Network Fabric to enable Resource Disaggregation in Datacenters

Vishal Shrivastav (Cornell University)

A – Exam

Nov 6, 2017
Common theme:
Hardware Software co-design for building next generation datacenter network
Outline

- Problem Definition, Motivation and Challenges
- Shoal: Network Fabric for Rack-scale Architecture
- Ongoing and Future Work
Part I: Problem Definition, Motivation and Challenges
What is Resource Disaggregation?

- Decouple compute, memory, storage, peripheral (FPGA, GPU)
  - Communication over the network fabric

Physical disaggregation

Virtual disaggregation
What is Resource Disaggregation?

- A real world disaggregated FPGA fabric

[Microsoft Build 2017]
Why Resource Disaggregation?

- More efficient (fine granular) resource provisioning
  - Reduced resource fragmentation

A new app arrives

Enough resource available, but still can’t allocate to the new app!!
Why Resource Disaggregation?

6 FPGAs now handling the load from 22 software instances
Efficiency gain of 73%

Unused FPGAs

[Microsoft Build 2017]
Why Resource Disaggregation?

[Microsoft Build 2017]

- 6 FPGAs now handling the load from 22 software instances
- Efficiency gain of 73%
- Freed capacity allocated to new services, e.g. DNN
Why Resource Disaggregation?

- Communication to *faster remote memory/storage* can take less time than *slower local storage*.

```
<table>
<thead>
<tr>
<th>Delay (ns)</th>
<th>Delay (μs)</th>
<th>Delay (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>10000</td>
<td>10000</td>
<td>100</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>10</td>
</tr>
</tbody>
</table>
```

Baseline RTT = 20us

![Diagram of DCN with CPU and storage devices showing latency times](image)
Why Resource Disaggregation?

- Communication to *faster remote memory/storage* can take less time than *slower local storage*
  - e.g. Memory Intensive Applications

Using remote memory instead of local storage can give 12X better performance [Gu et al. *NDSI’17*]
Why Resource Disaggregation?

• Seamless scaling of both capacity and bandwidth
  
• Physical decoupling allows each resource technology to evolve independently
  – Reduces time-to-adoption
Resource Disaggregation: a holistic view
Resource Disaggregation: a holistic view

FOCUS OF THIS TALK

Networking research:
High performant fabric

PL/SYS research:
New programming models and APIs

System/File system/Architecture research:
Novel memory hierarchy, memory and storage protocols etc.

Memory and Storage Management, isolation
N/W OS
N/W driver

OS research:
Distributed resource management, isolation

Application

Computing

Disaggregated Workload Characteristics

• Comprises memory, storage, peripheral atop IP traffic
  – “Converged” traffic -- high network load

• “Small” flows account for significant bandwidth
  – Memory flows can be as small as a cache-line (64B)
  – Storage flows can be as small as a block size (4KB)

• Skewed traffic pattern, high degree of incast
  – Shared resource (mem, storage, FPGA etc.) likely to be hotspots
  – Analysis in [Gao et al. OSDI’16] shows >84% of flows generated among 1/3rd of nodes
Ideal N/W Requirements for Disaggregation

Across any traffic pattern and flow size distribution, simultaneously achieve

• Low and predictable latency
  – Much smaller latencies than traditional DC workloads
  – Tight bounds on the worst-case performance, small variance

• High network utilization
  – “Converged” traffic increases the network load significantly

• Lossless fabric
  – Packet loss at high loads significantly reduces flow throughput
  – RDMA for low latency requires losslessness for high performance
Ideal N/W Requirements for Disaggregation

Across any traffic pattern and flow size distribution, simultaneously achieve

- Tight bounds on the worst-case performance
- High network utilization

Key Insight:
Tightly coupling end host (network stack) with network fabric (dataplane)

- Packet loss at high loads significantly reduces flow throughput
- RDMA for low latency requires losslessness for high performance
Unit of Disaggregation

- **Rack**
  - ~2m
  - 10ns + 300ns
  - Limited resources

- **Pod**
  - Few 10s of m
  - 200ns + 1us
  - More resources to pool

- **Datacenter**
  - Few km
  - 10us + 2us
  - Entire datacenter resource
Thesis Question

How to build a network fabric carrying the “converged” traffic that can simultaneously achieve low latency, high network utilization and bounded queuing, across any traffic pattern and flow size distribution?

- Building the fabric to enable disaggregation at rack scale
- Building the fabric to enable disaggregation across multiple racks
Part II:
Shoal: Network Fabric for Rack-scale Architecture

Joint with
Asaf Valadarsky (The Hebrew University of Jerusalem)
Hitesh Ballani, Paolo Costa (Microsoft Research Cambridge)
Ki Suh Lee, Han Wang, Rachit Agarwal, Hakim Weatherspoon (Cornell)
Rack-scale Architecture (RSA)

• High-density, disaggregated racks
  – High density achieved via power and space efficient SoC designs
  – Disaggregated processing (CPU, TPU, GPGPU), memory, storage
  – Interconnected via a very high capacity (50-100+ Tbps) fabric

• Racks new unit of computation – “rack scale computer”
  – Provide the abstraction of a standalone (super-)computer
  – Shipped as a single unit (pre-assembled at the manufacturer site)
  – Easy to integrate and manage
  – More compute power (performance) per $
Rack-scale Architecture (RSA)

- High-density, disaggregated tracks
  - High density achieved via power and space efficient SoC designs
  - Disaggregated processing (CPU, TPU, GPGPU), memory, storage
  - Interconnected via a very high capacity (50-100+ Tbps) fabric

- Racks as a new unit of computation
  - “rack scale computer”
  - Provide the abstraction of a standalone (super)computer
  - Shipped as a single unit (pre-assembled at the manufacturer site)
  - Easy to integrate and manage
  - More compute power (performance) per $22

Challenge:
Building a high capacity network fabric subject to rack’s power and space constraints
How to build a very high capacity network fabric that can accommodate high density and efficiently support resource disaggregation within a rack?

- Small power, space consumption to enable high compute density
- Low latency
- High utilization
- Lossless fabric
Building a high capacity RSA Network Fabric

- **Strawman Designs -**
  
  1. **ToR Chasis Switch**
     - High cost, power and space
     - E.g. 20 Tbps CISCO 7700 switch consumes 26% and 54% of rack’s power and space

  2. **Clos of Packet Switches**
     - Large port count for full bisection bandwidth
     - E.g. 25 Tbps bisection bandwidth using 32x100 Gbps Mellanox switches consumes 56% of rack’s power
Building a high capacity RSA Network Fabric

- Strawman Designs -

1. ToR Chasis Switch
   - High cost, power and space
   - 20 Tbps CISCO 7700 switch consumes 26% power and 54% space of a track

Packet switches consume too much power and space, making them unsuitable for achieving high density

- Large port count for full bisection bandwidth
- 25 Tbps bisection bandwidth using 32x100 Gbps Mellanox switches consumes 56% of rack power
State-of-the-art RSA Network Fabric

- **Direct connect fabric -- R2C2** [Costa et al. *SIGCOMM’15]*
  - Oversubscribed
  - Multi-hop routing
  - Higher, less predictable latency
  - Static topology

- **Circuit switch fabric -- XFabric** [Legtchenko et al. *NSDI’16]*
  - Central controller
  - Multi-hop routing
  - High reconfiguration overhead
topology calculation \(\sim 1\text{sec}\)
dataplane reconfig \(\sim 1\text{-}10\text{ms}\)
Lesson learned

• Circuit switched fabric is well suited for RSA
  ✓ Extremely power efficient
  ✓ Reconfigurable fabric
  ✓ Small reconfiguration delay (~2ns)
  ✓ Minimal switching delay

Right compromise between resource consumption & flexibility
Shoal: Fabric

- Clos of circuit switches

Challenges for building an efficient circuit switch fabric:
1. No central controller to set up circuits
   - Scalable, no scheduling and reconfiguration overhead
2. Small path length (# of hops)
   - Less wasted throughput, smaller latency
Shoal: Fabric Design Insight

- Abstract the circuit switch fabric as one giant switch
  - Behaves like a typical input-buffered crossbar switch

- Crossbar reconfigured using dynamic matching e.g. i-SLIP
- Need a control plane

Key Observation:
If the input traffic is uniformly load-balanced, a static permutation matching will achieve 100% throughput
Shoal: Fabric Design Insight

- Load-balanced Birkhoff-von Neumann switch
  
  [Chang et al. Comp Comm’02]

  - Crossbar reconfigured using static permutation matching
  - No control plane needed
  - 100% throughput
Shoal: Fabric Design Insight

- Load balancing via a crossbar
  - Uniform load balancing can be achieved using a static permutation matching

![Diagram of Stage I and Stage II with an example matching matrix]

<table>
<thead>
<tr>
<th>Time slot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7 8</td>
</tr>
<tr>
<td>1 2 3 4 5 6 7 8</td>
</tr>
<tr>
<td>... ... ... ... ... ... ... ...</td>
</tr>
<tr>
<td>8 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>
Shoal: Going from a switch to a fabric

Fabric schedule -- very tight synchronization

Individual switch schedule

---

1 2 3 4 5 6 7
a d c d c d c d b
c d d d c d c
b a b a b a b
a b a b a b

Time slot

1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
3 4 5 6 7 8 1
8 1 2 3 4 5 6 7
Shoal: Routing

Packets are sent as fixed size cells

Flow 1 -> 4
Shoal: Routing Properties

• No central controller for scheduling

• Each packet traverses the fabric *at most* twice
  – Max path length of 2 (1 hop)
  – Bounded worst-case network utilization of 50%
    (for full permutation traffic matrix e.g. 1->2, 2->3, 3->4, 4->1)

• Provision network with 2X capacity
  – Cost < equivalent pkt sw fabric with 1X capacity
    *circuit switches are cheap* !!
  – Power (50Tbps) < 72% of 25Tbps pkt sw fabric
    *circuit switches are power efficient* !!
Shoal: Congestion Control

**Invariant:**
Each queue holds *at most* 1 packet from any given flow

**Results in optimal worst-case queue bound**

- **For local flows**
  - Schedule next packet from flow only after the previous packet has been transmitted (FIFO ordering) -- **Bound of N-1**

- **For remote flows**
  - Leverage periodic connections set up by the fabric between any two nodes -- **Bound of N-2**

1 -> 2
Send pkt

2 -> 1
Recv feedback

1 -> 2
Send(?) nxt pkt

and so on ....
Shoal: Congestion Control

Invariant:
Each queue holds *atmost* 1 packet from any given flow

Results in optimal worst-case queue bound

- For local flows
  - Schedule next packet from flow only after the previous packet has been transmitted (FIFO ordering) -- **Bound of N-1**

Queue length bound:
No queue grows larger than 2N-3 across *any* traffic pattern

any two nodes -- **Bound of N-2**

<table>
<thead>
<tr>
<th>1 -&gt; 2</th>
<th>2 -&gt; 1</th>
<th>1 -&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send pkt</td>
<td>Recv feedback</td>
<td>Send(?) nxt pkt</td>
</tr>
</tbody>
</table>

and so on ....
Shoal: Implementation

Terasaic Altera DE5-Net boards with Stratix V FPGA

- Used Bluespec System Verilog (BSV)
- ~1000 LOCs
Shoal: Evaluation

- **Baseline**: Ideal Packet Switch Network (PSN)
  - Per packet uniform load balancing
  - Per flow queuing at switches
  - Per flow hop-by-hop flow control

- Shoal provisioned with 2X bandwidth compared to PSN

- Shoal uses 256B cell size
  - Guardband = 0.5 (sync) + 2 (reconfig) = 2.5ns (10% overhead)

- How far away is Shoal from PSN in terms of
  - Latency for short flows (<100KB) ?
  - Throughput for long flows (>1MB) ?
Shoal: Evaluation

99-th perc. FCT (us)

Big Data Benchmark
GraphLab
Memcached
Wordcount
Terasort
Spark

Goodput (Gbps)

Big Data Benchmark
GraphLab
Memcached
Wordcount
Terasort
Spark
Shoal: Summary

- Extremely power efficient, enabling high density
  - 50 Tbps fabric consumes just 15% of rack’s total power

- Low latency fabric
  - Within 2X of an ideal fabric across varied workloads
  - Further improve by using smaller cell sizes

- High network utilization
  - Bounded 50% worst-case utilization -- Provision with 2X capacity

- Achieves Fairness & Bounded queuing (lossless fabric)
  - No queue grows larger than 2N-3 cells across any traffic pattern
Part III:
Ongoing and Future Work
Shoal-v2

• Reduce latency further by using novel virtual topology
  • Shoal’s virtual topology a fully connected graph
    • Diameter = 1
    • Degree = N-1
    • Epoch len = N-1

• A different virtual topology
  • Diameter = 2
  • Degree = 2(√N-1)
  • Epoch len = 2(√N-1)

Reduces latency at the cost of throughput. A hybrid solution??
How to build a network fabric carrying the “converged” traffic that can simultaneously achieve low latency, high network utilization and bounded queuing, across any traffic pattern and flow size distribution?

- Building the fabric to enable disaggregation at rack scale
  - Shoal, Shoal-v2(?)
- Building the fabric to enable disaggregation across multiple racks
Going beyond a Rack: Pros and Cons

**Rack scale**

- Small physical distance
  - Very small propagation delay
  - Can use copper cables
    - Less $ and power
  - Enables custom technology
    - Direct connect
    - Circuit switching
  - Shipped as a single unit
    - Easy mgmt & integration

- Limited power/space budget
  - Can only fit so much resource
  - Hard to accommodate power hungry accelerators like GPU

**Pod scale**

- Larger physical distance
  - Higher propagation delay
    - Still not too high
  - Will need to use optic fiber
    - More $ and power
  - Challenging to enable custom technology that don’t scale

- Larger power/space budget
  - More resources to pool
  - Can better accommodate accelerators like GPU

**Pros outweigh cons**
Network Fabric across Disaggregated Racks

Joint with
Hitesh Ballani, Paolo Costa (Microsoft Research Cambridge)
Rachit Agarwal, Hakim Weatherspoon (Cornell)
Why not use circuit switches like Shoal?

- Achieving tight synchronization beyond a rack is hard!
  - Necessary for any co-ordination free scheduler for circuit sw
  - Need uniform wire lengths etc.
  - Less synchronization precision →
    - Higher throughput overhead (for small slot sizes)
    - Higher latency (for large slot sizes)

Use Packet Switches

Will need a very efficient transport protocol
How to design a transport protocol that can provide low latency, high utilization and bounded queuing, across any traffic pattern and flow size distribution, at the scale of a pod?
Review: Datacenter Transport Protocols

### Congestion prevention

- **Centralized**
  - Fastpass
    - ✓ Zero network queuing
    - ✗ Central scheduling
  - QJump
    - ✓ Bounded queuing
    - ✗ Pessimistic queuing estimation

- **Decentralized**
  - DCQCN, TIMELY
    - ✓ Lossless by design
    - ✗ Relies on PFC – collateral damage, deadlocks

### Congestion detection & recovery

- Sender Push
  - TCP, DCTCP, HULL
    - ✓ Low protocol overhead
    - ✓ Rate convergence over multiple RTTs
    - ✗ Can’t mitigate flow syncs even in subsequent RTTs
    - ✗ Lossy by default

- Receiver Pull
  - EyeQ, pHost, NDP
    - ✓ Convergence in a RTT
    - ✓ Can mitigate synced flow effects in a RTT
    - ✗ High protocol overhead
    - ✗ Prone to underutilization
    - ✗ Lossy by default

### Lossless

- pFabric, PDQ, D²TCP
  - ✓ Reduced in-network interference
  - ✗ Rely on applications to specify priority
  - ✗ Not useful in high priority flow incast

### In-network Priority
Review: Datacenter Transport Protocols

Congestion prevention

Centralized
- Fastpass
  - Not scalable, high scheduling overhead
- Central scheduling

Decentralized
- Slump
  - Fundamental trade-off between low latency and high utilization

Congestion detection & recovery

Sender Push
- TCP, DCTCP, HUL
  - Not optimized for incast
  - Rate convergence over multiple RTTs
  - Can't mitigate flow syncs even in subsequent RTTs
  - Lossy by default

Receiver Pull
- High overhead for short flow, possible link underutilization
- Can mitigate synced flow
- Lossy by default

Lossless
- DCQCN, TIMELY
  - Low network utilization at high loads
  - Damage, deadlocks

In-network Priority
- pFabric, PDQ, D²TCP
  - Not useful in event of short flow (high priority) incast
  - Not useful in high priority flow incast
  - Convergence in a RTT
  - Can mitigate synced flow effects in a RTT
  - High protocol overhead
  - Prone to underutilization
  - Lossy by default
  - High overhead for short flow, possible link underutilization
  - Can't mitigate flow syncs even in subsequent RTTs

Fundamental trade-off between low latency and high utilization estimation.

Even small amounts of packet loss at high loads cause disproportionate slowdown in flow throughput [Alizadeh et al. NSDI’12]

Even small amounts of packet loss at high loads cause disproportionate slowdown in flow throughput [Alizadeh et al. NSDI’12]
School: Fabric across disaggregated racks

Forward cells to destination based on corresponding leaf’s schedule (needs synchronization b/w leaf & spine)

Exactly one point of queuing along the path

n spines with r ports each
r*n virtual queues at each spine

Full visibility of congestion

Leaf-pulled protocol

Congestion control loop

forward cells based on a static permutation schedule
• Uniform per pkt load balancing on upstream
• No in-network contention on downstream

Bounded worst-case queuing (N cells) and latency (N*slot size) at 100% utilization across any traffic pattern
Broader Picture and Next Steps

Broadly interested in datacenter networking

1. Dataplane programmability
   - P4FPGA [SOSR 2017]

2. End host functionalities
   - Highly scalable QoS in hardware [ONGOING]

3. Datacenter services
   - Clock synchronization protocol (DTP) [SIGCOMM 2016]

4. Network fabric and transport protocols
   - Shoal [submitted to NSDI 2018]
   - Shoal-v2 [ONGOING]
   - School [submit to SIGCOMM 2018]
Broader Picture and Next Steps

Broader interested in datacenter networking

1. Dataplane programmability
   - P4FPGA [SOSR 2017]

2. End host functionalities
   - Highly scalable QoS in hardware [ONGOING]

3. Datacenter services
   - Clock synchronization protocol (DTP) [SIGCOMM 2016]

4. Network fabric and transport protocols
   - Shoal [submitted to NSDI 2018]
   - Shoal-v2 [ONGOING]
   - School [submit to SIGCOMM 2018]

Common theme:
Hardware Software co-design for building next generation datacenter network
Conclusion

- Resource disaggregation promises to greatly improve the efficiency of datacenters
- Network will be a key enabling (or blocking) factor for disaggregation
- State-of-the-art network fabric falls short to meet the requirements of disaggregation
- Shoal presents a novel network fabric for disaggregated, high-density racks
- School proposes a novel network fabric across multiple disaggregated racks
Thank you!