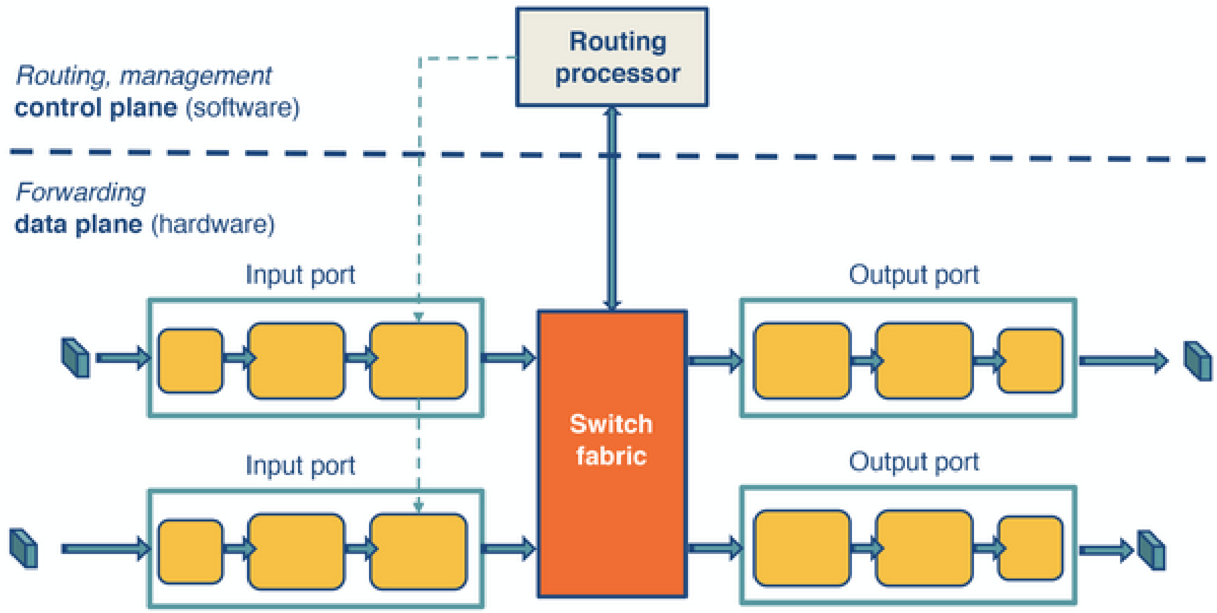
Router Design and Algorithms (Part 1)

Introduction

1. Routers forward data along the path from source to destination
    - Cover router architecture
    - Routers transfer the received packet from an input link interface to the appropriate output link interface towards the packet's destination
        - Determine output link by looking at the destination IP address and consulting the forwarding table
    - Need to handle packets based on multiple criteria
        - Flags
        - Quality of service
        - Security

What's Inside a Router?

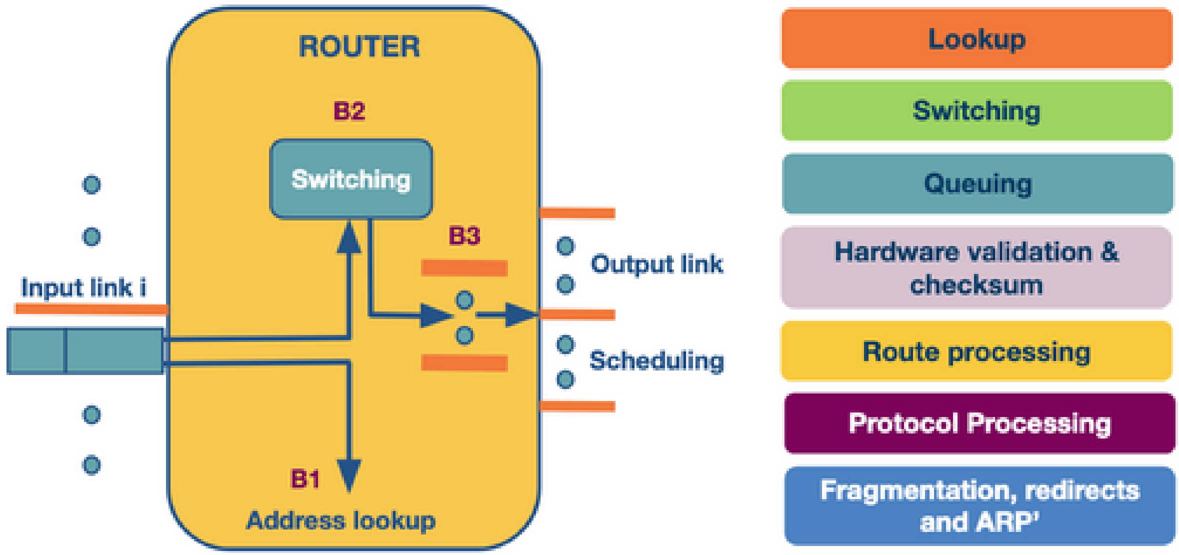
1. What are the basic components of a router?
    - Routers implement forwarding plane and control plane functions
    - Forwarding (or switching) function
        - Transfer packet from input link interface to output link interface
        - Very short timescales (nanoseconds)
        - Typically implemented in hardware
2. Basic Components
    - Input ports
        - Physically terminate the incoming links to the router
        - Data link processing unit decapsulates the packets
        - Input ports perform the lookup function (consult forwarding table to ensure each packet is forwarded to the appropriate output port through the switch fabric)
    - Switching fabric
        - Moves packets from input ports to output ports
        - Three types
            * Memory
            * Bus
            * Crossbar
    - Output ports
        - Receive and queue the packets from the switching fabric and send them over to the outgoing link
    - Control plane functions (processor)
        - Implement the routing protocols
        - maintain the routing tables
        - Compute the forwarding table
        - Implemented in software in the routing processor, or by a remote controller in the case of SDN



Router Components

Router Architecture

1. Model of a router
   - Lookup: When a packet arrives at the input link, the router looks at the destination UP address and determines the output link by looking at the forwarding table
     - FIB: Forwarding Information Base
     - FIB provides a mapping between destination prefixes and output links
     - Routers use the longest prefix matching algorithms to resolve any ambiguities
     - Packet classification: Performing lookup based on destination or source IP addresses, port, and other criteria
   - Switching: Transfers the packet from the input link to the output link
     - Modern routers use corssbar switches for this task
     - Scheduling the switch is difficult because multiple inputs may want to send packets to the same output
   - Queueing: After the packet has been switched to a specific output, it will need to be queued if the link is congested
     - Could be FIFO
     - Could be more complex (weighted fair queueing) to provide delay guarantees or fair bandwidth allocation
2. Time sensitive tasks
   - Header validation and checksum: Router checks the packet's version number, decrements the time-to-live (TTL) field, recalculates the header checksum
   - Route processing: Routers build forwarding tabels using routing protocols like RIP, OSPF, and BGP
   - Protocol processing: Need to implement the following protocols
     - Simple Network Management Protocol (SNMP) for remote inspection
     - TCP and UDP for remote communication with the router
     - Internet Control Message Protocol (ICMP) for sending error messages, e.g., when time-to-live (TTL) time is exceeded



Router Architecture

Different Types of Switching

1. Switching fabric is the brain of the router (forwards packets from input to output)
    - Switching via memory
        - Input/output ports operate as I/O devices in an operating system, controlled by the routing processor
        - When an input receives a packet, it sends an interrupt to the routing processor and the packet is copied to the processor's memory
        - Processor extracts the destination address and looks into the forward table to find the output port
        - Packet is copied into that output's port buffer
    - Switching via bus
        - Routing processor does not intervene as with memory
        - Input port puts an internal header that designates the output port and sends the packet to the shared bus
        - All output ports receive the packet, but only the designated one will keep it
        - When the packet arrives at the designated output port, the internal header is removed from the packet
        - Speed of the bus limits the speed of the router
    - Switching via interconnection network
        - Crossbar switch is an interconnection network that connects N input ports to N output ports using 2N buses
        - Horizontal buses meet the vertical buses at crosspoints controlled by the switching fabric
        - Crossbar network can carry multiple packets at the same time, as long as they are using different input and output ports

Quiz 1

1. The data plane functions of a traditional router are implemented in hardware.
2. The control plane functions of a traditional router are implemented in software.
3. Which plane operates on a shorter timescale?
    - Control

- Data (true)
- Management
- All planes opearte on the same timescale

4. Classify each function as an operation of either the data plane or control plane
    - Computing paths based on a protocol = Control
    - Forwarding packets at layer 3 = Data
    - Switching packets at layer 2 = Data
    - Running protocols to build a routing table = Control
    - Running the Spanning Tree protocol = Control
    - Decrementing Time To Live (TTL) = Data
    - Computing an IP header checksum = Data
    - Running a protocol/logic to configure a middle box device for load balancing = Control
    - Forwarding packets according to installed rules in a middlebox device = Data
5. Which, if any, of the following types of switching can send multiple packets across the fabric in parallel?
    - Interconnection Network/Crossbar

The Challenges Routers Face

1. Fundamental problems routers face
    - Bandwidth and Internet population scaling
        - Increasing number of devices that connect to the Internet
        - Increasing volumes of network traffic due to new applications
        - New technologies such as optical links that can accomodate higher volumes of traffic
    - Services at high speeds
        - New applications require services such as protection against delays in the presence of congestion and protection during attacks or failures
2. Router bottlenecks
    - Longest prefix matching: Increasing number of Internet hosts and networks has made it impossible for routers to have explicit entries for all possible destinations
        - Instead, routers group destinations into prefixes
        - Algorithms for efficient longest prefix matching become more complex
    - Service differentiation: Routers can also offer service differentiation which means different quality of service (or security) guarantees
        - Requires routers to classify packets based on more complex criteria beyond destination
    - Switching limitations: Crossbar switching employs parallelism to deal with high-speed traffic
        - Causes problems (head of line blocking)
    - Bottlenecks and services: Providing performance guarantees (quality of service) at high speeds is nontrivial, as is providing support for new services such as measurements or security guarantees

Bottleneck	Cause	Sample Solution
Exact lookups	Link speed scaling	Parallel hashing
Prefix lookups	Link speed scaling Prefix database size scaling	Compressed multibit tries
Packet classification	Service differentiation Link speed and size scaling	Decision tree algorithms Hardware parallelism (CAMS)
Switching	Optical-electronic speed gap Head-of-line blocking	Crossbar switches Virtual output queues
Fair queueing	Service differentiation Link speed scaling Memory scaling	Weighted fair queueing Deficit round robin DiffServ, Core Stateless
Internal bandwidth	Scaling of internal bus speeds	Reliable striping
Measurement	Link speed scaling	Juniper's DCU
Security	Scaling in number and intensity of attacks	Traceback with bloom filters Extracting worm signatures

Router Bottlenecks

Prefix-Match Lookups

1. What is prefix matching?
   - As the Internet grows in terms of networks (AS numbers) and IP addresses, one of the challenges that a router faces is scalability
     - Help this by grouping multiple IP addresses by the same prefix
2. Prefix notation
   - Dot decimal
     - 132.234
     - 1000010011101010*
   - Slash notation
     - 132.234.0.0/16
     - 16 denotes only the first 16 bits are relevant for prefixing
   - Masking
     - 132.234.0.0 with mask 255.255.0.0
     - 255.255.0.0 denotes that only the first 16 bits are important
3. Why do we need variable-length prefixes?
   - In the early days of the Internet, we used an IP addressing model based on classes (fixed-length prefixes)
   - As IP addresses were exhausted, the Classless Internet Domain Routing (CIDR) came into effect in 1993
     - CIDR assigns IP addresses using arbitrary-length prefixes and has helped to decrease the router table size
     - New problem: longest-matching prefix lookup
4. Why do we need (better) lookup algorithms?
   - Router challenges when looking up output port:
     - Lookup speed
     - Memory

    - Update time
- Observations
    - Measurement studies on network traffic had shown a large number of concurrent flows of short duration; caching solutions will not work efficiently
    - Important element of any lookup opration is how fast it is done (lookup speed) and a large part of the cost for computation is accessing memory
    - Unstable routing protocol may adversely impact the udpate time in the table to add, delete, or replace a prefix
        * Inefficient routing protocols increase this value up to additional milliseconds
    - Vital trade-off is memory usage
        * Expensive, fast memory: Cache in software, SRAM in hardware
        * Cheaper, slow memory: DRAM, SDRAM

Quiz 2

1. Consider a router with the following forwarding table

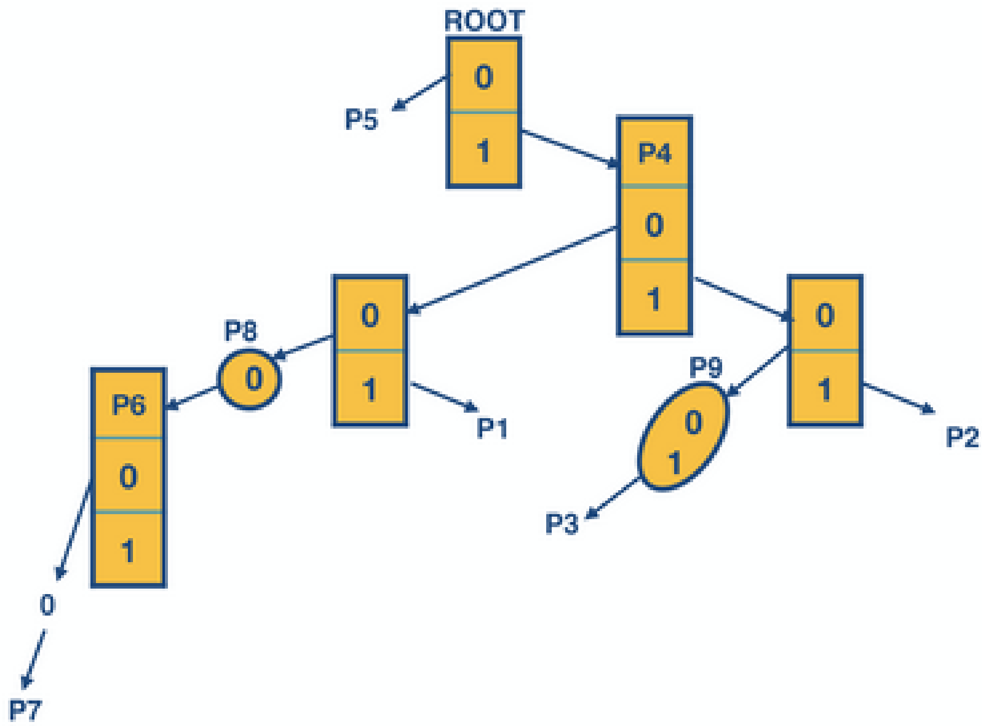
Prefix Match	Output Link
101*	A
111*	B
11001*	C
otherwise	D

```
* Given that the router uses longest prefix matching, determine the output
link for packet with given destination IP address. Type the letter of the
output link.
    - 11100001 10000000 0001 0001 0111 1001 (B)
    - 1111 0001 1111 0000 1010 0001 0111 0111 (B)
    - 1010 1010 1010 1010 1010 1010 1010 1010 (A)
    - 1100 1001 1000 0000 0001 0001 0111 0111 (C)
```

2. Determine the mask for the address 192.168.0.1/24.
    - 255.255.255.0

Unibit Tries

1. One of the simplest techniques for prefix lookup is the unibit trie
    - Every node has a 0 or 1 pointer
        - 0-pointer points to a subtrie for all prefixes that begin with 0, and similarly, 1-pointer points to a subtrie for all prefixes that start with 1
    - How to prefix match
        - Begin the search for a longest prefix match by tracing the trie path
        - Continue the search until we fail (no match or an empty pointer)
        - When our search fails, the last known successful prefix traced in the path is our match and our returned value
    - Notes
        - If a prefix is a substring of another prefix, the smaller string is stored in the path to the longer (more specific) prefix
        - There may be nodes that contain only one pointer; compress these one-way branches to a single text string with 2 bits



Unibit Trie

Quiz 3

1. For each prefix look up, determine the node we return.
   - 0* (A)
   - 1* (B)
   - 01* (C)
   - 00* (A)
   - 0000* (E)
   - 00011* (H)

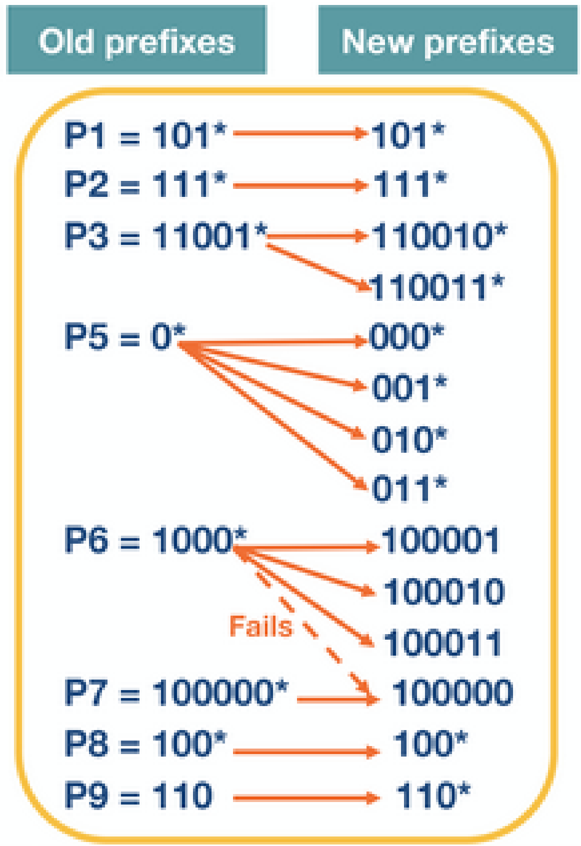
Multibit Tries

1. Why do we need multibit tries?
   - Unibit trie is very efficient and offers advantages such as fast lookup and easier updates, its most significant problem is the number of memory accesses required to perform a lookup
     - For 32-bit addresses, we can see that looking up the address in a unibit trie might require 32 memory accesses in the worst case
     - Assuming 60 nsec latency, worst-case search time is 1.92 microseconds
   - Instead, implement lookups using a stride
     - Stride: number of bits that we check at each step
   - Multibit trie: Trie where each node has 2^k children, where k is the stride
     - Fixed-length and variable-length stride tries exist

Prefix Expansion

1. Controlled prefix expansion
   - Expand a given prefix to more prefixes

- Ensure that the expanded prefix is a multiple of the chosen stride length
- Remove all lengths that are not multiples of the chosen stride length



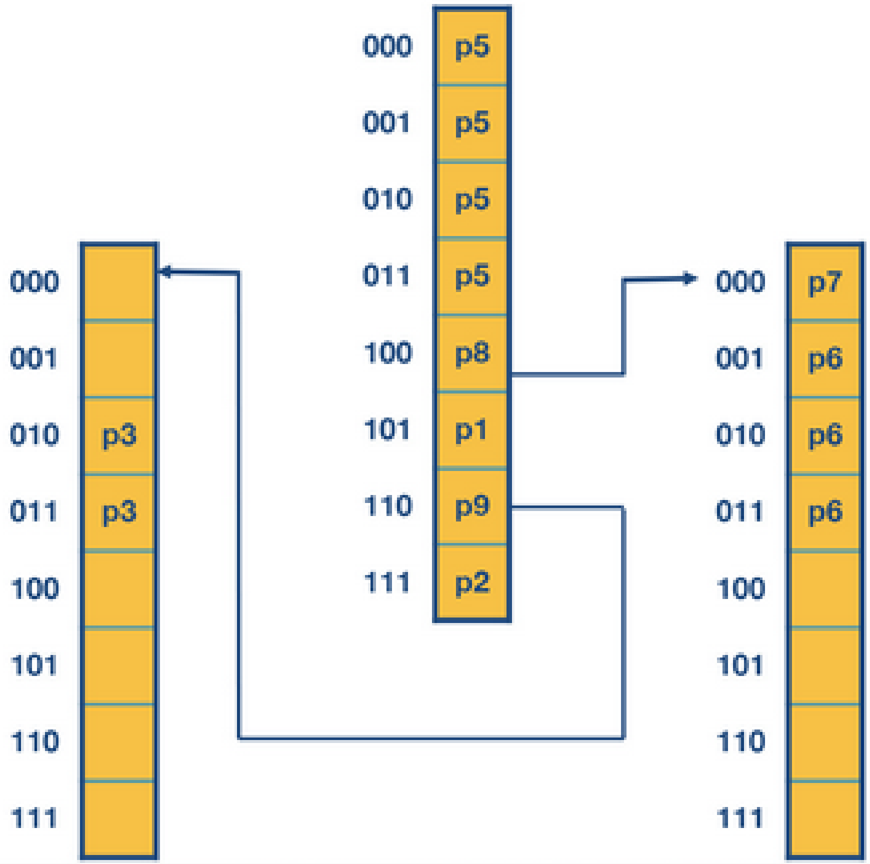
Prefix Expansion

Quiz 4

1. Which of the following prefixes are associated with the prefix 1*?
   - 110* (true)
   - 10*
   - 100* (true)
   - 101*
   - 001*
   - 011*
   - 111* (true)

Multibit Tries: Fixed Stride

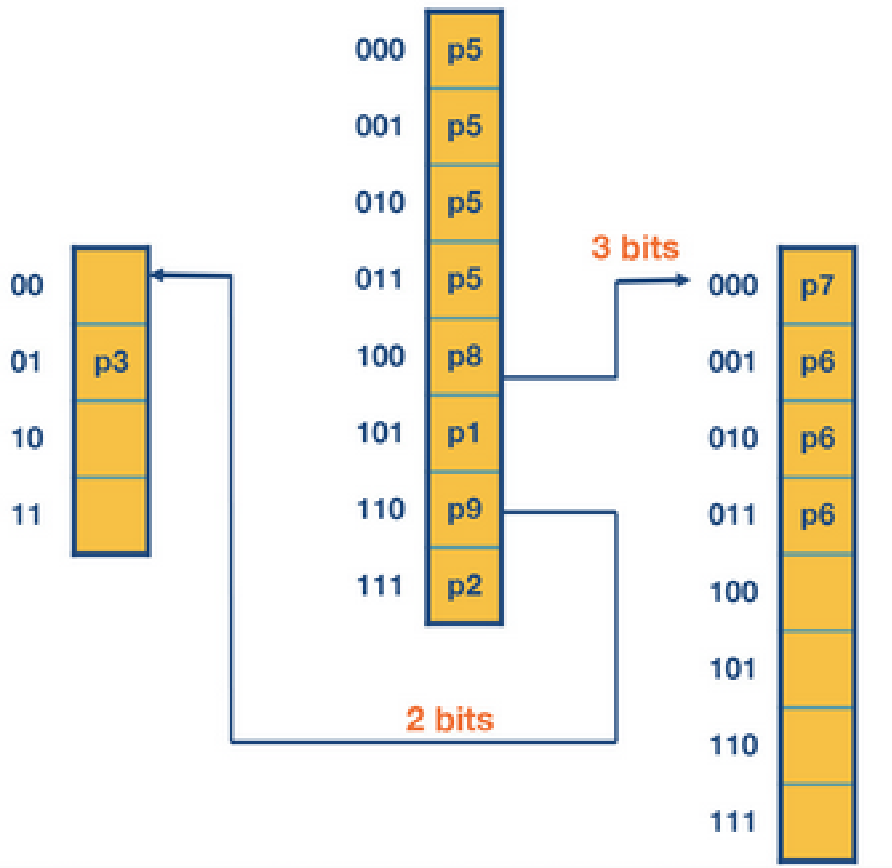
1. Every element in a trie represents two pieces of information: a pointer and a prefix value
2. The prefix search moves ahead with the preset length in n-bits (3 in this case)
3. When the path is traced by a pointer, we remember the last matched prefix (if any)
4. Our search ends when an empty pointer is met. At that time, we return the last matched prefix as our final prefix match.



Fixed Stride

Multibit Tries: Variable Stride

1. Variable stride allows us to examine a different number of bits every time
   - Encode the stride of the trie using a pointer to the node
     - Root node stays as is
   - The rightmost node still needs to examine 3 bits because of P7
   - The leftmost node only needs to examine 2 bits because P3 has 5 bits in total. Can rewrite the leftmost node as in the figure below
     - Have four fewer entries than our fixed stride scheme
     - By varying stride, can make prefix database smaller and optimize for memory
2. Key points
   - Every node can have a different number of btis to be explored
   - The optimizations to the stride length for each node are all done to save trie memory and the least memory accesses
   - An optimum variable stride is selected by using dynamic programming



Variable Stride

Quiz 5

1. A multibit trie is shorter than a unibit trie representing the same prefix database and requires fewer memory accesses to perform a lookup.
2. Fixed-length multibit tries can support an arbitrary number of prefix lengths.
   - False