Computer Networks COL 334/672

Data Plane

Slides adapted from KR

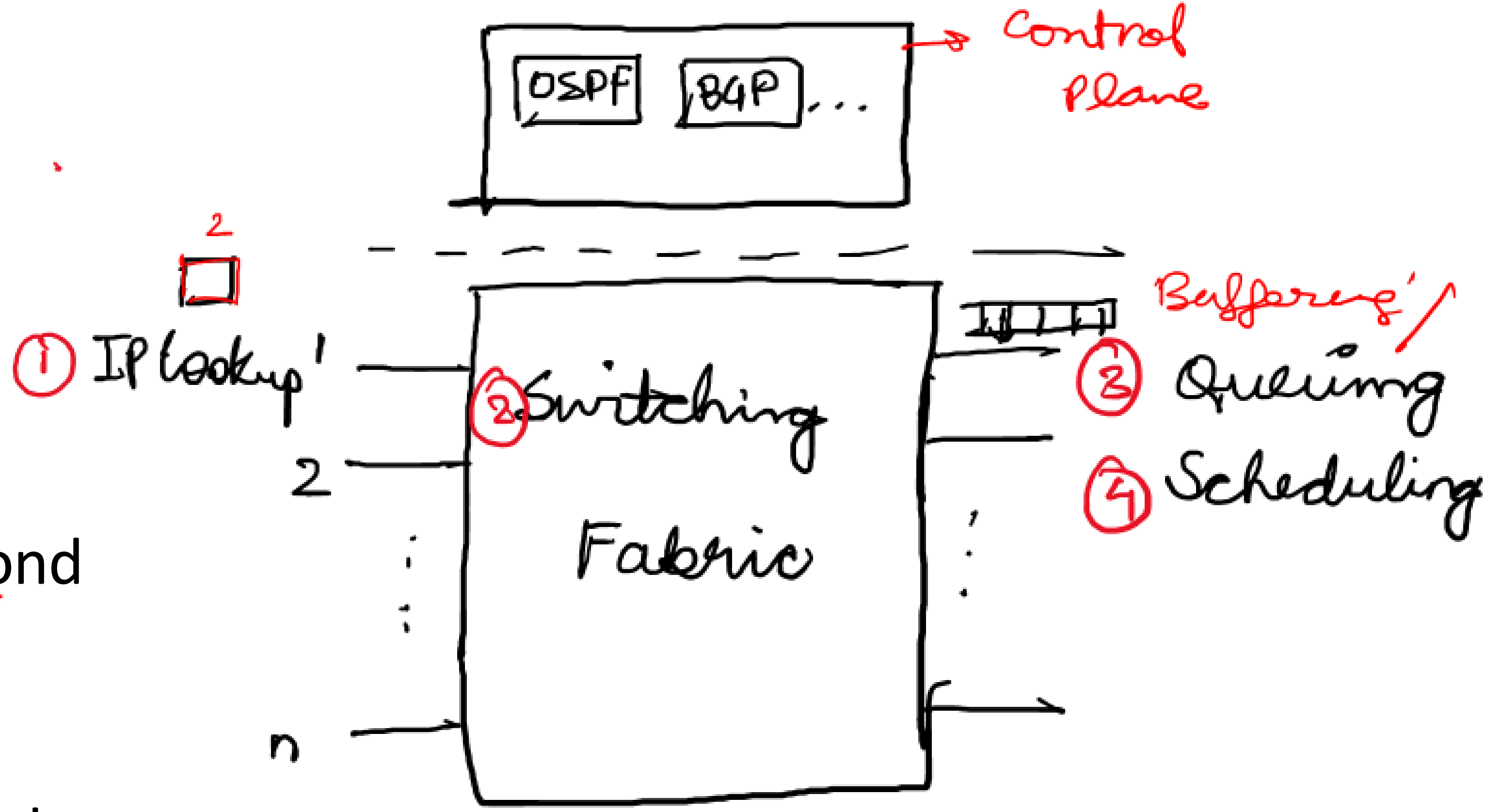
Sem 1, 2025-26

Today's Class: Data Plane Functions

- IP lookup
- Switching
- Queuing
- Scheduling

- **Challenge:** micro/nanosecond timescales

- **Learning goal:** Understand the problem statement, challenges, and a sample solution



IP Lookup

First 6-bits are same

00000000 1 16 11.11.0/24 – 1
00000001 2 11.11.1/24 – 2
00000010 2 11.11.2/24 – 2
00000011 2 11.11.3/24 – 2

→

11.11.0/24 – 1
11.11.0/22 – 2

- **Problem statement**: Finding the output port for a packet with destination IP, **D**

1. Prefixes of arbitrary length / (Classless Inter-domain Routing - CIDR)

2. A prefix can be a subset of another prefix (route aggregation / supernetting)

Longest prefix match

DST PREFIX	PORT
192.11/16	1
11.11.1/24	2
11.11.2/24	2
12.12.1/22	3
11.11.3./24	2
11.11.0/24	1

192.11/16	1
12.12.1/22	3
11.11.0/22	2
11.11.0/24	1

Longest prefix matching

longest prefix match

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011*** ********	2
otherwise	3

examples:

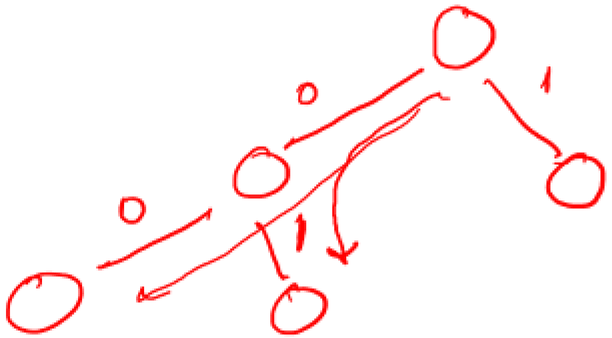
11001000 00010111 00010110 10100001 which interface? → 0

11001000 00010111 00011000 10101010 which interface? → 1

LPM Techniques

$\rightarrow O(n)$

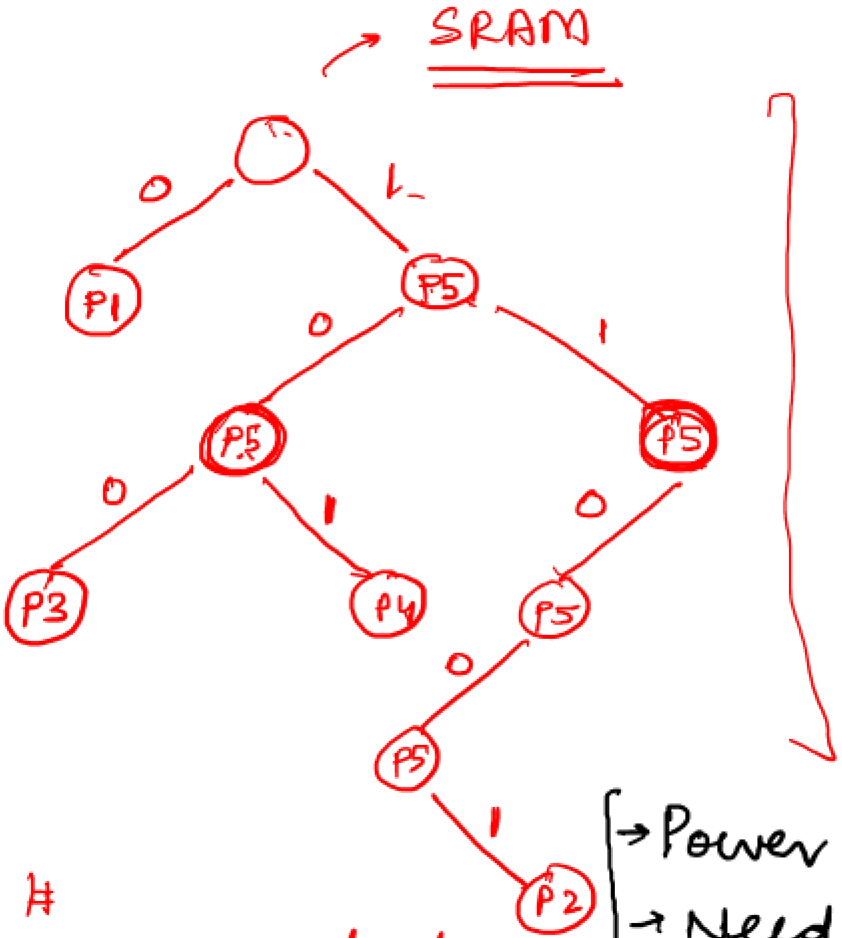
- Brute force: Search through all forwarding table entries iteratively
  - Too slow!
- Can we do a faster lookup?
- Solution: Use Trie or Prefix tree



01
00

Example: Unibit Tries

Prefix	Port
0*	P1
1*	P5
101*	P4
100*	P3
11001*	P2



How many memory lookups in the worst case?

Worst case # memory lookup

↳ 32 bit

Can we think of faster techniques?

→ can you do better?

option 2 → Uso multi-bit tries

→ Power hungry

→ Need more gates than SRAM

Content Addressable Memory

→ fully associative memory

→ O(1) lookups

Data Plane Functions

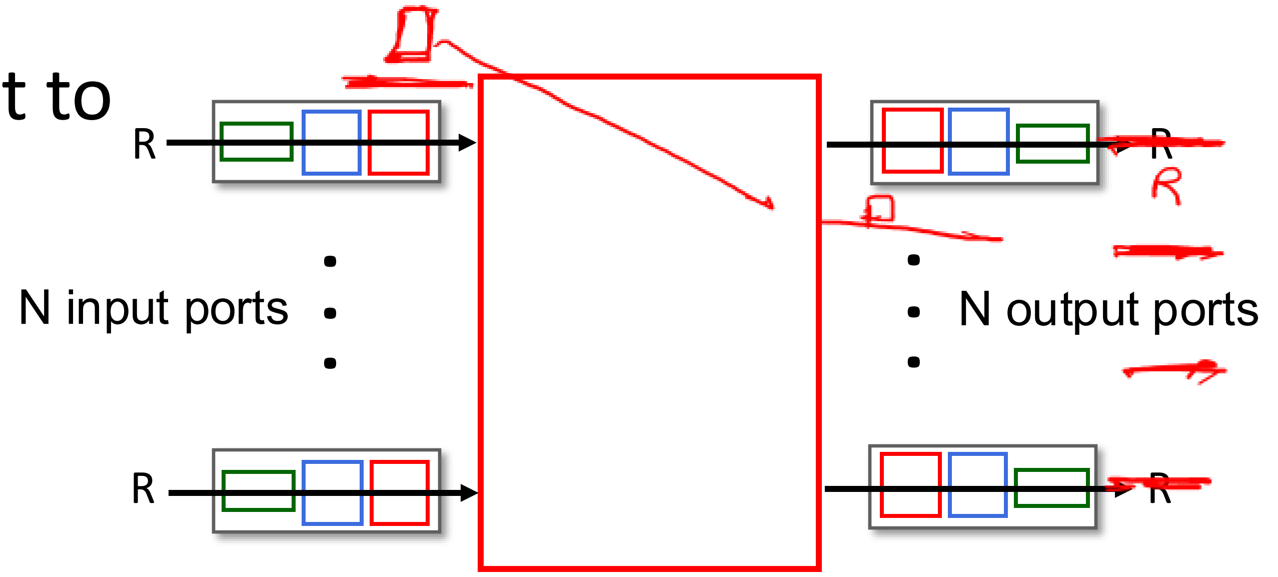
- Prefix lookup
- **Switching**
- Queuing
- Scheduling

Switching

Switching

- Transfer packets from input port to appropriate output port
- Switching rate: rate at which packets can be transfer from inputs to outputs
- Often measured as multiple of input/output line rate
- What is the desired switching rate in this case?
- Switching rate depends on switching fabric and switching algorithm at the router

N/W



switching rate ≥ Σ output rate of all links

≥ N×R

(Don't want switching to be the bottleneck)

Switching via memory

Router is an ASIC : App specific Integrated Circuit designed for routing & fwding

first generation routers:

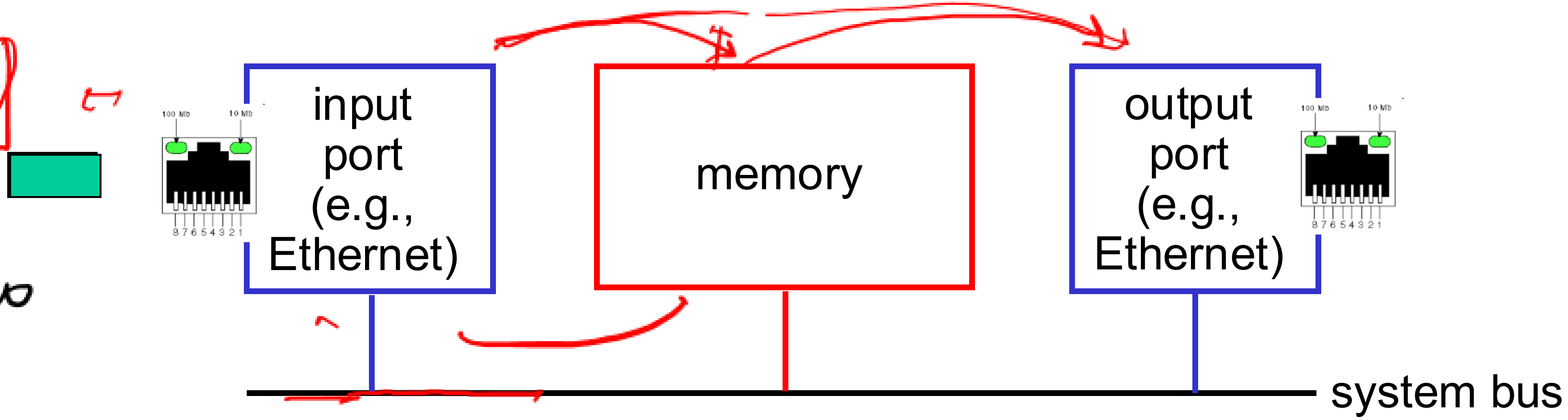
IP lookup in the processor/CPU & then forward the packet to output port

- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings)
- What is the switching rate if the bus rate is B?

Router processor

#2 [IP lookup here] ←

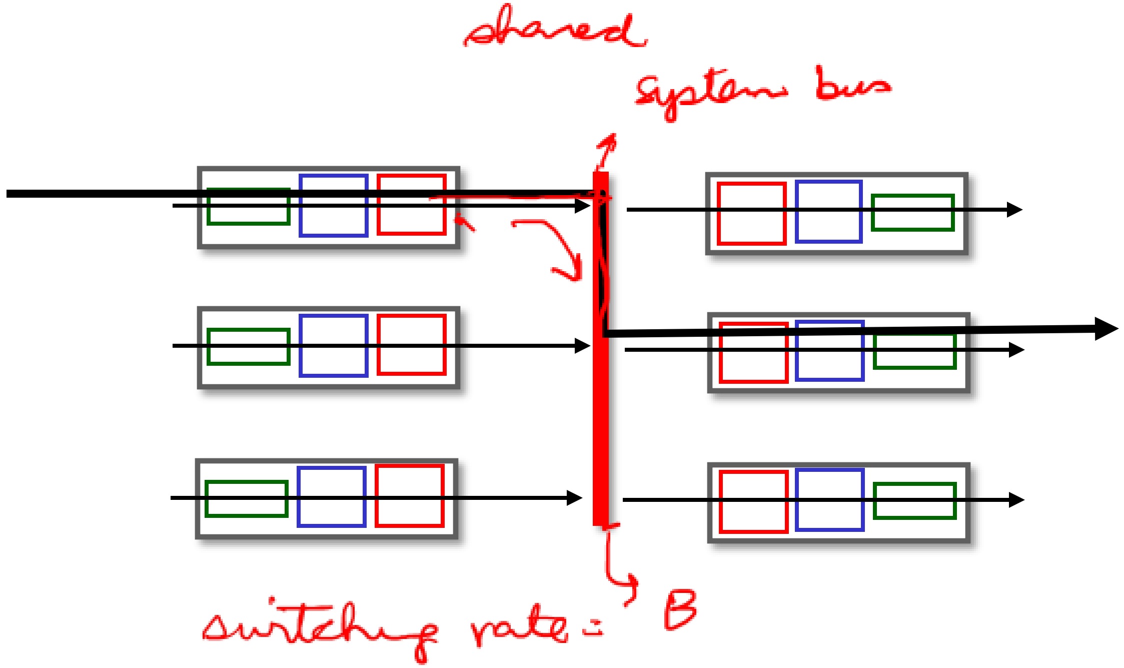
↑ Can we do better?



switching rate $= \frac{B}{2}$ (Need to send packet twice on the bus for forwarding)

Switching via a bus

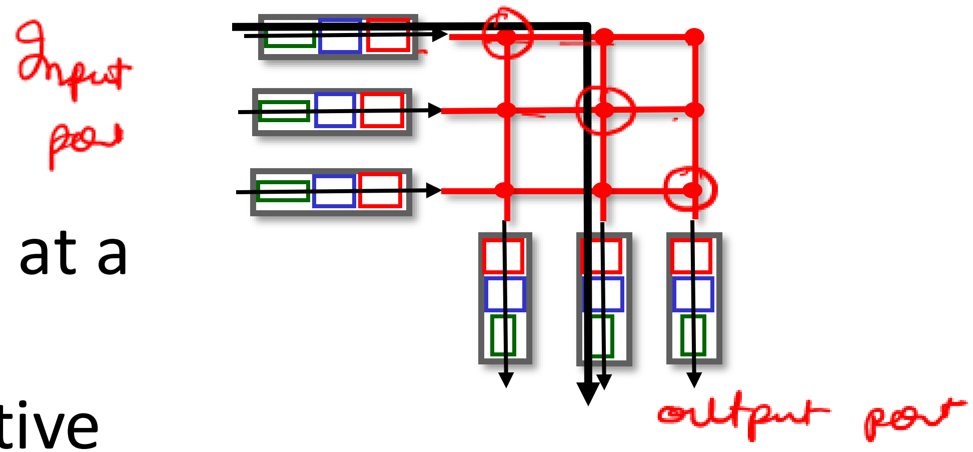
- datagram from input port memory to output port memory via a shared bus
- *bus contention:* switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access routers



Switching via interconnection network

- Grid of N inputs, N outputs
- How it works:
  - Each input can connect to one output at a time
  - Multiple input-output pairs can be active at the same time
- How to decide which input-output should be connected at each timeslot
  - Need to develop efficient switching algorithms

switching algorithm


Input
port
output port