15-744: Computer Networking

L-8 Routers

Outline
• IP router design
• IP route lookup
• Variable prefix match algorithms
• Alternative methods for packet forwarding

Forwarding and Routers
• Forwarding
• IP lookup
• High-speed router architecture
• Readings
  • [McK97] A Fast Switched Backplane for a Gigabit Switched Router
  • [KCY03] Scaling Internet Routers Using Optics
  • Know RIP/OSPF
• Optional
  • [D+97] Small Forwarding Tables for Fast Routing Lookups
  • [BV01] Scalable Packet Classification

IP Router Design
• Different architectures for different types of routers
• High speed routers incorporate large number of processors
• Common case is optimized carefully
What Does a Router Look Like?

- Currently:
  - Network controller
  - Line cards
  - Switched backplane

- In the past?
  - Workstation
  - Multiprocessor workstation
  - Line cards + shared bus

Line Cards

- Network interface cards
- Provides parallel processing of packets
- Fast path per-packet processing
  - Forwarding lookup (hardware/ASIC vs. software)

Network Processor

- Runs routing protocol and downloads forwarding table to line cards
  - Some line cards maintain two forwarding tables to allow easy switchover
- Performs “slow” path processing
  - Handles ICMP error messages
  - Handles IP option processing

Switch Design Issues

- Have N inputs and M outputs
  - Multiple packets for same output – output contention
  - Switch contention – switch cannot support arbitrary set of transfers
    - Crossbar
    - Bus
      - High clock/transfer rate needed for bus
    - Banyan net
      - Complex scheduling needed to avoid switch contention
- Solution – buffer packets where needed
Switch Buffering

- Input buffering
  - Which inputs are processed each slot – schedule?
  - Head of line packets destined for busy output blocks other packets
- Output buffering
  - Output may receive multiple packets per slot
  - Need speedup proportional to # inputs
- Internal buffering
  - Head of line blocking
  - Amount of buffering needed

Line Card Interconnect

- Virtual output buffering
  - Maintain per output buffer at input
  - Solves head of line blocking problem
  - Each of MxN input buffer places bid for output
- Crossbar connect
  - Challenge: map of bids to schedule for crossbar

ISLIP

Round 1, Iteration 1

REQUEST

Round 1, Iteration 2

REQUEST

Round 2, Iteration 1

REQUEST

Round 2, Iteration 2

REQUEST

Round 3, Iteration 1

REQUEST
What Limits Router Capacity?

Approximate power consumption per rack

Power density is the limiting factor today

Multi-rack Routers Reduce Power Density

Crossbar
Linecards
Switch
Linecards

Examples of Multi-rack Routers

Alcatel 7670 RSP
Juniper TX8/T640
Avici TSR
Chiaro

Limits to Scaling

- Overall power is dominated by linecards
  - Sheer number
  - Optical WAN components
  - Per packet processing and buffering.
- But power density is dominated by switch fabric
Multi-rack Routers Reduce Power Density

- Limit today ~2.5Tb/s
- Electronics
- Scheduler scales <2x every 18 months
- Opto-electronic conversion

Question
- Instead, can we use an **optical** fabric at 100Tb/s with 100% throughput?

- Conventional answer: **No**
  - Need to reconfigure switch too often
  - 100% throughput requires complex electronic scheduler.

If Traffic is Uniform...

Real Traffic is Not Uniform
Two-stage Load-Balancing Switch

Load-balancing stage

Switching stage

100% throughput for weakly mixing, stochastic traffic

[C.-S. Chang, Valiant]

Static WDM Switching

Array Waveguide Router (AWGR)
Passive and Almost Zero Power

4 WDM channels, each at rate 2R/N
Outline

- IP router design
- IP route lookup
- Variable prefix match algorithms
- Alternative methods for packet forwarding

Original IP Route Lookup

- Address classes
  - A: 0 | 7 bit network | 24 bit host (16M each)
  - B: 10 | 14 bit network | 16 bit host (64K)
  - C: 110 | 21 bit network | 8 bit host (255)
- Address would specify prefix for forwarding table
  - Simple lookup

Original IP Route Lookup – Example

- www.cmu.edu address 128.2.11.43
- Class B address – class + network is 128.2
- Lookup 128.2 in forwarding table
- Prefix – part of address that really matters for routing
- Forwarding table contains
  - List of class+network entries
  - A few fixed prefix lengths (8/16/24)
- Large tables
  - 2 Million class C networks
  - 32 bits does not give enough space encode network location information inside address – i.e., create a structured hierarchy
CIDR Revisited
- Supernets
  - Assign adjacent net addresses to same org
  - Classless routing (CIDR)
- How does this help routing table?
  - Combine routing table entries whenever all nodes with same prefix share same hop
  - Routing protocols carry prefix with destination network address
  - Longest prefix match for forwarding

CIDR Illustration
Provider is given 201.10.0.0/21

CIDR Shortcomings
- Multi-homing
- Customer selecting a new provider

Outline
- IP router design
- IP route lookup
- Variable prefix match algorithms
- Alternative methods for packet forwarding
Trie Using Sample Database

Sample Database
- P1 = 10*
- P2 = 111*
- P3 = 11001*
- P4 = 1*
- P5 = 0*
- P6 = 1000*
- P7 = 100000*
- P8 = 1000000*

How To Do Variable Prefix Match

- Traditional method – Patricia Tree
  - Arrange route entries into a series of bit tests
- Worst case = 32 bit tests
  - Problem: memory speed is a bottleneck

Speeding up Prefix Match (P+98)

- Cut prefix tree at 16 bit depth
  - 64K bit mask
  - Bit = 1 if tree continues below cut (root head)
  - Bit = 1 if leaf at depth 16 or less (genuine head)
  - Bit = 0 if part of range covered by leaf

Prefix Tree
Speeding up Prefix Match (P+98)

- Each 1 corresponds to either a route or a subtree
  - Keep array of routes/pointers to subtree
  - Need index into array – how to count # of 1s
  - Keep running count to 16bit word in base index + code word (6 bits)
  - Need to count 1s in last 16bit word
    - Clever tricks
  - Subtrees are handled separately

Speeding up Prefix Match - Alternatives

- Route caches
  - Temporal locality
  - Many packets to same destination
- Other algorithms
  - Waldvogel – Sigcomm 97
    - Binary search on prefixes
    - Works well for larger addresses
  - Bremler-Barr – Sigcomm 99
    - Clue = prefix length matched at previous hop
    - Why is this useful?
  - Lampson – Infocom 98
    - Binary search on ranges
**Speeding up Prefix Match - Alternatives**

- Content addressable memory (CAM)
  - Hardware based route lookup
  - Input = tag, output = value associated with tag
  - Requires exact match with tag
    - Multiple cycles (1 per prefix searched) with single CAM
    - Multiple CAMs (1 per prefix) searched in parallel
  - Ternary CAM
    - 0,1,don't care values in tag match
    - Priority (i.e. longest prefix) by order of entries in CAM

**Outline**

- IP router design
- IP route lookup
- Variable prefix match algorithms
- Alternative methods for packet forwarding

**Techniques for Forwarding Packets**

- Source routing
  - Packet carries path
- Table of virtual circuits
  - Connection routed through network to setup state
  - Packets forwarded using connection state
- Table of global addresses (IP)
  - Routers keep next hop for destination
  - Packets carry destination address

**Source Routing**

- List entire path in packet
- Driving directions (north 3 hops, east, etc.)
- Router processing
  - Examine first step in directions
  - Strip first step from packet
  - Forward to step just stripped off
Source Routing

- Advantages
  - Switches can be very simple and fast
- Disadvantages
  - Variable (unbounded) header size
  - Sources must know or discover topology (e.g., failures)
- Typical use
  - Ad-hoc networks (DSR)
  - Machine room networks (Myrinet)

Virtual Circuits/Tag Switching

- Connection setup phase
  - Use other means to route setup request
  - Each router allocates flow ID on local link
  - Creates mapping of inbound flow ID/port to outbound flow ID/port
- Each packet carries connection ID
  - Sent from source with 1st hop connection ID
- Router processing
  - Lookup flow ID – simple table lookup
  - Replace flow ID with outgoing flow ID
  - Forward to output port

Virtual Circuits Examples

Virtual Circuits

- Advantages
  - More efficient lookup (simple table lookup)
  - More flexible (different path for each flow)
  - Can reserve bandwidth at connection setup
  - Easier for hardware implementations
- Disadvantages
  - Still need to route connection setup request
  - More complex failure recovery – must recreate connection state
- Typical uses
  - ATM – combined with fix sized cells
  - MPLS – tag switching for IP networks
### IP Datagrams on Virtual Circuits

- **Challenge** – when to setup connections
  - At bootup time – permanent virtual circuits (PVC)
    - Large number of circuits
  - For every packet transmission
    - Connection setup is expensive
  - For every connection
    - What is a connection?
    - How to route connectionless traffic?

- **Traffic pattern**
  - Few long lived flows
  - Flow – set of data packets from source to destination
  - Large percentage of packet traffic
  - Improving forwarding performance by using virtual circuits for these flows
  - Other traffic uses normal IP forwarding

### Summary: Addressing/Classification

- Router architecture carefully optimized for IP forwarding
- Key challenges:
  - Speed of forwarding lookup/classification
  - Power consumption
- Some good examples of common case optimization
  - Routing with a clue
  - Classification with few matching rules
  - Not checksumming packets

### Open Questions

- Fanout vs. bandwidth
- MPLS vs. longest prefix match
- More vs. less functionality in routers
- Hardware vs. software
  - CAMs vs. software
- Impact of router design on network design
Global Addresses (IP)
- Each packet has destination address
- Each switch has forwarding table of destination \(\rightarrow\) next hop
  - At v and x: destination \(\rightarrow\) east
  - At w and y: destination \(\rightarrow\) south
  - At z: destination \(\rightarrow\) north
- Distributed routing algorithm for calculating forwarding tables

Global Address Example

Router Table Size
- One entry for every host on the Internet
  - 100M entries, doubling every year
- One entry for every LAN
  - Every host on LAN shares prefix
  - Still too many, doubling every year
- One entry for every organization
  - Every host in organization shares prefix
  - Requires careful address allocation

Global Addresses
- Advantages
  - Stateless – simple error recovery
- Disadvantages
  - Every switch knows about every destination
    - Potentially large tables
  - All packets to destination take same route
**Summary**

<table>
<thead>
<tr>
<th>Source Routing</th>
<th>Global Addresses</th>
<th>Virtual Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Size</td>
<td>Worst</td>
<td>OK – Large address</td>
</tr>
<tr>
<td>Router Table Size</td>
<td>None</td>
<td>Number of hosts (prefixes)</td>
</tr>
<tr>
<td>Forward Overhead</td>
<td>Best</td>
<td>Prefix matching</td>
</tr>
<tr>
<td>Setup Overhead</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Error Recovery</td>
<td>Tell all hosts</td>
<td>Tell all routers</td>
</tr>
</tbody>
</table>

**Binary Search on Ranges**

- Encode each prefix as range and place all range endpoints in binary search table or tree. Need two next hops per entry for > and = case. [Lampson, Srinivasan, Varghese]

- Problem: Slow search \( \log_2 N + 1 = 20 \) for a million prefixes and update \( O(n) \).
  - Some clever implementation tricks to improve on this

**Packet Classification**

- **Typical uses**
  - Identify flows for QoS
  - Firewall filtering

- **Requirements**
  - Match on multiple fields
  - Strict priority among rules
    - E.g.
      1. no traffic from 128.2.*
      2. ok traffic on port 80

**Skip Count vs. Path Compression**

- Removing one way branches ensures # of trie nodes is at most twice # of prefixes
- Using a skip count requires exact match at end and backtracking on failure \( \rightarrow \) path compression simpler
Complexity

• N rules and k header fields for k > 2
  • $O(\log N^{k-1})$ time and $O(N)$ space
  • $O(\log N)$ time and $O(N^k)$ space
  • Special cases for $k = 2$ → source and destination
    • $O(\log N)$ time and $O(N)$ space solutions exist

• How many rules?
  • Largest for firewalls & similar → 1700
  • Diffserv/QoS → much larger → 100k (?)

Bit Vectors

<table>
<thead>
<tr>
<th>Rule</th>
<th>Field1</th>
<th>Field2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00*</td>
<td>00*</td>
</tr>
<tr>
<td>1</td>
<td>00*</td>
<td>01*</td>
</tr>
<tr>
<td>2</td>
<td>10*</td>
<td>11*</td>
</tr>
<tr>
<td>3</td>
<td>11*</td>
<td>10*</td>
</tr>
</tbody>
</table>

Field 1

Field 2

Observations [GM99]

• Common rule sets have important/useful characteristics
  • Packets rarely match more than a few rules (rule intersection)
    • E.g., max of 4 rules seen on common databases up to 1700 rules
Aggregating Rules [BV01]
• Common case: very few 1’s in bit vector → aggregate bits
• OR together A bits at a time → N/A bit-long vector
  • A typically chosen to match word-size
  • Can be done hierarchically → aggregate the aggregates
• AND of aggregate bits indicates which groups of A rules have a possible match
  • Hopefully only a few 1’s in AND’ed vector
  • AND of aggregated bit vectors may have false positives
• Fetch and AND just bit vectors associated with positive entries

Rearranging Rules [BV01]
• Problem: false positives may be common
• Solution: reorder rules to minimize false positives
  • What about the priority order of rules?
• How to rearrange?
  • Heuristic → sort rules based on single field’s values
    • First sort by prefix length then by value
    • Moves similar rules close together → reduces false positives