15-441 Computer Networking

14 - Router Design

Based on slides from Dave Andersen and Nick Feamster

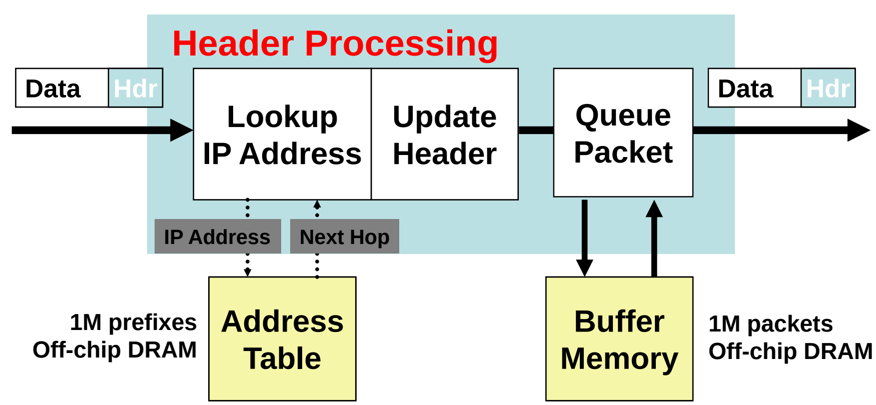
Router Architecture

- Data Plane
  - Moving the data, i.e., the packets
  - How packets get forwarded
- Control Plane
  - How routing protocols establish routes/etc.

Today's Lecture: Data Plane

- The design of big, fast routers
- Partridge *et al.*, *A 50 Gb/s IP Router*
- Design constraints
  - Speed
  - Size
  - Power consumption
- Components
- Algorithms
  - Lookups and packet processing (classification, etc.)
  - Packet queuing
  - Switch arbitration

Generic Router Architecture



What's In A Router

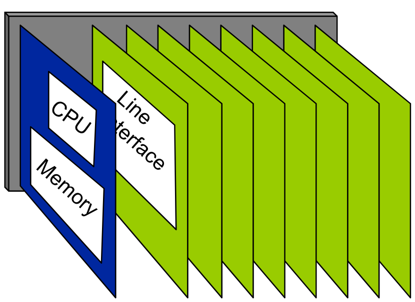
- Interfaces
  - Input/output of packets
- Switching fabric
  - Moving packets from input to output
- Software
  - Routing
  - Packet processing
  - Scheduling
  - Etc.

Summary of Routing Functionality

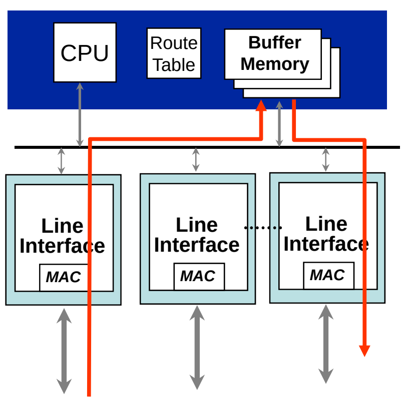
- Router gets packet
- Looks at packet header for destination
- Looks up routing table for output interface
- Modifies header   Why?
- Passes packet to output interface

First Generation Routers

Shared Bus



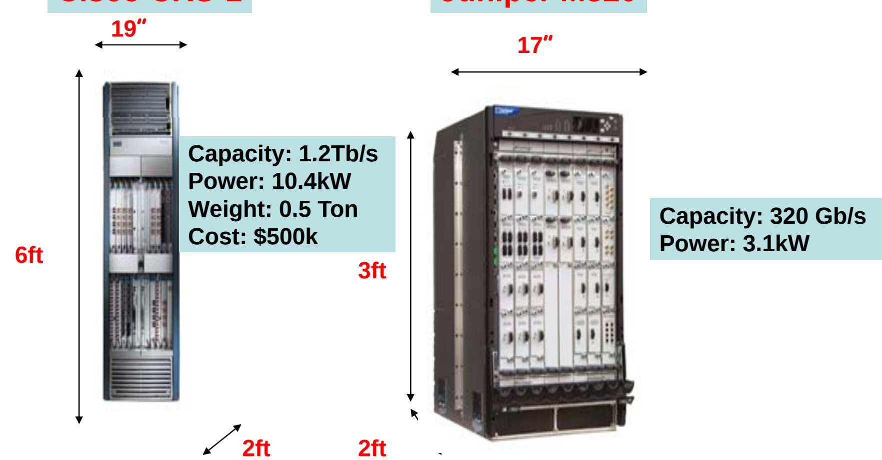
Off-chip Buffer



Typically <0.5Gb/s aggregate capacity

What a Router Chassis Looks Like

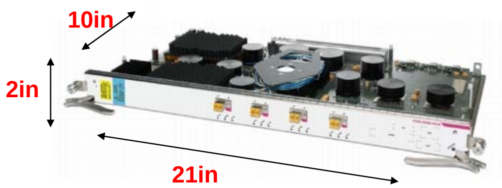
Cisco CRS-1   Juniper M320



What a Router Line Card Looks Like

1-Port OC48 (2.5 Gb/s)
(for Juniper M40)

4-Port 10 GigE
(for Cisco CRS-1)



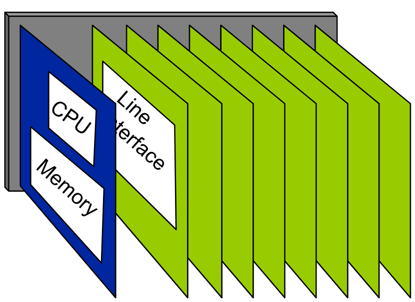
Power: about 150 Watts

Big, Fast Routers: Why Bother?

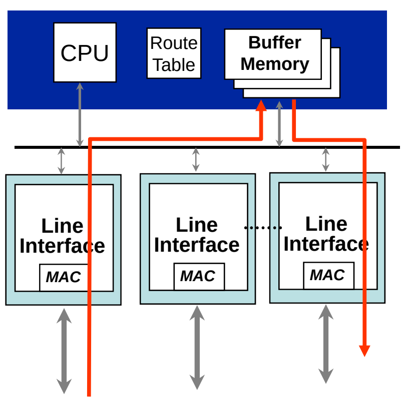
- Faster link bandwidths
- Increasing demands
- Larger network size (hosts, routers, users)
- More cost effective

First Generation Routers

Shared Bus



Off-chip Buffer



Typically <0.5Gb/s aggregate capacity

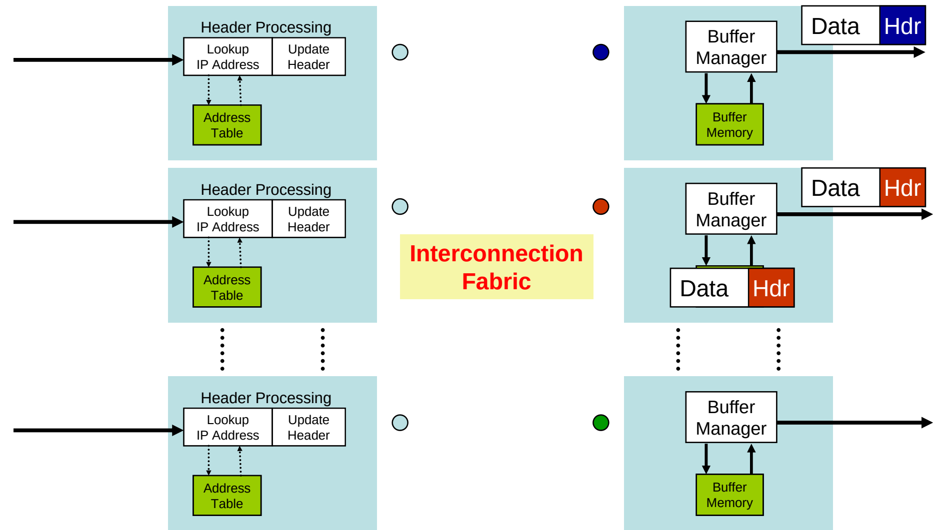
Innovation #1: Each Line Card Has the Routing Tables

- Prevents central table from becoming a bottleneck at high speeds
- **Complication:** Must update forwarding tables on the fly.

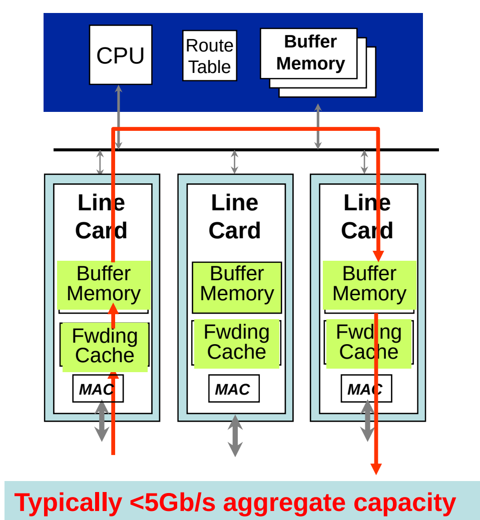
Control Plane & Data Plane

- Control plane must remember lots of routing info (BGP tables, etc.)
- Data plane only needs to know the "FIB" (Forwarding Information Base)
  - Smaller, less information, etc.
  - Simplifies line cards vs the network processor

Generic Router Architecture



Second Generation Routers



Bypasses memory bus with direct transfer over bus between line cards

Moves forwarding decisions local to card to reduce CPU pain

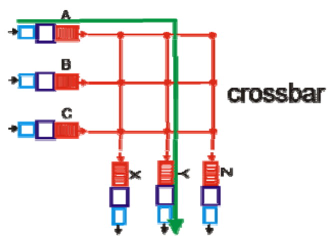
Punt to CPU for "slow" operations

Bus-based

- Some improvements possible
  - Cache bits of forwarding table in line cards, send directly over bus to outbound line card
- But shared bus was big bottleneck
  - E.g., *modern* PCI bus (PCIx16) is only 32Gbit/sec (in theory)
  - Almost-modern cisco (XR 12416) is 320Gbit/sec.
  - Ow! How do we get there?

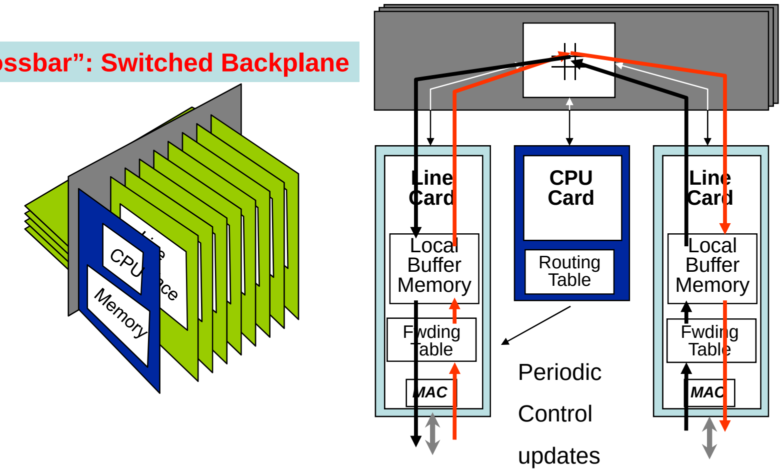
Innovation #2: Switched Backplane

- Every input port has a connection to every output port
- During each timeslot, each input connected to zero or one outputs
- **Advantage:** Exploits parallelism
- **Disadvantage:** Need scheduling algorithm



Third Generation Routers

"Crossbar": Switched Backplane



Typically <50Gb/s aggregate capacity

What's so hard here?

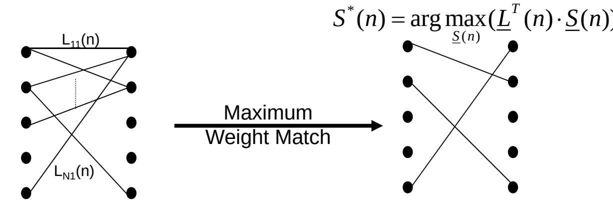
- Back-of-the-envelope numbers
  - Line cards can be 40 Gbit/sec today (OC-768)
    - Undoubtedly faster in a few more years, so scale these #s appropriately!
  - To handle minimum-sized packets (~40b)
    - 125 Mpps, or 8ns per packet
    - But note that this can be deeply pipelined, at the cost of buffering and complexity. Some lookup chips do this, though still with SRAM, not DRAM. Good lookup algos needed still.
- For every packet, you must:
  - Do a routing lookup (where to send it)
  - Schedule the crossbar
  - Maybe buffer, maybe QoS, maybe filtering by ACLs

Crossbar Switching

- **Conceptually:** *N* inputs, *N outputs*
  - Actually, inputs are also outputs
- In each timeslot, one-to-one mapping between inputs and outputs.
- Crossbar constraint: If input I is connected to output j, no other input connected to j, no other output connected to input I
- **Goal:** Maximal matching

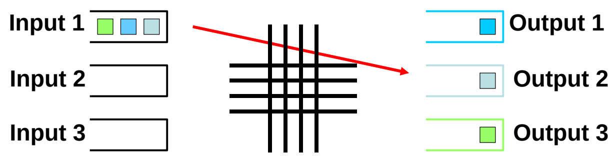
Traffic Demands **Bipartite Match**

$$S^*(n) = \arg\max_{\underline{S}(n)} (\underline{L}^T(n) \cdot \underline{S}(n))$$



Head-of-Line Blocking

Problem: The packet at the front of the queue experiences contention for the output queue, blocking all packets behind it.

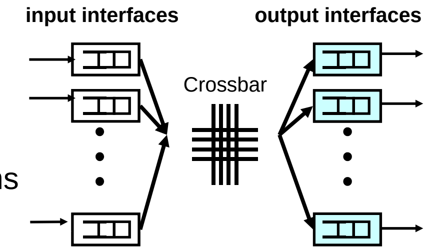


Maximum throughput in such a switch: 2 – sqrt(2)

M.J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input Versus Output Queuing on a Space-Division Packet Switch," *IEEE Transactions On Communications*, Vol. Com-35, No. 12, December 1987, pp. 1347-1356.

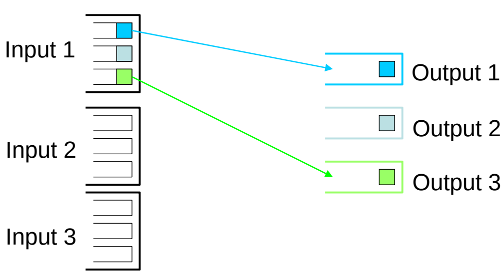
Combined Input-Output Queuing

- **Advantages**
  - Easy to build
  - Better throughput
- **Disadvantages**
  - Harder to design algorithms
    - Two congestion points



Solution: Virtual Output Queues

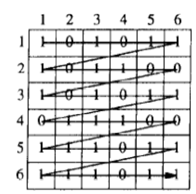
- Maintain N virtual queues at each input
  - one per output



N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% Throughput in an Input-Queued Switch," IEEE Transactions on Communications, Vol. 47, No. 8, August 1999, pp. 1260-1267.

Early Crossbar Scheduling Algorithm

- Wavefront algorithm



(a)

$A_{ij = 1}$ indicates that card i has a packet to send to card j

Problems: Fairness, speed, …

Alternatives to the Wavefront Scheduler

- PIM: Parallel Iterative Matching
  - **Request:** Each input sends requests to all outputs for which it has packets
  - **Grant:** Output selects an input at random and grants
  - **Accept:** Input selects from its received grants
- **Problem:** Matching may not be maximal
- **Solution:** Run several times
- **Problem:** Matching may not be "fair"
- **Solution:** Grant/accept in round robin instead of random

Scheduling and Fairness

- What is an appropriate definition of fairness?
  - One notion: Max-min fairness
  - Disadvantage: Compromises throughput
- Max-min fairness gives priority to low data rates/small values
- An ill-behaved flow only hurts itself

Max-Min Fairness

- A flow rate x is **max-min fair** if any rate x cannot be increased without decreasing some y which is smaller than or equal to x.
- How to share equally with different resource demands
  - small users will get all they want
  - large users will evenly split the rest
- More formally, perform this procedure:
  - resource allocated to customers in order of increasing demand
  - no customer receives more than requested
  - customers with unsatisfied demands split the remaining resource

Example

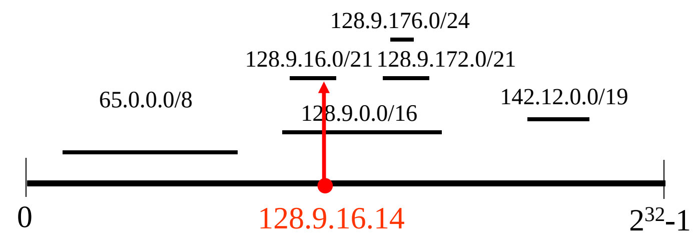
- Demands: 2, 2.6, 4, 5; capacity: 10
  - 10/4 = 2.5
  - **Problem:** 1st user needs only 2; excess of 0.5,
- Distribute among 3, so 0.5/3=0.167
  - now we have allocs of [2, 2.67, 2.67, 2.67],
  - leaving an excess of 0.07 for cust #2
  - divide that in two, gets [2, 2.6, 2.7, 2.7]
- Maximizes the minimum share to each customer whose demand is not fully serviced

IP Address Lookup

Challenges:

1. **Longest-prefix match (not exact).**
2. Tables are large and growing.
3. Lookups must be fast.

IP Lookups find Longest Prefixes



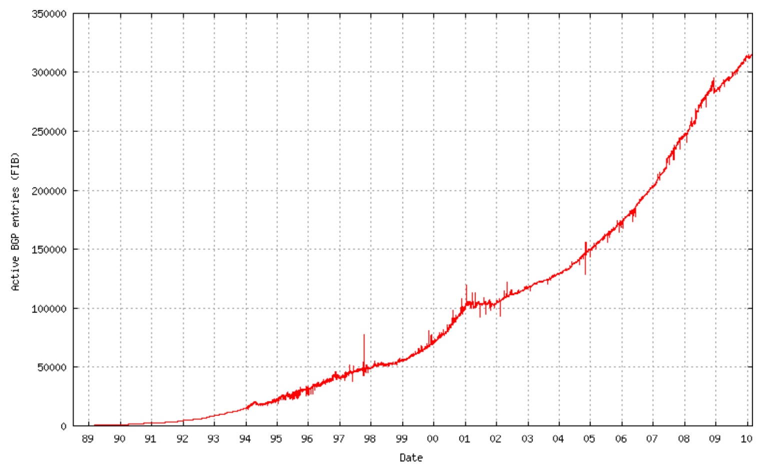
Routing lookup: Find the longest matching prefix (aka the most specific route) among all prefixes that match the destination address.

IP Address Lookup

Challenges:

1. Longest-prefix match (not exact).
2. **Tables are large and growing.**
3. Lookups must be fast.

Address Tables are Large



IP Address Lookup

Challenges:

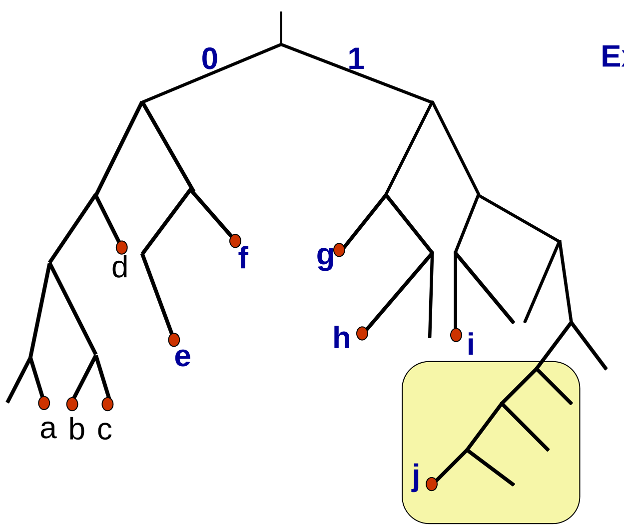
1. Longest-prefix match (not exact).
2. Tables are large and growing.
3. **Lookups must be fast.**

Lookups Must be Fast

Cisco CRS-1 1-Port OC-768C (Line rate: 42.1 Gb/s)

Year	Line	40B packets (Mpkt/s)	
1997	622Mb/s	1.94	OC-12
1999	2.5Gb/s	7.81	OC-48
2001	10Gb/s	31.25	OC-192
2003	40Gb/s	125	OC-768

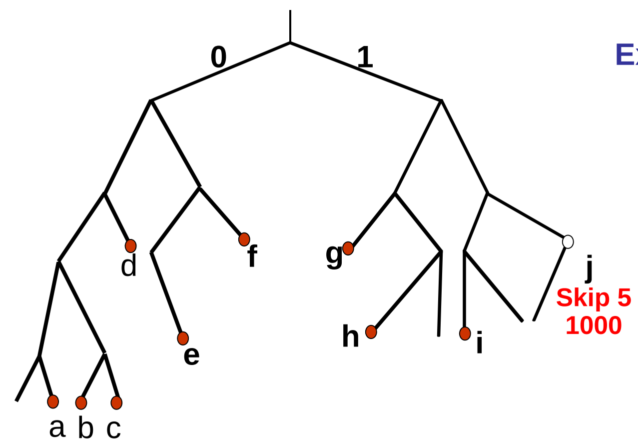
IP Address Lookup: Binary Tries



Example Prefixes:

a) 00001
b) 00010
c) 00011
d) 001
e) 0101
f) 011
g) 100
h) 1010
i) 1100
j) 11110000

IP Address Lookup: Patricia Trie



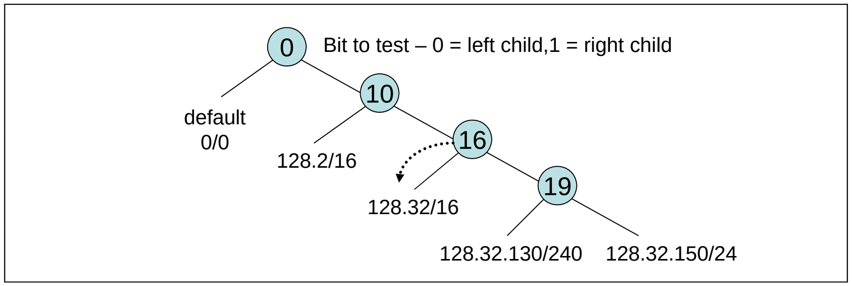
Example Prefixes

a) 00001
b) 00010
c) 00011
d) 001
e) 0101
f) 011
g) 100
h) 1010
i) 1100
j) 11110000

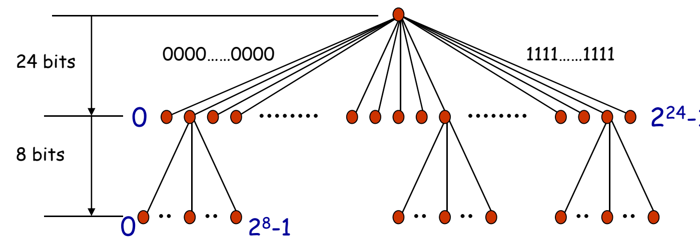
Problem: Lots of (slow) memory lookups

LPM with PATRICIA Tries

- Traditional method – Patricia Tree
  - Arrange route entries into a series of bit tests
- Worst case = 32 bit tests
  - Problem: memory speed, even w/SRAM!



Address Lookup: Direct Trie



- When pipelined, one lookup per memory access
- **Inefficient use of memory**

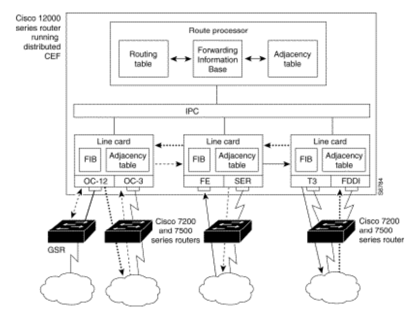
Faster LPM: Alternatives

- Content addressable memory (CAM)
  - **Hardware-based** route lookup
  - Input = tag, output = value
  - Requires exact match with tag
    - Multiple cycles (1 per prefix) 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

Historically, this approach has not been very economical.

Faster Lookup: Alternatives

- Caching
  - Packet trains exhibit temporal locality
  - Many packets to same destination
- Cisco Express Forwarding

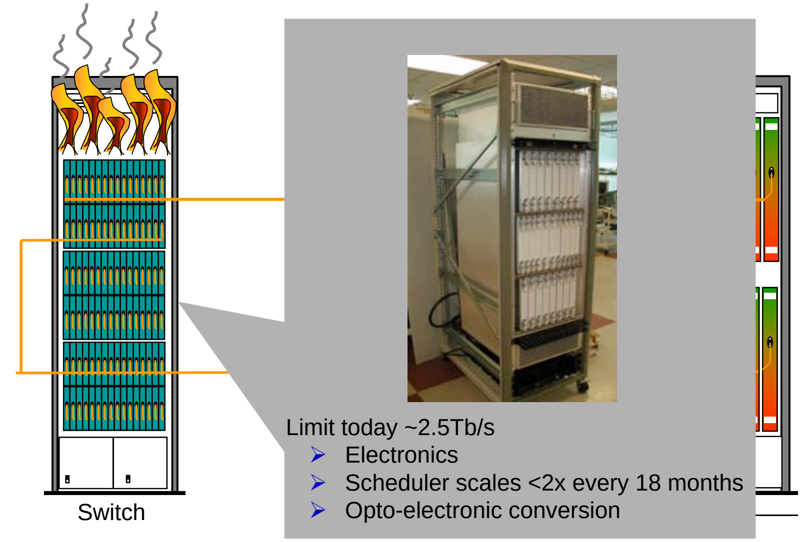


IP Address Lookup: Summary

- Lookup limited by memory bandwidth.
- Lookup uses high-degree trie.

- State of the art: 10Gb/s line rate.
- Scales to: 40Gb/s line rate.

Fourth-Generation Routers



Router Design

- Many trade-offs: power, $$$, throughput, reliability, flexibility
- Move towards distributed architectures
  - Line-cards have forwarding tables
  - Switched fabric between cards
  - Separate Network processor for “slow path” & control
- Important bottlenecks on fast path
  - Longest prefix match
  - Cross-bar scheduling
- Beware: lots of feature creep