Vision: Toward 10 Tbps NDN Forwarding with Billion Prefixes by Programmable Switches

2021/9/23

Junji Takemasa, Yuki Koizumi, Toru Hasegawa
Osaka University

This work has been supported by JSPS KAKENHI Grant Number JP20H04176.
Background

- **Goal**
  - 10-Tbps NDN router w/ billion prefixes FIB

- **Approach**
  - Leveraging *switching speed* of programmable switch and large *DRAM capacity* of commodity computers

- **Issue**
  - The DRAM accesses consume the switch’s bandwidth, thereby degrading the router’s throughput.
Key Idea

- **Forwarding Data packets by switch alone, Interest packets by computers**
  
  - Leveraging fast switching speed to accommodate a large amount of Data traffic
  
  - Leveraging large DRAM capacity of computers to store billion prefixes of FIB
Throughput Estimation

- **Result**: 10.8-Tbps throughput w/ 2 computers
- **Method**: calculating ideal number of computers $n$
  - $n$ determines the bottleneck in a router as well as throughput.
  - Large $n$ lacks bandwidth for Data forwarding $B^D [\text{bps}]$ (due to connection of wastefully many computers)
  - Small $n$ lacks computing capacity for Interest forwarding $n \times \lambda^I [\text{pps}]$
  - Condition for ideal $n$: $B^D \approx n \times \lambda^I \times S$ (S: Data size)
  - The *bandwidth* & the *computing capacity* are balanced.

*Details of estimation condition in Section 2 of the paper*
Packet Processing Design

- **Packet processing flows:**
  - **Interest flow**
    1. Looking up FIB of computer’s DRAM to decide the outgoing port
    2. Recording the incoming port at PIT of switch ASIC
  - **Data flow**
    1. Looking up and deleting the incoming port of Interest at PIT of switch ASIC

- **Challenge: PIT in switch ASIC**
  - Limited SRAM capacity & Arithmetic and Logic Units (ALUs)
Challenges of PIT in Switch ASIC

- **C1: store a few millions of names at SRAM**
  - $2^{21} \times 64 = 128$ MB $>> O(10)$-MB SRAM
  - **Design:** Hashing names of PIT at switch ASIC
    - Compressing 64-B name into about 21-bit name’s hash
    - Resolving hash collisions via name-based PIT at computers

- **C2: complete all the operations by one pipeline pass**
  - Packet re-circulation via loopback reduces bandwidth by half.
  - **Design:** Multi-staged pipeline layout for PIT
    - Splitting entry’s fields into distinct stages to fully leverage ALUs

D1: Hashing Names of PIT

- **Interest recording**
  - In-face is recorded at switch or computer.
    - At switch if same hash is not found.
    - At computer with name if same hash is found.
  - Number of pending Interests is counted for collision handling.
    - +1 when Interest is successfully forwarded.
    - –1 when Data is successfully forwarded.

I: /X/Y

<table>
<thead>
<tr>
<th>Hash</th>
<th>IFace</th>
<th>Pending</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x01</td>
<td>#1</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

I: /a/b

Switch’s PIT

<table>
<thead>
<tr>
<th>Name</th>
<th>IFace</th>
</tr>
</thead>
<tbody>
<tr>
<td>/a/b</td>
<td>#2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Computer’s PIT

\[*\text{Hash}(/X/Y) = \text{Hash}(/a/b)*\]
D1: Hashing Names of PIT

Data forwarding

- Switch (hash) can itself forward Data packet just after arrival of only one Interest of same hash.
  - The hash is obviously created by Interest of same name.
- Switch must involve computer (name) for Data packet after arrivals of 2 or more than Interests of same hash.
  - The hash cannot validate whether of same or different name.
  - Collision detection based on number of pending Interests at switch (details in the paper)
D2: Pipeline Layout

- **Splitting in-face & pending fields to 2 stages**
  - ALUs of a single stage is not enough to handle both fields.
- **Distributing hashes to multiple stages**
  - 10s of MB-capacity only results from SRAMs of all the stages.

![Pipeline Layout Diagram]

<table>
<thead>
<tr>
<th>Hash</th>
<th>IFace</th>
<th>Pending</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x03</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Switch’s PIT Diagram]
Prototype Implementation

- **Objectives**
  - Evaluation of throughput compared to Naïve
    - Naïve: running PIT and FIB at computer, dispatching packets to computer at switch
  - Validation of Data forwarding by switch alone

- **Router**: switch & 1 computer
  - Switch: Tofino ASIC & 32x 100 Gbps ports
  - Computer: 2x 22-cores CPUs & 2x 40 Gbps ports

- **Traffic generator**: up to 500 Gbps Interest-Data traffic by 3 computers

*Implementation details in Section 4 of the paper*
Results

**Total throughput**

- Proposal: nearly equal to the upper bound of traffic generator
- Naïve: limited by 80 Gbps bandwidth b/w switch & computer
  - Bandwidth consumption due to DRAM accesses for Data packets

<table>
<thead>
<tr>
<th>Router Architecture</th>
<th>bits/s</th>
<th>packets/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
<td>470 Gbps</td>
<td>94.4 MPPS</td>
</tr>
<tr>
<td>Naive</td>
<td>79 Gbps</td>
<td>15.8 MPPS</td>
</tr>
</tbody>
</table>

**Validation in Proposal**

- 98% of Data packets are forwarded by switch alone.
- The rest 2% are successfully forwarded via computer.

*Measurement condition (details in the paper)
1135-B Data, $2^{30}$ prefixes of FIB, $2^{21}$ hashes of PIT in SRAM
Summary & Open Issues

- **10-Tbps NDN router w/ billion prefixes FIB is feasible with a switch and a few computers.**
  - Data forwarding by switch alone for efficient bandwidth usage, whereas Interest forwarding with computer’s large DRAMs
  - Compact hash-based PIT structure for switch ASIC
  - 470 Gbps throughput in prototype router w/ one Tofino switch & one computer

- **Open issues (details in the paper)**
  - Tb-scale traffic generator (ccnGen in poster session)
  - Formal verification for PIT’s behavior
  - TLV handling in stateful parser of switch ASIC
  - Performance against unexpected traffic patterns