Fast packet switching

• Random access: Low throughput
• Packet switching: Increasing the throughput?
• What’s inside a router?

• Suggested reading:
  – Kurose and Ross, Sec.4.6
Packet switches

• A packet switch consists of a routing engine (table look-up), a switch scheduler, and a switch fabric.
• The routing engine looks-up the packet address in a routing table and determines which output port to send the packet.
  – Packet is tagged with port number
  – The switch uses the tag to send the packet to the proper output port
First Generation Switches

- Computer with multiple line cards
  - CPU polls the line cards
  - CPU processes the packets
- Simple, but performance is limited by processor speeds and bus speeds
- Examples: Ethernet bridges and low end routers
Second Generation switches

- Most of the processing is now done in the line cards
  - Route table look-up, etc.
  - Line cards buffer the packets
  - Line card send packets to proper output port

- Advantages: CPU and main Memory are no longer the bottleneck

- Disadvantage: Performance limited by bus speeds
  - Bus BW must be N times LC speed (N ports)

- Example: CISCO 7500 series router
Third generation switches

- Replace shared bus with a switch fabric
- Performance depends on the switch fabric, but potentially can alleviate the bus bottleneck
Switch Architectures

- Distributed buffer
- Output buffer
- Input buffer
Distributed buffer

- Modular Architecture

  Basic module is a 2x2 switch, which can be either in the through or crossed position

  ![2x2 Switch Diagram]

- Switch buffers: None, at input, or at output of each module

  Switch fabric consists of many 2x2 modules

![Switch Fabric Diagram]
Interconnection networks

- N input
- \(\log(N)\) stages with \(N/2\) modules per stage

Example: Omega (shuffle exchange network)

- Notice the order of inputs into a stage is a shuffle of the outputs from the previous stage: \((0,4,1,5,2,6,3,7)\)
- Easily extended to more stages
- Any output can be reached from any input by proper switch settings
  - Not all routes can be done simultaneously
  - Exactly one route between each SD pair
  - Self-routing network
Self Routing

- Use a tag: n bit sequence with one bit per stage of the network
  - E.g., Tag = $b_3b_2b_1$

- Module at stage $i$ looks at bit $i$ of the tag ($b_i$), and sends the packet up if $b_i=0$ and down if $b_i=1$

- In omega network, for destination port with binary address abc the tag is cba
  - Example: output 100 => tag = 001
  - Notice that regardless of input port, tag 001 will get you to output 100
Baseline network

- Another Example of a multi-stage interconnection network
- Built using the basic 2x2 switch module
- Recursive construction
  - Construct an N by N switch using two N/2 by N/2 switches and a new stage of N/2 basic (2x2) modules
  - N by N switch has $\log_2(N)$ stages each with N/2 basic (2x2) modules
Contention

- Two packets may want to use the same link at the same time (same output port of a module)
- Hot spot effect
- Solution: Buffering
Throughput analysis of interconnection networks

- Assume no buffering at the switches
- If two packets want to use the same port one of them is dropped
- Suppose switch has \( m \) stages
- Packet transmit time = 1 slot (between stages)
- New packet arrival at the inputs, every slot
  - Saturation analysis (for maximum throughput)
  - Uniform destination distribution independent from packet to packet
Let $P(m)$ be the probability that a packet is transmitted on a stage $m$ link

$P(0) = 1$

$P(m+1) = 1 - P\text{(no packet on stage } m+1 \text{ link (link c) )}$

$= 1 - P(\text{neither inputs to stage } m+1 \text{ chooses this output})$

Each input has a packet with probability $P(m)$ and that packet will choose the link with probability $1/2$. Hence,

$$P(m \mid 1) = 1 \cdot \left(1 - \frac{1}{2}P(m)\right)^2$$

We can now solve for $P(m)$ recursively

For an $m$ stage network, throughput (per output link) is $P(m)$, which is the probability that there is a packet at the output
Throughput can be significantly improved by adding buffers at the stages.
- Buffers increase delay
- Tradeoff between delay and throughput
Advantages/Disadvantages of multi-stage architecture

• Advantages
  – Modular
  – Scalable
  – Bus (links) only needs to be as fast as the line cards

• Disadvantages
  – Delays for going through the stages
    Cut-through possible when buffers empty
  – Decreased throughput due to internal blocking

• Alternatives: Buffers that are external to the switch fabric
  – Output buffers
  – Input buffers
Output buffer architecture

- As soon as a packet arrives, it is transferred to the appropriate output buffer
- Assume slotted system (cell switch)
- During each slot the switch fabric transfers one packet from each input (if available) to the appropriate output
  - Must be able to transfer $N$ packets per slot
  - Bus speed must be $N$ times the line rate
  - No queueing at the inputs
    Buffer at most one packet at the input for one slot
Advantages/Disadvantages of Output buffer architecture

- **Advantages:** No delay or blocking inside switch
- **Disadvantages:**
  - Bus speed must be $N$ times line speed
    Imposes practical limit on size and capacity of switch

- **Shared output buffers:** output buffers are implemented in shared memory using a linked list
  - Requires less memory (due to statistical multiplexing)
  - Memory must be fast
Input buffer architecture

- Packets buffered at input rather than output
  - Switch fabric does not need to be as fast

- During each slot, the scheduler established the crossbar connections to transfer packets from the input to the outputs
  - Maximum of one packet from each input
  - Maximum of one packet to each output

- Head of line (HOL) blocking – when the packet at the head of two or more input queues is destined to the same output, only one can be transferred and the other is blocked
Throughput analysis of input queued switches

- HOL blocking limits throughput because some inputs (consequently outputs) are kept idle during a slot even when they have other packet to send in their queue.

- Consider an NxN switch and again assume that inputs are saturated (always have a packet to send).

- Uniform traffic $\Rightarrow$ each packet is destined to each output with equal probability ($1/N$).

- Now, consider only those packets at the head of their queues (there are $N$ of them!).
Summary of input queued switches

• The maximum throughput of an input queued switch, is limited by HOL blocking to 58% (for large N)
  – Assuming uniform traffic and FCFS service

• Advantages of input queues:
  – Simple
  – Bus rate = line rate

• Disadvantages: Throughput limitation
Overcoming HOL blocking

- If inputs are allowed to transfer packets that are not at the head of their queues, throughput can be substantially improved (not FCFS)

Example:

```
input 1  1  2
input 2  3  2
input 3  4  3
input 4  4  2
```

- How does the scheduler decide which input to transfer to which output?
Each entry in the backlog matrix represent the number of packets in input i’s queue that are destined to output j.

During each slot the scheduler can transfer at most one packet from each input to each output.
- The scheduler must choose one packet (at most) from each row, and column of the backlog matrix.
- This can be done by solving a bi-partite graph matching algorithm.
- The bi-partite graph consists of N nodes representing the inputs and N nodes representing the outputs.
Bi-partite graph representation

- An edge in the graph from an input to the output if there is a packet in the backlog matrix to be transferred from that input to the output
- The bi-partite graph for the previous example

- A matching is a set of edges such that no two nodes share an edge
  - Finding a matching in a bi-partite graph is equivalent to finding a set of packets such that no two packets share a row or column in the backlog matrix
- A maximum matching is a matching with the maximum possible number of edges
  - Finding a maximum matching is equivalent to finding a maximum number of packets which can be transmitted simultaneously
Maximum Matchings

• Algorithms for finding maximum matching exist
  • The best known algorithms takes $O(N^{2.5})$ operations
    – Too long for large N

• Alternatives
  – Sub-optimal solutions
  – Maximal matching: A matching that cannot be made any larger for a given backlog matrix

  – For previous example:

    (1-1,3-3) is maximal
    
    (2-1,1-2,3-3) is maximum

• Fact: The number of edges in a maximal matching • 1/2 the number of edges in a maximum matching
Achieving 100% throughput in an input queued switch

• Finding a maximum matching during each time slot does not eliminate the effects of HOL blocking
  – Must look beyond one slot at a time in making scheduling decisions

• Definition: A weighted bi-partite graph is a bi-partite graph with costs associated with the edges

• Definition: A maximum weighted matching is a matching with the maximum edge weights

• Theorem: A scheduler that chooses during each time slot the maximum weighted matching where the weight of link \((i,j)\) is equal to the length of queue \((i,j)\) achieves full utilization (100% throughput)
Summary

- Functionality
- Architecture
- Performance
- Analysis