Topological Transformation Approaches to Optimizing TCAM-Based Packet Classification Systems

Chad Meiners, Alex Liu, Eric Torng
Department of Computer Science
Michigan State University
{meinersc, alexliu, torng}@cse.msu.edu
Background: Packet Classification

- Why do we need packet classification:
  - different service for different applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Filtering</td>
<td>Deny all udp traffic from 1.2.<em>.</em></td>
</tr>
<tr>
<td>Policy Routing</td>
<td>Send all VoIP traffic arriving from domain 1 on interface Y and destined to domain 2 via a separate ATM network</td>
</tr>
<tr>
<td>Accounting and Billing</td>
<td>Treat all video traffic to domain 1 as highest priority and perform accounting for the traffic</td>
</tr>
<tr>
<td>Traffic Rate Limiting</td>
<td>Ensure ISP2 doesn’t inject more than 30Mbps of e-mail and 100Mbps of total traffic on interface X</td>
</tr>
<tr>
<td>Traffic Shaping</td>
<td>Ensure no more than 100Mbps of web traffic is injected into ISP2 on interface X</td>
</tr>
</tbody>
</table>
Background (Access Control List)

Packet:

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Dst IP</th>
<th>Src Port</th>
<th>Dst Port</th>
<th>Protocol</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.3.20</td>
<td>192.168.0.1</td>
<td>6000</td>
<td>8000</td>
<td>TCP</td>
<td></td>
</tr>
</tbody>
</table>

Classifier:

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Dst IP</th>
<th>Src Port</th>
<th>Dst Port</th>
<th>Protocol</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.3.0/24</td>
<td>192.168.0.1</td>
<td>[1,65534]</td>
<td>[1,65534]</td>
<td>TCP</td>
<td>accept</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>discard</td>
</tr>
</tbody>
</table>

Conflict Resolution: first-match

Packet classification problem:
Background: TCAM-Based Packet Classification

- Ternary Content Addressable Memory (TCAM)
  - Traditional memory: input is the index, output is the entry
  - TCAM: input is the entry, output the index (or the entry)
  - Every bit is 0, 1, or * (don’t care)
- Search is performed against all entries in parallel and the first result (i.e., the index or the entry) is returned

**Search Key: 11101101**

<table>
<thead>
<tr>
<th>Search Key</th>
<th>TCAM</th>
<th>SRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00011000</td>
<td>accept</td>
</tr>
<tr>
<td>2</td>
<td>1110110*</td>
<td>discard</td>
</tr>
<tr>
<td>3</td>
<td>111***</td>
<td>discard</td>
</tr>
<tr>
<td>4</td>
<td>***</td>
<td>accept</td>
</tr>
</tbody>
</table>

- Widely deployed in routers and middle boxes
## Key issues of TCAM-Based Packet Classification

<table>
<thead>
<tr>
<th>Hardware Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Space</strong></td>
</tr>
<tr>
<td>– 36 Mbits limit</td>
</tr>
<tr>
<td>– Larger chips are slower</td>
</tr>
<tr>
<td><strong>Power Consumption</strong></td>
</tr>
<tr>
<td>– Affects the power budget</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
</tr>
<tr>
<td>– TCAMs cost more than NP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application Demands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classifier size</strong></td>
</tr>
<tr>
<td>– Growing</td>
</tr>
<tr>
<td><strong>Range expansion</strong></td>
</tr>
<tr>
<td>– Ranges to multiple prefixes</td>
</tr>
<tr>
<td>– Source and Destination Ports</td>
</tr>
<tr>
<td>– ([1,65534]) = 30 prefixes</td>
</tr>
<tr>
<td>– 30 * 30 = 900 rules</td>
</tr>
</tbody>
</table>

### Research Goal:
Minimize the number TCAM bits that represent the classifier
Encoded Classification

- Bit patterns in TCAM
  - Represent encoded classifier
  - Smaller
- Must encode header
  - Key = Encode(header)
  - TCAM(Key) = decision

How do we encode classifiers for TCAMs?
Range Encoding

• Idea
  – Reencode troublesome ranges
  – Two approaches
    • Database independent
      – Encoding function does not depend on the rules
    • Database dependent
      – Encoding function tailored to eliminate bad ranges

• Limitations of Prior Art
  – Focus on individual rules and ranges
  – Do not consider the rules’ decisions
Our approach: Classifier Encoding

• Idea
  – Reencode the whole classifier
    • Range intervals
    • Decisions on intervals
  – Use topological transformation functions
    • Hyperrectangle to better hyperrectangle
    • Map header to new hyperrectangle

• Benefits
  – Sub-linear compression
  – Reduced domain space
  – Excellent range encoding
Classifier Encoding via Topological Transformations

1. Domain Compression
   - Reduce rule size
     - 104 bits to less than 36
   - Reduce the number of rules
     - Eliminate equivalent regions
   - Prefix Alignment
     - Mitigate range expansion
     - Align intervals to prefix boundary
Domain Compression-Algorithm (1 of 3)

- Classifier with 4 rules and 3 decisions
- Map space to 3 points with 3 decisions

\[ [0,1] \rightarrow 0 \]
\[ [2,3] \rightarrow 1 \]
\[ [4,4] \rightarrow 2 \]
\[ [5,7] \rightarrow 0 \]

- \([0,1] \rightarrow \text{black}\)
- \([0,3] \rightarrow \text{red}\)
- \([4,4] \rightarrow \text{green}\)
- \([5,7] \rightarrow \text{black}\)
Domain Compression-Algorithm (2 of 3)

• How do we change the classifier rules

• For each decision in the mapping table
  – Choose a landmark

• Intervals become the [min, max] landmarks in rule

- [0,1] → black
- [0,3] → red
- [4,4] → green
- [5,7] → black

- [0,0] → black
- [0,1] → red
- [2,2] → green
- [-,-] → black
Domain Compression-Algorithm (3 of 3)

- [1,1],[2,3] -> blue
- [6,6],[2,3] -> blue
- [1,6],[4,4] -> yellow
- [4,5],[0,7] -> red
- [0,7],[0,7] -> grey

- How do we generalize?
- Construct FDD
- Roots show isomorphisms
Topological Transformation Approaches to Optimizing TCAM-Based Packet Classification Systems: Meiners, Liu, Torng
Can we do any better?

- Yes, we can
- The yellow, red, grey rules each take 2
- Let’s align them to the prefix boundaries
Prefix Alignment

- Prefix intervals
  - take a single TCAM
- Other intervals
  - Sum of prefix interval
- Stretch and shift intervals
  - Intervals into prefixes
- Dynamic Programming
  - Optimal for 1-D
Prefix Alignment

• Stretch interval to align with prefix boundaries

• Stretch y intervals
  – Fixes red rule
  – Fixes grey rule

• Stretch x intervals
  – Fixes yellow rule

• 4 TCAM rules
Prefix Alignment-Hill climbing

Use a combining dynamic programming and hill climbing

- Dynamic Programming
  - Optimal 1-D
- Hill climbing
  - Iterate over dimensions
  - Usually perfect
Domain Compression + Prefix Alignment

- Domain compression and prefix alignment compose

- 9 TCAM rules to 4

- How is this different?
- Semantic v.s. rule encoding
  - Blue rules
Architecture

• Transformers in TCAM
  – Parallel but small TCAMs
  – One TCAM --- multiple lookups

• Range expansion
  – No multiplication effect
  – DP finds a minimal prefix list

• Compact
  – All tables fits within 36 or 40 bits
Results

• For a data set of 25 real-world classifiers
  – On average
    • Encoded classifiers use 2.6 % of original TCAM
    • Transformers use 7.7 % of original TCAM
    • Total 10.3 %
  – Versus per work
    • Best range encode can achieve is 1.0 expansion ratio
    • We achieve on average an expansion ratio of 0.3
Conclusions

• We present the first classifier encoding method
• First step reduces the size and number of rules
• Second step mitigates ranges expansion

• We achieve sub-linear compression
Questions?

• Thank you for listening!
Domain Compression (1 of 3)

- Map isomorphic axis intervals to points
- Transformation function corrects header

\((2,4)\)  \(\Rightarrow\)  \(T_Y(y)\)  \((2,T_Y(4))\)

\((2,2)\)  \(\Rightarrow\)  \(\text{Yellow}\)
Domain Compression (2 of 3)

• Every axis is minimized

\[(2, T_Y(4))\]
\[(2, 2)\]

\[(T_X(2), T_Y(4))\]
\[(2, 2)\]
Domain Compression (3 of 3)

- 5 rules (2 blue)
- 9 TCAM rules
  - 2 red, 4 yellow
- 4 rules
- 7 TCAM rules
  - 2 yellow, 2 red, 2 grey
Domain Compression-Algorithm (2 of 2)

- Choose landmarks
- Convert rules
- Remove empty rules

$T_X(x)$
- $[0,0] \rightarrow 0$
- $[1,1] \rightarrow 1$
- $[2,3] \rightarrow 2$
- $[4,5] \rightarrow 3$
- $[6,6] \rightarrow 1$
- $[7,7] \rightarrow 0$

$T_Y(y)$
- $[0,1] \rightarrow 0$
- $[2,3] \rightarrow 1$
- $[4,4] \rightarrow 2$
- $[5,7] \rightarrow 0$

- $[1,1],[2,3] \rightarrow$ blue
- $[6,6],[2,3] \rightarrow$ blue
- $[1,6],[4,4] \rightarrow$ yellow
- $[4,5],[0,7] \rightarrow$ red
- $[0,7],[0,7] \rightarrow$ grey