Controlling the Delay of Small Flows in Datacenters

Damiano Carra

Computer Science Department
University of Verona
Verona, Italy
Outline

Background
Design
Evaluation
Conclusions
Datacenter networking

- Hundreds / thousands machines

- Initial approach → Hierarchical structure
Datacenter networking (cont’d)

- **Main issue** → Bandwidth bottleneck
  - Due to oversubscription

- **Possible solutions**
  - At the *protocol* level
    - e.g., Multi-path TCP (MPTCP), DataCenter TCP (DCTCP)
  - At the *topology* level
    - e.g., Fat-tree topologies
Hybrid topologies

- **Mixed packet-/circuit-switched topologies**
  - Circuit-switched topology carries large flows
  - Packet-switched topology carries the rest of the traffic

- **Circuits have long set-up time**
  - e.g., optical switches

- **The solution requires:**
  - Traffic matrix estimation
  - Centralized hot-spot flow scheduling algorithm
Open issues

- Fat-tree topologies are expensive
  - Switches, cabling ... → oversubscription to reduce costs

- Solution based on new protocols require changes at the protocol stack

- Bandwidth is not the only important performance metric
  - Delay-sensitive applications
Outline

Background

Design

Evaluation

Conclusions
Mice and elephants

- 90/10 rule
  - 90% of the flows carry 10% of the traffic
  - 10% of the flows carry 90% of the traffic
  - Confirmed by recent measurements in different datacenters

- Small flows → Delay sensitive
- Large flows → Bandwidth sensitive
Main idea

- Mice and elephants on separate paths
  - Mice path → Simple topology to control delay
  - Elephant path → Favors bandwidth intensive transfers

- Different from previous two-tier architectures
  - We focus on supporting small delays for small flows
    (rather than considering only the routing of the large flows)
Architecture

Tier-1
Fully Connected Topology

Tier-2
Switched Topology

ToR switch
Servers
Fully connected topology

- Each ToR switch is connected to all the other ToR switches
  - Number of hops is minimized
  - Delay due to only one queue

- Single high-end switch can be used
  - 256 ports, 10Gb Ethernet each
  - Target: container-size datacenter
    - 20-40 machine per ToR → 5k-10k machines
Fully connected topology (cont’d)

- Alternative approach based on optical networks
  - WDM ring, with different channels
    - corresponding to a $\lambda$, or a fraction of it

- Single Source Broadcast Channel (SSBC)
  - In each channel, a single ToR switch can transmit
    - While all the other ToR switches listen
  - This solution requires $N$ distinct channels
    - $N \rightarrow$ number of ToR switches
Optical module

Optical ring

from/to ToR switch
Considerations

- Flexible resource assignment
  - $\lambda$ can be fractioned with TDM
  - Optical rings can be stacked

- The ring can be implemented inside a single switch
  - In case ToR switches are spatially close
  - The optical module would be a port of such a switch
Routing flows

- How to identify mice and elephants?

- Many possible solutions
  - Application-assisted
  - Per-flow statistics at the ToRs
  - Random sampling

- Any solution can be used
  - Each solution has pros and cons
Routing flows (cont’d)

- We consider a simple solution
  - Elephant detection at the ToRs
  - Based on byte counting
    - and a threshold $T_s$

- Routing
  - A new flow goes to the fully connected topology
  - If the flow has more than $T_s$ bytes, it goes to the other tier
Routing flows (cont’d)

- **SDN implementation**
  - Default route
    - fully connected topology
  - ToR switch communicates with the Control Center if a flow has more than $T_s$ bytes

- The SDN Control Center can:
  - Route the elephant maximizing the throughput
  - Compute and set periodically $T_s$
Settings

- Simulation-based study
  - Based on the simulator developed for MPTCP

- Different topologies implemented
  - VL2
    - Single tier, implements Clos
  - cThrough
    - 2 tier architecture, with optical circuit-switched tier
  - Our solution → mDelay
Traffic matrices

- The system works in heavy load regime
- Three different traffic patterns
  - Element \((i,j)\) ➔ Traffic from ToR switch \(i\) to ToR switch \(j\)
Average flow delay and throughput

Delay (w.r.t. service time)

Throughput (% link bandwidth)

-17%
Flow completion time: small flows

- Diagonal Heavy Tail traffic pattern
  - similar results with other patterns
- Threshold $T_s = 35$ pkts
- Small flows represent 90% of the flows
Flow completion time: medium flows

- Medium flows are treated as in VL2
  - The second tier is actually VL2, but with less bandwidth
- cThrough penalizes small and medium flows
Flow completion time: large flows

- cThrough better only for very large flows

- In mDelay, the bandwidth dedicated to small flows has a moderated impact on large flows
Discussion

- Open issues that needs more investigation
  - Adaptive threshold
    - Based on the instantaneous load of the fully connected topology
  - Bandwidth dimensioning
    - How many resources should be assigned to the fully connected topology?
  - Comparison with strict priority
    - Single tier, with different traffic classes
Conclusions

- We proposed an hybrid datacenter architecture
  - Focused on the delay of small flows

- No modification to the protocol stack
  - ToR switches should be flow-aware

- Preliminary results indicate interesting properties
  - and open different research directions
Thanks!