(*) Packet Flow:

(a) Ingress Packet Processing:

→ When an IP packet arrives at a network, it contains enough information for route lookup after encapsulation.
→ This packet is sent to forwarding engine in the line card.
→ Forwarding engine searches a table (forwarding table) to determine next hop.
→ The next hop info contains egress line card and the outgoing port.
→ On completion of other functions, this packet is sent to the backplane interface.
→ It contains info. that helps the backplane interface to figure out to which interface card it is to be sent.
→ It then schedules the packet for transmission.

(b) Egress Packet Processing:

→ When the packet reaches egress line card, the backplane interface stores it in the memory.
→ Meanwhile the packet is updated with the address of new memory location and sent to the buffer queue.
→ In the buffer queue, the packets are transmitted out of the queue according to their priorities.
→ If the queue is full and there is a problem of congestion, the queue manager drops the packet with low priority.
→ Finally the packet arrives at the n/w interface and then the TTL value and checksum is updated.
→ Packet is transferred to appropriate Port.

(*). Packet Processing:

The tasks performed by the router can be divided into:
(a). Time Critical
(b). Non time Critical

Time critical tasks can be broadly grouped into header processing & forwarding.
Header processing includes validation of packets, TTL, checksum calculation, etc.
Non-time critical: Maintenance, Management, Error Handling.

Time critical operations must have High priority under any circumstances.

Packets → CPU (route lookup) → destination

Fast Path functions include: Header processing
Packet forwarding
Packet Classification
Packet Queuing & Scheduling

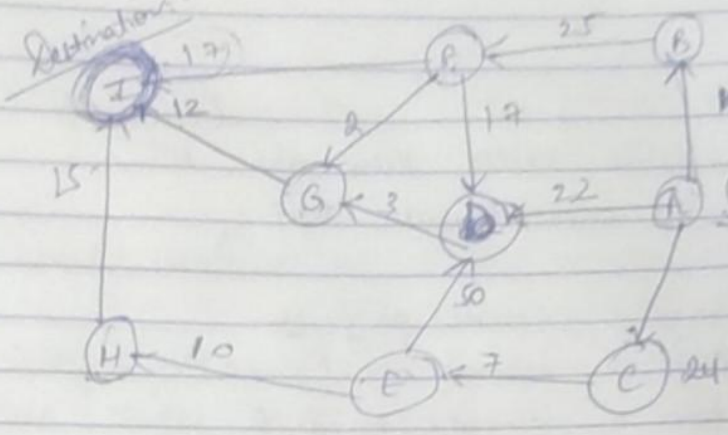
Slow Path Func: ARP processing
Fragmentation
Reassembly
Advanced IP processing.

given more priority.

(*) Widest Path Algo:

2 approaches: ①. Extension of Dijkstra
②. Extension of Bellman ford.

K - Shortest Path Algorithm: 1st shortest Path + 2nd shortest Path




| A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|
| 0 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |
| | 10 | 24 | 22 | ∞ | ∞ | ∞ | ∞ | ∞ |
| | | 24 | 22 | ∞ | 25 | ∞ | ∞ | ∞ |
| | | 24 | | ∞ | 25 | 3 | ∞ | ∞ |
| | | 24 | | ∞ | 25 | | ∞ | 12 |
| | | 24 | | ∞ | 25 | | ∞ | |

| A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|
| 0 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |
| | 10 | 24 | 22 | ∞ | ∞ | ∞ | ∞ | ∞ |
| | | 24 | 22 | ∞ | 35 | ∞ | ∞ | ∞ |
| | | 24 | | ∞ | 35 | 25 | ∞ | ∞ |
| | | | | 31 | 35 | 25 | ∞ | ∞ |
| | | | | 31 | 35 | | ∞ | 37 |
| | | | | | 35 | | 41 | 37 |

35 + 2 + 12 = 49

Nested Loops:

Iterate on I, G, D, A.

{

Iterate on reachable nodes those are not belonging to the shortest path itself.

3

From I, iterate on F, H
From G, iterate on F
From D, iterate on F, E.

In each iteration, we should compute the difference b/w the shortest distance from A to both vertex plus the length of the edge connecting them.

AF − AI + FI
35 − 37 + 15
−2 + 17 = 15

AH − AI + HI
41 − 37 + 15
= 19

AF − AG + FG
35 − 25 + 2 = 12

AF − AD + FD
35 − 22 + 17 = 40

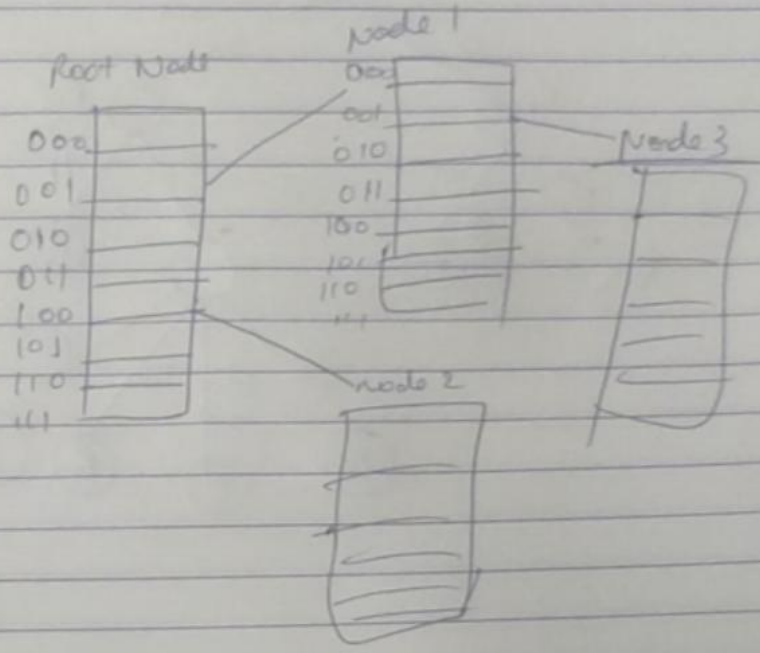
AF − AD + ED
31 − 22 + 50
= 59

A → F → G → I
35 + 12 + 12
47

Eg: Stride length = 3

Prefix Database:

| | |
|---|---|
| P1 | * |
| P2 | 1* |
| P3 | 00* |
| P4 | 101* |
| P5 | 111* |
| P6 | 1000* |



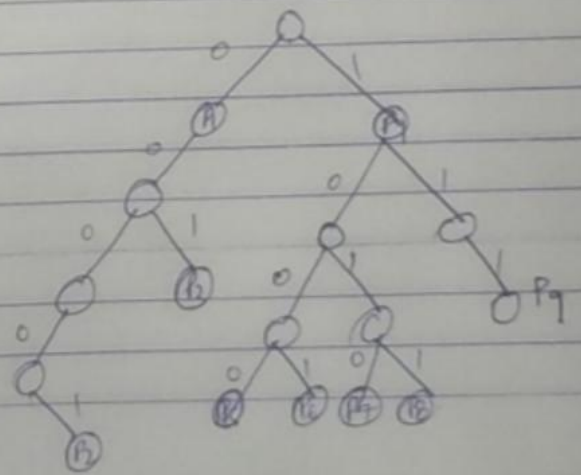

Compression Multibit Strides

→ exists for Multibit tries (compression algo)

→ Multibit tries uses prefix expansion to reduce the no. of levels.

→ ↑ storage space

g:



Compressed