CALIBERS
A Bandwidth Calendaring Paradigm For Science Workflows

Nathan Hanford, Dipak Ghosal
Eric Pouyoul, Mariam Kiran
Fatemah Alali
Raj Kettimuthu
Ben Mack-Crane
Should the user have to do resource allocation?
Motivation

Mission-Critical Science Workflows: Hurricane tracking, Astronomy, etc.

Data needs to be in SAN storage or a burst buffer by a strict deadline

Negative consequences to missing deadline

Goal of predictability over raw performance
Talk Outline

1. Background
2. Implementation
3. Results
4. Conclusion
Background
Building blocks

TCP: survivable, scalable and fair (for the most part)  
(But fairness isn’t always desired)  
Software-Defined Networks: rapidly reconfigurable  
Switch-based shaping: avoids interference  
End-system pacing: efficient throughput control  
Intent-driven network for deadline awareness  
ESnet’s transcontinental 10 Gbps SDN Testbed and OSCARS circuits
Contemporary Solutions

TEMPUS: Performance-oriented
DNA/AMOeba: Uses traffic classification
B4: Performance-focused
SWAN: Dynamic dataplane reconfiguration

Our contributions:
1. Considering end-systems we can’t control
2. Exclusively dealing with elephant flows
Implementation
CALIBERS Architecture

Currently single-controller implemented as a RESTful python orchestrator.

Participating DTNs run a RESTful Python client and shape using CoDel.

Corsa DP2000 Series edge switches use 3-color meters to guarantee non-participating clients don’t interfere with bandwidth reservations, and are dynamically controlled through a REST API.

GridFTP (Globus) provides the actual transfers.

Runs on OSCARS circuits.
High-level Architecture

**Central Controller**
1. Poll SNMP data from the switches, routers, and historical databases
2. Run ML and routing algorithms
3. Run optimization algorithm
4. Actuate rate control in TBNs
5. Actuate AQM in n switches and routers
Solution Approach

1. Find the minimum rate, $R_{\text{min}} = \frac{\text{file size}}{\text{deadline}}$
2. Find the maximum residual rate ($R_{\text{resid}}$)
   a. Assign $R_{\text{resid}}$ to the new request as long as $R_{\text{resid}} \geq R_{\text{min}}$
   b. Transfer the file as fast as possible to free up resources for future requests
3. If $R_{\text{min}}$ is not available
   a. Reduce rate of other flows
4. When a flow completes, redistribute its bandwidth to ongoing flows
5. Pacing and bandwidth redistribution are performed based on four heuristic algorithms combining two concepts:
   a. Global and local optimization
   b. Shortest Job First (SJF) and Longest Job First (LJF)
Dynamic Pacing Algorithm

1) Determine which flows should be considered for pacing:
   • Global approach:
     • the scheduler consider all flows when distributing any residual capacity
   • Local approach:
     • The scheduler consider only flows that span the bottleneck link when distributing residual capacity
     • Bottleneck link defined as the link with a flow that has the longest completion time, i.e., the link that will stay busy the longest

2) Based on the selected flows, determine which flow should be paced first
   • Shortest Job First (SJF):
     • Start with the flow with the smallest remaining data to be transferred
   • Longest Job First (LJF):
     • Start with the flow with the largest remaining data to be transferred
Evaluation: Metrics

Network Utilization
Reject Ratio
Performance Index: the difference between network utilization and reject ratio
The larger the difference the better
Ideally we want 100% utilization and a reject ratio of 0%
Simulated Algorithm Evaluation

As arrival rate increases:
- Utilization increases
- Reject ratio increases

Negligible difference between the 4 algorithms with small epoch

Lower performance even though reject ratio is because utilization is low

Based on the simulated network (G-scale), local approach optimization is sufficient
The difference in performance between SJF and LJF becomes more apparent with a longer epoch duration:
- with LJF the makespan time of all flows reduced
- hence resources are freed up faster for future requests

Lower performance with larger epoch as arrival rate increases:
- requests are aggregated making the scheduler less flexible

At low arrival rate, higher performance with 5-min:
- The utilization is higher because requests are aggregated, hence higher performance
Comparison with TCP Fairness

GridFTP Flow Completion Times to Washington, DC

- Amsterdam
- New York
- Denver

Completion time (s)

Achieved throughput

- rate_Mbps
- retrans (right)

Transmissions

Throughput (Mbps)
Our Live Demonstrations

- Two simultaneous tests: one with unpaced TCP, the other with CALIBERS
- 6 senders per test, for 12 total senders from around the United States and the world
- Receiver will be the SCinet DTN in the NOC booth # 1081
- Controllers will be located in Atlanta, and operated from the DOE booth # 613
- Goal is to meet or exceed deadlines beyond the capability of TCP
Conclusions

- Do resource allocation for the user
- Allow jobs to “sprint” past others to meet their deadlines
- Offer a different kind of service from OSCARS circuits
  - (Which, in turn, offer a different kind of service from dark fiber connections).
- CALIBERS does pacing, metering, and shaping
  - Prevents interference
- All pacing, metering, and shaping is done in hardware for scalability
Future Work

- **Very Near Future: Our Demo!**
  - DOE Booth # 613:
  - 4PM Tuesday
  - 11AM Wednesday
  - 1PM Thursday

- **Longer-term**
  - Distributed controller
  - Routing
  - Algorithm refinement

- **Questions? nhanford@ucdavis.edu**