15-744: Computer Networking

Data Center Networking II
Overview

- Data Center Topology Scheduling
- Data Center Packet Scheduling
Current solutions for increasing data center network bandwidth

1. Hard to construct
2. Hard to expand
An alternative: hybrid packet/circuit switched data center network

Goal of this work:
– Feasibility: software design that enables efficient use of optical circuits
– Applicability: application performance over a hybrid network
## Optical circuit switching v.s. Electrical packet switching

<table>
<thead>
<tr>
<th></th>
<th>Electrical packet switching</th>
<th>Optical circuit switching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switching technology</td>
<td>Store and forward</td>
<td>Circuit switching</td>
</tr>
<tr>
<td>Switching capacity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Switching time</td>
<td></td>
<td>e.g. MEMS optical switch</td>
</tr>
</tbody>
</table>

- **Electrical packet switching**
  - Store and forward
  - e.g. MEMS optical switch

- **Optical circuit switching**
  - Circuit switching
  - e.g. Calient FiberConnect

- **Electrical packet switching**
  - At high end: 16x40Gbps e.g. Cisco CRS
  - 32x100Gbps on market, e.g. Calient FiberConnect

- **Switching capacity**
  - Less than 10ms e.g. MEMS optical switch
Optical Circuit Switch

- Does not decode packets
- Needs take time to reconfigure

Output 1
Output 2
Input 1

Glass Fiber Bundle
Lenses
Fixed Mirror

Rotate Mirror
Mirrors on Motors
## Optical circuit switching v.s. Electrical packet switching

<table>
<thead>
<tr>
<th></th>
<th>Electrical packet switching</th>
<th>Optical circuit switching</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Switching technology</strong></td>
<td>Store and forward</td>
<td>Circuit switching</td>
</tr>
<tr>
<td><strong>Switching capacity</strong></td>
<td>16x40Gbps at high end e.g. Cisco CRS-1</td>
<td>320x100Gbps on market, e.g. Calient FiberConnect</td>
</tr>
<tr>
<td><strong>Switching time</strong></td>
<td>Packet granularity</td>
<td>Less than 10ms</td>
</tr>
<tr>
<td><strong>Switching traffic</strong></td>
<td>For bursty, uniform traffic</td>
<td>For stable, pair-wise traffic</td>
</tr>
</tbody>
</table>

- **Packet granularity** for bursty, uniform traffic.
- **Circuit switching** for stable, pair-wise traffic.

*Electrical packet switching* is used for bursty, uniform traffic, while *optical circuit switching* is better suited for stable, pair-wise traffic.
Optical circuit switching is promising despite slow switching time

- [IMC09][HotNets09]: “Only a few ToRs are hot and most their traffic goes to a few other ToRs. …”
- [WREN09]: “…we find that traffic at the five edge switches exhibit an ON/OFF pattern…”

Full bisection bandwidth at packet granularity may not be necessary
Hybrid packet/circuit switched network architecture

Optical circuit-switched network for **high capacity** transfer

Electrical packet-switched network for **low latency** delivery

Optical paths are provisioned rack-to-rack
- A simple and cost-effective choice
- Aggregate traffic on per-rack basis to better utilize optical circuits
Design requirements

Control plane:
- Traffic demand estimation
- Optical circuit configuration

Data plane:
- Dynamic traffic de-multiplexing
- Optimizing circuit utilization (optional)
c-Through (a specific design)

- No modification to applications and switches
- Leverage end-hosts for traffic management
- Centralized control for circuit configuration
c-Through - traffic demand estimation and traffic batching

1. Transparent to applications.
2. Packets are buffered per-flow to avoid HOL blocking.

Accomplish two requirements:
- Traffic demand estimation
- Pre-batch data to improve optical circuit utilization
c-Through - optical circuit configuration

- Use Edmonds’ algorithm to compute optimal configuration
- Many ways to reduce the control traffic overhead
c-Through - traffic de-multiplexing

VLAN-based network isolation:
- No need to modify switches
- Avoid the instability caused by circuit reconfiguration

Traffic control on hosts:
- Controller informs hosts about the circuit configuration
- End-hosts tag packets accordingly
Augmenting via Wireless Links

• Key benefit: on-demand links
  – Create links on-the-fly at congestion hotspots
  – Adapt to traffic dynamics

• New wireless technology: 60 GHz beamforming
  – Multi-Gbps data rate
  – Small interference footprint
Our Goal:
Any-to-any Communication

• Traffic hotspots can appear between any rack pair
  → Connect any rack pair wirelessly

Hard to do using existing 60GHz beamforming!
Challenge #1: Link Blockage

• 60GHz transmissions are blocked by small obstacles (anything larger than 2.5mm!)

• Confirmed by our testbed measurements
  – Signal strength dropped by 10-30dB
  – Up to 15-90% throughput loss

Must use multi-hop forwarding
  – Antenna rotation delay
  – Reduce throughput by at least half
Challenge #2: Radio Interference

• Beam interferes with racks in its direction
  – Exacerbated by dense rack deployment
  – Signal leakage makes it worse

• Verified via testbed measurements
  – A single link causes 15-20dB drop in signal quality for 15 nearby links

😊 Links interfere with each other
  – Very few links can run concurrently
  – Put a hard limit on aggregate bandwidth
3D Beamforming

Connect racks by reflecting signal off the ceiling!
Overview

• Data Center Topologies Scheduling

• Data Center Packet Scheduling
Datacenters and OLDIs

- OLDI = OnLine Data Intensive applications
  - e.g., Web search, retail, advertisements
- An important class of datacenter applications
- Vital to many Internet companies

OLDIs are critical datacenter applications
OLDIs

• Partition-aggregate
  • Tree-like structure
  • Root node sends query
  • Leaf nodes respond with data

• Deadline budget split among nodes and network
  • E.g., total = 200 ms, parents-leaf RPC = 30 ms

• Missed deadlines
  → incomplete responses
  → affect user experience & revenue
Challenges Posed by OLDIs

Two important properties:

1) **Deadline bound** (e.g., 200 ms)
   - Missed deadlines affect revenue

2) **Fan-in bursts**
   - Large data, 1000s of servers
     - Tree-like structure (high fan-in)
     - Fan-in bursts $\rightarrow$ long “tail latency”
   - Network shared with many apps (OLDI and non-OLDI)

Network must meet deadlines & handle fan-in bursts
Current Approaches

**TCP**: deadline agnostic, long tail latency
- Congestion $\rightarrow$ timeouts (slow), ECN (coarse)

**Datacenter TCP (DCTCP)** [SIGCOMM '10]
- first to *comprehensively* address tail latency
- Finely vary sending rate based on *extent* of congestion
- shortens tail latency, but is *not* deadline aware
  - $\sim 25\%$ missed deadlines at high fan-in & tight deadlines

DCTCP handles fan-in bursts, but is not deadline-aware
D²TCP

Deadline-aware and handles fan-in bursts

Key Idea: Vary sending rate based on both deadline and extent of congestion

- Built on top of DCTCP
- Distributed: uses per-flow state at end hosts
- Reactive: senders react to congestion
  - no knowledge of other flows
**D²TCP’s Contributions**

1) **Deadline-aware** and handles **fan-in bursts**
   - *Elegant gamma-correction* for congestion avoidance
     - far-deadline → back off more
     - near-deadline → back off less
   - Reactive, decentralized, state (end hosts)

2) **Does not** hinder long-lived (non-deadline) flows

3) **Coexists** with TCP → incrementally deployable

4) **No** change to switch hardware → deployable today

D²TCP achieves 75% and 50% fewer missed deadlines than DCTCP and D³
Coflow Definition

A collection of parallel flows

*Distributed* endpoints

*Each flow is independent*

Completion time depends on *the last flow* to complete

---

Coflow: A Networking Abstraction for Cluster Applications, HotNets’2012
How to schedule coflows ... … for faster completion of coflows? #1 completion of coflows? #1 to meet more deadlines? #2
Enables coflows in data-intensive clusters

1. Simpler Frameworks
   Zero user-side configuration using a simple coflow API

2. Better performance
   Faster and more predictable transfers through coflow scheduling
Benefits of Inter-Coflow Scheduling

Coflow 1

Link 2
3 Units

Link 1

Coflow 2
6 Units
3-ε Units

Fair Sharing

Flow-level Prioritization

The Optimal

Coflow 1 comp. time = 6
Coflow 2 comp. time = 6

Coflow 1 comp. time = 3
Coflow 2 comp. time = 6

Inter-Coflow Scheduling

Concurrent Open Shop Scheduling
- Tasks on independent machines
- Examples include job scheduling and caching blocks
- Use a ordering heuristic

Varys

Employs a two-step algorithm to minimize coflow completion times

1. Ordering heuristic
   Keeps an ordered list of coflows to be scheduled, preempting if needed

2. Allocation algorithm
   Allocates minimum required resources to each coflow to finish in minimum time
Ordering Heuristic: **SEBF**

<table>
<thead>
<tr>
<th></th>
<th>$C_1$</th>
<th>$C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Width</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Size</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>Bottleneck</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

- **Shortest-First**
- **Narrowest-First**
- **Smallest-First**
- **Smallest-Effective-Bottleneck-First**
A coflow cannot finish before its very last flow.

Finishing flows faster than the bottleneck cannot decrease a coflow’s completion time.

Ensure minimum allocation to each flow for it to finish at the desired duration; for example, at bottleneck’s completion, or at the deadline.
Varys

Enables frameworks to take advantage of coflow scheduling

1. Exposes the coflow API
2. Enforces through a centralized scheduler
Data Center Summary

• Topology
  • Easy deployment/costs
  • High bi-section bandwidth makes placement less critical
  • Augment on-demand to deal with hot-spots

• Scheduling
  • Delays are critical in the data center
  • Can try to handle this in congestion control
  • Can try to prioritize traffic in switches
  • May need to consider dependencies across flows to improve scheduling
Review

- Networking background
  - OSPF, RIP, TCP, etc.
- Design principles and architecture
  - E2E and Clark
- Routing/Topology
  - BGP, powerlaws, HOT topology
Review

• Resource allocation
  • Congestion control and TCP performance
  • FQ/CSFQ/XCP

• Network evolution
  • Overlays and architectures
  • Openflow and click
  • SDN concepts
  • NFV and middleboxes

• Data centers
  • Routing
  • Topology
  • TCP
  • Scheduling