Outline

• Where IP routers sit in the network
• What IP routers look like
• What do IP routers do?
• Some details:
  – The internals of a “best-effort” router
    • Lookup, buffering and switching
  – The internals of a “QoS” router
Outline (next time)

• The way routers are *really* built.
• Evolution of their internal workings.
• What limits their performance.
• The way the network is built today
Outline

• Where IP routers sit in the network
• What IP routers look like
• What do IP routers do?
• Some details:
  – The internals of a “best-effort” router
    • Lookup, buffering and switching
  – The internals of a “QoS” router
• Can optics help?
The Internet is a mesh of routers (in theory)
What do they look like?

Access routers
e.g. ISDN, ADSL

Core router
e.g. OC48c POS

Core ATM switch
Basic Architectural Components of an IP Router

Control Plane

Routing Protocols
Routing Table

Datapath

Forwarding Table Switching

per-packet processing
Per-packet processing in an IP Router

1. Accept packet arriving on an incoming link.
2. Lookup packet destination address in the forwarding table, to identify outgoing port(s).
3. Manipulate packet header: e.g., decrement TTL, update header checksum.
4. Send packet to the outgoing port(s).
5. Buffer packet in the queue.
6. Transmit packet onto outgoing link.
A General Switch Model

- **Input Receiver**
- **Input Buffer**
- **Interconnect**
- **Control Routing, Scheduling**
- **Output Buffer Transmitter**
- **Output Ports**
IP Switch Model

1. Ingress
   - Forwarding Table
   - Forwarding Decision

2. Interconnect

3. Egress
   - Forwarding Table
   - Forwarding Decision
Forwarding Engine

Packet
- payload
- header

Router

Routing Lookup Data Structure

Forwarding Table

<table>
<thead>
<tr>
<th>Dest-network</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>65.0.0.0/8</td>
<td>3</td>
</tr>
<tr>
<td>128.9.0.0/16</td>
<td>1</td>
</tr>
<tr>
<td>149.12.0.0/19</td>
<td>7</td>
</tr>
</tbody>
</table>
The Search Operation is *not* a Direct Lookup

IP addresses: 32 bits long $\Rightarrow 4G$ entries
The Search Operation is also not an Exact Match Search

Exact match search: search for a key in a collection of keys of the same length.

Relatively well studied data structures:

- Hashing
- Balanced binary search trees
### Example Forwarding Table

<table>
<thead>
<tr>
<th>Destination IP Prefix</th>
<th>Outgoing Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>65.0.0.0/8</td>
<td>3</td>
</tr>
<tr>
<td>128.9.0.0/16</td>
<td>1</td>
</tr>
<tr>
<td>142.12.0.0/19</td>
<td>7</td>
</tr>
</tbody>
</table>

**Prefix length**

**IP prefix: 0-32 bits**

![Diagram showing IP prefix ranges and outgoing ports](chart.png)
Prefixes can Overlap

Routing lookup: Find the longest matching prefix (the most specific route) among all prefixes that match the destination address.
<table>
<thead>
<tr>
<th>Prefixes</th>
<th>Prefix Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.9.16.0/21</td>
<td>32</td>
</tr>
<tr>
<td>128.9.172.0/21</td>
<td>24</td>
</tr>
<tr>
<td>128.9.176.0/24</td>
<td>24</td>
</tr>
<tr>
<td>128.9.16.14</td>
<td>128.9.0.0/16</td>
</tr>
<tr>
<td>142.12.0.0/19</td>
<td>8</td>
</tr>
</tbody>
</table>

The graph represents the difficulty of the longest prefix match, with prefix lengths on the y-axis and prefixes on the x-axis.
## Lookup Rate Required

<table>
<thead>
<tr>
<th>Year</th>
<th>Line</th>
<th>Line-rate (Gbps)</th>
<th>40B packets (Mpps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1998-99</td>
<td>OC12c</td>
<td>0.622</td>
<td>1.94</td>
</tr>
<tr>
<td>1999-00</td>
<td>OC48c</td>
<td>2.5</td>
<td>7.81</td>
</tr>
<tr>
<td>2000-01</td>
<td>OC192c</td>
<td>10.0</td>
<td>31.25</td>
</tr>
<tr>
<td>2002-03</td>
<td>OC768c</td>
<td>40.0</td>
<td>125</td>
</tr>
</tbody>
</table>

31.25 Mpps $\Rightarrow$ 33 ns

DRAM: 50-80 ns, SRAM: 5-10 ns
Size of the Forwarding Table

Source: http://www.telstra.net/ops/bgptable.html
Types of Internal Interconnects

1. Multiplexers

2. Tri-State Devices

3. Shared Memory
Where do packets go post output port selection?
Two basic techniques

Input Queueing

Output Queueing

Usually a non-blocking switch fabric (e.g. crossbar)

Usually a fast bus
Shared Memory Bandwidth

- 5ns per memory operation
- Two memory operations per packet
- Therefore, up to 160Gb/s
- In practice, closer to 80Gb/s

Shared Memory

5ns SRAM

200 byte bus
Input buffered switch

- Independent routing logic per input
  - FSM
- Scheduler logic arbitrates each output
  - priority, FIFO, random
- Head-of-line blocking problem
Input Queueing

Head of Line Blocking

Load

Delay

Load

58.6%
100%
Head of Line Blocking
(Virtual) Output Buffered Switch

N buffers per input

- How would you build a shared pool?
Solving HOL with Input Queueing

*Virtual output queues*
Input Queueing
Virtual Output Queues

Load

Delay

100%
Output scheduling

- $n$ independent arbitration problems?
  - static priority, random, round-robin
- simplifications due to routing algorithm?
- general case is max bipartite matching
Finding a maximum size match

• How do we find the maximum size (weight) match?
Finding a maximum size bipartite matching is equivalent to solving a network flow problem with capacities and flows of size 1.
Network flows and bipartite matching

Maximum Size Matching:
Complexity of Maximum Matchings

• Maximum Size Matchings:
  – Algorithm by Dinic $O(N^{5/2})$

• Maximum Weight Matchings
  – Algorithm by Kuhn $O(N^3)$

• In general:
  – Hard to implement in hardware
  – Too Slow in Practice
    • But gives nice theory and upper bound
Arbitration

- Maximal Matches
- Wavefront Arbiter (WFA)
- Parallel Iterative Matching (PIM)
- iSLIP
Maximal Matching

• A maximal matching is one in which each edge is added one at a time, and is not later removed from the matching.
• i.e. no augmenting paths allowed (they remove edges added earlier).
• No input and output are left unnecessarily idle.
Example of Maximal Size Matching

Maximal Size Matching

Maximum Size Matching
Maximal Matchings

- In general, *maximal* matching is simpler to implement, and has a faster running time.
- A maximal size matching is at least half the size of a maximum size matching.
- A maximal weight matching is defined in the obvious way.
- A maximal weight matching is at least half the weight of a maximum weight matching.
Routing Strategies?

- Architecture of the middle of the switch?
  - Wavefront
  - Slip/PIM
  - Butterfly/Benes networks

- Goal in each case is a conflict-free schedule of inputs to outputs given the output is already determined
Wave Front Arbiter
(Tamir)

Requests

Match
Wave Front Arbiter

Requests

Match
Wavefront Arbiters

*Properties*

- Feed-forward (i.e. non-iterative) design lends itself to pipelining.
- Always finds maximal match.
- Usually requires mechanism to prevent inputs from getting preferential service.
  - What the 50Gbs router does:
    - Scramble (permute) inputs each cycle
Parallel Iterative Matching

# Iteration

ϕ1: Requests

ϕ2: Grant

ϕ3: Accept/Match

selection

selection
PIM Properties

- Guaranteed to find a maximal match in at most $N$ iterations.
- In each phase, each input and output arbiter can make decisions independently.
- In general, will converge to a maximal match in $< N$ iterations.
Parallel Iterative Matching

PIM with a single iteration
Parallel Iterative Matching

![Graph showing performance of PIM with 4 iterations vs. offered load.](image)
iSLIP

Φ1: Requests

Φ2: Grant

Φ3: Accept/Match
iSLIP Operation

- **Grant phase:** Each output selects the requesting input at the pointer, or the next input in round-robin order. *It only updates its pointer if the grant is accepted.*

- **Accept phase:** Each input selects the granting output at the pointer, or the next output in round-robin order.

- **Consequence:** Under high load, grant pointers tend to move to unique values.
iSLIP

Properties

- Random under low load
- TDM under high load
- Lowest priority to MRU
- 1 iteration: fair to outputs
- Converges in at most $N$ iterations. (On average, simulations suggest $< \log_2 N$)
- Implementation: $N$ priority encoders
- 100% throughput for uniform i.i.d. traffic.
- But…some pathological patterns can lead to low throughput.
iSLIP

- FIFO
- iSLIP with 4 iterations
- Output Buffering

Graph showing Avg Cell Latency (Cells) vs. Offered Load (%)
iSLIP

Implementation
Alternative Switching

- Crossbars are expensive
- Alternative networks can match inputs to outputs:
  - Ring
  - Tree
  - K-ary N-cubes
  - Multi-stage logarithmic networks
    - Each cell has constant number of inputs and outputs
Example: Butterfly

Butterfly Network

| 000 | 000 |
| 001 | 001 |
| 010 | 010 |
| 011 | 011 |
| 100 | 100 |
| 101 | 101 |
| 110 | 110 |
| 111 | 111 |

split on MSB

split on LSB
Butterfly

Inputs

Outputs

A “Butterfly”
Benes Network

Butterfly #1

Butterfly #2
Benes networks

• Any permutation has a conflict free route
  – Useful property
  – Offline computation is difficult

• Can route to random node in middle, then to destination
  – Conflicts are unlikely under uniform traffic
  – What about conflicts?
Sorting Networks

- Bitonic sorter recursively merges sorted sublists.
- Can switch by sorting on destination.
  - additional components needed for conflict resolution
Flow Control

• What do you do when push comes to shove?
  – ethernet: collision detection and retry after delay
  – FDDI, token ring: arbitration token
  – TCP/WAN: buffer, drop, adjust rate
  – any solution must adjust to output rate

• Link-level flow control
Link Flow Control Examples

- **Short Links**

- **long links**
  - several flits on the wire
Smoothing the flow

• How much slack do you need to maximize bandwidth?