IEEE ANTS 2012 Tutorial

Data Center Networking

Malathi Veeraraghavan
Charles L. Brown Dept. of Elec. & Comp. Engr.
University of Virginia
Charlottesville, VA 22904-4743, USA
mvee@virginia.edu
http://www.ece.virginia.edu/mv

Jogesh K. Muppala
Dept. of Computer Sc. and Engr.
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
muppala@cse.ust.hk
http://www.cse.ust.hk/~muppala/
Tutorial Outline

• Part I: Background, Topologies and Research Literature
  – Jogesh Muppala

• Part II: Current Technologies: Protocols
  – Malathi Veeraraghavan
Tutorial Part I Outline

- Introduction to Data Center Networks
- Data Centers Networks Requirements
- Data Center Network Topologies
  - The Real World
  - Research and Academic Proposals
Introduction to Data Center Networks
Tutorial Part I Outline

- Introduction to Data Center Networks
  - Data Centers Networks Requirements
  - Data Center Network Topologies
    - The Real World
    - Research and Academic Proposals
Cloud and Data Centers

- **Cloud: The Next Generation of Large-Scale Computing**
  - Infrastructure as a Service (IaaS)
  - Platform as a Service (PaaS)
  - Software as a Service (SaaS)

- **Cloud needs support of large-scale elastic data centers**
  - Massive number of servers
  - Massive amount of storage
  - Orchestrated together with a Data Center Network
  - Virtual Machine support

- **Example: Google and its Services**
Inside Google’s Data Center
A Server Room in Council Bluffs, IA Data Center
Inside Google’s Data Center
A Campus Network Room in Council Bluffs, IA Data Center
Inside Google’s Data Center

Central Cooling Plant in Google’s Douglas County, GA Data Center
Data Center Application Requirements

• Data centers typically run two types of applications
  – outward facing (e.g., serving web pages to users)
  – internal computations (e.g., MapReduce for web indexing)

• Workloads often unpredictable:
  – Multiple services run concurrently within a DC
  – Demand for new services may spike unexpectedly

    • Spike of demands for new services mean success!
    • But this is when success spells trouble (if not prepared)!

• Failures of servers are the norm
Data Center Costs [Greenberg 2008]

- Total cost varies
  - upwards of $1/4 B for mega data center
  - server costs dominate
  - network costs significant

- Long provisioning timescales:
  - new servers purchased quarterly at best

<table>
<thead>
<tr>
<th>Amortized Cost*</th>
<th>Component</th>
<th>Sub-Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>~45%</td>
<td>Servers</td>
<td>CPU, memory, disk</td>
</tr>
<tr>
<td>~25%</td>
<td>Power infrastructure</td>
<td>UPS, cooling, power distribution</td>
</tr>
<tr>
<td>~15%</td>
<td>Power draw</td>
<td>Electrical utility costs</td>
</tr>
<tr>
<td>~15%</td>
<td>Network</td>
<td>Switches, links, transit</td>
</tr>
</tbody>
</table>
Overall Data Center Design Goal [Greenberg 2008]

- Agility – Any service, Any Server
- Turn the servers into a single large fungible pool
  - Let services “breathe” : dynamically expand and contract their footprint as needed
    - We already see how this is done in terms of Google’s GFS, BigTable, MapReduce
  - Equidistant end-points with non-blocking core
  - Unlimited workload mobility
- Benefits
  - Increase service developer productivity
  - Lower cost
  - Achieve high performance and reliability

- These are the three motivators for most data center infrastructure projects!
Data Center Networks Requirements
Tutorial Part I Outline

• Introduction to Data Center Networks
  ➢ Data Centers Networks Requirements
• Data Center Network Topologies
  – The Real World
  – Research and Academic Proposals
Data Center Network Requirements

• Uniform high capacity
  – Capacity between servers limited only by their NICs
  – No need to consider topology when adding servers
    • => In other words, high capacity between any two servers no matter which racks they are located!

• Performance isolation
  – Traffic of one service should be unaffected by others

• Ease of management: “Plug-&-Play” (layer-2 semantics)
  – Flat addressing, so any server can have any IP address
  – Server configuration is the same as in a LAN
  – Legacy applications depending on broadcast must work
Data Center Network Requirements

• Requirements for scalable, easily manageable, fault tolerant and efficient Data Center Networks (DCN):
  – R1: Any VM may migrate to any physical machine without a change in its IP address
  – R2: An administrator should not need to configure any switch before deployment
  – R3: Any end host should efficiently communicate with any other end hosts through any available paths
  – R4: No forwarding loops
  – R5: Failure detection should be rapid and efficient

• Implication on network protocols:
  – A single layer2 fabric for entire data center (R1&R2)
  – Mac forwarding tables with hundreds of thousands entries (R3)
  – Efficient routing protocols which disseminate topology changes quickly to all points (R5)
Tutorial Part I Outline

• Introduction to Data Center Networks
• Data Centers Networks Requirements
  ➢ Data Center Network Topologies
    – The Real World
    – Research and Academic Proposals
A Typical Data Center Network

Data Center Network

North-South Traffic

East-West Traffic

TOR Switches

Server Racks
Typical Data Center Topology Today

- **Data Center Network topology:**
  - End hosts connects to top of rack (ToR) switches
  - ToR switches contains 48 GigE ports and up to 4 10 GigE uplinks
  - ToR switches connect to one or more end of row (EoR) switches

- **Forwarding:**
  - Layer 3 approach:
    - Assign IP addresses to hosts hierarchically based on their directly connected switch.
    - Use standard intra-domain routing protocols, eg. OSPF.
    - Large administration overhead
Typical Data Center Topology Today

– Layer 2 approach:
  • Forwarding on flat MAC addresses
  • Less administrative overhead
  • Bad scalability
  • Low performance

– Middle ground between layer 2 and layer 3:
  • VLAN
  • Feasible for smaller scale topologies
  • Resource partition problem

• End host virtualization:
  – Needs to support large addresses and VM migrations (e.g. vMotion)
  – In layer 3 fabric, migrating the VM to a different switch changes VM’s IP address
  – In layer 2 fabric, migrating VM incurs scaling ARP and performing routing/forwarding on millions of flat MAC addresses.
Full Mesh Network
Basic Tree Topology

- Core
- Aggregation
- Edge
- TOR Switches
- Server Racks
An Example from Cisco’s Recommendation

Internet

Core Router

Access Router

Data Center
Layer 3

Layer 2

Load Balancer

L2 Switches

TOR Switches

Server Racks

IEEE ANTS 2012 Tutorial Data Center Networking
An Example from Cisco’s Recommendation

- Hierarchical Network: 1+1 redundancy
- Equipment higher in the hierarchy handles more traffic, more expensive, more efforts made at availability ➔ scale-up design
- Servers connect via 1 Gbps UTP to Top of Rack switches
- Other links are mix of 1G, 10G; fiber, copper
Data Center Topologies: The Real World
Tutorial Part I Outline

- Introduction to Data Center Networks
- Data Centers Networks Requirements
  - Data Center Network Topologies
    - The Real World
      - Research and Academic Proposals
Clos Networks

- Charles Clos 1953
- Circuit Switching
- Non-blocking
- Multistage

Source: http://upload.wikimedia.org/wikipedia/en/9/9a/Closnetwork.png
Clos Networks

- 3-stage Clos
Clos Networks

- Folded Clos: Leaf and Spine
An Example Clos Network

Source: https://s3.amazonaws.com/bradhedlund2/2012/40G-10G-leaf-spine/clos-40G.png
Clos Networks

- 5-stage Clos
Clos Networks

- Folded Clos: Leaf and Spine
Data Center Fabric

- Abstraction of the data center network into a single orchestrated entity
  - Definition depends on different vendors, defined to their convenience
- Typical attributes of a fabric: Open (standards based), Scalable (cost and power), Intelligent (auto discovery), Secure (isolation and virtual network support), Resilient (fault-tolerant, stateless), Flexible (auto provisioning), Modular
Some Terms You Will Hear

- Shortest Path Bridging (SPB)
- Virtual Chassis
- Multi-chassis link Aggregation (MLAG)
- Transparent Interconnection of Lots of Links (TRILL)
Data Center Fabric Example: Juniper QFabric
Data Center Fabric Example: Juniper QFabric

- **QFabric Node (QFX 3500):**
  - 1U ToR ultra low latency, high port density (48 10GbE ports)
  - Layer 2/3 switch: packet forwarding, QoS, ACL management
  - 4 x 40 GbE links to interconnect

- **QFabric Interconnect:**
  - High-speed 3-stage Clos 10Tbps interconnection
  - 21U eight-slot (16 40GbE ports per slot)
  - Four interconnects per QFabric switch

- **QFabric Director:**
  - Distributed control plane running multiple routing engines
  - 2U server (x86 based with 36GB mem, 4TB storage)
  - Dedicated 1GbE network for control to nodes and interconnects
Data Center Fabric Example: Juniper QFabric

Single logical switch
Data Center Topologies: Research and Academic
Tutorial Part I Outline

• Introduction to Data Center Networks
• Data Centers Networks Requirements
  ➢ Data Center Network Topologies
    – The Real World
    ✓ Research and Academic Proposals
DCN Topology Taxonomy

Data Center Networks

Fixed Topology
- Tree-based
  - Basic tree
  - Fat tree
  - Clos network
- Recursive
  - DCell
  - BCube
  - MDCube
  - FiConn

Flexible Topology
- Fully Optical
  - OSA
- Hybrid
  - c-Through
  - Helios
DCN Topology Taxonomy

Data Center Networks

Fixed Topology

Tree-based

Basic tree Fat tree Clos network

DCell BCube MDCube FiConn

Recursive

Flexible Topology

Fully Optical

OSA

Hybrid

c-Through Helios
Tree-based Topologies

- Single port on each computer
- Scale up by adding more levels of switches or more ports on switches
- Basic tree, fat tree, Clos Network, ...
- Also can be classified as switch-centric topologies
Fat-Tree Topology
Fat-Tree Topology

- **Fat-Tree**: a special type of Clos Network
  - K-ary fat tree: three-layer topology (edge, aggregation and core)
  - Split fat tree into k pods
  - Each pod consists of \((k/2)^2\) servers & 2 layers of \(k/2\) k-port switches
  - Each edge switch connects to \(k/2\) servers & \(k/2\) aggr. switches
  - Each aggr. switch connects to \(k/2\) edge & \(k/2\) core switches
  - \((k/2)^2\) core switches: each connects to \(k\) pods
  - Each pod supports non-blocking operation among \((k/2)^2\) hosts
  - Each source and destination have \((k/2)^2\) paths
Fat-Tree Topology

- **Fat-Tree Properties**
  - Identical bandwidth at any bisection
  - Each layer has the same aggregated bandwidth

- Can be built using cheap devices with uniform capacity
  - Each port supports same speed as end host
  - All devices can transmit at line speed if packets are distributed uniform along available paths

- Great scalability: k-port switch supports $k^2/4$ servers
Fat-tree Topology

- Layer 2 switch algorithm: data plane flooding!
- Layer 3 IP routing:
  - shortest path IP routing will typically use only one path despite the path diversity in the topology
  - if using equal-cost multi-path routing at each switch independently and blindly, packet re-ordering may occur; further load may not necessarily be well-balanced
  - Aside: control plane flooding!
Example: PortLand

- A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric
- Fat-tree network topology
- Uses a single “logical layer 2” data center network fabric that scales to millions of endpoints
- Runs on commodity switch hardware with unmodified hosts
PortLand Design

• Design:
  – Fabric manager
  – Positional pseudo MAC addresses
  – Proxy-based ARP
  – Distributed Location Discovery
  – Loop free forwarding
  – Fault tolerant routing
PortLand Addressing

- Internally separates host identity from host location
  - uses IP address as host identifier
  - introduces “Pseudo MAC” (PMAC) addresses internally to encode endpoint location
- PMAC: balance the pros and cons of flat- vs. topology-dependent addressing
- PMACs are “topology-dependent,” hierarchical addresses
  - But used only as “host locators,” not “host identities”
  - IP addresses used as “host identities” (for compatibility w/ apps)
PortLand Addressing

• Pros and Cons:
  ✓ Small switch state & Seamless VM migration
  ✓ “Eliminate” flooding in both data & control planes
  – But requires a IP-to-PMAC mapping and name resolution
    • a location directory service
    – Location discovery protocol & fabric manager
    • for support of “plug-&-play”
Positional Pseudo MAC Addresses

- Pseudo MAC (PMAC) addresses encodes the location of the host
  - 48-bit: pod.position.port.vmid
  - Pod (16 bit): pod number of the edge switch
  - Position (8 bit): position in the pod
  - Port (8 bit): the port number it connects to
  - Vmid (16 bit): VM id of the host
PMAC Addressing Scheme

- PMAC (48 bits): pod.position.port.vmid
  - Pod: 16 bits; position and port (8 bits); vmid: 16 bits
- Assign only to servers (end-hosts) – by switches
Distributed Location Discovery

- Switches periodically send Location Discovery Message (LDM) out all of their ports to set their positions and to monitor liveness.
- LDM contains: switch identifier, pod number, position, tree level, up/down.
- Find position number for edge switch:
  - Edge switch randomly proposes a value in [0, k/2-1] to all aggregation switches in the same pod.
  - If it is verified as unused and not tentatively reserved, the proposal is finalized.
- Find tree level and up/down state:
  - Port states: disconnected, connected to end host, connected to another switch.
  - A switch with at least half of ports connected to end hosts is an edge switch, and it infers on subsequent LDM that the corresponding incoming port is upward facing.
  - A switch getting LDM from edge switch is aggregation switch and corresponding incoming port is downward facing port.
  - A switch with all ports connecting to aggregation switch is core switch, all ports are downward.
Location Discovery Protocol

<table>
<thead>
<tr>
<th>Switch Identifier</th>
<th>Pod Number</th>
<th>Position</th>
<th>Tree Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0:B1:FD:56:32:01</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
</tbody>
</table>
PortLand: Name Resolution

- Edge switch listens to end hosts, and discover new source MACs
- Installs <IP, PMAC> mappings, and informs fabric manager
PortLand: Name Resolution

- Edge switch intercepts ARP messages from end hosts
- send request to fabric manager, which replies with PMAC

ARP replies contain only PMAC

<table>
<thead>
<tr>
<th>Address</th>
<th>HWtype</th>
<th>HWAddress</th>
<th>Flags</th>
<th>Mask</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5.1.2</td>
<td>ether</td>
<td>00:00:01:02:00:01</td>
<td>C</td>
<td></td>
<td>eth1</td>
</tr>
</tbody>
</table>
Fabric Manager

• Characteristics:
  – Logically centralized user process running on a dedicated machine
  – Maintains soft state about network configuration information
  – Responsible for assisting with ARP resolution, fault tolerance and multicast

• Why centralized?
  – Eliminate the need for administrator configuration
PortLand: Fabric Manager

- Fabric manager: logically centralized, multi-homed server
- Maintains topology and \( \text{<IP,PMAC>} \) mappings in “soft state”

<table>
<thead>
<tr>
<th>IP</th>
<th>Pseudo MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5.1.2</td>
<td>00:00:01:02:00:01</td>
</tr>
<tr>
<td>10.2.4.5</td>
<td>00:02:00:02:00:01</td>
</tr>
</tbody>
</table>

![Network map]

**ARP mappings**

**Soft state**

**Administrator configuration**
Portable Loop Free Forwarding

- Forwarding based on PMAC (pod.position.port.vmid):
  - Core switches get pod value from PMAC, and send to corresponding port
    - Core switches learn the pod number of directly-connected aggregation switches
  - Aggregation switches get the pod and position value, if in the same pod, send to the port correspond to the position value, if not, send to the core switch
    - Aggregation switches learn the position number of all directly connected edge switches
Loop-free Forwarding and Fault-Tolerant Routing

- Switches build forwarding tables based on their position
  - edge, aggregation and core switches
- Use strict “up-down semantics” to ensure loop-free forwarding
  - Load-balancing: use any ECMP path via flow hashing to ensure packet ordering
- Fault-tolerant routing:
  - Mostly concerned with detecting failures
  - Fabric manager maintains logical fault matrix with per-link connectivity info; inform affected switches
  - Affected switches re-compute forwarding tables
Clos Network

- Folded Clos: Leaf and Spine
Example: VL2

- **Main Goal:** support agility & be cost-effective
- A virtual (logical) layer 2 architecture for connecting racks of servers (network as a big “virtual switch”)
  - employs a 3-level Clos topology (full-mesh in top-2 levels) with non-uniform switch capacities
- Also provides identity and location separation
  - “application-specific” vs. “location-specific” addresses
  - employs a directory service for name resolution
  - but needs direct host participation (thus mods at servers)
- Explicitly accounts for DC traffic matrix dynamics
  - employs the Valiant load-balancing (VLB) technique
    - using randomization to cope with volatility
VL2 Topology Design

• Scale-out vs. scale-up
• Argue for and exploit the gap in switch-to-switch capacity vs. switch-to-server capacities
  – Current: 10Gbps vs. 1Gbps; future: 40 Gpbs vs. 10 Gbps
• A scale-out design with broad layers
  – E.g., a 3-level Clos topology with full-mesh in top-2 levels
    • ToR switches, aggregation switches & intermediate switches
    • Less wiring complexity, and more path diversity
  – Same bisection capacity at each layer
    • No oversubscription
  – Extensive path diversity
    • Graceful degradation under failure
VL2 Topology: Example

Node degree (D) of available switches & # servers supported

<table>
<thead>
<tr>
<th>D</th>
<th># Servers in pool</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>80</td>
</tr>
<tr>
<td>24</td>
<td>2,880</td>
</tr>
<tr>
<td>48</td>
<td>11,520</td>
</tr>
<tr>
<td>144</td>
<td>103,680</td>
</tr>
</tbody>
</table>

[D^2/4] * 20 Servers
VL2: Valiant Load Balancing

- Use Randomization to Cope with Volatility
  - Every flow “bounced” off a random intermediate switch
  - Provably hotspot free for any admissible traffic matrix
  - Servers could randomize flow-lets if needed
**VL2 Summary**

- VL2 achieves agility at scale via
  - L2 semantics
  - Uniform high capacity between servers
  - Performance isolation between services

- Lessons
  - Randomization can tame volatility
  - Add functionality where you have control
  - There’s no need to wait!
DCN Topology Taxonomy

Data Center Networks

Fixed Topology

Tree-based

Basic tree

Fat tree

Clos network

Recursive

DCell

BCube

MDCube

FiConn

Flexible Topology

Fully Optical

OSA

c-Through

Helios

Hybrid
Recursive Topologies

• Multiple ports on each server
  – Servers act as both computation nodes, and
  – Implement traffic forwarding functionality among its ports
• Scale up by adding more ports on servers and switches
• FiConn, DCell, BCube, MDCube, ...
• Also can be viewed as Server-centric Topologies
DCell

DCell₀
BCube
BCube

- Main Goal: network architecture for shipping-container based modular data centers
- Designed for shipping-container modular DC
- BCube construction: level structure
  - $\text{BCube}_k$ recursively constructed from $\text{Bcube}_{k-1}$
- server-centric:
  - servers perform routing and forwarding
- Consider a variety of communication patterns
  - one-to-one, one-to-many, one-to-all, all-to-all
  - single path and multi-path routing
BCube Topology Construction

- Recursive structure: $\text{BCube}_k$ is recursively constructed from $n$ $\text{BCube}_{k-1}$ and $nk$ $n$-port switches

$n=4$
One-to-All Traffic Forwarding

- Using a spanning tree
- Speed-up: $L/(k+1)$ for a file of size $L$

Two-edge disjoint (server) spanning trees in BCube$_1$ for one-to-traffic
MDCube
<table>
<thead>
<tr>
<th>Degree of Computers</th>
<th>Tree-based Topology</th>
<th>Recursive Topology</th>
<th>CamCube [23]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diameter</td>
<td>(2\log_{n-1} N)</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>No. of Switches</td>
<td>(\frac{n^2 + n + 1}{n}) (N)</td>
<td>(\frac{4N}{n^2} (n + 1))</td>
<td>(\frac{3}{2^n} + \frac{3^2}{4^n})</td>
</tr>
<tr>
<td>No. of Wires</td>
<td>(\frac{n}{n-1} (N - 1))</td>
<td>(N \log_2 \frac{N}{2})</td>
<td>(N + \frac{n^4}{w_{ToR}})</td>
</tr>
<tr>
<td>No. of Computers</td>
<td>((n - 1)^k)</td>
<td>(\frac{n^3}{4})</td>
<td>(n^2 \frac{n}{4} \times n_{ToR})</td>
</tr>
</tbody>
</table>

\(^{1}\) Typically \(k\) is smaller for DCell because it needs smaller \(k\) to connect the same number of computers compared to other recursive topologies.

\(^{2}\) \(d\) is the number of computers on each dimension of CamCube.
Comparisons – Number of Computers

Table 2: Number of Computers

<table>
<thead>
<tr>
<th>n</th>
<th>Tree-based Topology</th>
<th>Recursive Topology</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic Tree</td>
<td>Fat-tree</td>
</tr>
<tr>
<td>4</td>
<td>64</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>216</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4,096</td>
<td>1,024</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>110,592</td>
<td>27,648</td>
</tr>
</tbody>
</table>
## Comparisons - Performance

### Table 3: Performance Summary

<table>
<thead>
<tr>
<th></th>
<th>Tree-based Topology</th>
<th>Recursive Topology</th>
<th>CamCube [23]</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-to-one</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>One-to-several</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>One-to-all</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>All-to-all</td>
<td>$n$</td>
<td>$N$</td>
<td>$\frac{2N}{nT_{oR}}$</td>
</tr>
<tr>
<td>Bisection Width</td>
<td>$\frac{n}{2}$</td>
<td>$\frac{N}{2}$</td>
<td>$\frac{N}{nT_{oR}}$</td>
</tr>
</tbody>
</table>
Comparisons – Hardware Redundancy

Table 4: Hardware Redundancy Summary

<table>
<thead>
<tr>
<th>Redundancy Level</th>
<th>Switch</th>
<th>Edge/ToR</th>
<th>Aggregation</th>
<th>Core/Intermediate</th>
<th>Computer</th>
<th>Tree-based Topology</th>
<th>Recursive Topology</th>
<th>CamCube</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node-disjoint Paths</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 or 2</td>
<td>1</td>
<td>1 or 2</td>
<td>k + 1</td>
<td>k + 1</td>
</tr>
<tr>
<td>Edge-disjoint Paths</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 or 2</td>
<td>1</td>
<td>k + 1</td>
<td>k + 1</td>
<td>D</td>
</tr>
<tr>
<td>Redundancy Level</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>k</td>
<td>k</td>
<td>D - 1</td>
</tr>
</tbody>
</table>

1. D is the dimension of MDCube. Only the inter-container connections are considered for MDCube.
2. \( n_{core} \) is the number of core switches in the tree.
Routing Techniques

- **Addressing**
  - Related to the topology
  - Implemented on different layers

- **Routing function distribution**
  - Distributed
  - Centralized

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Address Level</strong></td>
<td>IP</td>
<td>MAC/IP</td>
<td>MAC/IP</td>
<td>MAC/IP</td>
<td>MAC/IP</td>
<td>MAC &amp; IP</td>
<td>MAC &amp; IP</td>
</tr>
<tr>
<td><strong>Address Resolution</strong></td>
<td>Distributed</td>
<td>Distributed</td>
<td>Distributed</td>
<td>Distributed</td>
<td>Distributed</td>
<td>Centralized</td>
<td>Centralized</td>
</tr>
<tr>
<td><strong>Route Decision</strong></td>
<td>Intermediate</td>
<td>Source</td>
<td>Source</td>
<td>Source</td>
<td>Source</td>
<td>Intermediate</td>
<td>Intermediate</td>
</tr>
<tr>
<td><strong>Compatibility to Different Topologies</strong></td>
<td>Specialized</td>
<td>Specialized</td>
<td>Specialized</td>
<td>Specialized</td>
<td>Specialized</td>
<td>Compatible</td>
<td>Compatible</td>
</tr>
</tbody>
</table>
Performance Enhancement

- Utilize the hardware redundancy to get better performance
- Flow scheduling
- Multipath routing

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reassign Paths for Large Flows</td>
<td>Fair Queueing</td>
<td>Metric-based Routing</td>
<td>Metric-based Routing</td>
<td>Random Path Selection</td>
<td>Simulated Annealing</td>
<td>Random Path Selection</td>
<td></td>
</tr>
<tr>
<td>Decision Maker of Multi-path Selection</td>
<td>Central Scheduler</td>
<td>$F$ Function</td>
<td>Intermediate Nodes</td>
<td>Source Node</td>
<td>Intermediate Nodes</td>
<td>Edge Switches</td>
<td>Source Node</td>
</tr>
</tbody>
</table>
Fault Tolerance

- Utilize the hardware redundancy to maintain performance in presence of failures

<table>
<thead>
<tr>
<th>Table 7: Fault Tolerant Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
</tr>
<tr>
<td>Failure Types</td>
</tr>
<tr>
<td>Failure Detecting Mechanisms</td>
</tr>
<tr>
<td>Area of Spreading Failure Information</td>
</tr>
</tbody>
</table>
DCN Topology Taxonomy

Data Center Networks

Fixed Topology
- Tree-based
  - Basic tree
  - Fat tree
  - Clos network
- Recursive
  - DCell
  - BCube
  - MDCube
  - FiConn

Flexible Topology
- Fully Optical
  - OSA
- Hybrid
  - c-Through
  - Helios
Optical and Hybrid Data Center Networks

• Problems with Fixed Topology Networks:
  – Inflexibility to traffic characteristics: Attempt to provide uniform high capacity between all the servers ➔ Underutilization
  – Bandwidth oversubscription
  – Higher bit-rate links ➔ Copper-wire links limited in distance due to cost and power requirements
• Is Optical switching the saviour?
• Two approaches
  – Hybrid Optical/Electrical switched networks: c-Through, Helios
  – Fully optical networks: OSA
Optical Circuit Switch

- Glass Fiber Bundle
- Lenses
- Fixed Mirror
- Rotating Mirrors on Motors

- Does not decode packets
- Needs take time to reconfigure
# 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>
c-Through

Optical Switch

Core

Aggregation

ToR
c-Through

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

Optical circuit-switched network for high capacity transfer

Electrical packet-switched network for low latency delivery
c-Through 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

Applications

Per-rack traffic demand vector

Socket buffers

Transparent to applications.

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
Helios
OSA
OSA Architecture

[Diagram of OSA Architecture showing optical switching matrix, couplers, WSS, MUX, DEMUX, and server racks connected to ToR1, ToRi, and ToRj.]
Part I Summary

- We reviewed typical Data Center Network requirements
- We looked at the current DCN approaches in the real world
- We have waded through the world of DCN current and future topologies
  - Taxonomy of the topologies
    - Fixed Topology: Electrical switching
    - Flexible Topology: Optical and Hybrid switching
- How do we orchestrate these topologies to accomplish what we set out to do? Move data between the servers
  - Protocols