High Speed Router Design

Shivkumar Kalyanaraman
Rensselaer Polytechnic Institute
shivkuma@ecse.rpi.edu
http://www.ecse.rpi.edu/Homepages/shivkuma

Based in part on slides of Nick McKeown (Stanford), S. Keshav (Cornell)

- Introduction
- Evolution of High-Speed Routers
- High Speed Router Components:
  - Lookup Algorithm
  - Classification
  - Switching

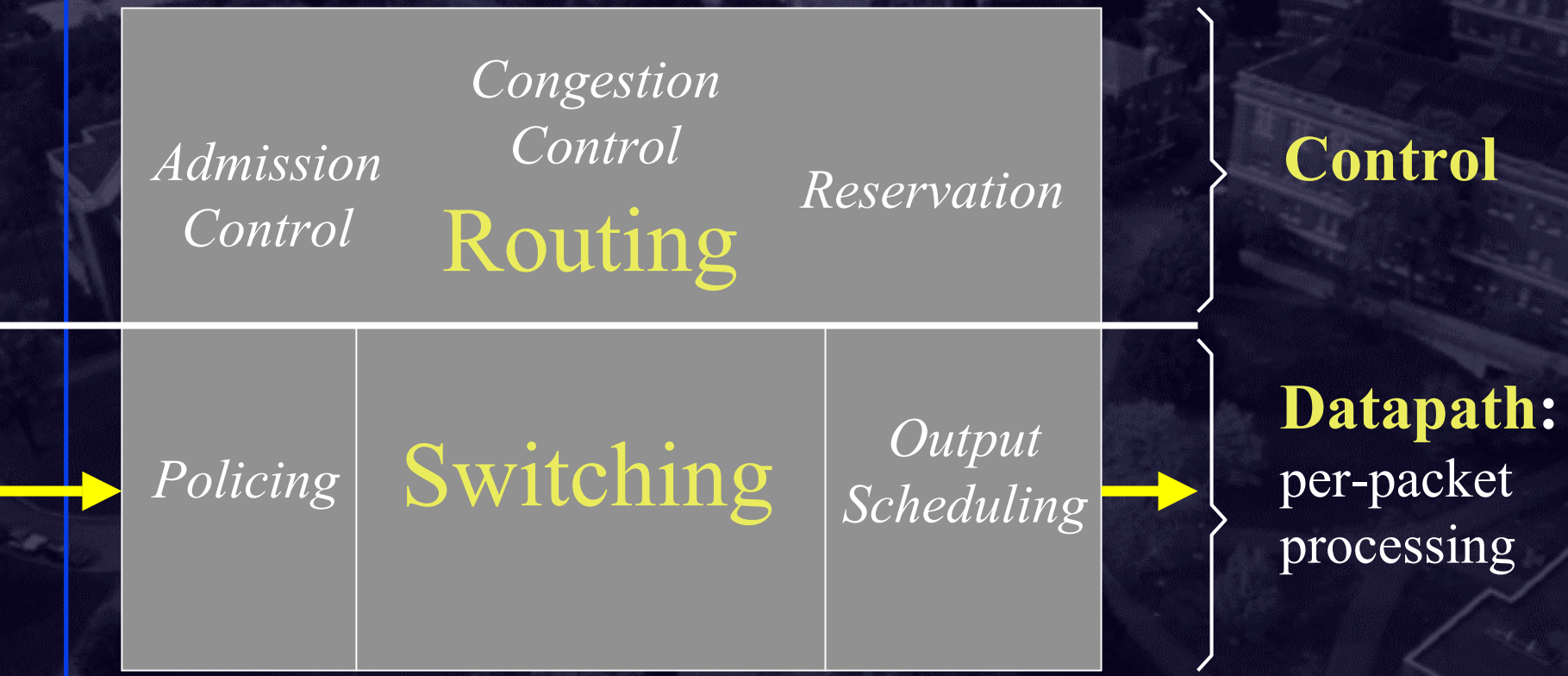
What do switches/routers look like?

Access routers
e.g. ISDN,
ADSL

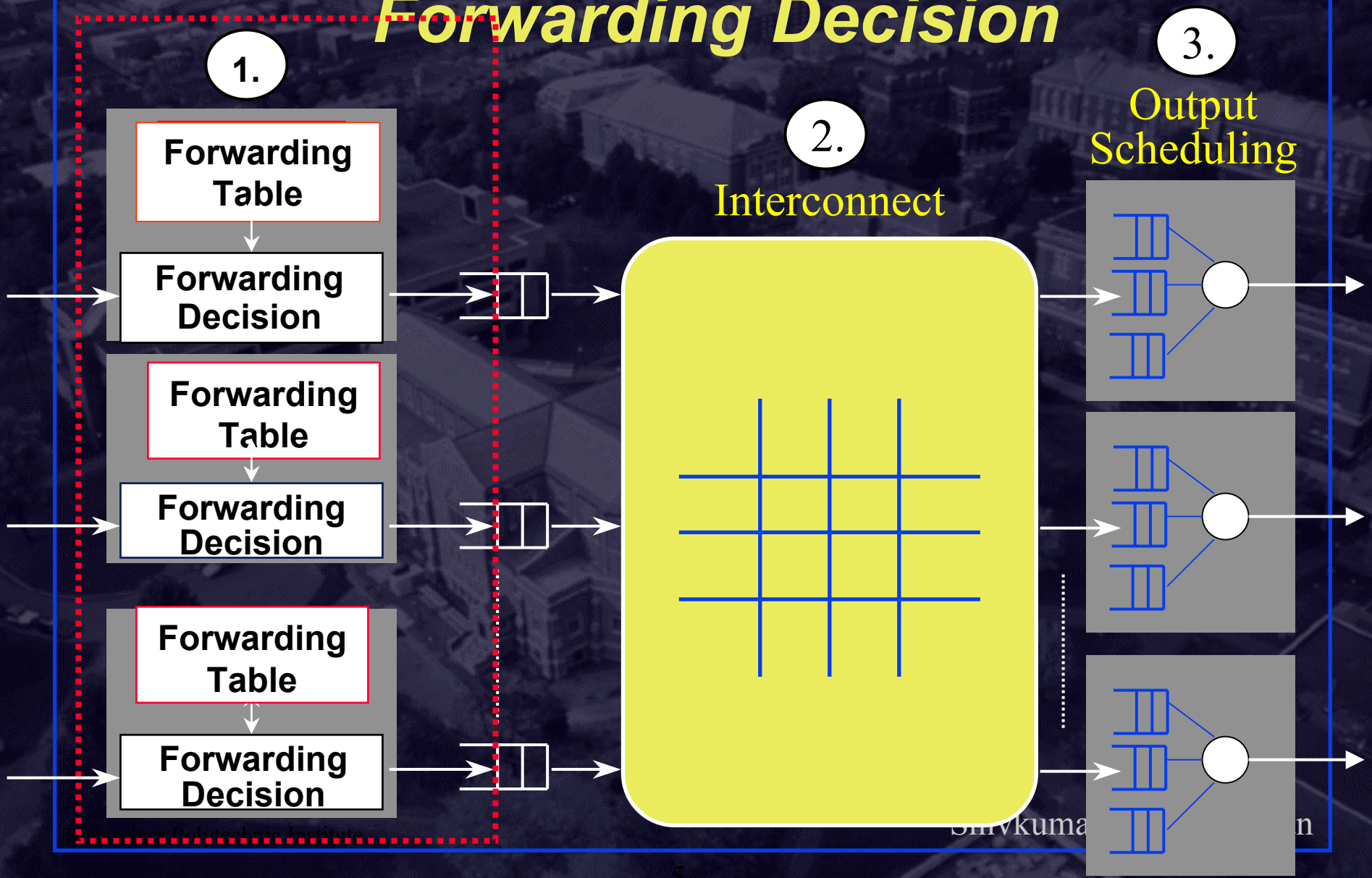
Core router
e.g. OC48c POS

Core ATM switch

Basic Architectural Components



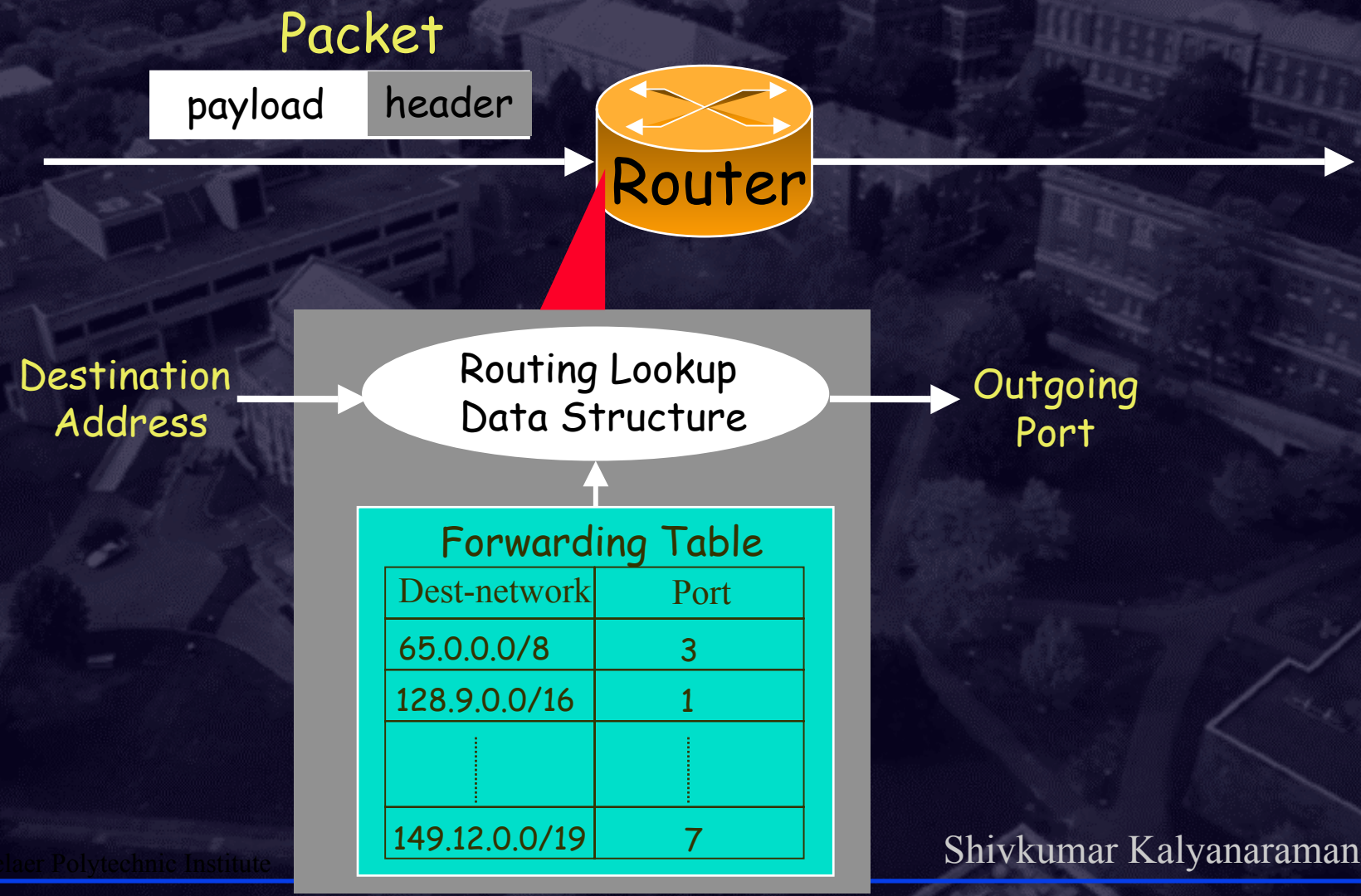
Basic Architectural Components:
Forwarding Decision



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 (switch)** packet to the outgoing port(s).
5. **Classify and buffer** packet in the queue.
6. Transmit packet onto outgoing link.

Lookup and Forwarding Engine

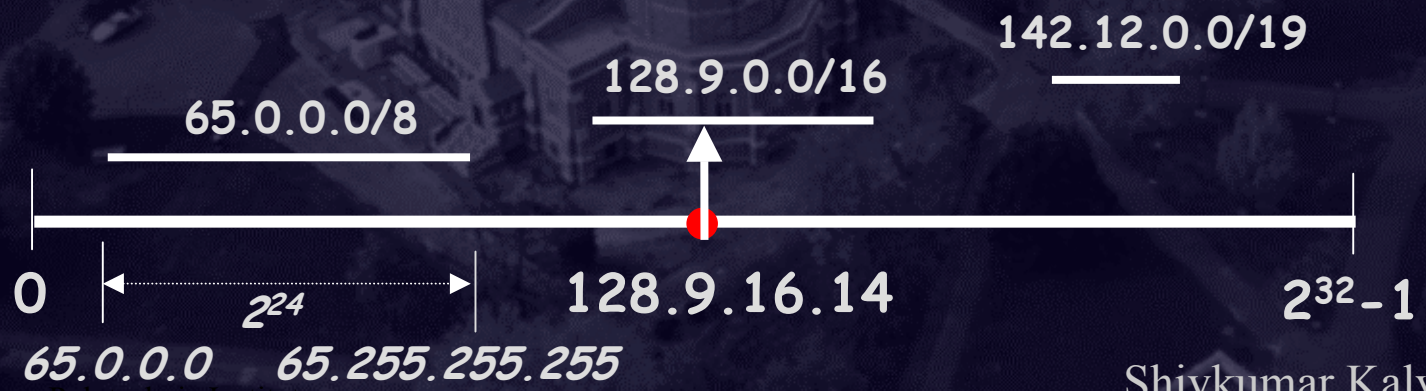


Example Forwarding Table

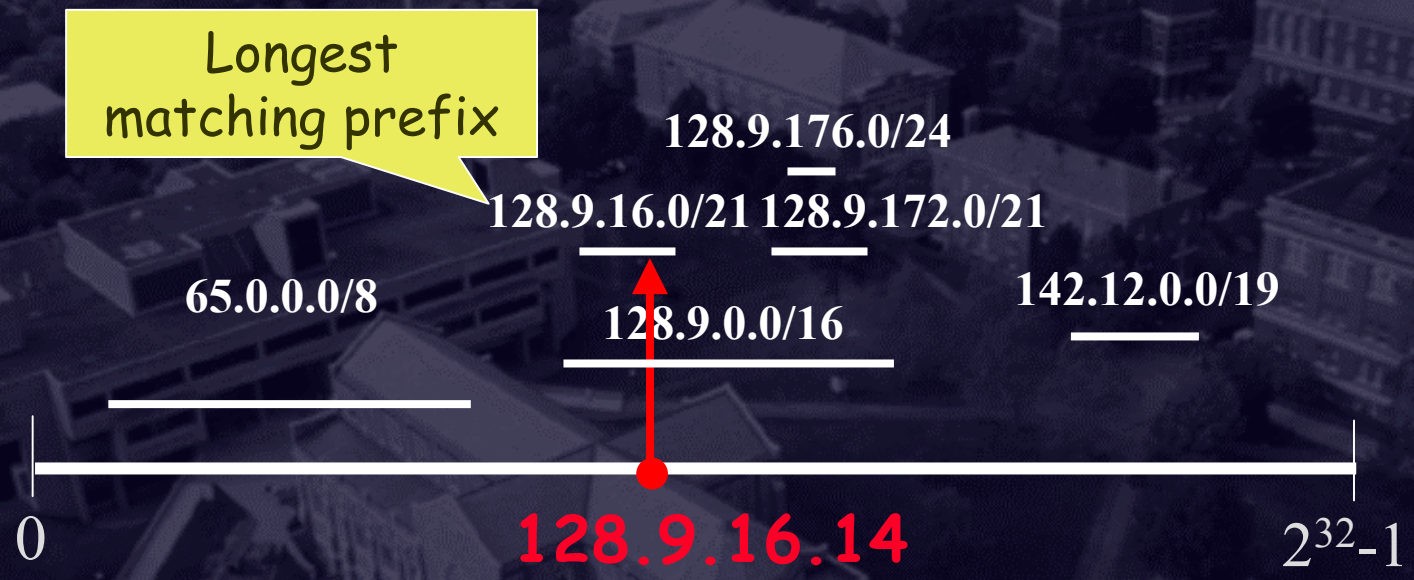
Destination IP Prefix	Outgoing Port
65.0.0.0/ 8	3
128.9.0.0/16	1
142.12.0.0/19	7

Prefix length

IP prefix: 0-32 bits



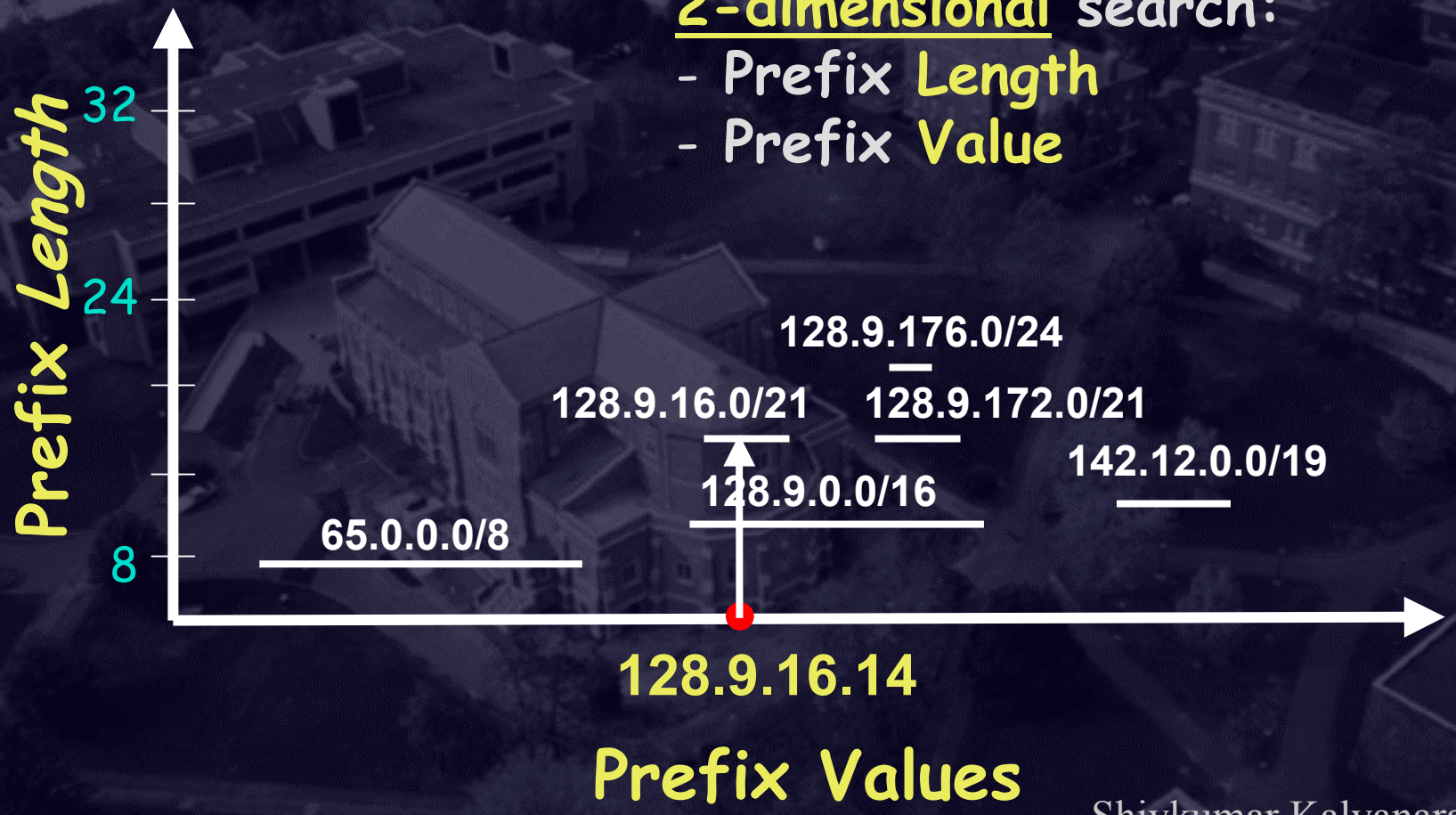
Prefixes can Overlap



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

Difficulty of Longest Prefix Match

2-dimensional search:
- Prefix Length
- Prefix Value



Lookup Rates Required

Year	Line	Line-rate (Gbps)	40B packets (Mpps)
1998-99	OC12c	0.622	1.94
1999-00	OC48c	2.5	7.81
2000-01	OC192c	10.0	31.25
2002-03	OC768c	40.0	125

31.25 Mpps ⇒ 33 ns

DRAM: 50-80 ns, SRAM: 5-10 ns

Update Rates Required

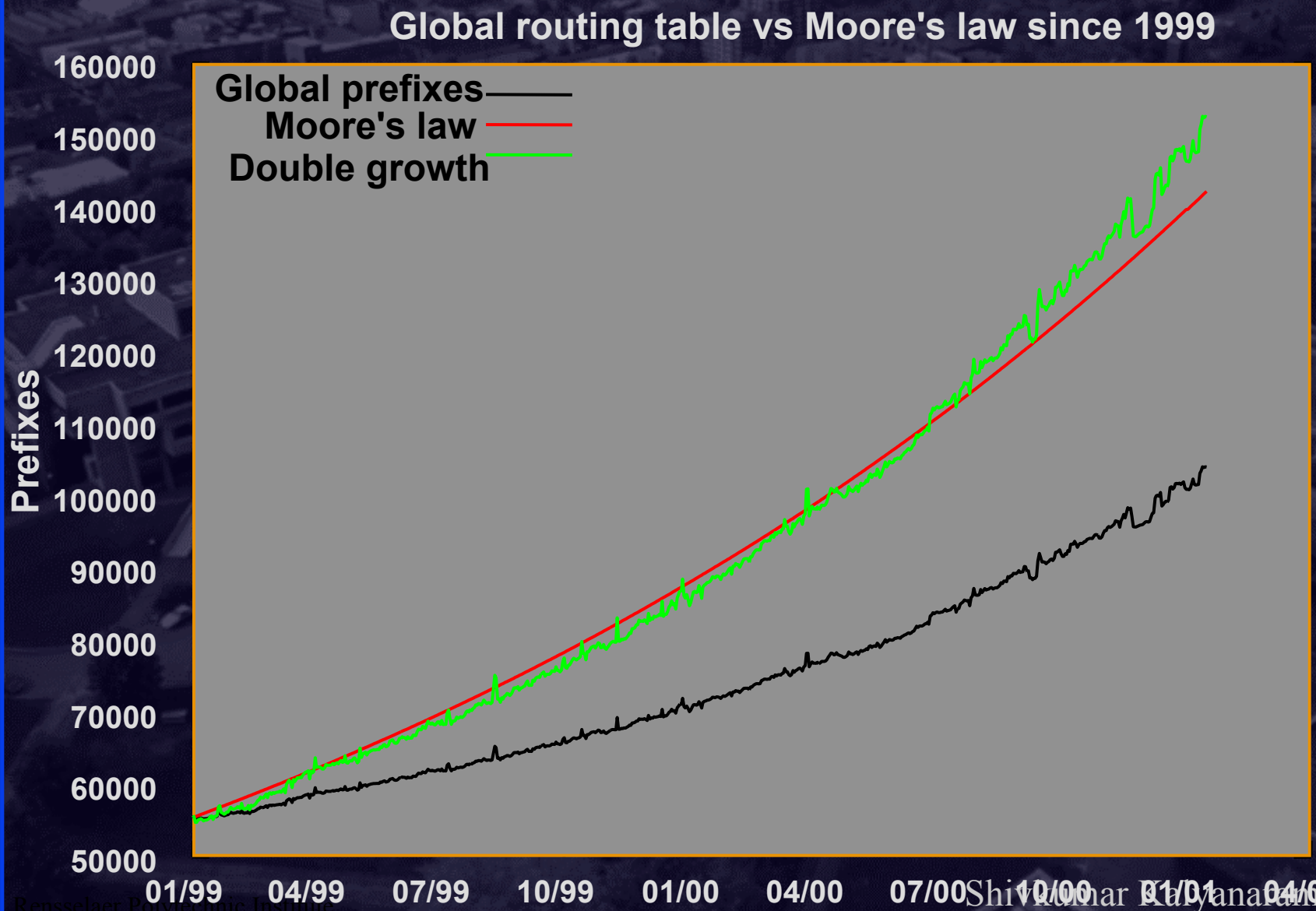
- Recent BGP studies show that updates can be:
  - **Bursty:** several 100s of routes updated/withdrawn => *insert/delete* operations
  - **Frequent:** Average 100+ updates per second
- Need data structure to be efficient in terms of lookup as well as update (insert/delete) operations.

Size of the Forwarding Table

Renewed growth due to multi-homing of enterprise networks!

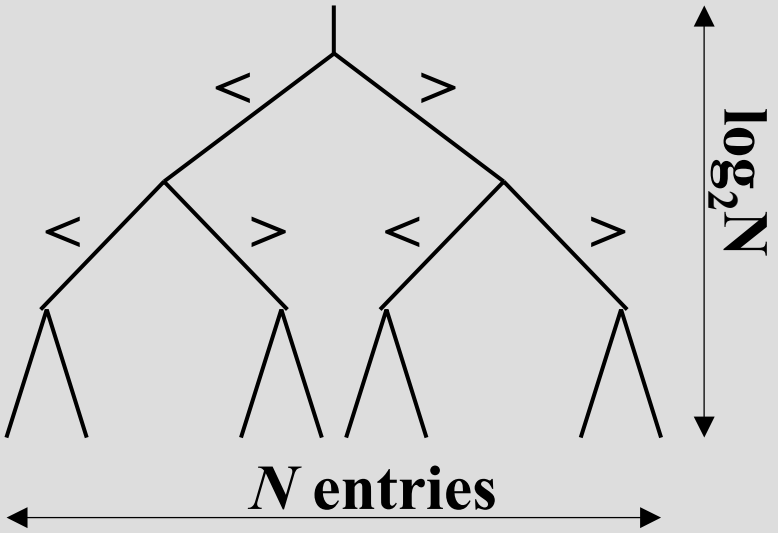
Source: http://www.telstra.net/ops/bgptable.html

Potential Hyper-Exponential Growth!

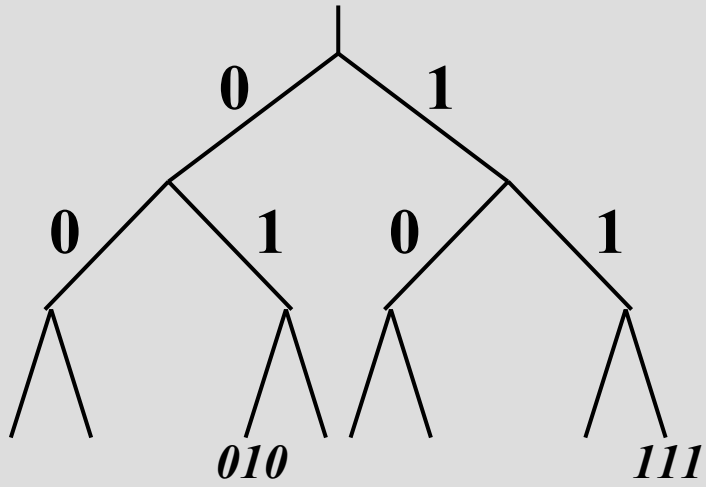


Trees and Tries

Binary Search Tree

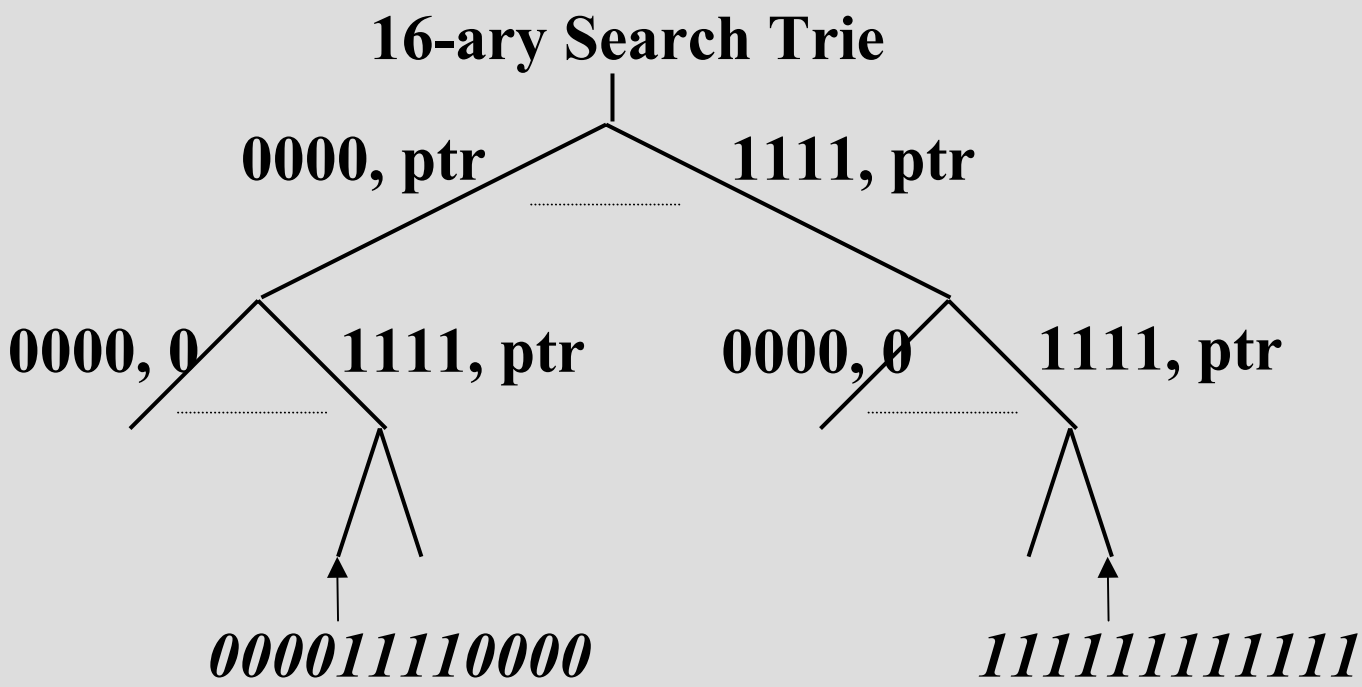


Binary Search Trie



Trees and Tries

Multiway tries



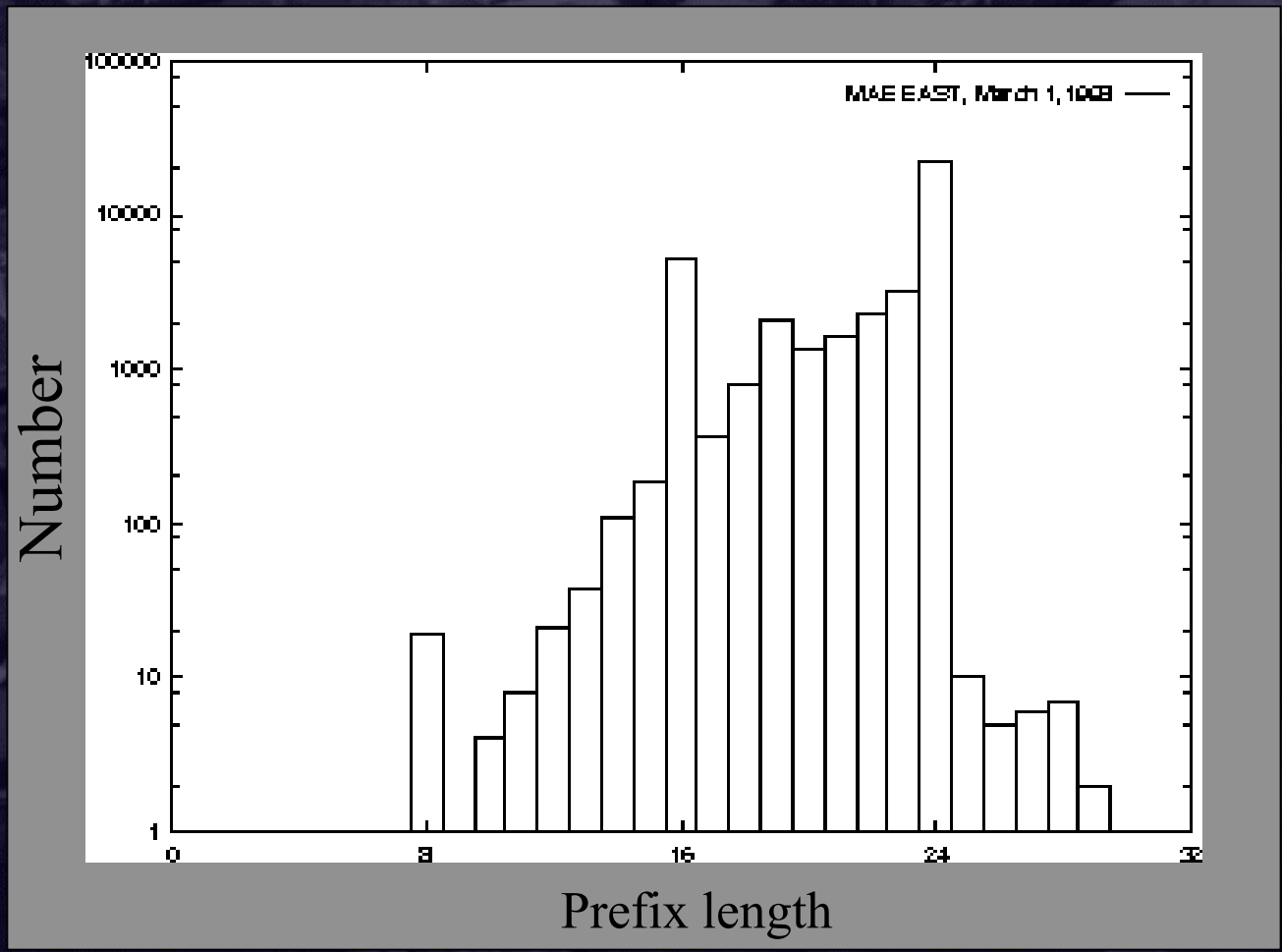
Lookup: Multiway Tries

Tradeoffs

Degree of Tree	*# Mem References*	*# Nodes ($x10^6$)*	*Total Memory (Mbytes)*	*Fraction Wasted (%)*
2	48	1.09	4.3	49
4	24	0.53	4.3	73
8	16	0.35	5.6	86
16	12	0.25	8.3	93
64	8	0.17	21	98
256	6	0.12	64	99.5

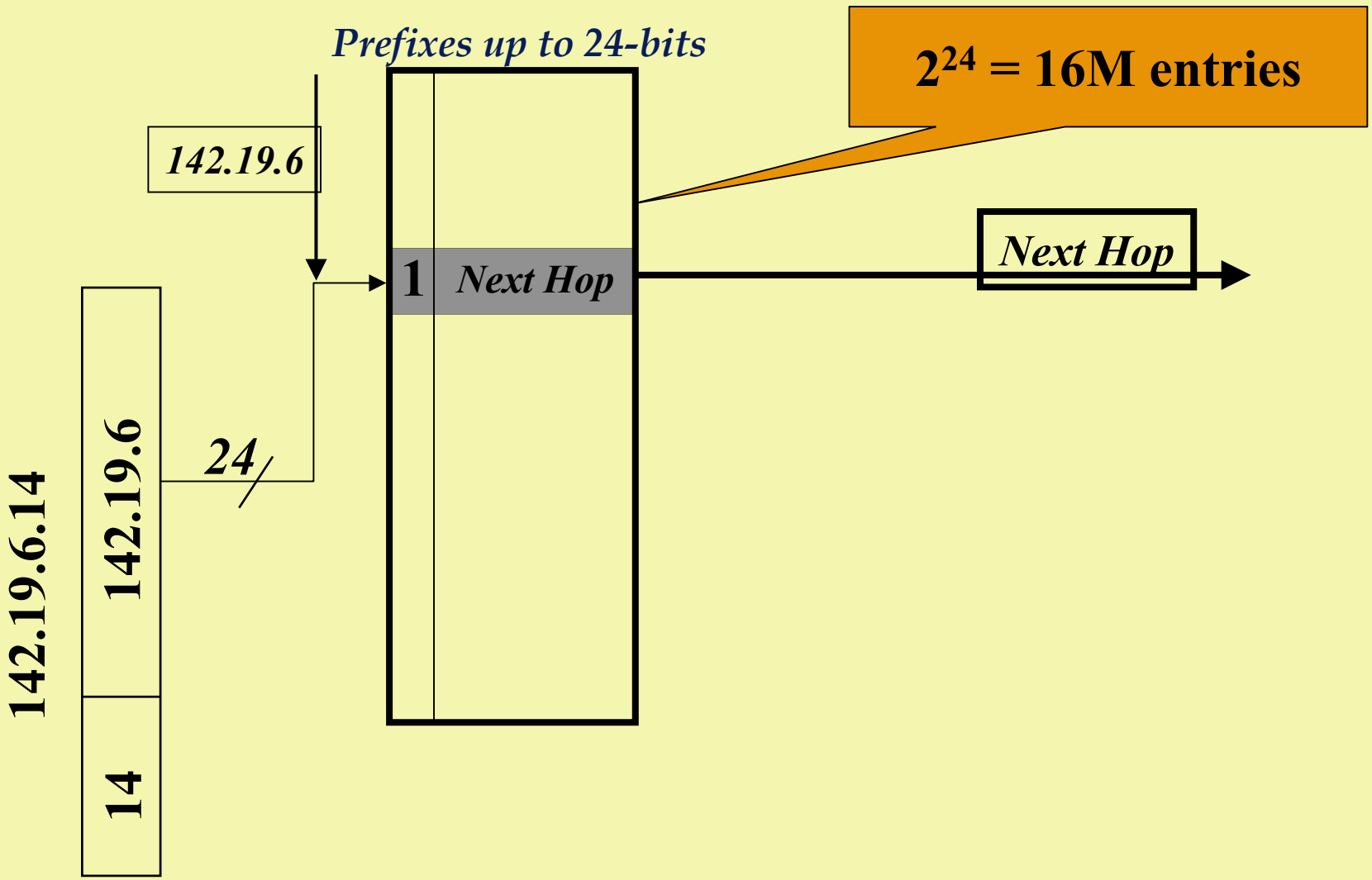
Table produced from 2^{15} randomly generated 48-bit addresses

Routing Lookups in Hardware

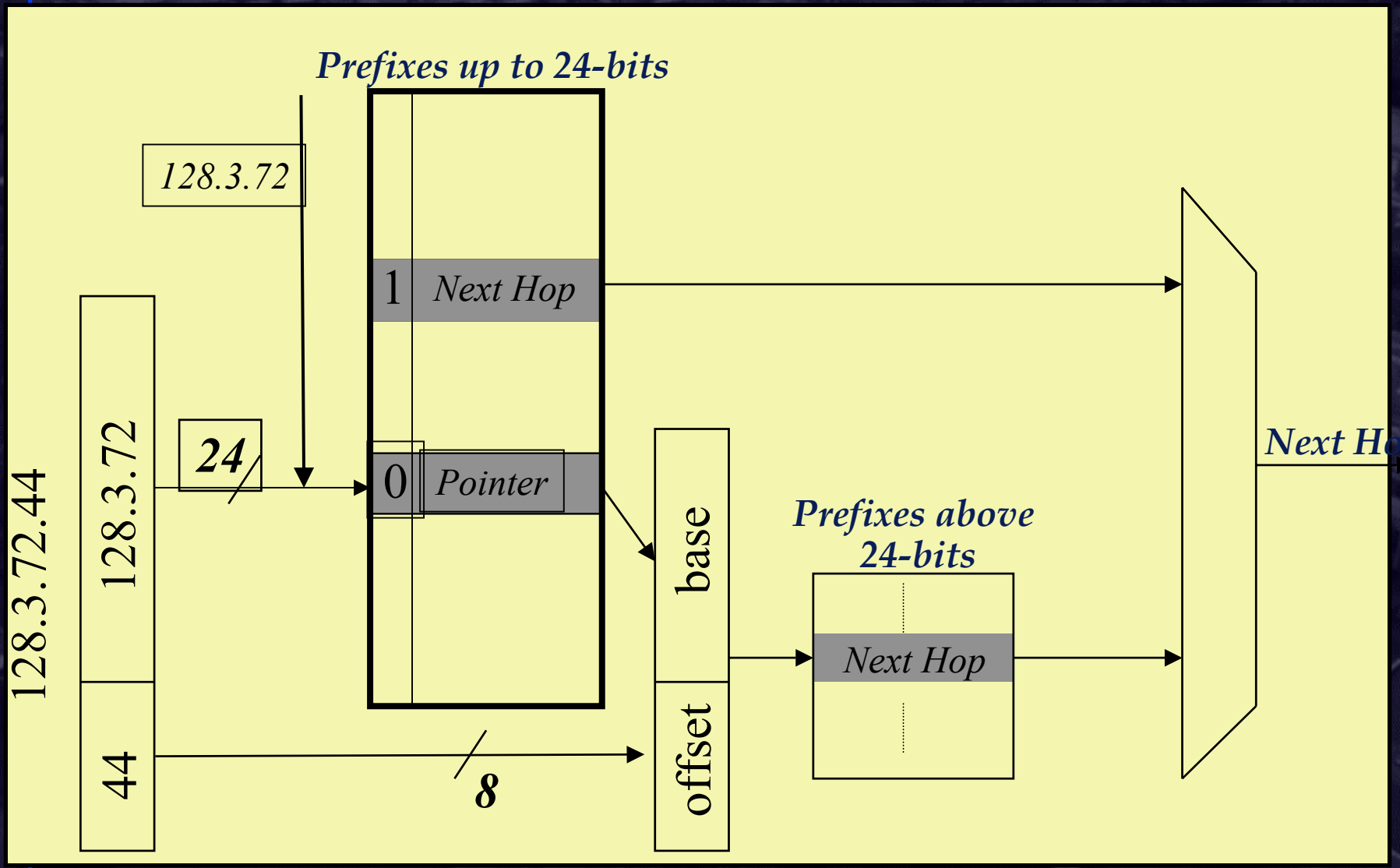


Most prefixes are 24-bits or shorter

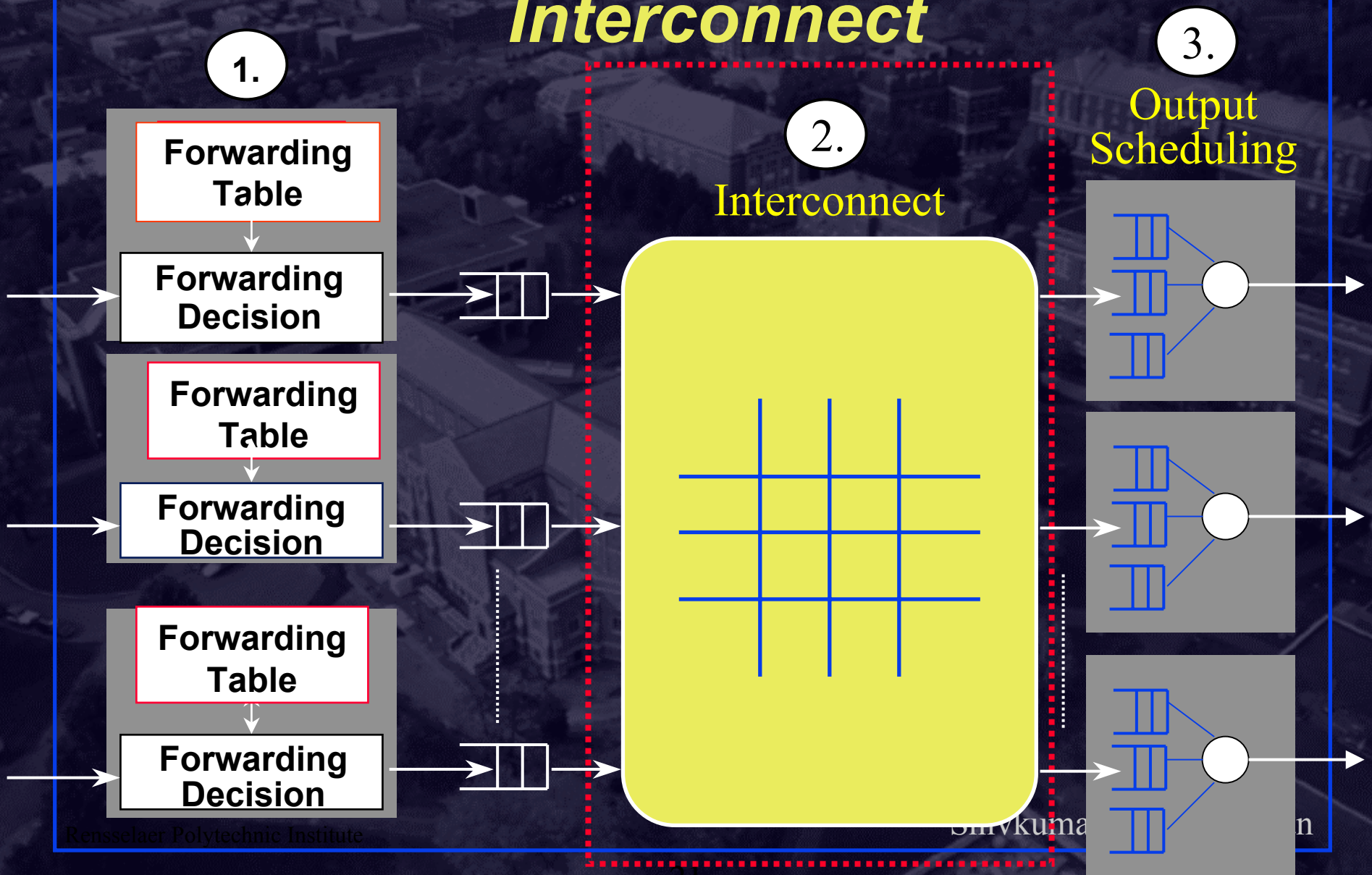
Routing Lookups in Hardware



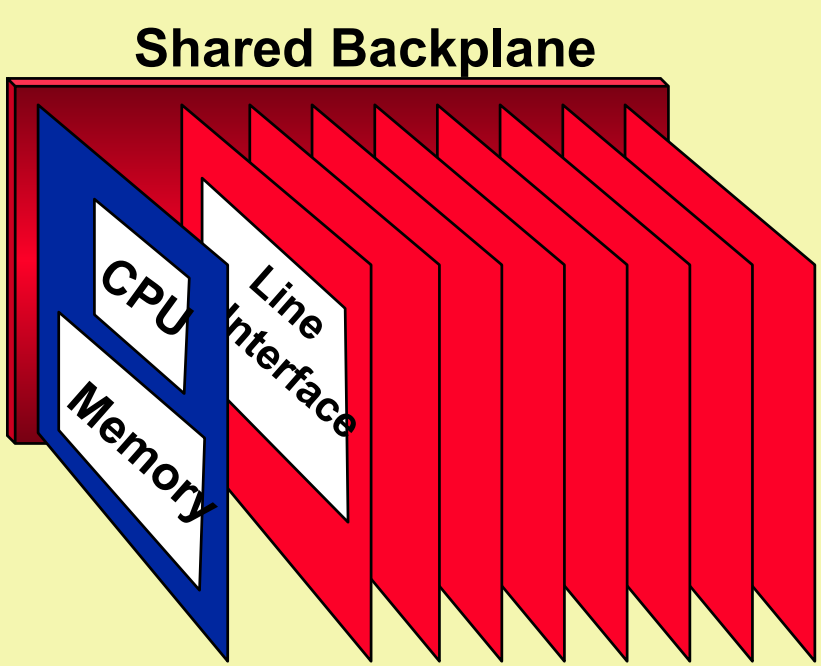
Routing Lookups in Hardware

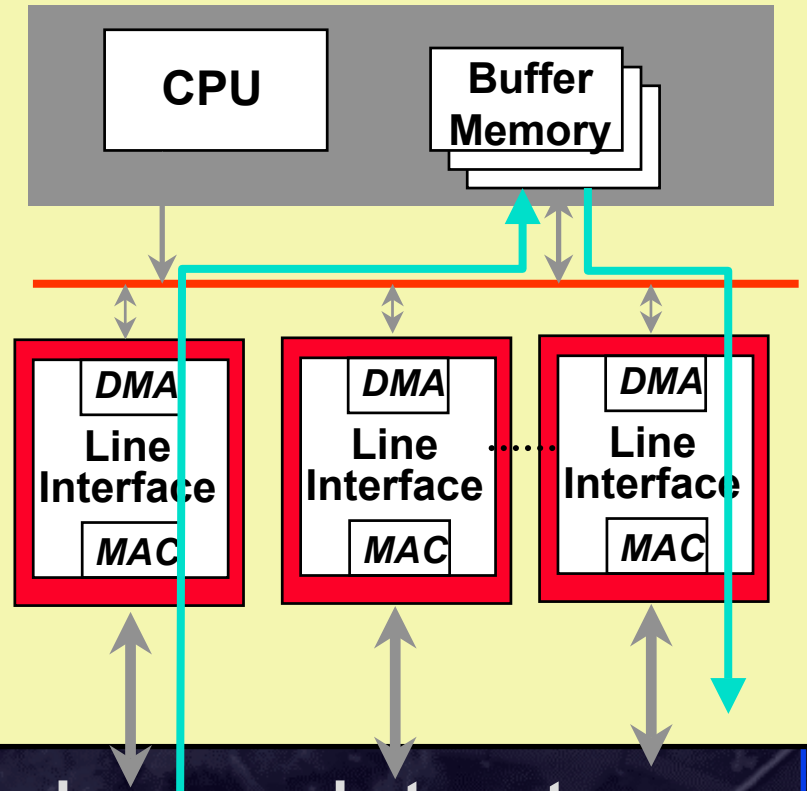


Basic Architectural Components: *Interconnect*



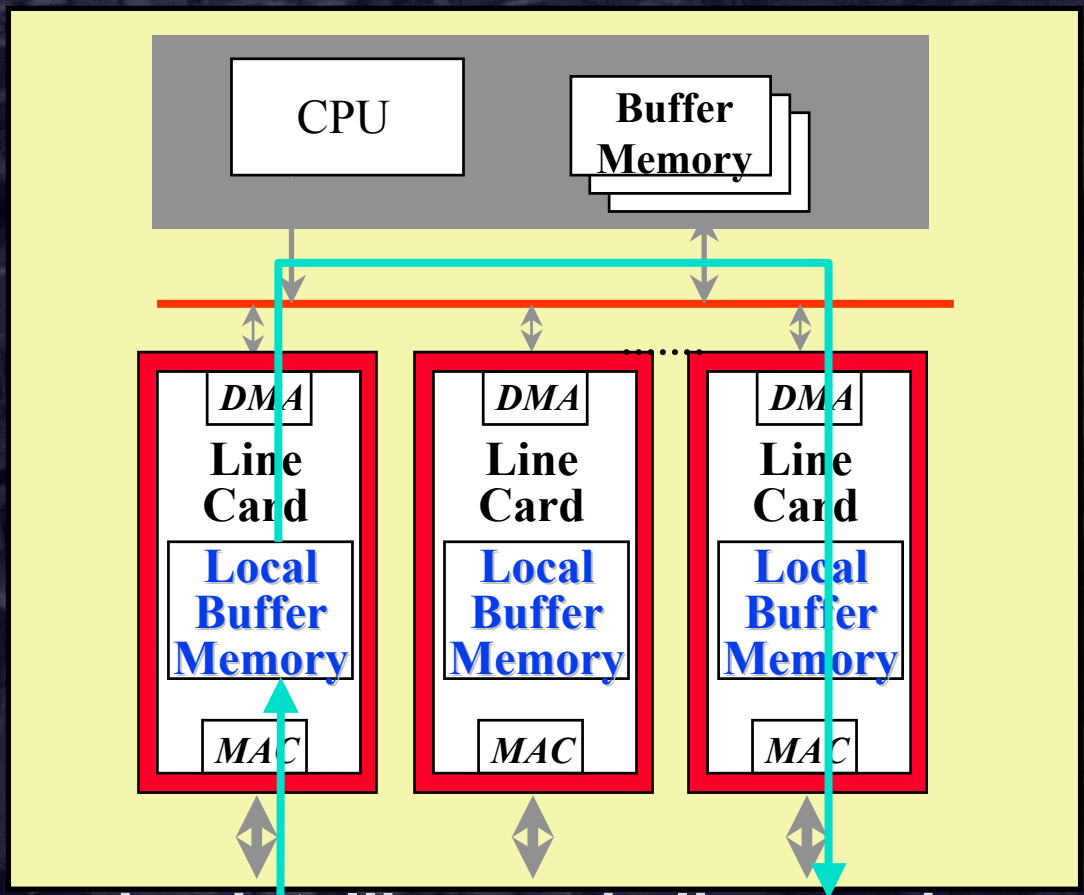
First-Generation IP Routers





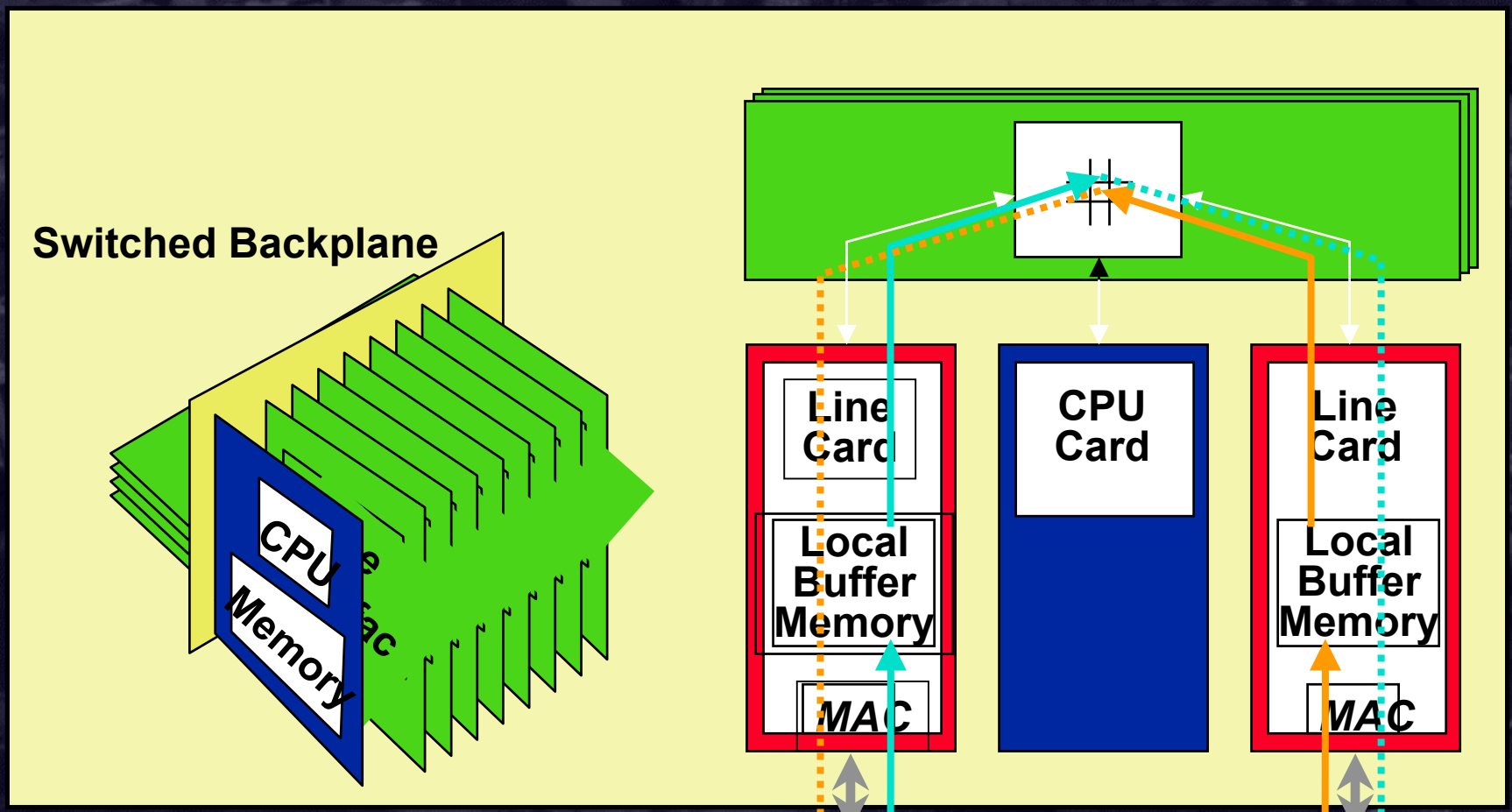
- Most Ethernet switches and cheap packet routers
- Bottleneck can be CPU, host-adaptor or I/O bus
- What is costly? Bus ? Memory? Interface? CPU?

Second-Generation IP Routers



- Port mapping intelligence in line cards
- Higher hit rate in local lookup cache
- What is costly? Bus ? Memory? Interface? CPU?

Third-Generation Switches/Routers



- Third generation switch provides parallel paths (fabric)
- What's costly? Bus? Memory, CPU?

Fourth-Generation Switches/Routers

Clustering and Multistage

Circuit switch

- A switch that can handle N **calls** has N logical inputs and N logical outputs
  - N up to 200,000
- Moves 8-bit samples from an input to an output port
  - Recall that **samples have no headers**
  - ***Destination*** of sample depends on ***time*** at which it arrives at the switch
- In practice, input trunks are ***multiplexed***
  - Multiplexed trunks carry ***frames*** = set of samples
- Goal: ***extract samples from frame***, and ***depending on position*** in frame, ***switch*** to output
  - each incoming sample has to get to the ***right output line*** and the ***right slot in the output frame***

Call blocking

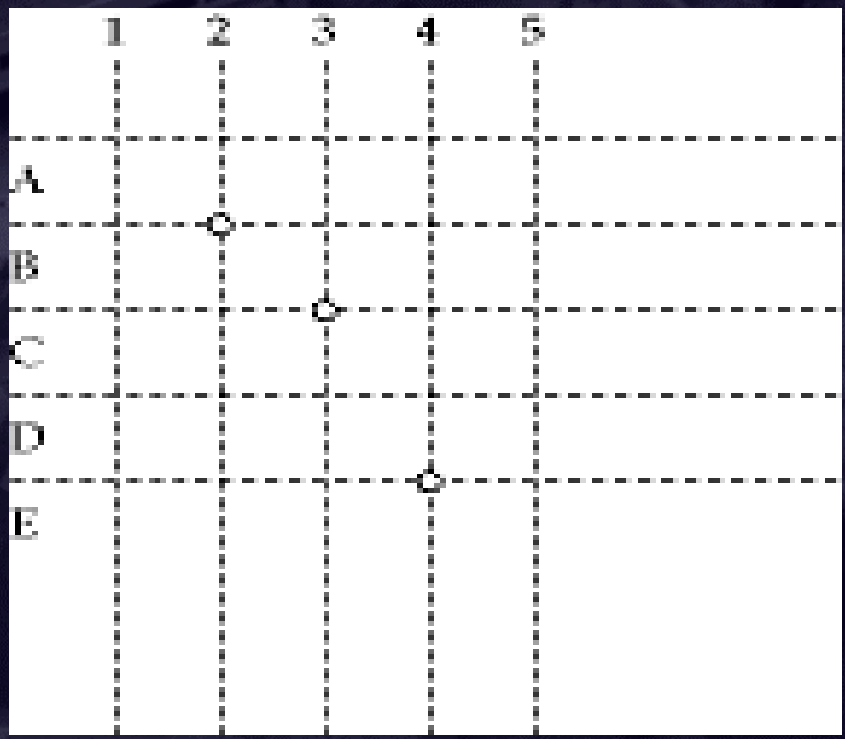
- Can't find a path from input to output
- **Internal** blocking
  - slot in output frame exists, but *no path*
- **Output** blocking
  - *no slot* in output frame is available
- Output blocking is reduced in *transit* switches
  - need to put a sample in one of *several* slots going to the desired next hop

Multiplexors and demultiplexors

- Most trunks time division multiplex voice samples
- At a central office, trunk is demultiplexed and distributed to active circuits
- Synchronous multiplexor
  - N input lines
  - Output runs N times as fast as input

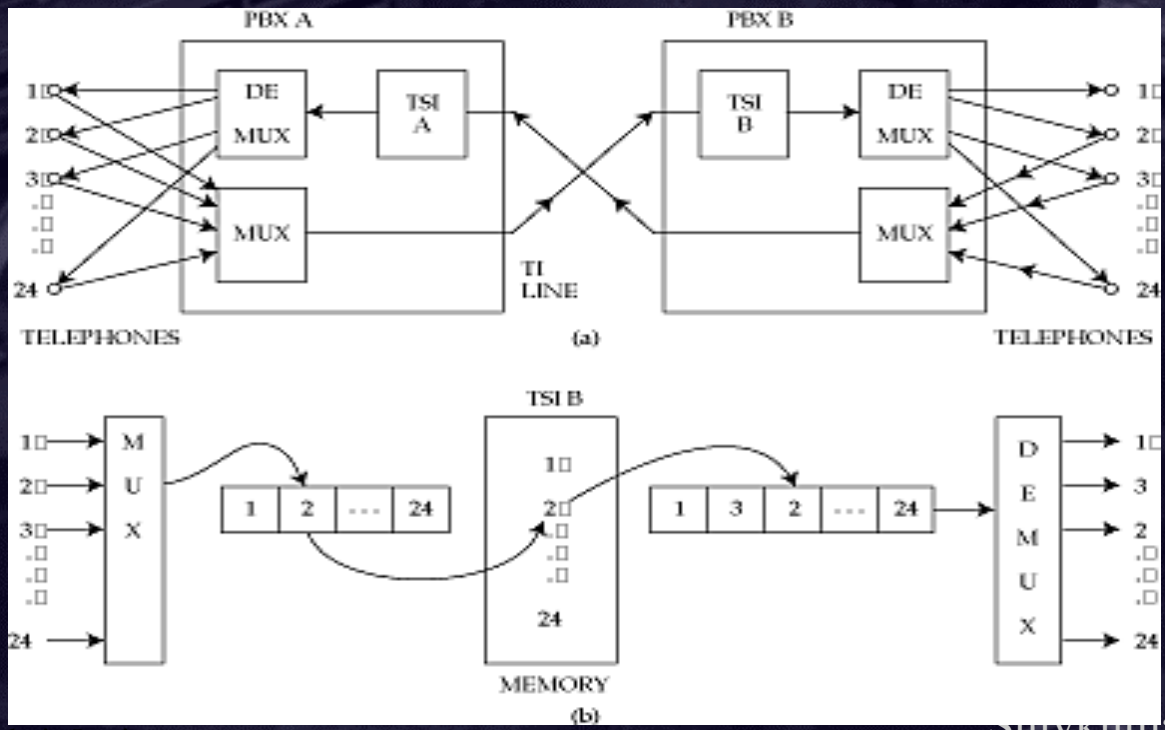
Switching: what does a switch do?

- **Transfers data from an input to an output**
  - many ports (density), high speeds
- Eg: Crossbar



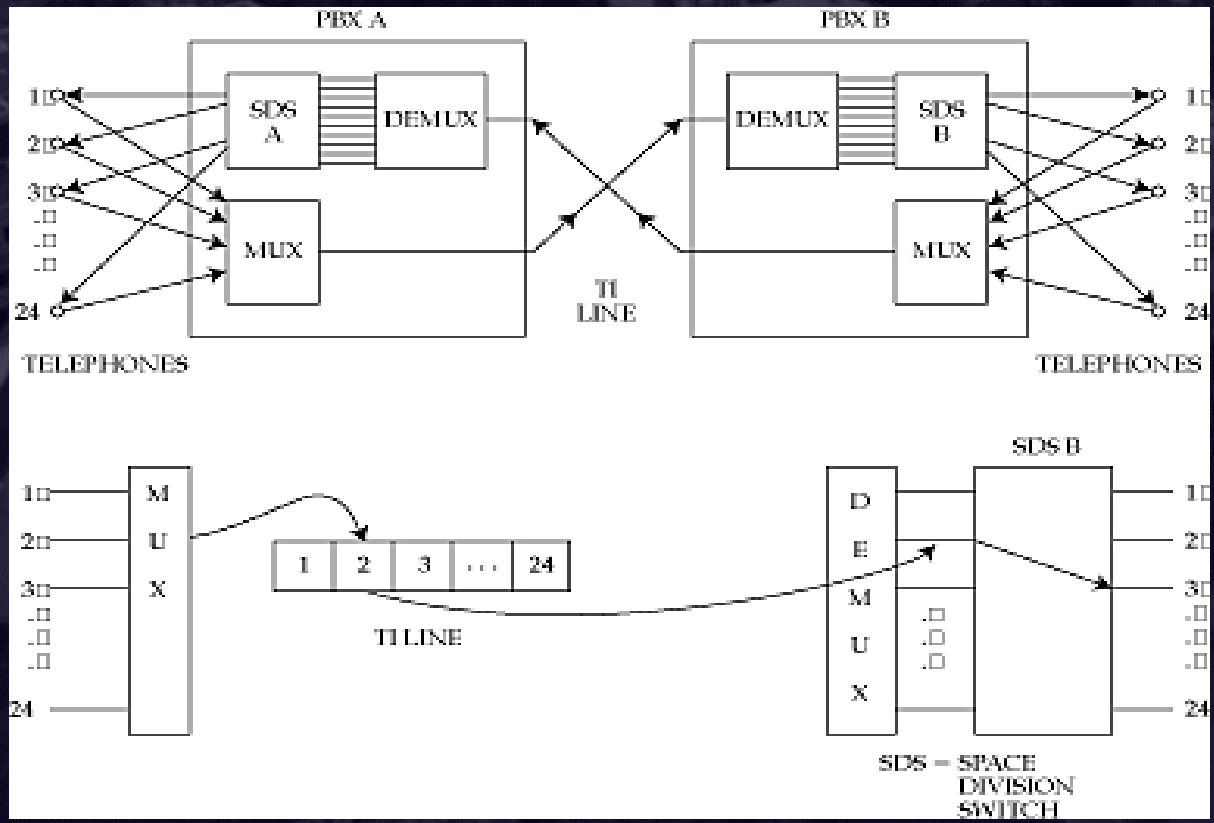
Time division switching

- Key idea: *when de-multiplexing, position in frame determines output trunk*
- Time division switching interchanges sample position within a frame: time slot interchange (TSI)



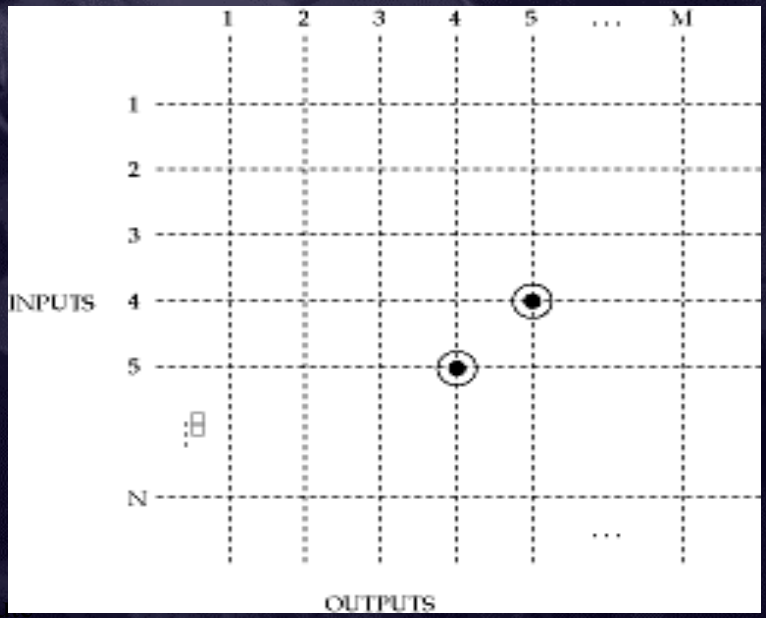
Space division switching

- *Each sample takes a different path through the switch*, depending on its destination



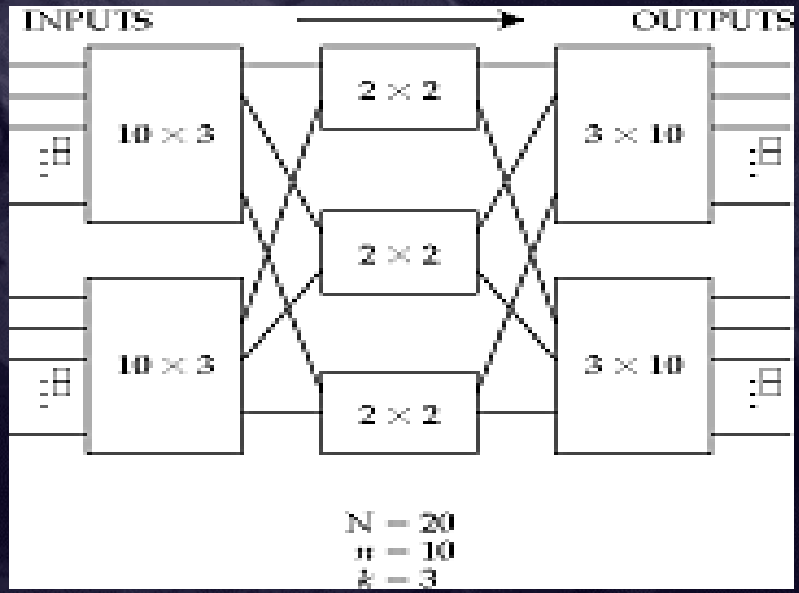
Crossbar

- Simplest possible space-division switch
- *Crosspoints* can be turned on or off, long enough to transfer a packet from an input to an output
- Internally nonblocking
  - but need N^2 crosspoints
  - time to set each crosspoint grows quadratically



Multistage crossbar

- In a crossbar during each switching time only one cross-point per row or column is active
- Can ***save crosspoints*** if a cross-point can ***attach to more than one input line*** (why?)
- This is done in a multistage crossbar
- Need to **rearrange connections every switching time**



Multistage crossbar

- Can suffer internal blocking
  - unless sufficient number of second-level stages
- Number of crosspoints < N^2
- Finding a path from input to output requires a depth-first-search
- Scales better than crossbar, but still not too well
  - 120,000 call switch needs ~250 million crosspoints

Packet switches

- In a circuit switch, path of a sample is determined at time of connection establishment
- No need for a sample header--position in frame used
- In a packet switch, packets carry a destination field or label
  - Need to *look up destination port on-the-fly*
- Datagram switches
  - lookup based on entire destination address (longest-prefix match)
- Cell or Label-switches
  - lookup based on VCI or Labels

Blocking in packet switches

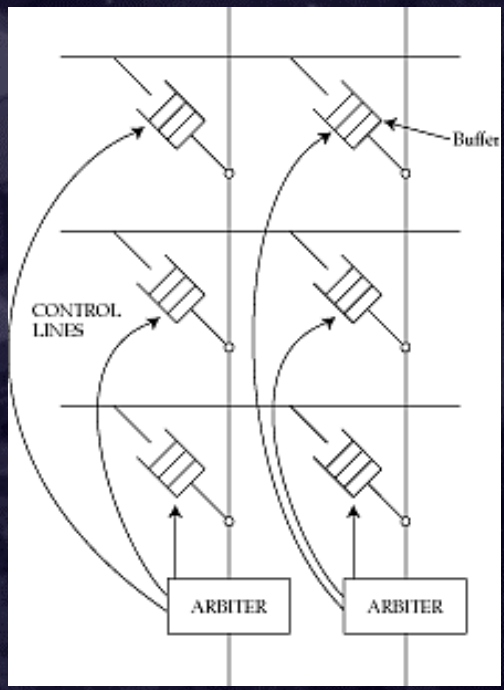
- Can have both internal and output blocking
- Internal
  - no path to output
- Output
  - trunk unavailable
- Unlike a circuit switch, *cannot predict if packets will block* (why?)
- If packet is blocked => must *either buffer or drop*

Dealing with blocking in packet switches

- Over-provisioning
  - internal links much faster than inputs
- Buffers
  - at input or output
- Backpressure
  - if switch fabric doesn't have buffers, prevent packet from entering until path is available
- Parallel switch fabrics
  - increases effective switching capacity

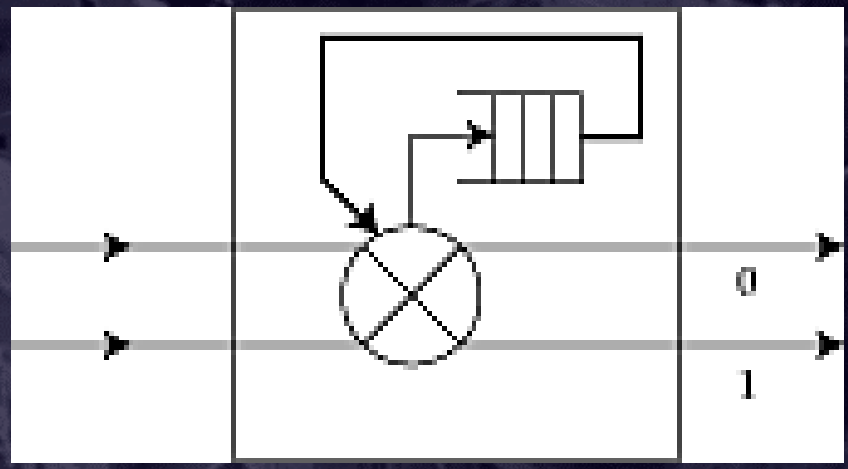
Switch Fabrics: Buffered crossbar

- What happens if packets at two inputs both want to go to same output?
- Can defer one at an input buffer
- Or, buffer cross-points: complex arbiter



Switch fabric element

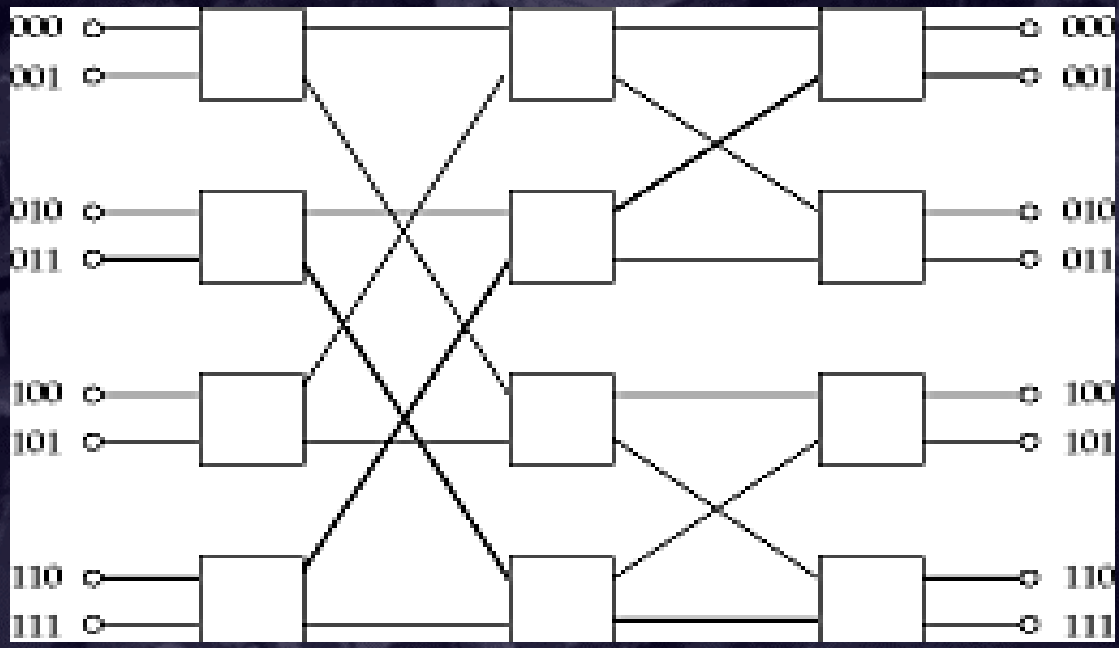
- Goal: towards building "self-routing" fabrics
- Can build complicated fabrics from a simple element



- Routing rule: if 0, send packet to upper output, else to lower output
  - If both packets to same output, buffer or drop

Banyan

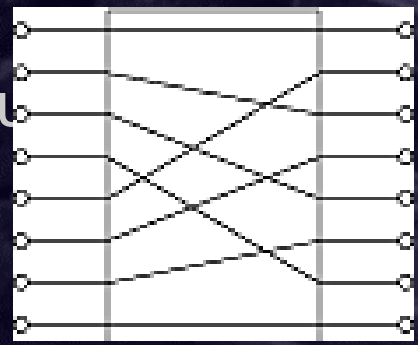
- Simplest self-routing recursive fabric



- What if two packets both want to go to the same output?
  - output blocking

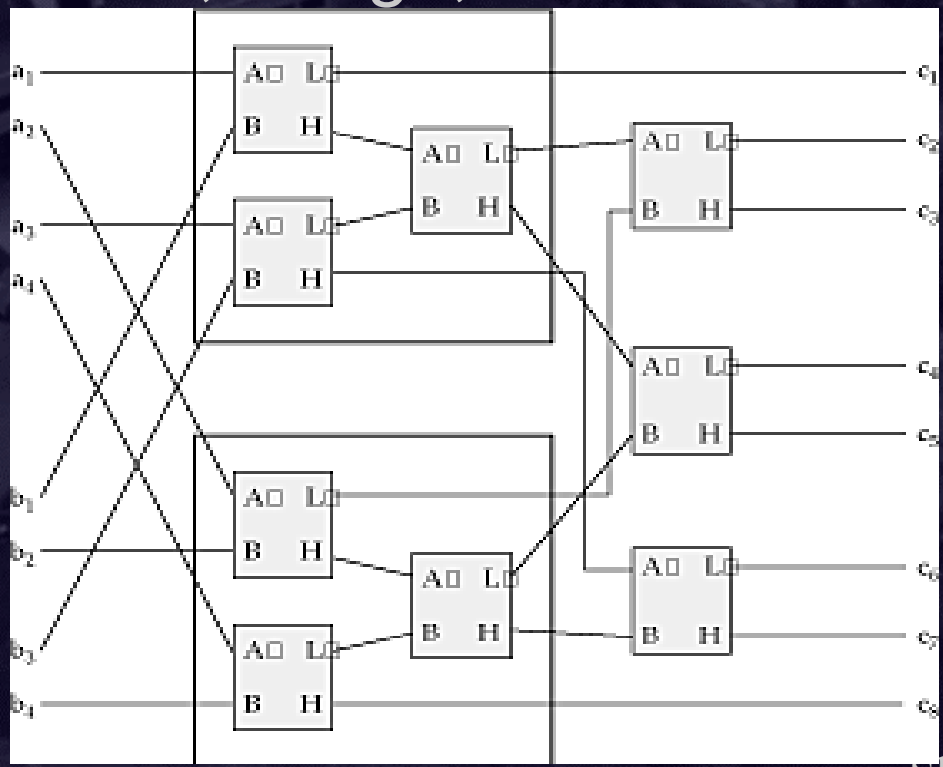
Blocking in Banyan S/ws: Sorting

- Can avoid blocking by choosing order in which packets appear at input ports
- If we can
  - present packets at inputs sorted by outpu
  - remove duplicates
  - remove gaps
  - precede banyan with a perfect shuffle stage
  - then no internal blocking
- For example: [X, 010, 010, X, 011, X, X, X]:
- Sort => [010, 011, 011, X, X, X, X, X]
- Remove dups => [010, 011, X, X, X, X, X, X]
- Shuffle => [010, X, 011, X, X, X, X, X]
- Need sort, shuffle, and trap networks

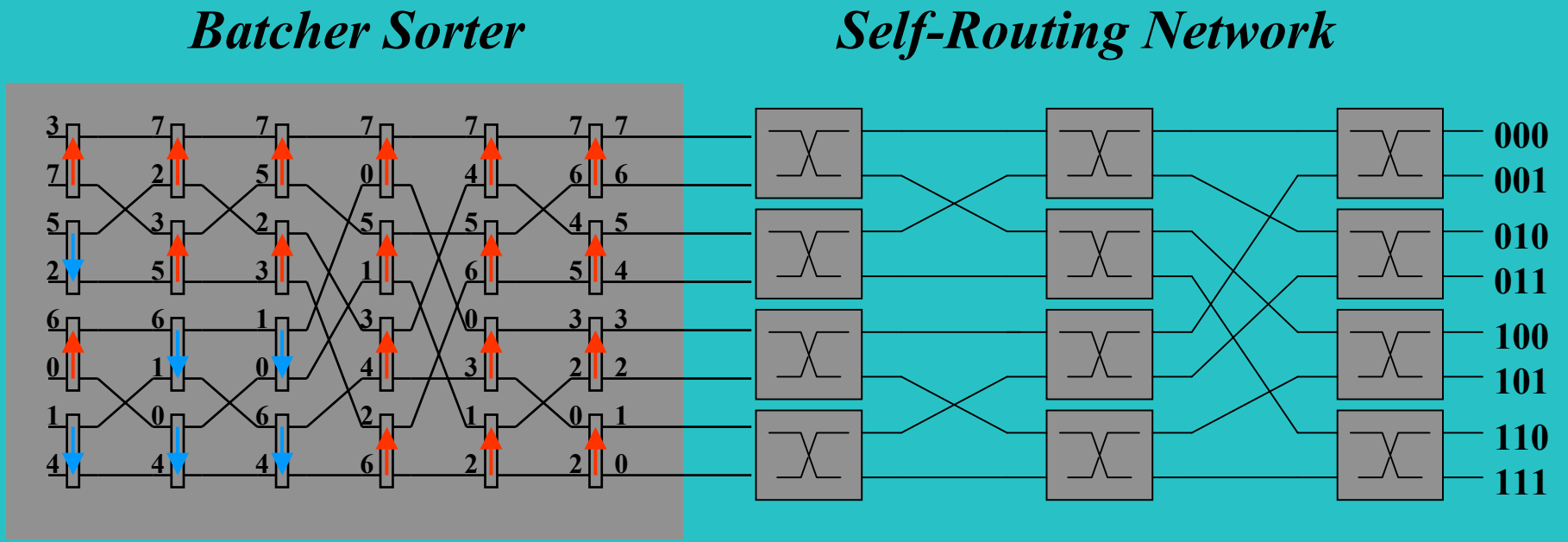


Sorting using Merging

- Build sorters from merge networks
- Assume we can merge two sorted lists
- Sort pairwise, merge, recurse

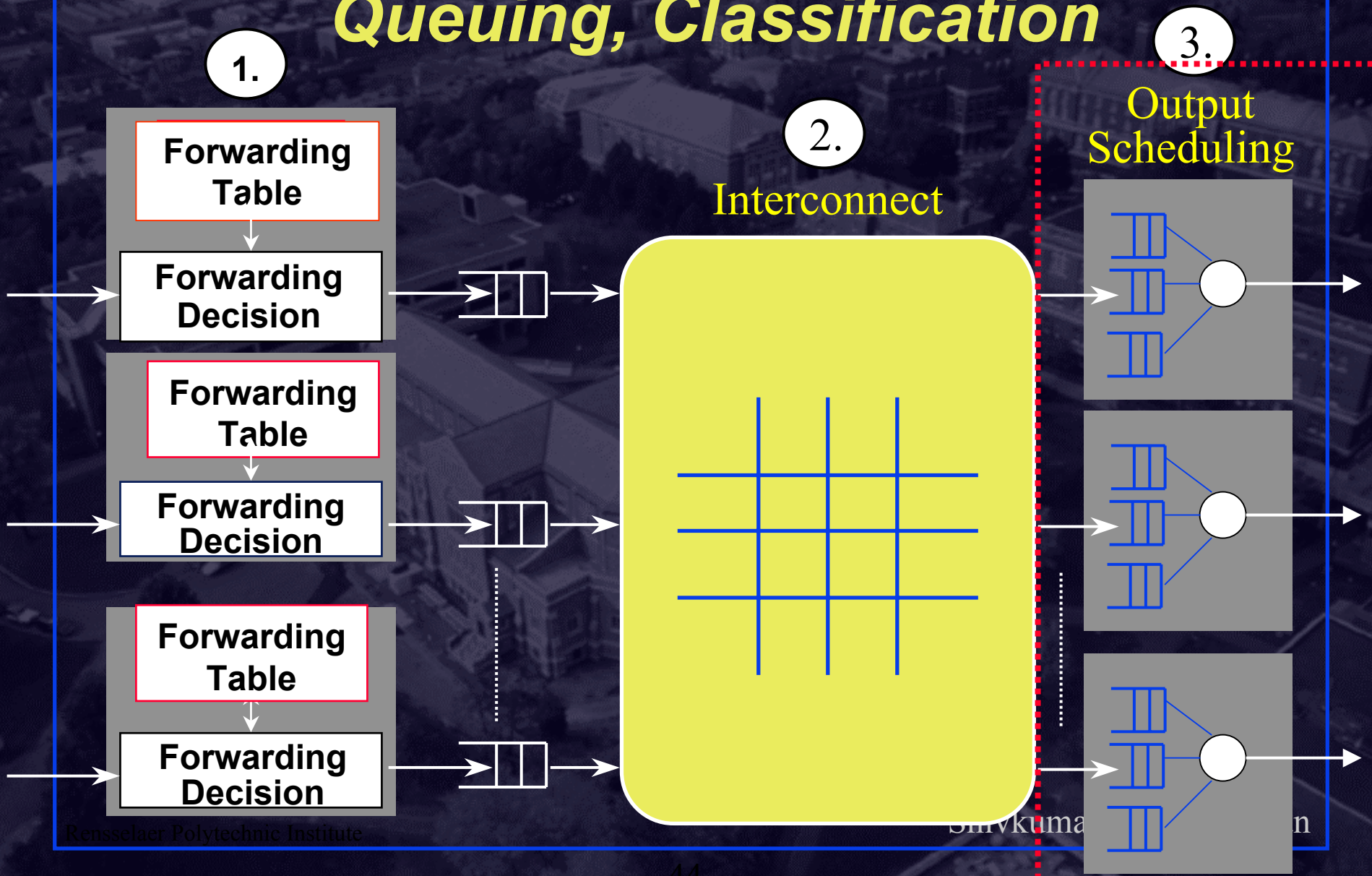


Non-Blocking Batcher-Banyan



- *Fabric can be used as scheduler.*
- *Batcher-Banyan network is blocking for multicast.*

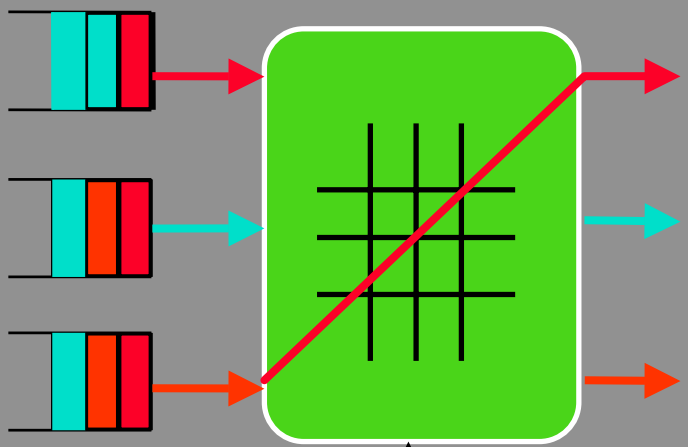
Basic Architectural Components: *Queuing, Classification*



Queuing:

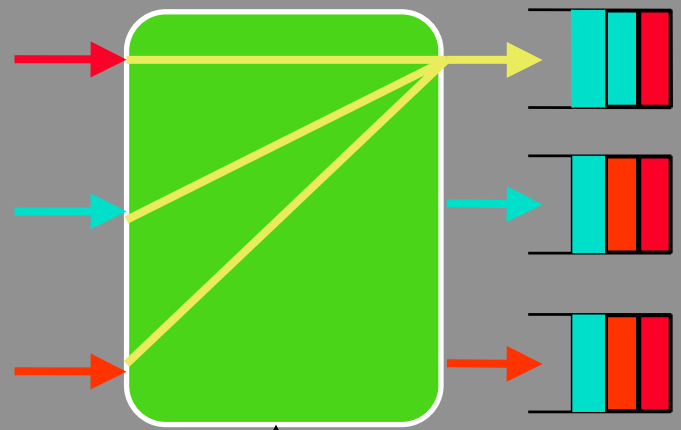
Two basic techniques

Input Queueing



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

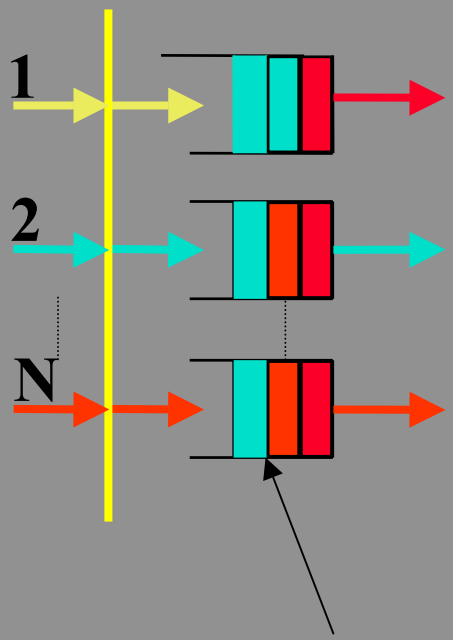
Output Queueing



Usually a fast bus

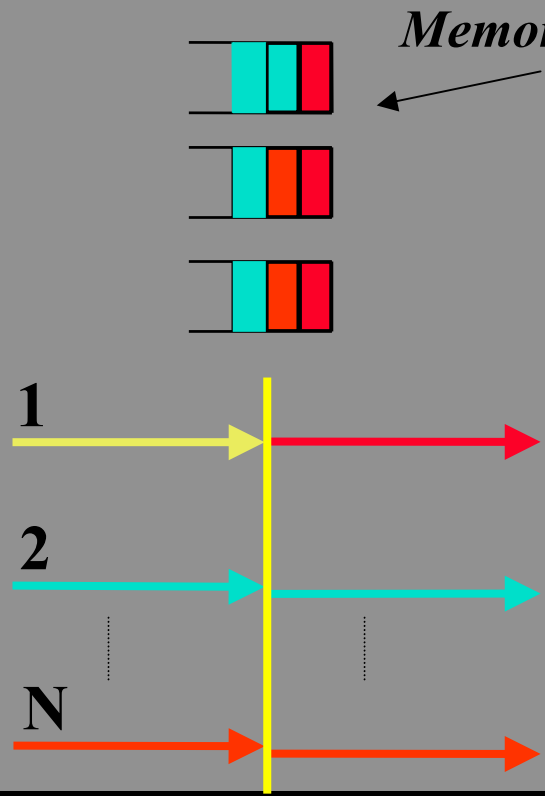
Queuing:
Output Queueing

Individual Output Queues



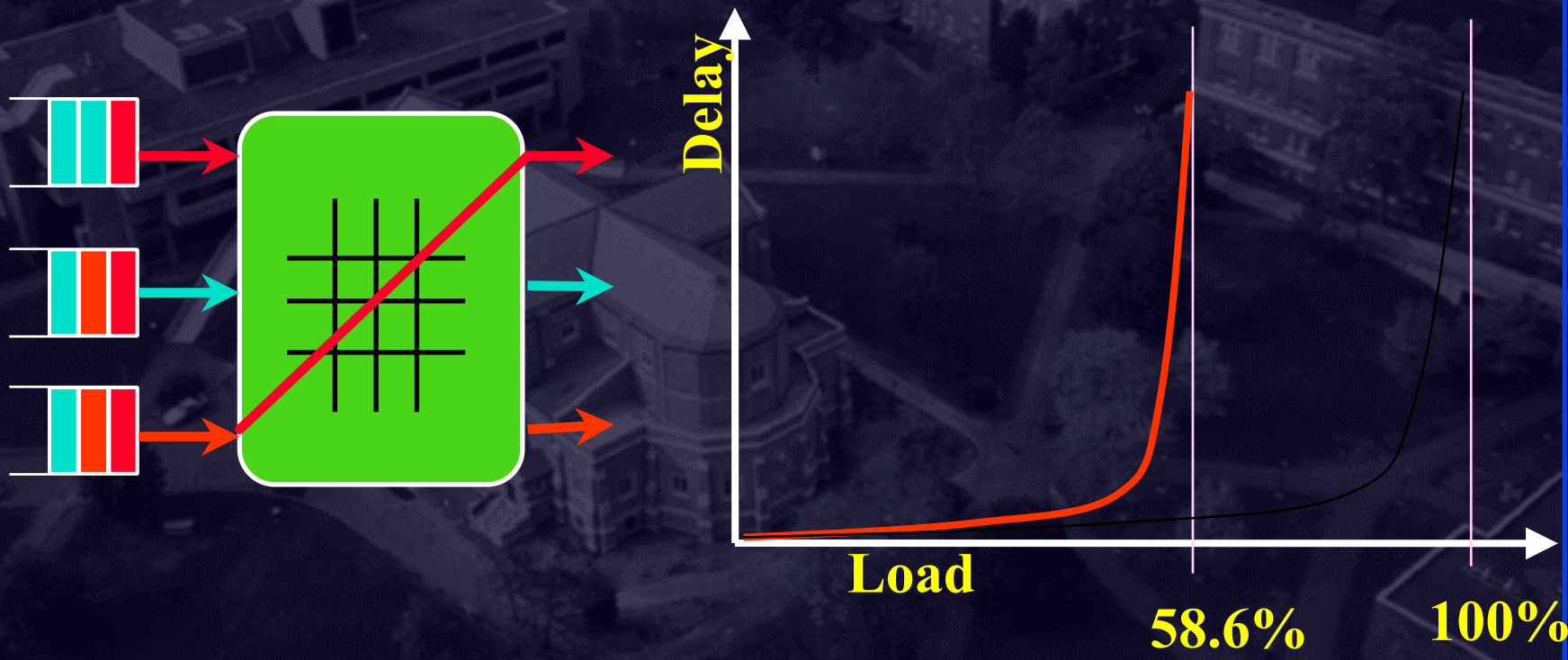
Memory b/w = (N+1).R

Centralized Shared Memory



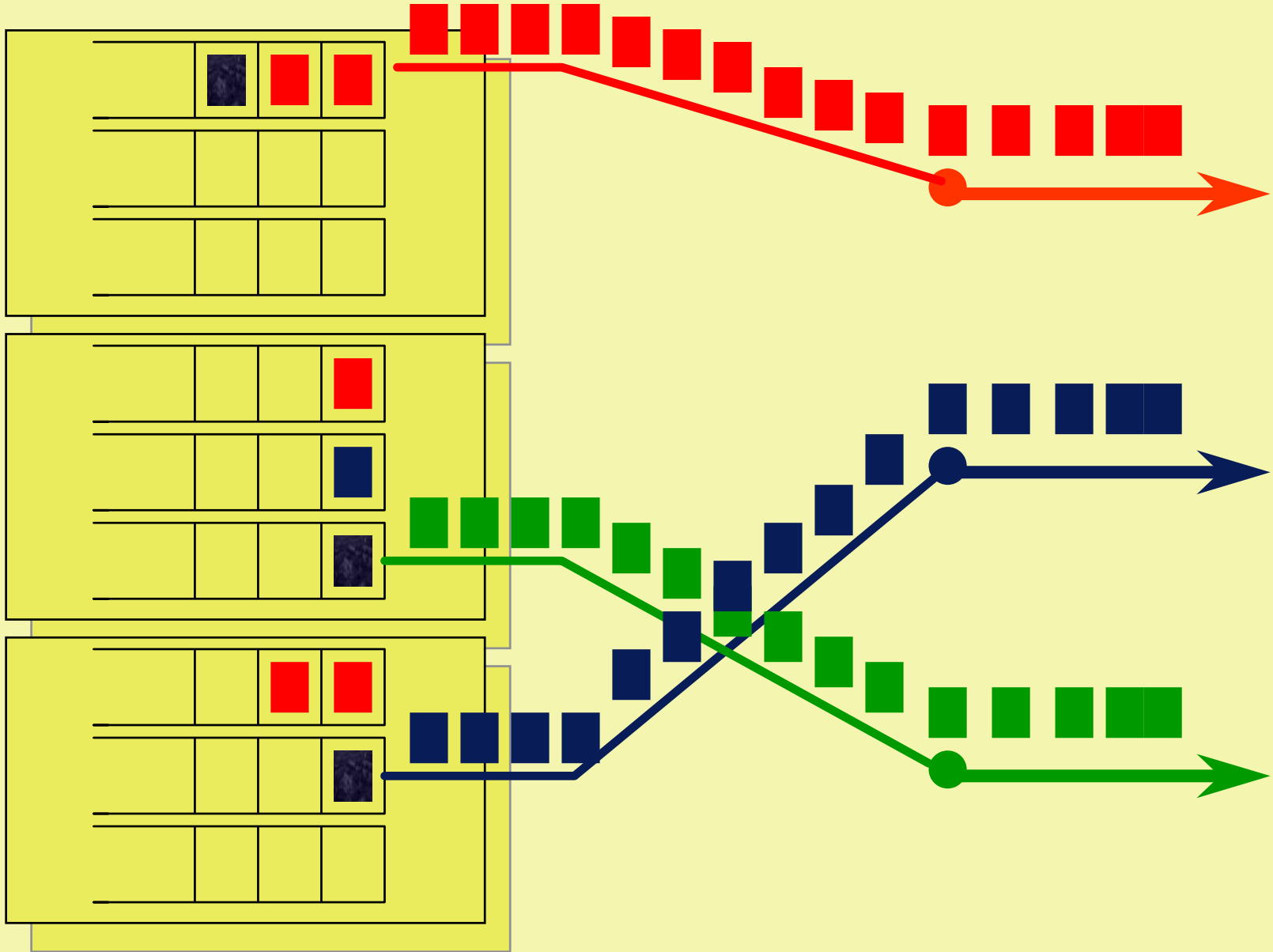
Input Queueing

Head of Line Blocking



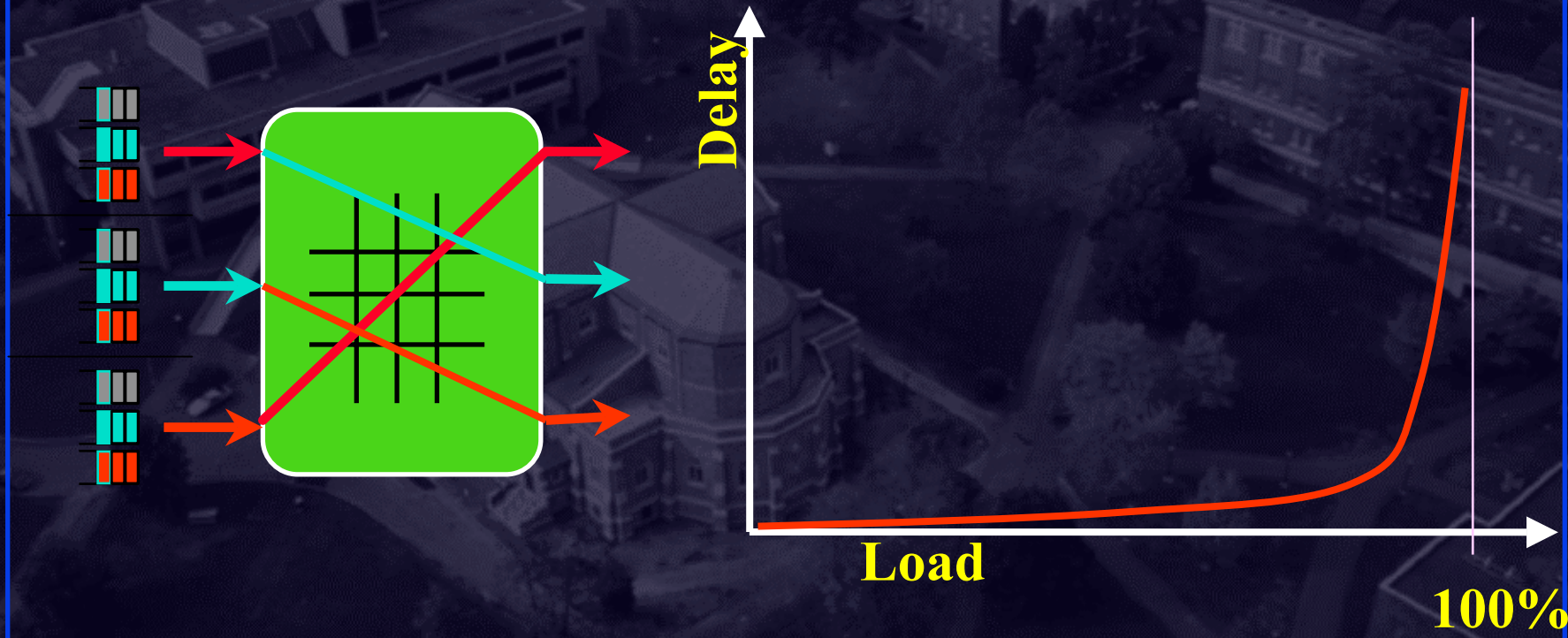
Solution: Input Queueing w/

Virtual output queues (VOQ)

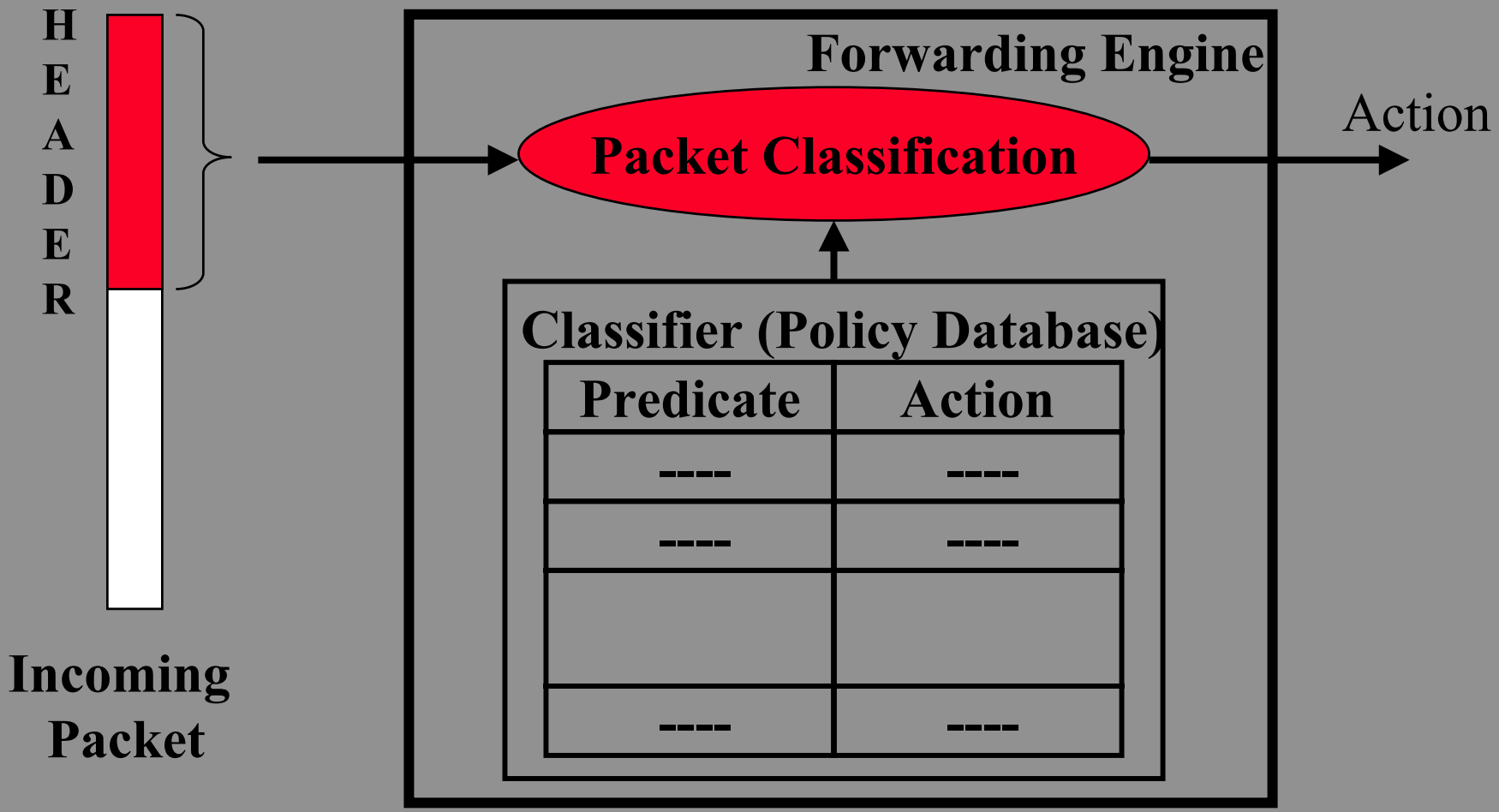


Input Queues

Virtual Output Queues



Packet Classification

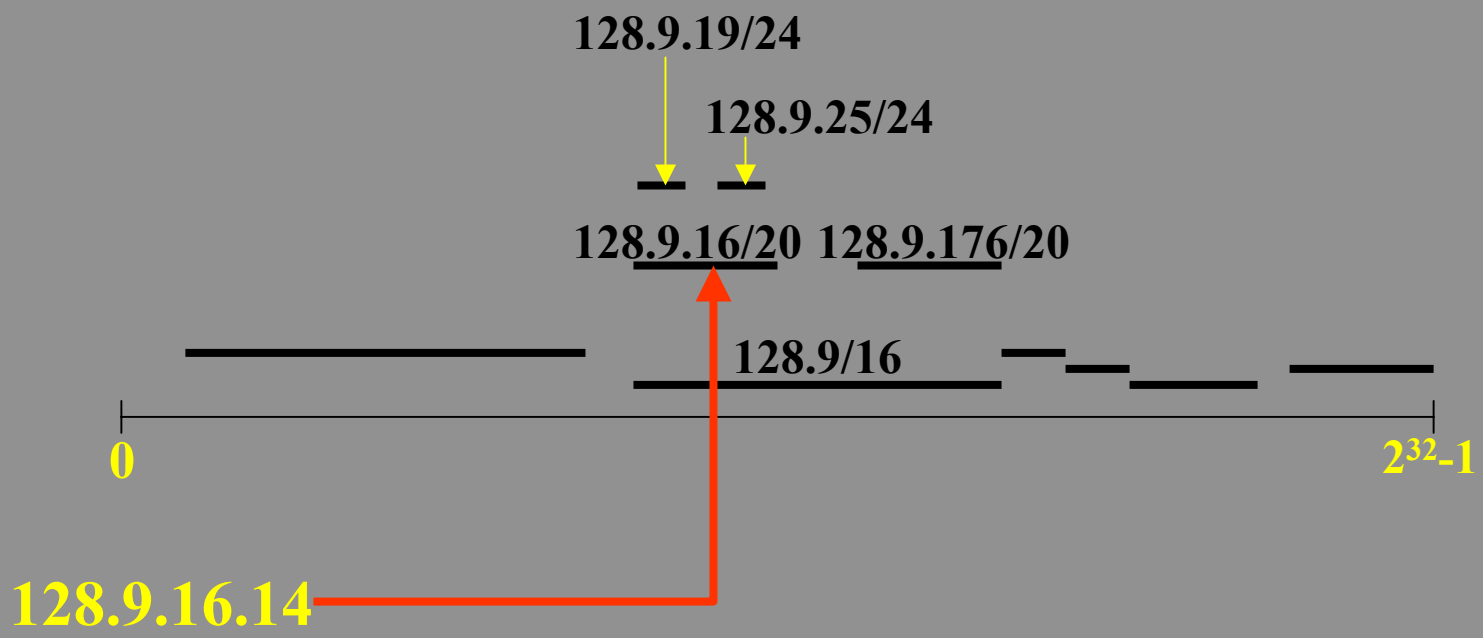


Multi-field Packet Classification

	Field 1	*Field 2*	*...*	*Field k*	*Action*
Rule 1	152.163.190.69/21	152.163.80.11/32	…	UDP	A1
Rule 2	152.168.3.0/24	152.163.0.0/16	…	TCP	A2
…	…	…	…	…	…
Rule N	152.168.0.0/16	152.0.0.0/8	…	ANY	An

Given a classifier with N rules, find the action associated with the highest priority rule matching an incoming packet.

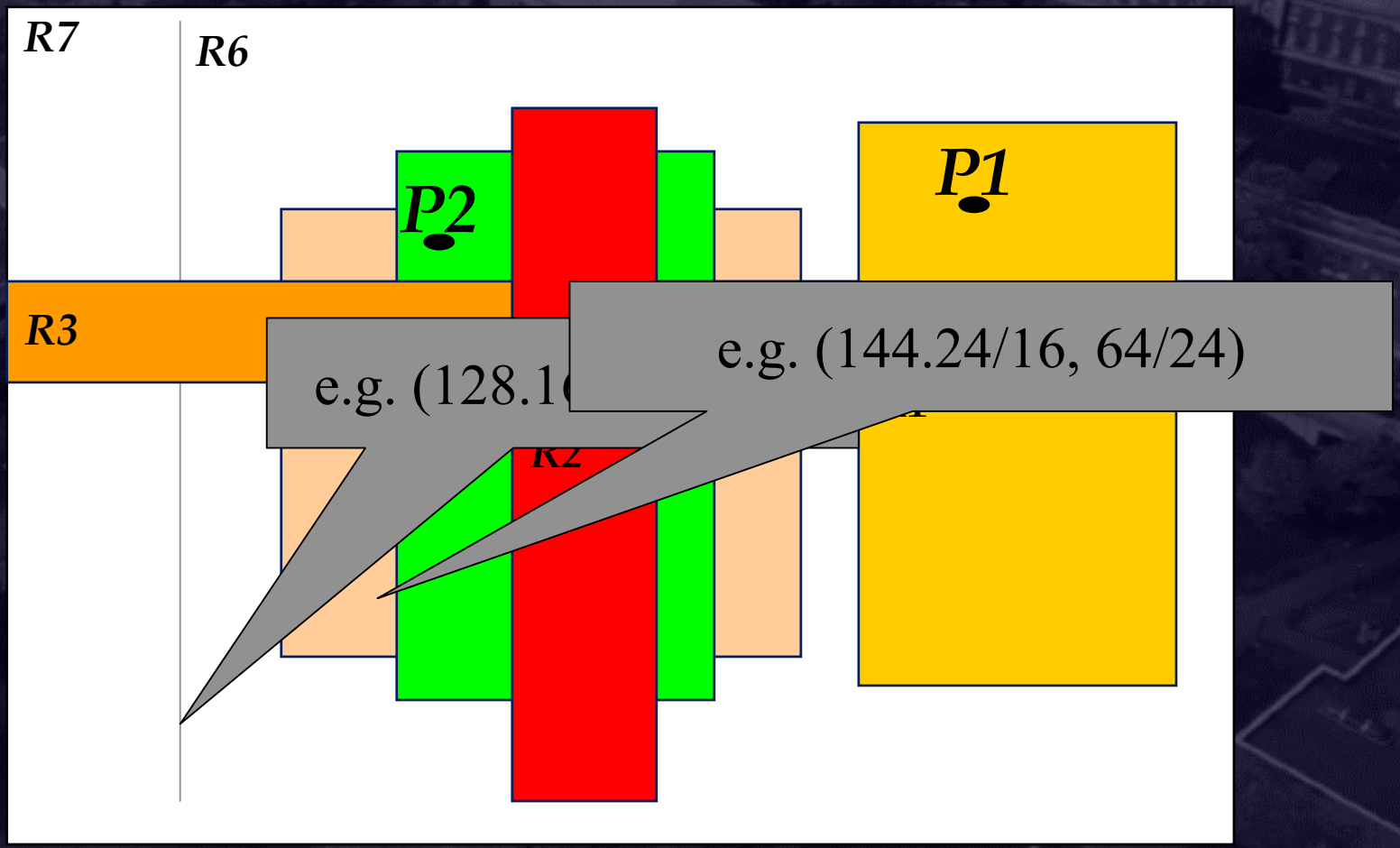
Prefix matching: 1-d range problem



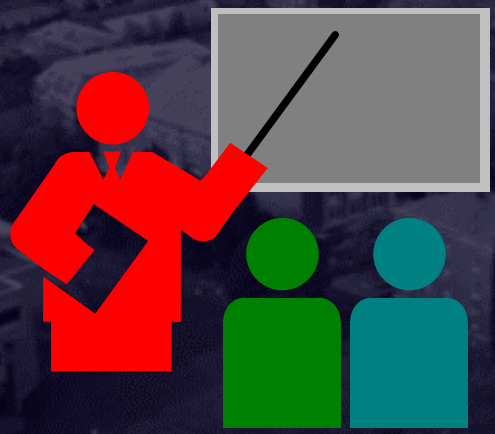
Most specific route = "longest matching prefix"

Classification: 2D Geometry problem

Field #1	Field #2	Data



Summary



- High speed routers: lookup, switching, classification, buffer management
- Lookup: Range-matching, tries, multi-way tries
- Switching: circuit s/w, crossbar, batcher-banyan,
- Queuing: input/output queuing issues
- Classification: Multi-dimensional geometry problem