HPCC++: Enhanced High Precision Congestion Control

draft-miao-iccrг-hpccplus-00

Rui Miao, Hongqiang Harry Liu, Rong Pan, Jeongkeun Lee, Changhoon Kim, Barak Gafni, Yuval Shpigelman

IETF-112 rtgwg
Nov 2021
Cloud desires hyper-speed networking

Today, clouds have

- bigger data to compute & store
- faster compute & storage devices
- more types of compute and storage resources

High-performance storage

- Storage-compute separation is norm
- HDD→SSD→NVMe
- Higher-throughput, lower latency
- 1M IOPS / 50~100us

High-performance computation

- Distributed deep learning, HPC
- CPU→GPU, FPGA, ASIC
- Faster compute, lower latency
- E.g. latency <10us

Resource disaggregation

- More network load
- Need ultra-lower latency: 3-5us, > 40Gbps (Gao Et.al. OSDI’16)
Hyper-speed network chips != hyper-speed networking

Hardware-offloading (e.g., RDMA)
Traditional software-based networking stacks cannot keep with the speed

Congestion control (CC)
Since, end hosts are aggressive, network is more vulnerable to congestion & packet loss
Realistic challenges in CC in high-speed networks

• Operation challenge-1: PFC storm & deadlock
  ➢ Running lossy networks is desired, but there is a convergence challenge!!!

• Operation challenge-2: running multiple applications
  ➢ QoS queues are scarce resources!!!

• Operation challenge-3: complex parameter tuning
  ➢ DCQCN has at least 15 parameters to tune!!!
In-band telemetry

- New commodity ASICs have in-band telemetry ability
- Mainly used for trouble-shooting purposes
Can we use in-band telemetry as precise feedback for congestion control?

**HPCC++: Enhanced High Precision Congestion Control**
In-band telemetry format

- HPCC++ defines the algorithm of using telemetry information
  - including queue length, transmitted bytes, timestamp, link capacity, etc.
- Yet, the actual packet format is up to the environment

<table>
<thead>
<tr>
<th>bits</th>
<th>31-24</th>
<th>23-16</th>
<th>15-8</th>
<th>7-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>Device-ID</td>
</tr>
<tr>
<td>1</td>
<td>TID</td>
<td>congestion</td>
<td>Tx Bytes Cnt[39:32]</td>
<td>TTL</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>Rx Timestamp Sec - Upper</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>Rx Timestamp Sec</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>Rx Timestamp Nano</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>Tx Timestamp Nano</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>Src-Sys-Port</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td>Tx Bytes Cnt[31:0]</td>
</tr>
</tbody>
</table>

*Example format of in-band telemetry used by HPCC++*
HPCC++ solves all there problems

Using in-band telemetry as the precise feedback

- Fast convergence
  - Sender knows the precise rate to adjust to, on every ACK
- Near-zero queue
  - Feedback does not only rely on queue
- Few parameters
  - Precise feedback, so no need for heuristics which requires many parameters
Smart routing with HPCC++

- Route condition changes drastically in DC
  - e.g., incast, failure, re-routing, ...
- HPCC++ offers a precise “view” of a route’s capacity
- Joint decision on routing and traffic allocation by combining views of routes
HPCC++ achieves lower FCT and near-zero queue

- In testbed, vs. DCQCN (hardware-based, widely used in industry)
  - Web search traffic at 50% load
- Vs. other CC (unavailable in HW) in simulation. HPCC performs better

- 95% lower FCT
- 55% lower FCT
Thank You