Networks Lecture 8

hlu8jm, jdk2fy

February 6th 2020

1 Review

Recall the router architecture.

- Input ports (Line Cards)
  - Line Termination - Psychical Layer signal reading
  - Link Layer receive protocol - CRC, Drop Ethernet frame, packet parsing
  - Forwarding/Queuing - Buffer values before sending if switch fabric/output is busy.
- Switching Fabric
  - Routes input port to correct output port
- Output ports
  - Similar to output port in opposite order

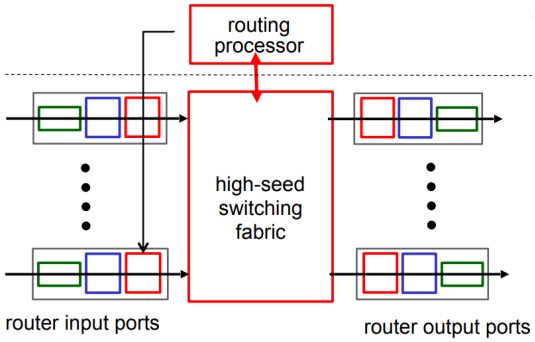

Figure 1: Router Architecture

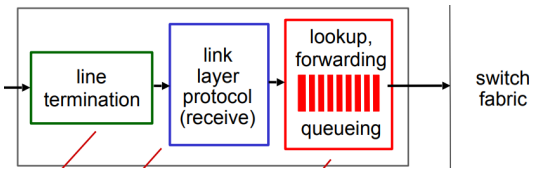

Figure 2:

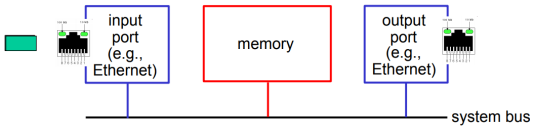

Figure 3: Memory Fabric

2 Switching Fabric

So how do we implement switching fabrics?

- Memory
  - CPU copies packet to memory, routes to output when ready
  - Con: Limited by CPU Bandwidth, lots of extra hardware
- Bus
  - All input/output ports use shared bus. Simplified hardware
  - Con: Only one input and one output access at a time, switching limited by bus bandwidth
- Interconnection network

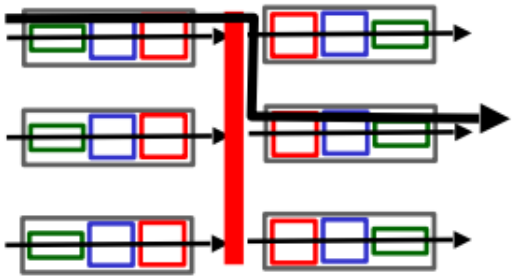

Figure 4: Bus Fabric

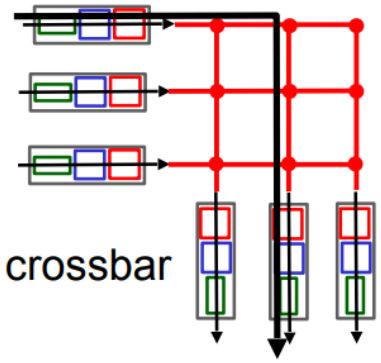

Figure 5: Crossbar Fabric

- Multiple connections can be made in parallel, overcoming bus bandwidth limitation. Typically less complex hardware than memory.
- Crossbar implementation
  - Grid of tri-state buffers, each tri-state buffer can be activated(1) to connect the row and column, or deactivated(0) to disconnect.
  - Each row and column should have up to one active tri-state buffer
- Banyan Tree implementation
  - Each switch acts like a Mux, routing the packet up or down based on the header.
  - Some collision avoidance methods needed

3 Queuing

Input Port:

Head of Line (HOL) blocking - Since each output port can only receive one packet at a time, an input queue may need to wait before sending. This delays all items in that queue.

Output Port

- Packets accumulate in the buffer if they arrive faster than they can be transmitted
- When the buffer is full, incoming data or data in the buffer must be discarded.
- Priority scheduling ensures that important packets are kept. This introduces bias, giving better performance to those with higher priority - net neutrality.

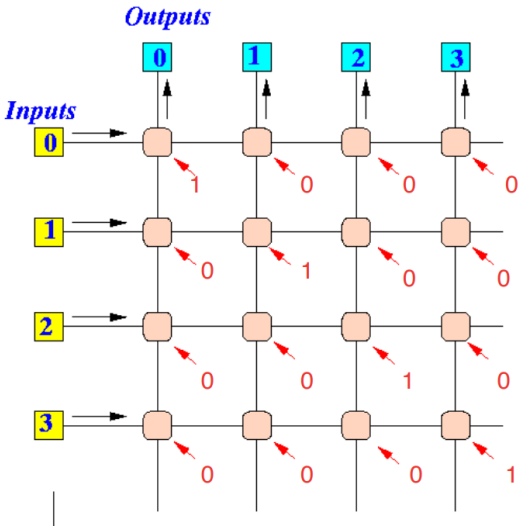

Figure 6: Crossbar Tri-state Buffer Grid

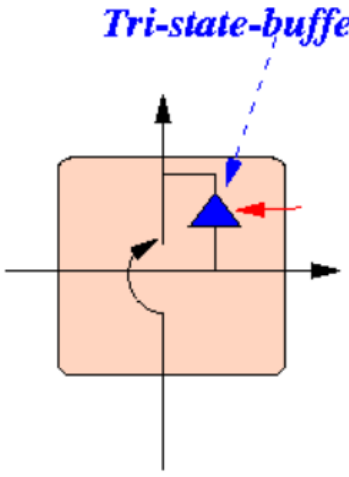

Figure 7: Tri-state Buffer

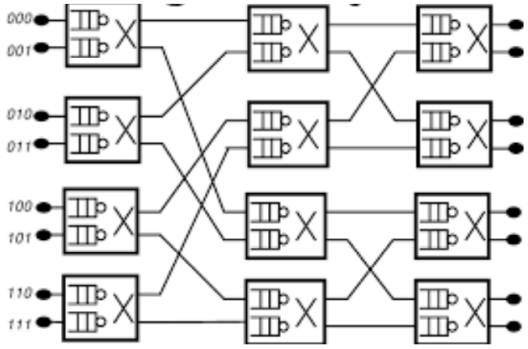

Figure 8: Banyan Tree System

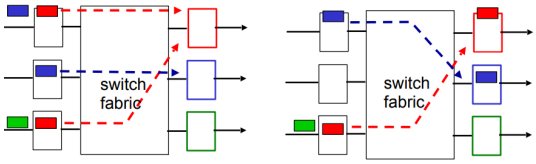

Figure 9: Head of Line Blocking

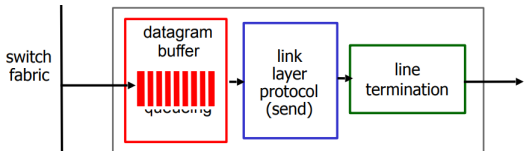

Figure 10: Output Port

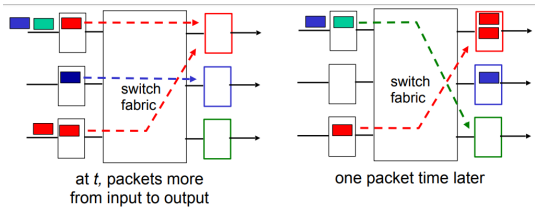

Figure 11: Output Buffer

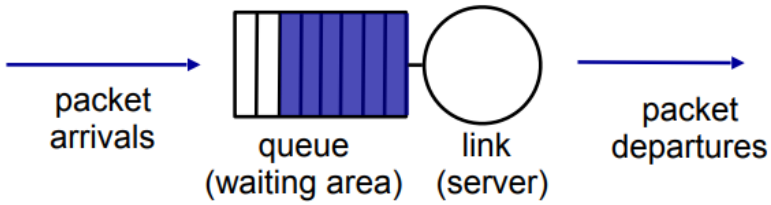

Figure 12: First In First Out Buffer

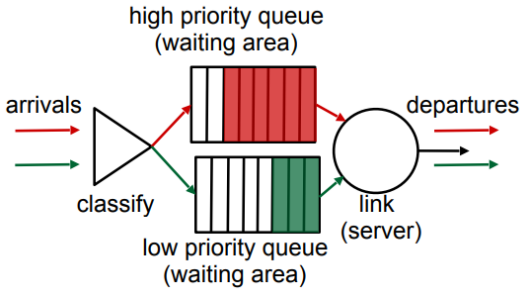

Figure 13: Priority Queue

Scheduling policy

- FIFO - first in first out - needs discard policy for dealing with full buffer
  - tail drop - drop incoming packet
  - priority - drop least priority packet
  - random - drop a random packet
- Priority Queues - Have multiple queues with different priority - send from the highest priority, non-empty queue.
- Round Robin - Have multiple queues by class, take turns sending from each queue evenly
- WFQ - Weighted Fair Queuing - Round Robin, but high priority classes get more packets sent on their turn

4 Per-router control plane

Each router implements a routing algorithm to interact with other routers and determine routing.

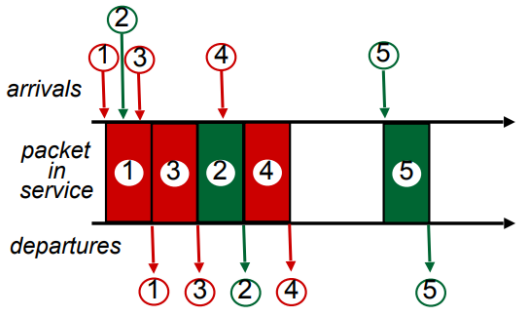

Figure 14: PQ Order

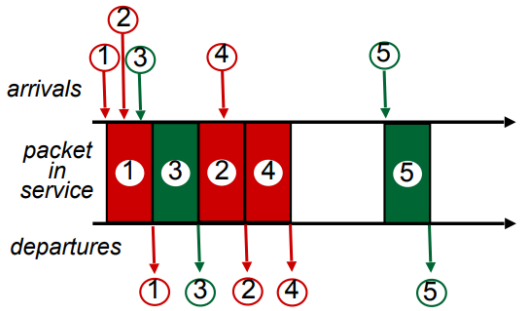

Figure 15: Round Robin

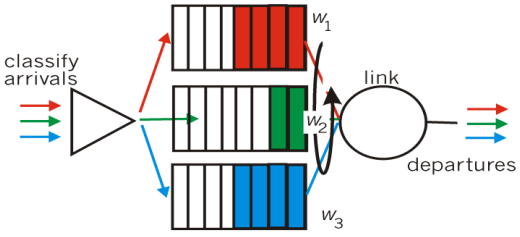

Figure 16: Weighted Fair Queuing

Use Bellman-Ford equation to calculate and maintain distances to other routers.

$$d_u(z) = min(for\ each\ adjacent\ node\ n\ : cost(u,n) + d_n(z)) \tag{1}$$

Steps:

1. Each router initializes known costs to their neighboring routers.
2. Each router shares their distance vector(distances from themselves to other nodes) to their neighboring routers. After receiving distance vectors, this is stored in a distance matrix, maintaining distance vectors of the known routers in the network.
3. Bellman-Ford equation is applied based on new information gained.
4. If distance vector changes from using the Bellman-Ford equation, share distance vector again.
5. repeat until no updates.