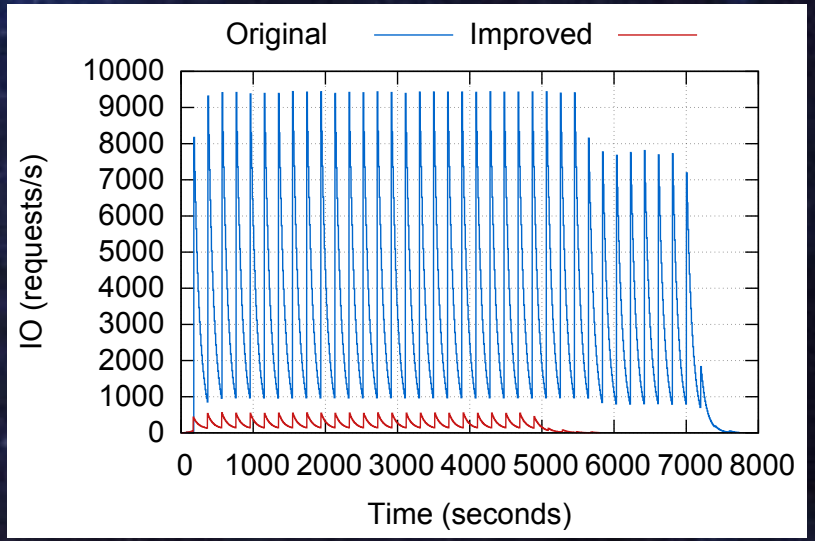
# Remora Summary

```
================================================================================
TACC:  Max Memory Used Per Node  : 8.52 GB
TACC:  Total Elapsed Time        : 0d 0h 0m 27s 64ms
TACC:  MDS Load (IO REQ/S)       : 0.00 (HOME) / 0.00 (WORK) / 2.00 (SCRATCH)
--------------------------------------------------------------------------------
TACC:  Sampling Period           : 2 seconds
TACC:  Complete Report Data      : /full/path/to/workdir/remora_5905747
================================================================================
```

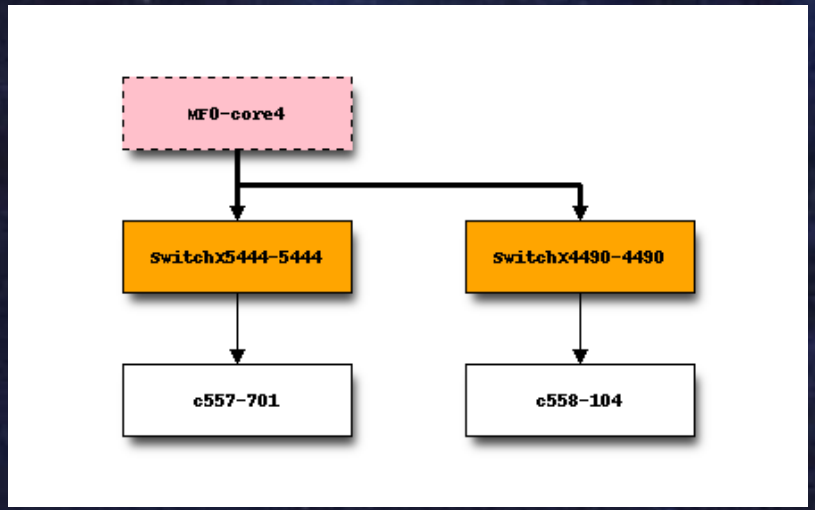
Plus additional lines for memory utilization is MICs or GPUs are used

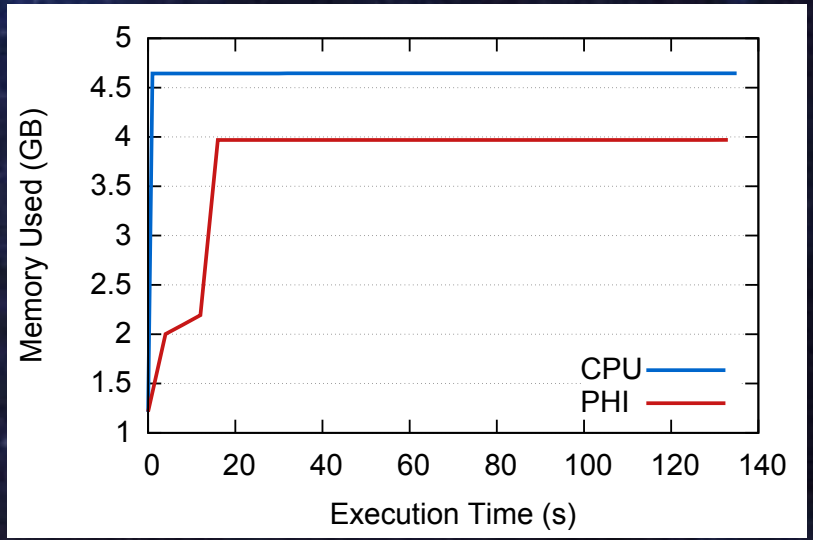
# Raw Data Analysis



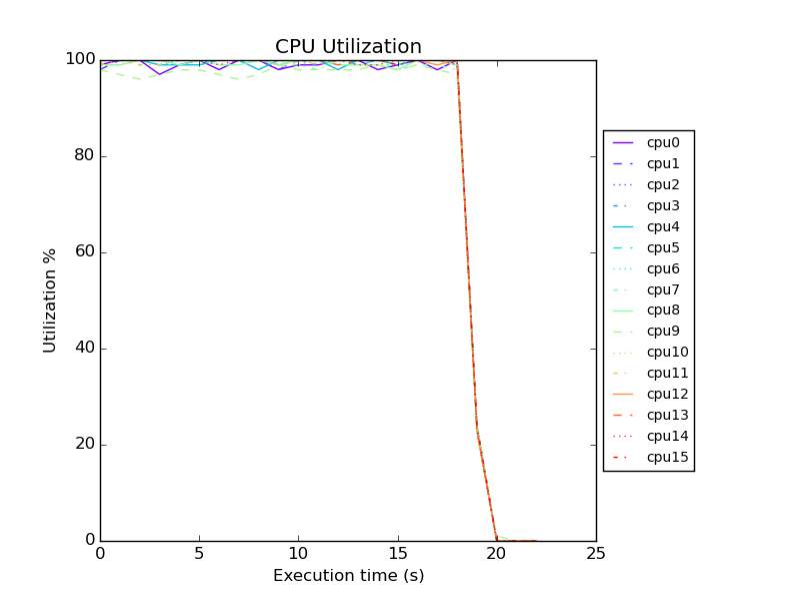



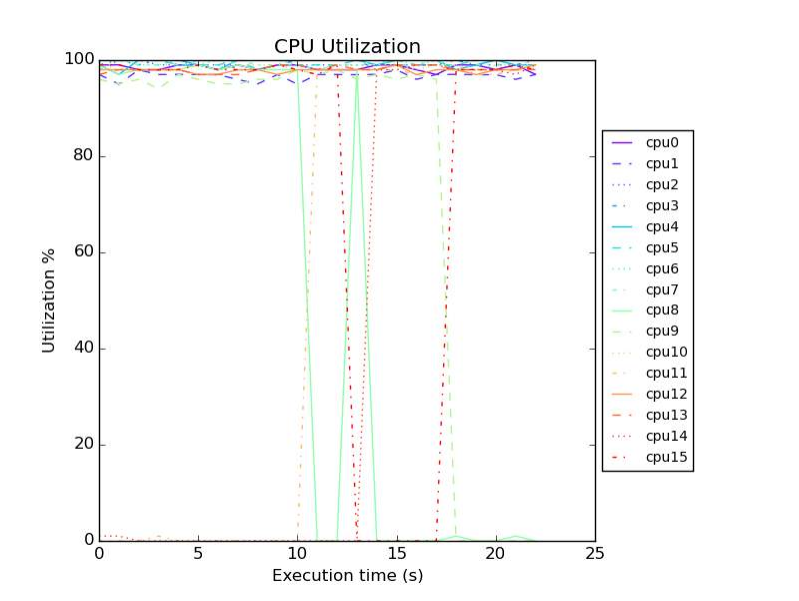
# Raw Data Analysis



# Raw Data Analysis



# Simple to Use

```
module load remora
remora ibrun mympi.code
```

```
module load remora
remora ./mycrazy.script
```

# Implementation

- Bash and python, plus some C xltop trickery by Antonio ☺
- Master starts flat tree ssh connection to all nodes
- Background task spawned in each node
- Background task collects data regularly
- IO data collected only from master node

# Implementation

**Programs**

- numastat
- mpstat,
- nvidia-smi
- ibtracert
- lbstatus
- xltop
- python

**Files**

- /proc/meminfo
- /proc/<pid>/status
- /proc/sys/lnet/stats
- /sys/class/infinband/…

# Portability

- Some hardcoded strings only applicable to TACC – easy fix (coming soon)
- Hardcoded MPI launcher (ibrun) – easy fix (coming soon)
- XPost-processing has some TACC specific entries – easy fix (coming soon)
- Itop requirement for Lustre IO report
- Need to expand on the way the hostlist is collected

# Future Plans

- Comprehensive report generation
- Identify egregious performance issues and generate appropriate warnings
- Add database for better comparative / historical data analysis
- Improve launch step for better scalabilty

# Thanks!

{carlos,agomez}@tacc.utexas.edu

www.github.com/TACC/remora

For more information:

www.tacc.utexas.edu