HULA: Scalable Load Balancing Using Programmable Data Planes

Naga Katta\textsuperscript{1}

Mukesh Hira\textsuperscript{2}, Changhoon Kim\textsuperscript{3}, Anirudh Sivaraman\textsuperscript{4}, Jennifer Rexford\textsuperscript{1}

Datacenter Load Balancing

- Multiple network paths
- High bisection bandwidth
- Volatile traffic
- Multiple tenants
A Good Load Balancer

• Multiple network paths
  – Track path performance
  – Choose best path
• High bisection bandwidth
• Volatile traffic
• Multiple tenants
A Good Load Balancer

• Multiple network paths
  – Track path performance
  – Choose best path

• High bisection bandwidth
  – Fine grained load balancing

• Volatile traffic

• Multiple tenants
A Good Load Balancer

• Multiple network paths
  – Track path performance
  – Choose best path
• High bisection bandwidth
  – Fine grained load balancing
• Volatile traffic
  – In-dataplane
• Multiple tenants
A Good Load Balancer

• Multiple network paths
  – Track path performance
  – Choose best path

• High bisection bandwidth
  – Fine grained load balancing

• Volatile traffic
  – In-dataplane

• Multiple tenants
  – In-network
Datapath LB: Challenges
Datapath LB: Challenges

- Large FIBs
- Per-link Congestion Measurement
- Congestion Feedback
- Tracks all paths
- Flowlet Detection
- Source Leaf
- LB Decision
- Spine Tier
- Destination Leaf
- Overlay Network
Datapath LB: Challenges

- Large FIBs
- Per-link Congestion Measurement
- Custom ASIC
- Tracks all paths
- Source Leaf
- Gap
- LB Decision
- Destination Leaf
- Spine Tier
- Overlay Network
- Lowlet Detection
- Tracks all paths
Programmable Commodity Switches

• Vendor agnostic
  – Uniform programming interface (P4)
  – Today’s trend → cheaper

• Reconfigurable in the field
  – Adapt or add dataplane functionality

• Examples
  – RMT, Intel Flexpipe, Cavium Xpliant, etc.
Programmable Switches - Capabilities

P4 Program

Compile

Memory

M A
m1 a1

Ingress Parser

Queue Buffer

Egress Deparser

More than OF 1.X
Programmable Switches - Capabilities

More than OF 1.X

Programmable Parsing

P4 Program

Compile

Ingress Parser

Queue Buffer

Egress Deparser
Programmable Switches - Capabilities

- Programmable Parsing
- Compile
- More than OF 1.X

Ingress Parser

Queue Buffer

Egress Deparser

Match-Action Pipeline
Programmable Switches - Capabilities

Programmable Parsing

P4 Program

Compile

More than OF 1.X

Register Array

Ingress Parser

Queue Buffer

Egress Deparser

Match-Action Pipeline

Memory

M

A

m1

a1

Memory

M

A

m1

a1

Memory

M

A

m1

a1

Memory

M

A

m1

a1

P4 Program

Compile

More than OF 1.X

Register Array

Ingress Parser

Queue Buffer

Egress Deparser

Match-Action Pipeline

Memory

M

A

m1

a1

Memory

M

A

m1

a1

Memory

M

A

m1

a1

Memory

M

A

m1

a1
Programmable Switches - Capabilities

Programmable Parsing

Ingress Parser

Switch Metadata

Queue Buffer

Match-Action Pipeline

Egress Deparser

Memory

P4 Program

Compile

More than OF 1.0

Stateful Memory

M

A

m1

a1

M

A

m1

a1

M

A

m1

a1

M

A

m1

a1
Hop-by-hop Utilization-aware Load-balancing Architecture

• Distance-vector like propagation
  – Periodic probes carry path utilization

• Each switch chooses best downstream path
  – Maintains only best next hop
  – Scales to large topologies

• Programmable at line rate
  – Written in P4.
### HULA: Scalable and Programmable

<table>
<thead>
<tr>
<th>Objective</th>
<th>P4 feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probe propagation</td>
<td>Programmable parsing</td>
</tr>
<tr>
<td>Monitor path performance</td>
<td>Link state metadata</td>
</tr>
<tr>
<td>Choose best path, route flowlets</td>
<td>Stateful memory and comparison operators</td>
</tr>
</tbody>
</table>
Probes carry path utilization

Probes originate

Probes replicate

Aggregates

Spines

ToR
Probes carry path utilization

- Probes originate
- Probes replicate

P4 primitives
- New header format
- Programmable Parsing
- RW packet metadata
Probes carry path utilization

ToR ID = 10
Max_util = 80%

ToR ID = 10
Max_util = 60%

ToR ID = 10
Max_util = 50%

ToR 1

S1

S3

S4

ToR 10
Each switch identifies best downstream path

Best hop table

<table>
<thead>
<tr>
<th>Dst</th>
<th>Best hop</th>
<th>Path util</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>S4</td>
<td>50%</td>
</tr>
<tr>
<td>ToR 1</td>
<td>S2</td>
<td>10%</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

ToR ID = 10
Max_util = 50%
Switches load balance flowlets

Flowlet table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Timestamp</th>
<th>Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>1</td>
<td>S4</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

Best hop table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Best hop</th>
<th>Path util</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>S4</td>
<td>50%</td>
</tr>
<tr>
<td>ToR 1</td>
<td>S2</td>
<td>10%</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
Switches load balance flowlets

Flowlet table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Timestamp</th>
<th>Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>1</td>
<td>S4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Best hop table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Best hop</th>
<th>Path util</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>S4</td>
<td>50%</td>
</tr>
<tr>
<td>ToR 1</td>
<td>S2</td>
<td>10%</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

P4 primitives

- RW access to stateful memory
- Comparison/arithmetic operators
Switches load balance flowlets

Flowlet table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Timestamp</th>
<th>Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>1</td>
<td>S4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Best hop table

<table>
<thead>
<tr>
<th>Dest</th>
<th>Best hop</th>
<th>Path util</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToR 10</td>
<td>S4</td>
<td>50%</td>
</tr>
<tr>
<td>ToR 1</td>
<td>S2</td>
<td>10%</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

P4 code snippet

```p4
if(curr_time - flowlet_time[flow_hash] > THRESH) {
    flowlet_hop[flow_hash] = best_hop[packet.dst_tor];
}
metadata.nxt_hop = flowlet_hop[flow_hash];
flowlet_time[flow_hash] = curr_time;
```
Evaluated Topology

- **S1** and **S2** are the top-level switches.
- **A1**, **A2**, **A3**, and **A4** connect to **S1** and **S2**.
- **L1**, **L2**, **L3**, and **L4** are the leaf-level switches.
- **40Gbps** links between **S1** and **A1**, **A2**, **A3**, and **A4**.
- **10Gbps** links between **L1**, **L2**, **L3**, and **L4**.
- **8 servers per leaf**.
- Link Failure indicated between **S1** and **S2**.

- **26**
Evaluation Setup

• NS2 packet-level simulator
• RPC-based workload generator
  – Empirical flow size distributions
  – Websearch and Datamining
• End-to-end metric
  – Average Flow Completion Time (FCT)
Compared with

- ECMP
  - Flow level hashing at each switch
- CONGA’
  - CONGA within each leaf-spine pod
  - ECMP on flowlets for traffic across pods

1. Based on communication with the authors
HULA handles high load much better
HULA keeps queue occupancy low
HULA is stable on link failure
HULA - Summary

• **Scalable** to large topologies
  • HULA distributes congestion state

• **Adaptive** to network congestion

• **Proactive** path probing

• **Reliable** when failures occur

• **Programmable** in P4!
Backup
# HULA: Scalable, Adaptable, Programmable

<table>
<thead>
<tr>
<th>LB Scheme</th>
<th>Congestion aware</th>
<th>Application agnostic</th>
<th>Dataplane timescale</th>
<th>Scalable</th>
<th>Programmable dataplanes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SWAN, B4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPTCP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONGA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HULA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>