

# Switching Hardware

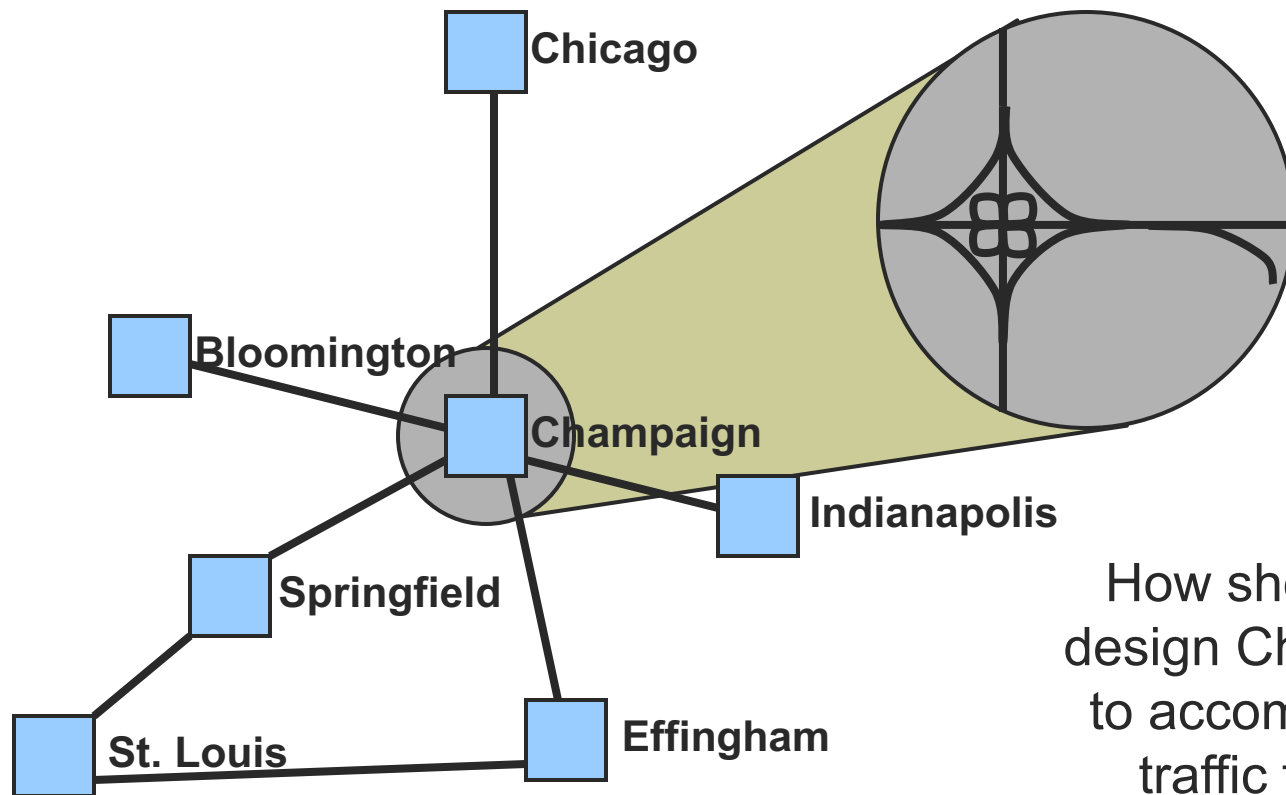
# [Where are we?

- Understand
  - Different ways to move through a network (forwarding)
    - Read signs at each switch (datagram)
    - Follow a known path (virtual circuit)
    - Carry instructions (source routing)
  - Bridge approach to extending LAN concept
- Next: how switches are built and contention within switches





# [ Switch Design



How should we design Champaign to accommodate traffic flows?



# [ Switch architecture ]



Juniper EX2200



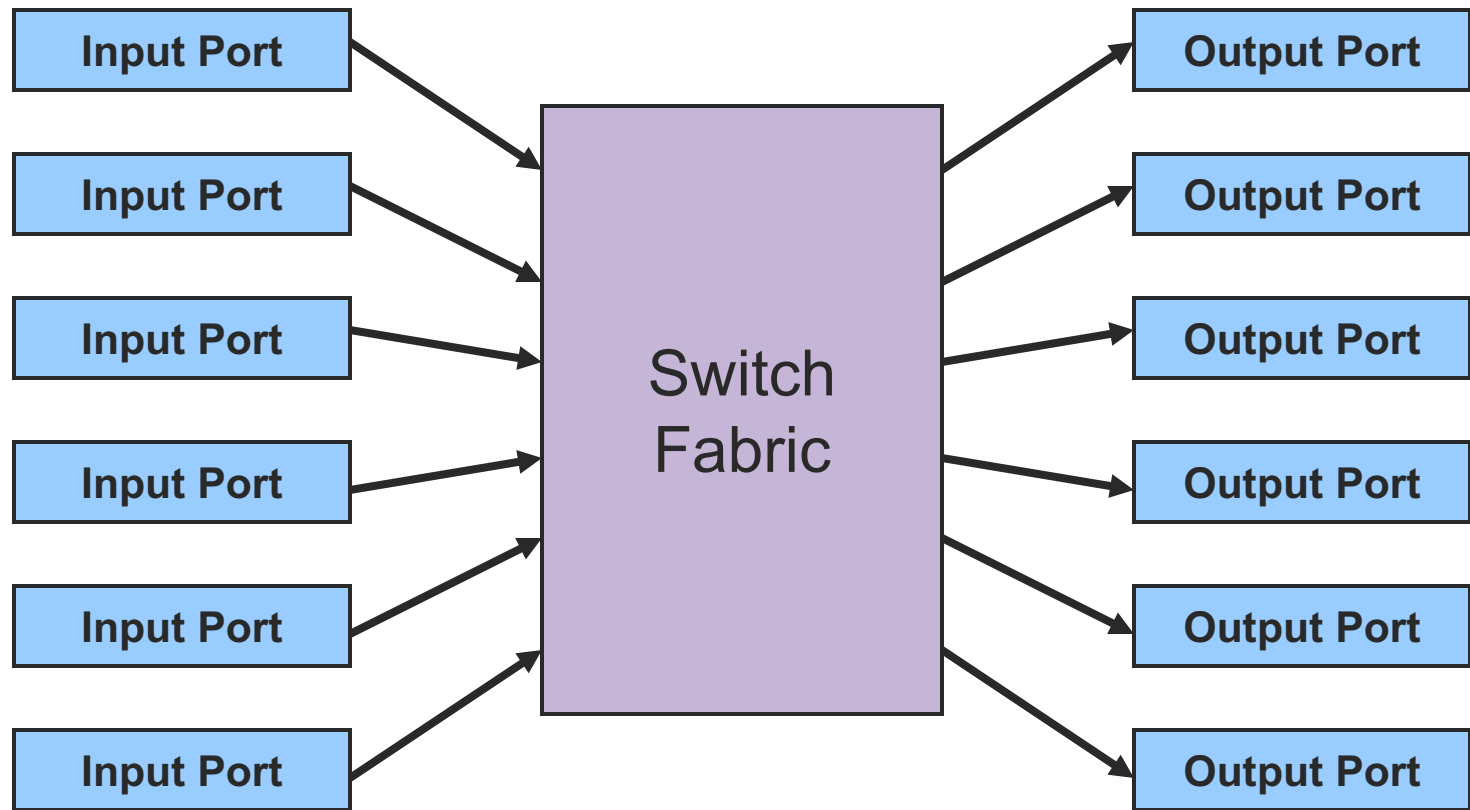
Juniper EX8200



Cisco Catalyst 6500



# [ Switch Design ]



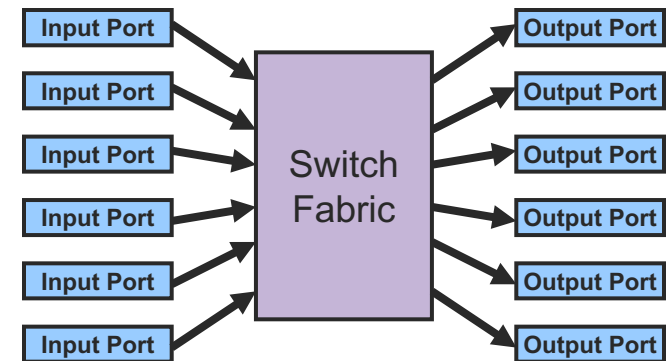
# Switch Architecture

## ■ Problem

- Connect  $N$  inputs to  $M$  outputs
  - $N \times M$  (“ $N$  by  $M$ ”) switch
  - Common case:  $N = M$

## ■ Goals

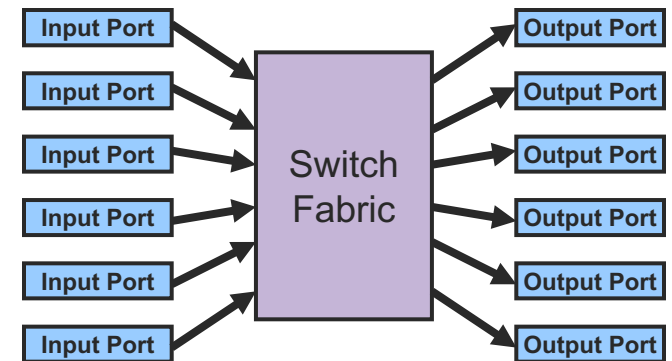
- Avoid contention
- High throughput
- Good scalability
  - Near-linear size/cost growth



# Switch high level architecture

- Ports handle complexity

- Forwarding decisions at input ports
- Buffering at output and possibly input ports



- Simple fabric (it seems...)

- Move packets from inputs to outputs
- May have a small amount of internal buffering



# [ Switch Design Goals ]

## ■ Minimize Contention

- Avoid contention through intelligent buffering
- Use output buffering when possible
- Apply back pressure through switch fabric
- Improve input buffering through non-FIFO buffers
  - Reduces head-of-line blocking
- Drop packets if input buffers overflow





# [ Switch Design Goals ]

- Maximize Throughput
  - Main problem is contention
  - Need a good traffic model
    - Arrival time
    - Destination port
    - Packet length
  - Telephony modeling is well understood
    - Until faxes and modems
  - Data traffic has different properties
    - E.g., phone call arrivals are “Poisson”, but packet arrivals are “heavy-tailed”

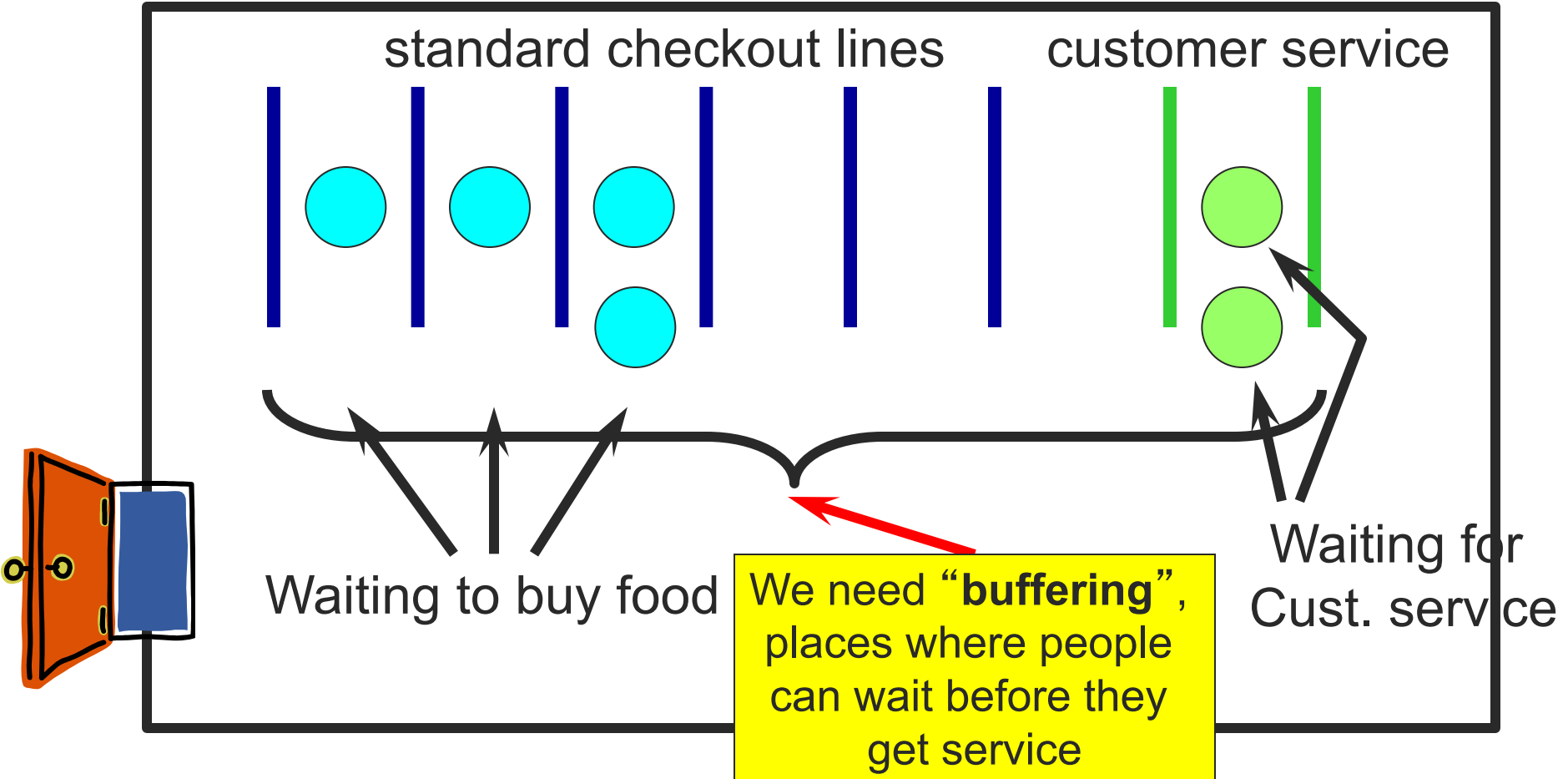


# [Contention]

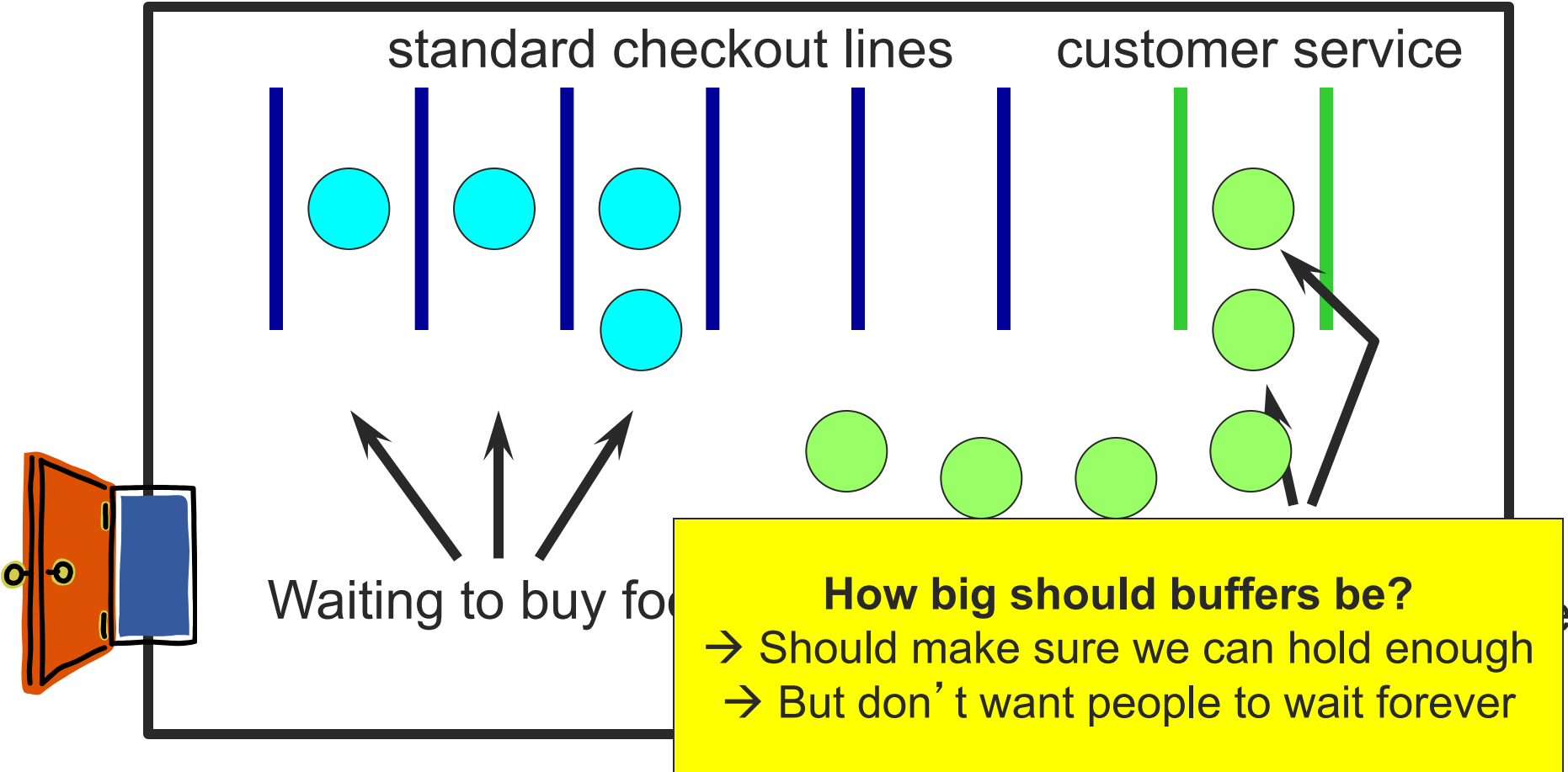
- Problem
  - Some packets may be destined for the same output port
- Solutions
  - One packet gets sent first
  - Other packets get delayed or dropped
- Delaying packets requires buffering
  - Buffers are finite, so we may still have to drop
  - Buffering at input ports
    - Increases, adds false contention
    - Sometimes necessary
  - Buffering at output ports
  - Buffering inside switch



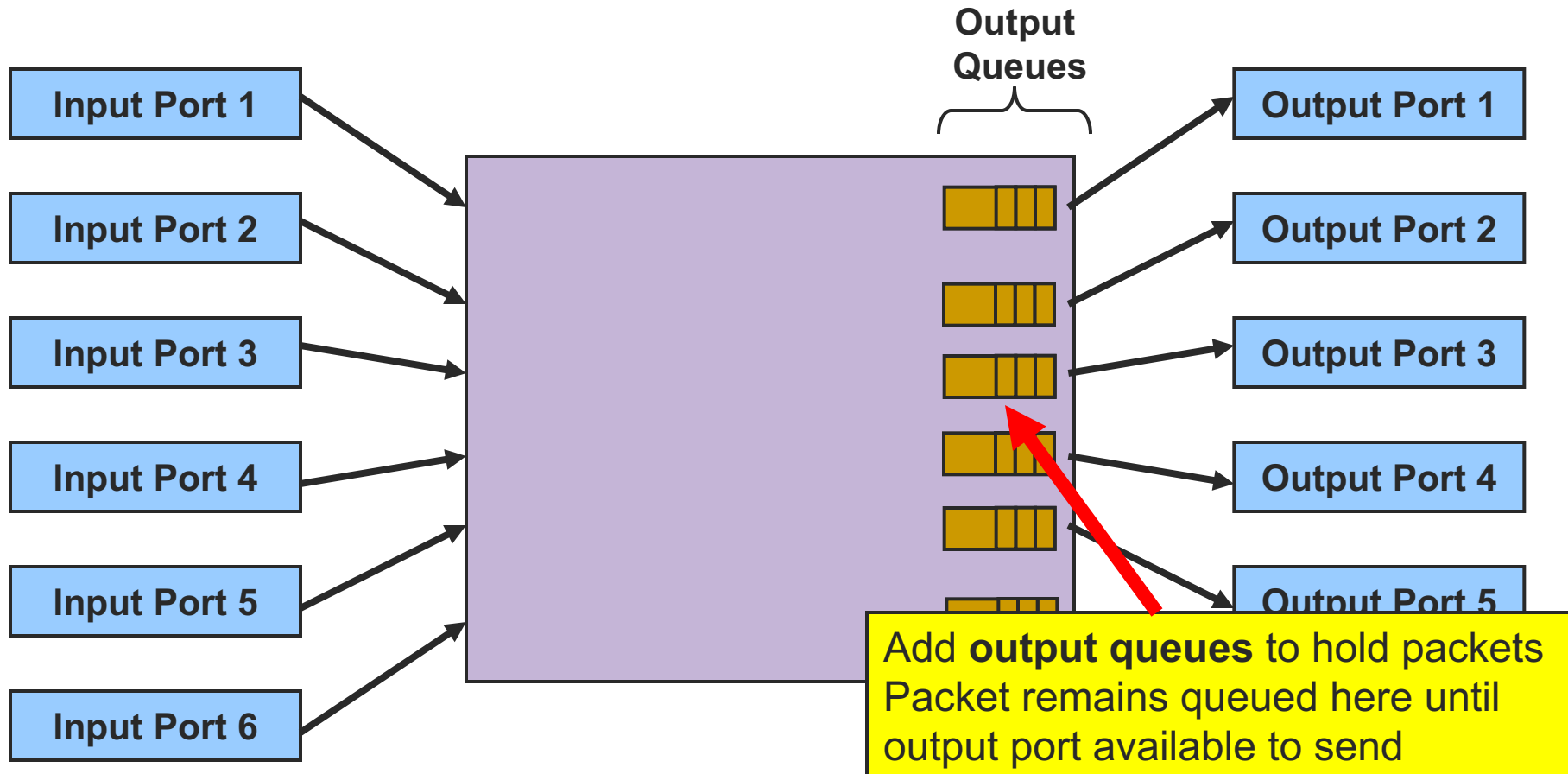
# Buffering



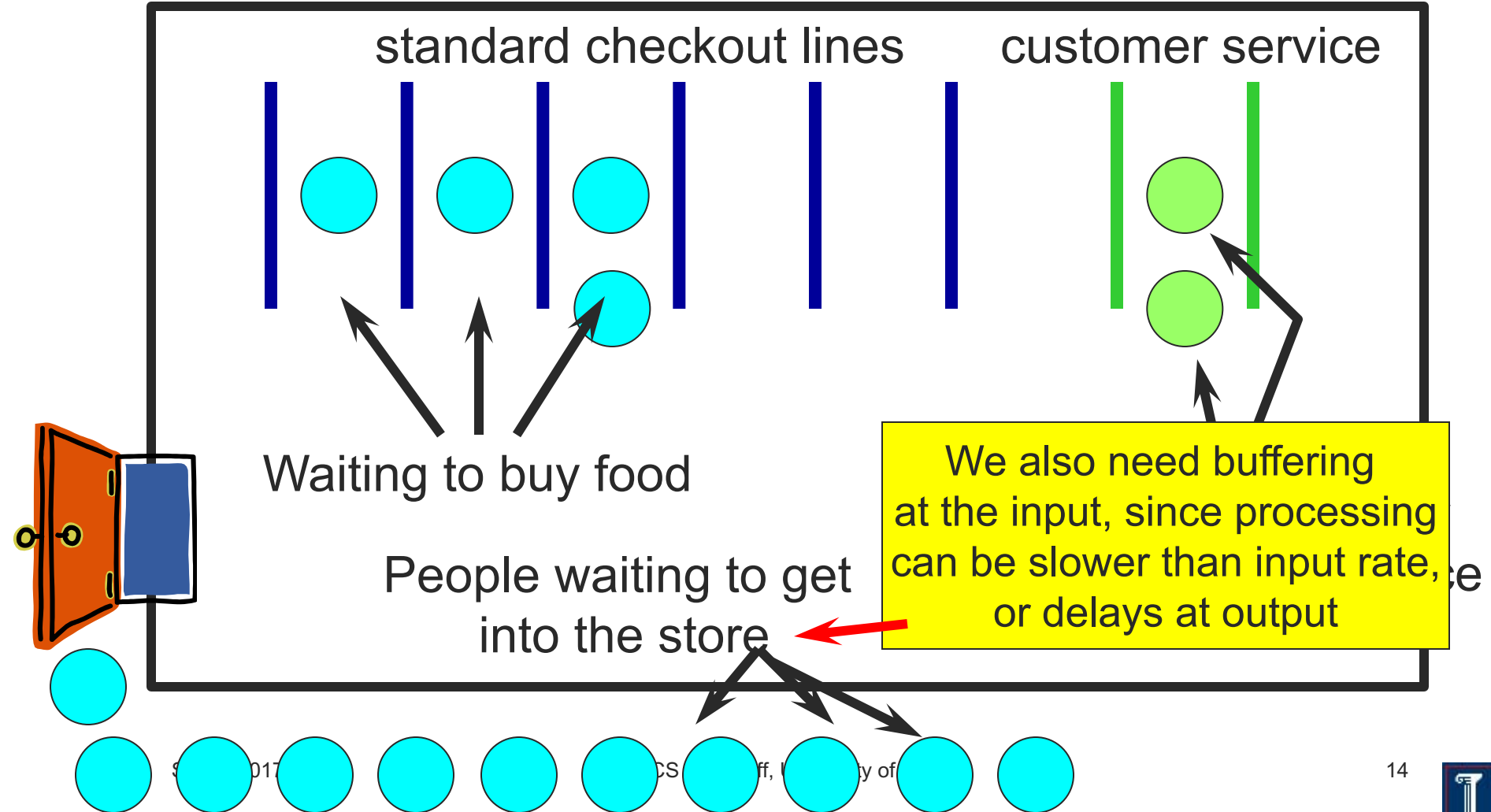
# Output Port Buffering



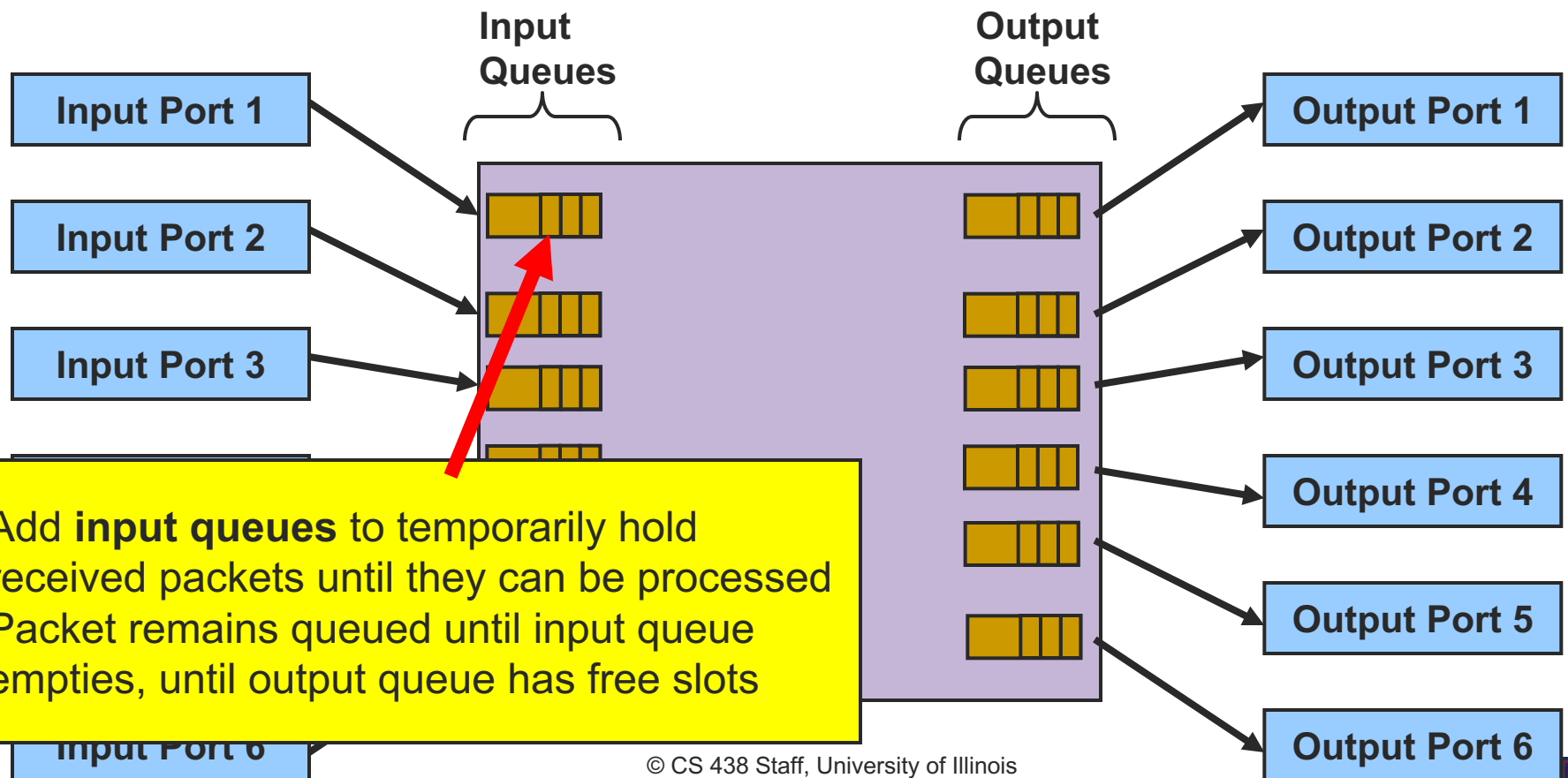
# [ Switch Design



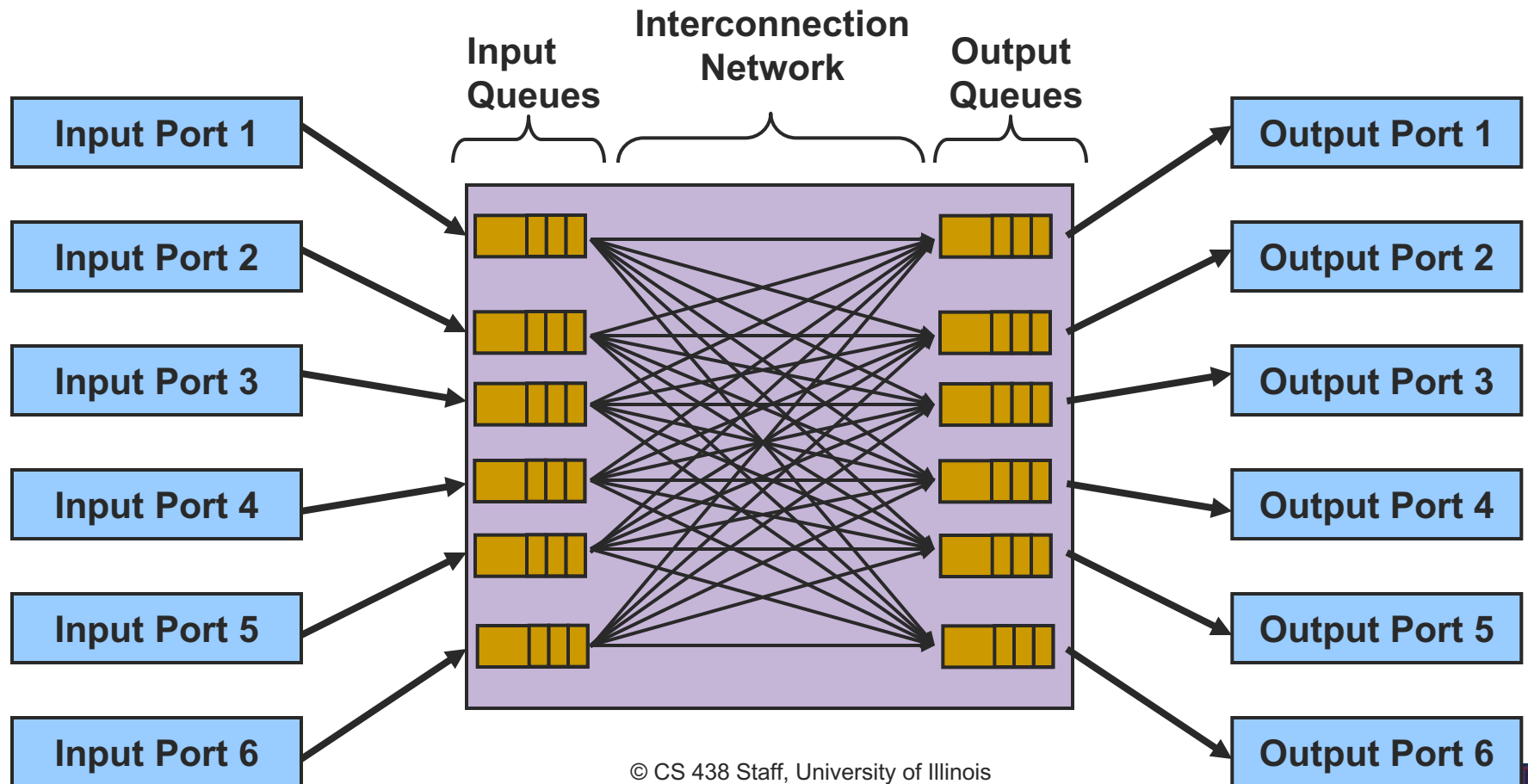
# Input Port Buffering



# Switch Design

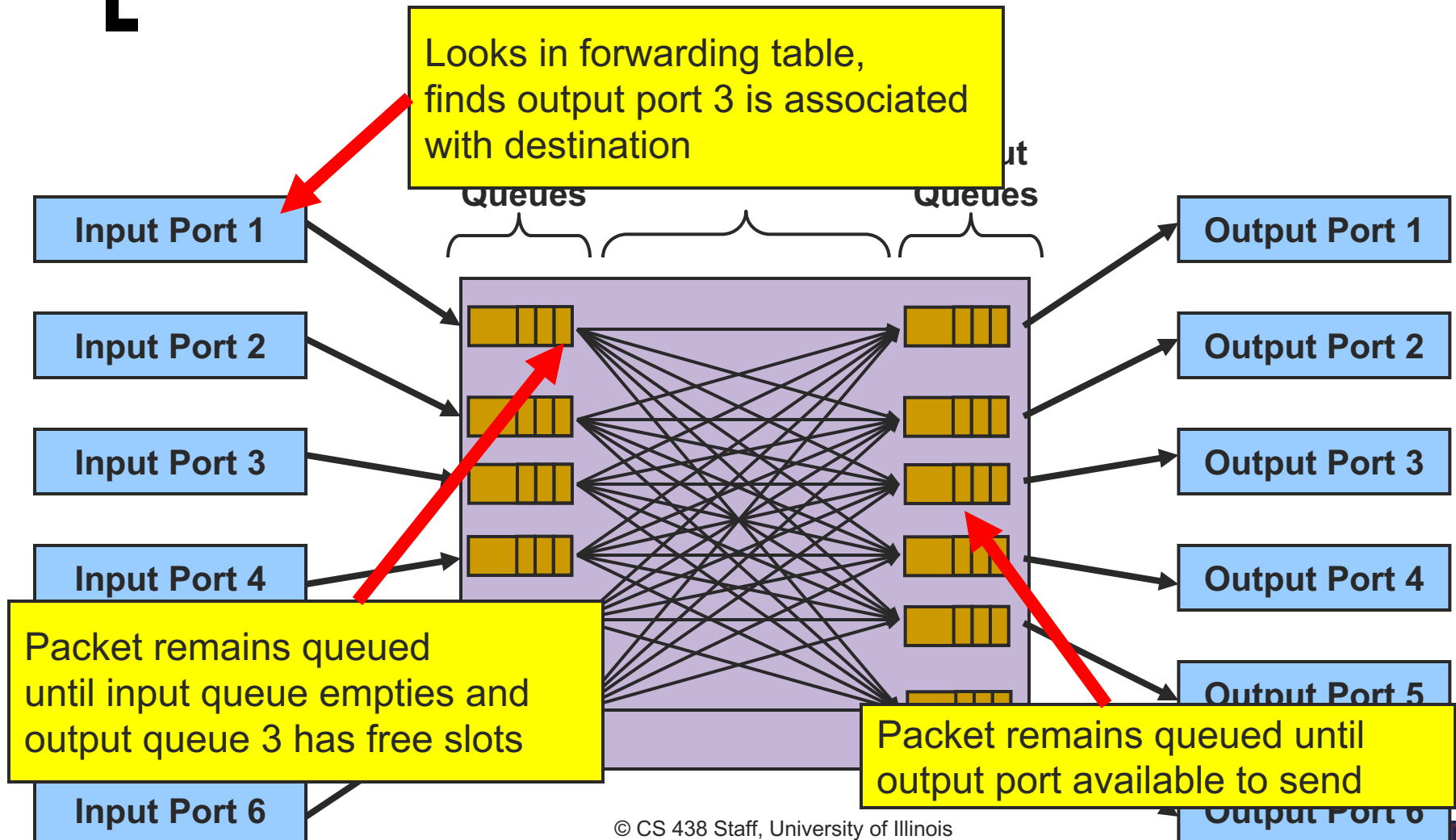


# Switch design: putting the pieces together

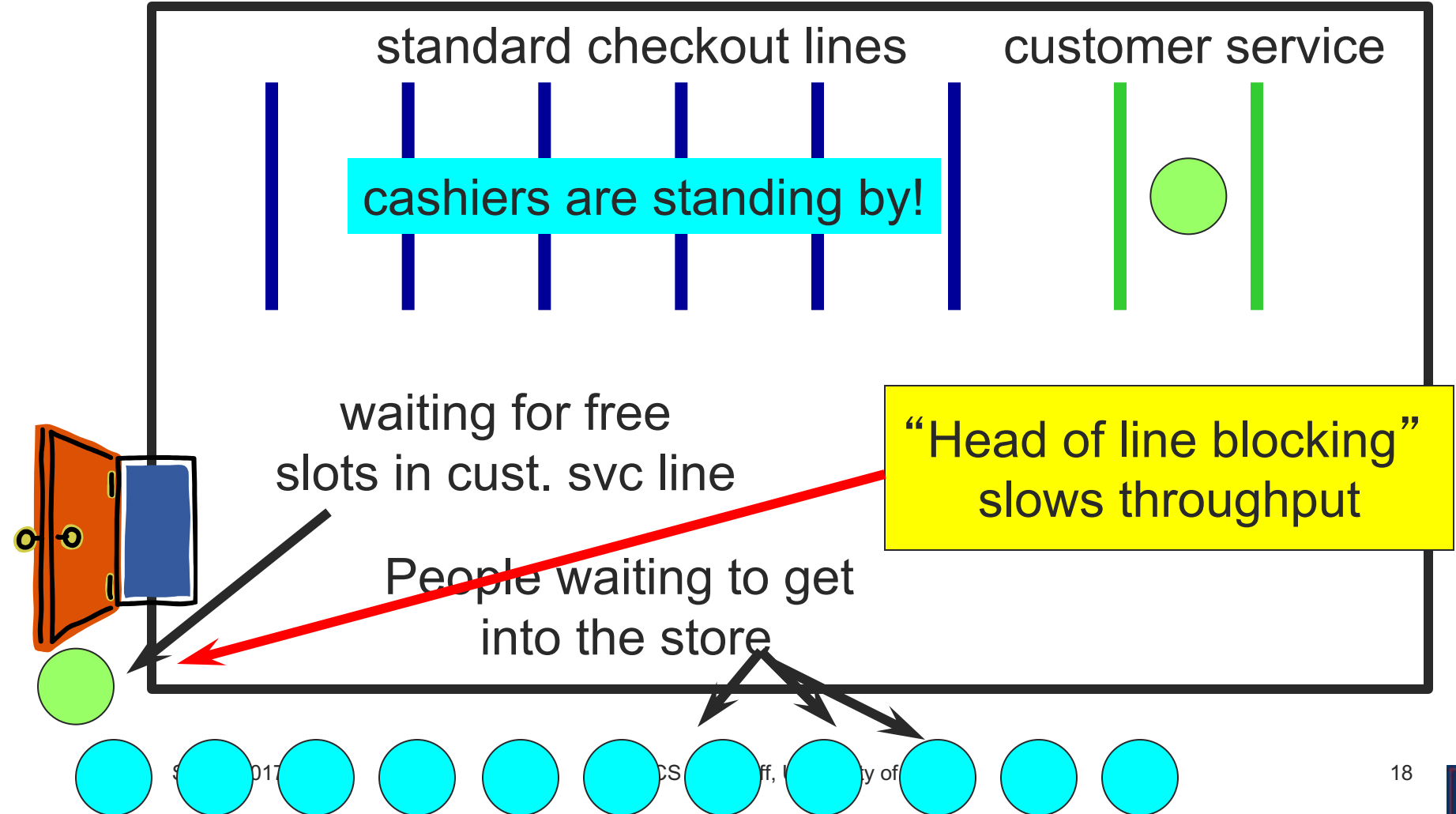




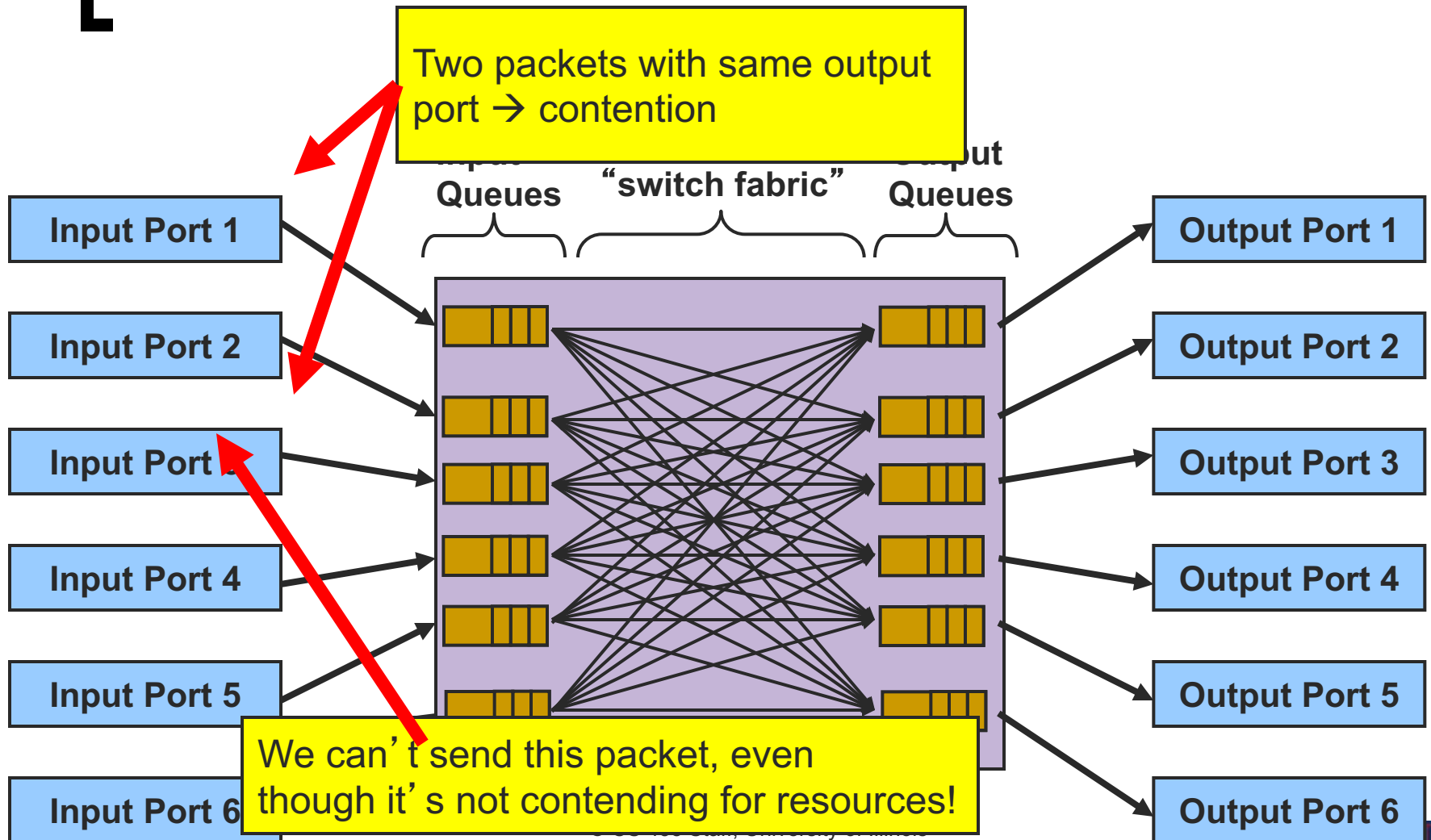
# Switch design: putting the pieces together



# Contention – Head of Line Blocking



# Head of Line Blocking

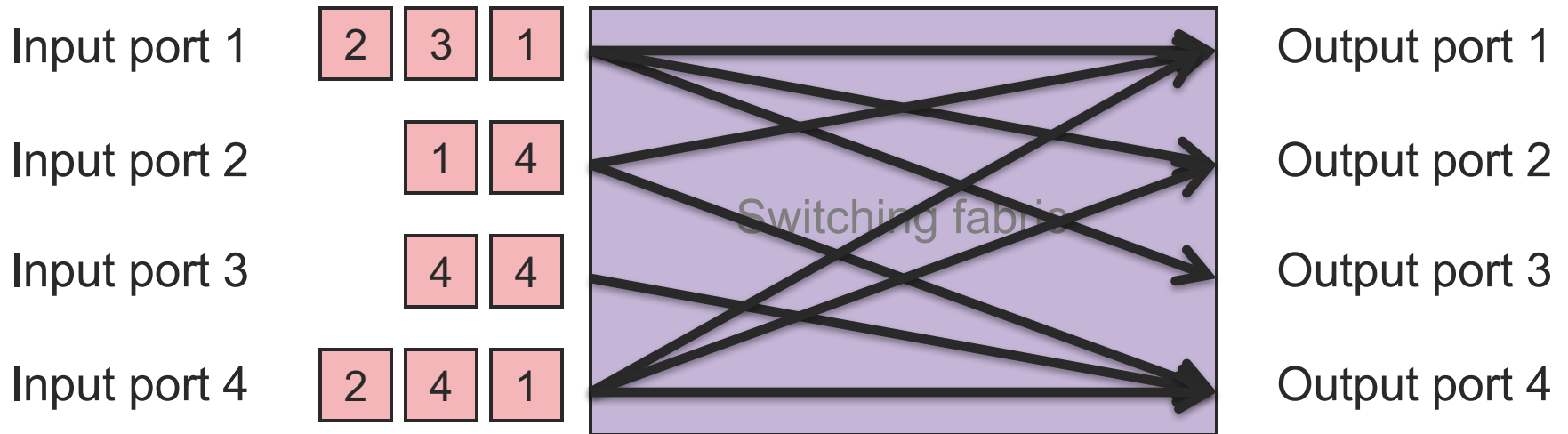


# Unblocking head of line blocking

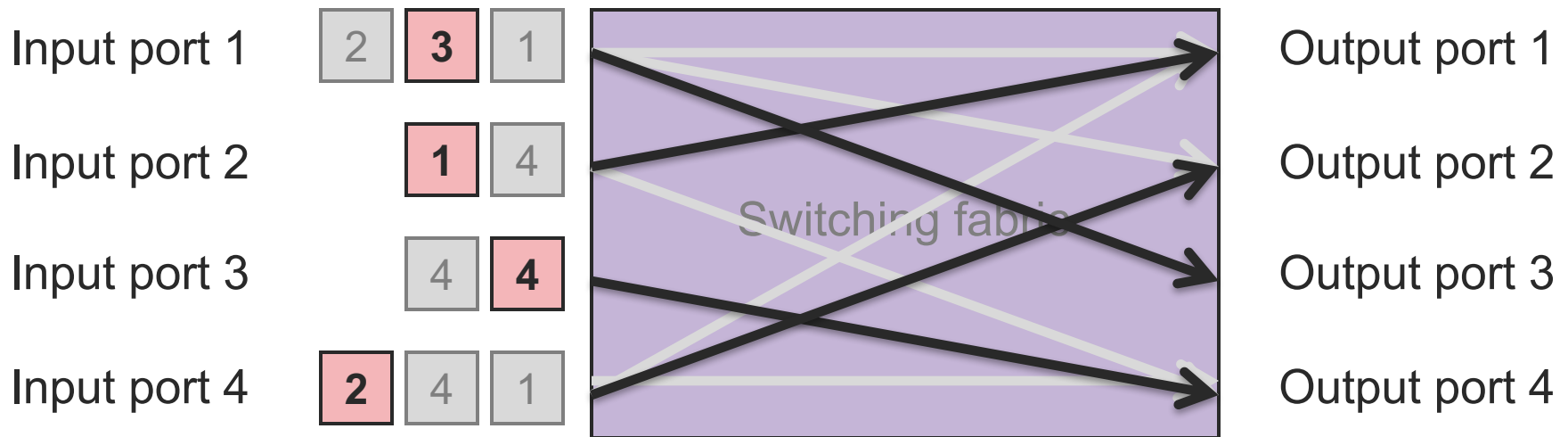
- Solution 1: No input queue
  - Switching fabric (hopefully) keeps up with input rate
- Solution 2: No need to always serve packet at head of queue. Could pick any!
  - Each input port has separate queue for each output port
- Next question: which packet do we pick?



# [ Picking packets' ports

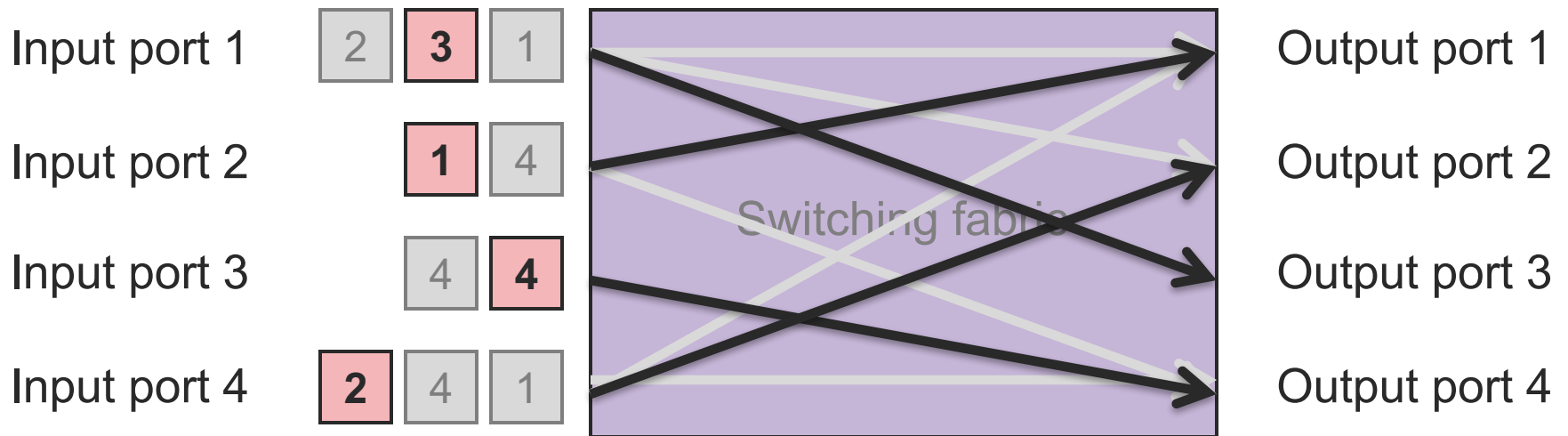


# [ Picking packets' ports



- Underlying problem for max throughput in single timestep: bipartite matching
  - Pick max subset of edges using 1 edge per node

# [ Picking packets' ports



- Switches may not find optimal solution: we also want
  - Fairness
  - Simplicity of implementation

# [What we know so far]

- Buffering masks temporary contention
- Need to carefully manage queues
  - Head-of-line blocking problem
  - Fairness
  - Throughput







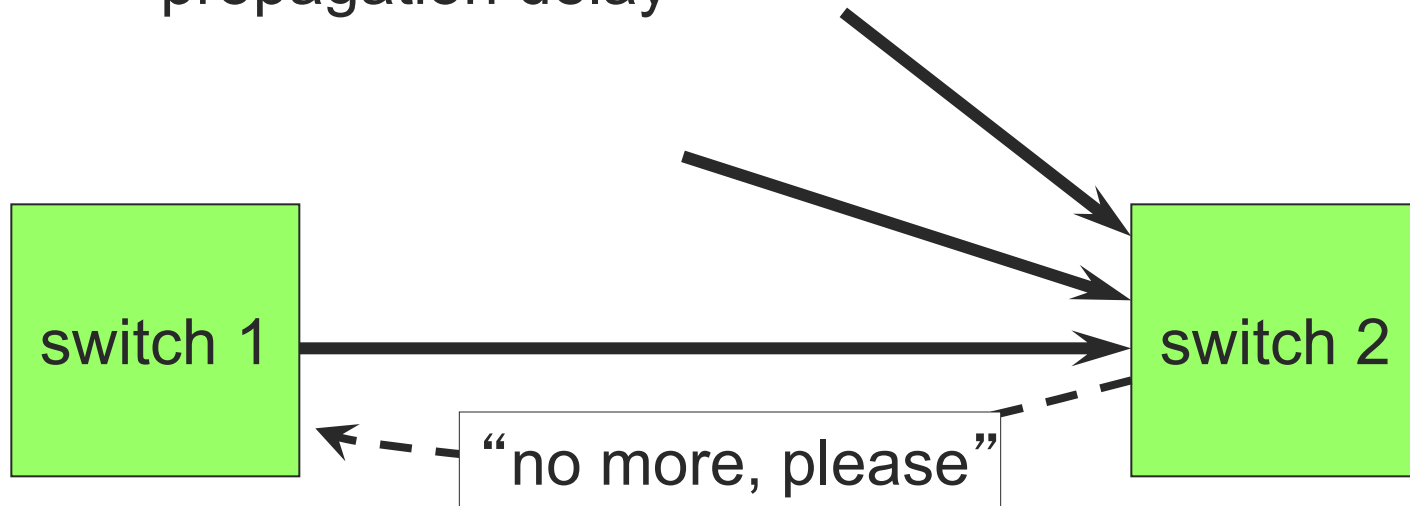
# [ What we know so far

- Did we completely solve contention problem? Could a packet ever be dropped?
  - Yes: queues can still overflow
  - Solution 1: plan allowed packet rates in advance – virtual circuit switching
  - Solution 2: dynamically request rate reduction – backpressure



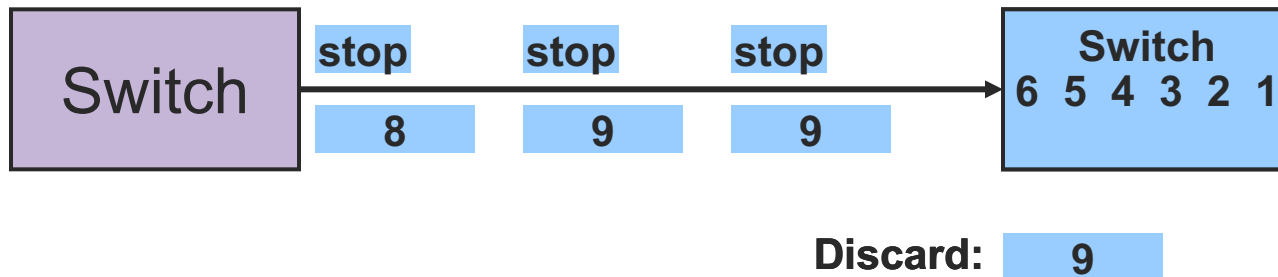
# Contention – Back Pressure

- Let the receiver tell the sender to slow down
  - Propagation delay requires that the receiver react before the buffer is full
  - Typically used in networks with small propagation delay



# Contention – Back Pressure

- Need to send backpressure *before* queue fills
- So, better when propagation delay small
  - e.g., switch fabrics
  - e.g., Ethernet pause-based flow control (IEEE 802.3x) used to run FibreChannel over Ethernet

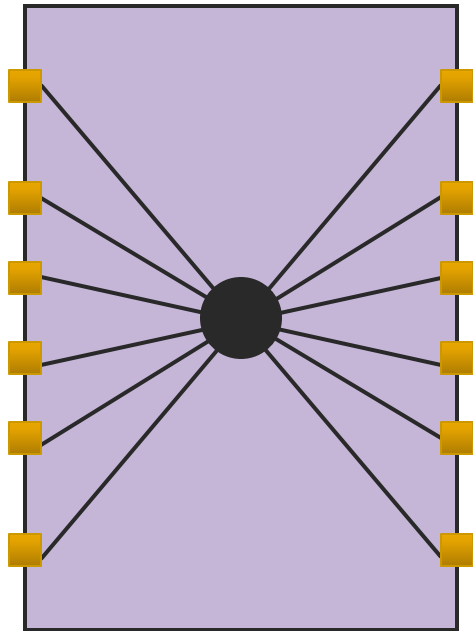


# [ Switch Design Goals ]

- High Throughput
  - Number of packets a switch can forward per second
- High Scalability
  - How many input/output ports can it connect
- Low Cost
  - Per port monetary costs

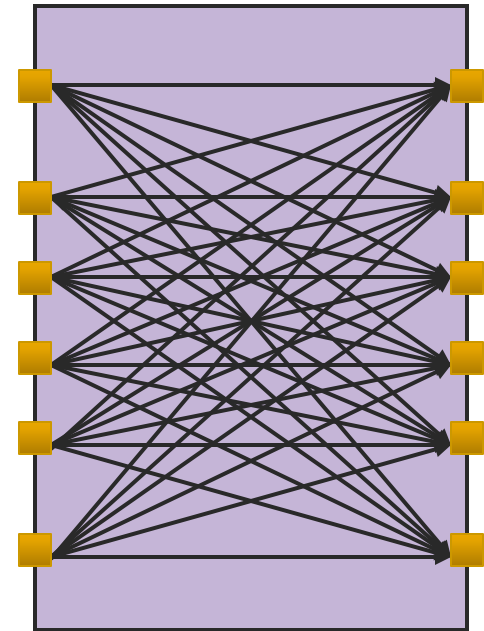


# [Two simple fabrics]



Shared bus or memory:  
Low \$, low throughput

Two simple  
fabrics for very  
large high-  
performance  
switches!



Full mesh:  
High \$, high throughput

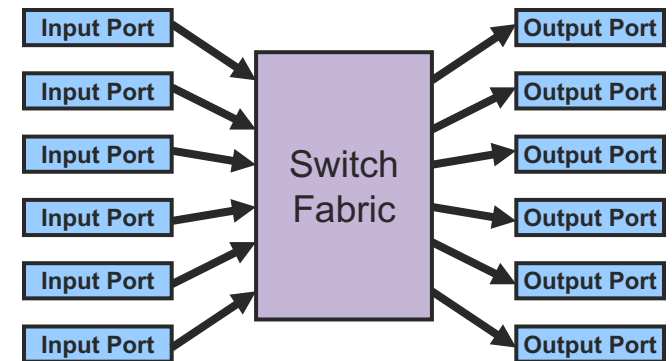
# Special Purpose Switches

## ■ Problem

- Connect N inputs to M outputs
  - NxM (“N by M”) switch
  - Often  $N = M$

## ■ Goals

- High throughput
  - Best is  $\text{MIN}(\text{sum of inputs}, \text{sum of outputs})$
- Avoid contention
- Good scalability
  - Linear size/cost growth



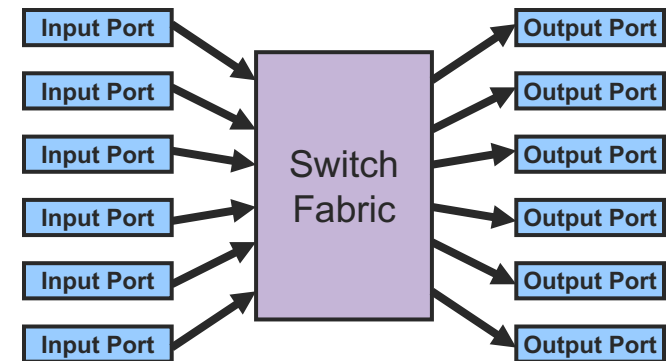
# [ Switch Design ]

- Ports handle complexity

- Forwarding decisions
- Buffering

- Simple fabric

- Move packets from inputs to outputs
- May have a small amount of internal buffering



# [ Switch Design Goals ]

- Throughput
  - Main problem is contention
  - Need a good traffic model
    - Arrival time
    - Destination port
    - Packet length
  - Telephony modeling is well understood
    - Until faxes and modems
  - Modeling of data traffic is new
    - Not well understood
    - Will good models help?





# [ Switch Design Goals ]

## ■ Contention

- Avoid contention through intelligent buffering
- Use output buffering when possible
- Apply back pressure through switch fabric
- Improve input buffering through non-FIFO buffers
  - Reduces head-of-line blocking
- Drop packets if input buffers overflow



# [ Switch Design Goals ]

- Scalability

- $O(N)$  ports
- Port design complexity  $O(N)$  gives  $O(N^2)$  for entire switch
- Port design complexity of  $O(1)$  gives  $O(N)$  for entire switch



# [ Switch Design ]

- Crossbar Switches
- Banyan Networks
- Batchner Networks
- Sunshine Switch

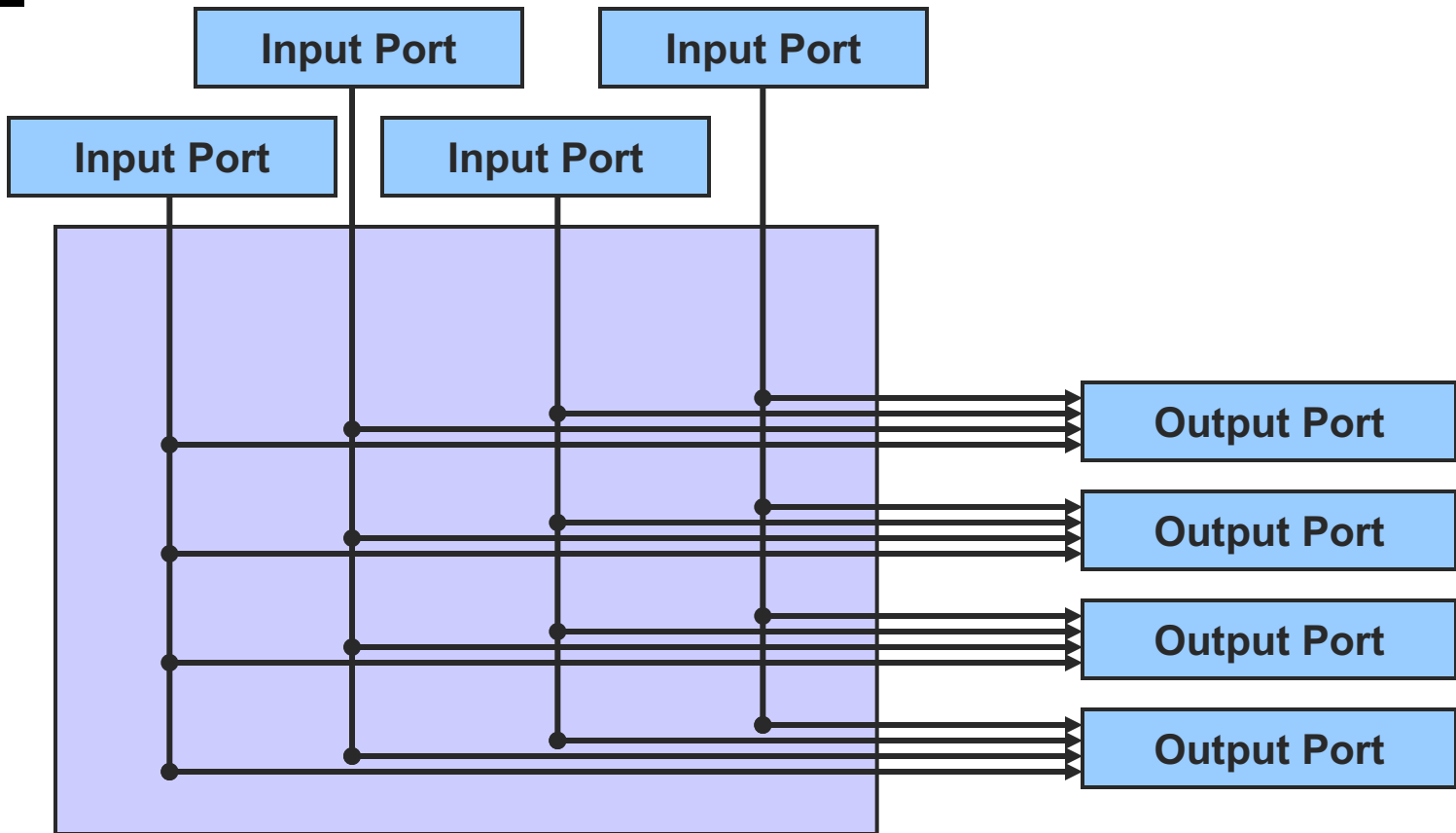


# [ Crossbar Switch ]

- Every input port is connected to every output port
  - $N \times N$
- Output ports
  - Complexity scales as  $O(N^2)$



# Crossbar Switch



# [Knockout Switch]

## ■ Problem

- Full crossbar requires each output port to handle up to  $N$  input packets

## ■ Assumption

- It is unlikely that  $N$  inputs will have packets destined for the same output port

## ■ Instead

- implement each port to handle  $L < N$  packets at the same time

## ■ Challenges

- What value of  $L$  to use
- Managing hotspots



# [ Knockout Switch ]

- Output port design
  - Packet filters
    - Recognize packets destined for a specific port
  - Concentrator
    - Selects up to  $L$  packets from those destined for this port
    - “Knocks out” (discards) excess packets
  - Queue
    - Length  $L$



# [ Knockout Switch ]

## ■ Goal

- Want some fairness
- No single input should have its packets always “knocked out”

## ■ Approach

- Essentially a “knock out” tennis tournament with each game of 2 players (packets) chosen randomly
- Overall winner is selected by playing  $\log N$  rounds, and keeping the winner



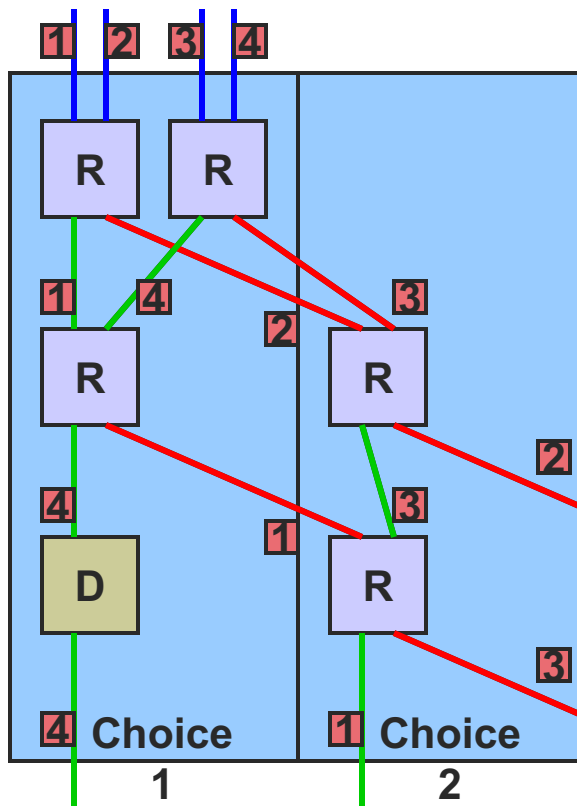


# [ Knockout Switch ]

- Pick  $L$  from  $N$  packets at a port
  - Output port maintains  $L$  cyclic buffers
  - Shifter places up to  $L$  packets in one cycle
  - Each buffer gets only one packet
  - Output port uses round-robin between buffers
  - Arrival order is maintained
- Output ports scale as  $O(N)$



# Knockout Switch



Choose L of N

Ex: 2 of 4



2x2  
random  
selector



Delay  
unit

Discard

What happens  
if more than L  
arrive?

Discard



# [ Self-Routing Fabrics ]

## ■ Idea

- Use source routing on “network” in switch
- Input port attaches output port number as header
- Fabric routes packet based on output port

## ■ Types

- Banyan Network
- Batchner-Banyan Network
- Sunshine Switch



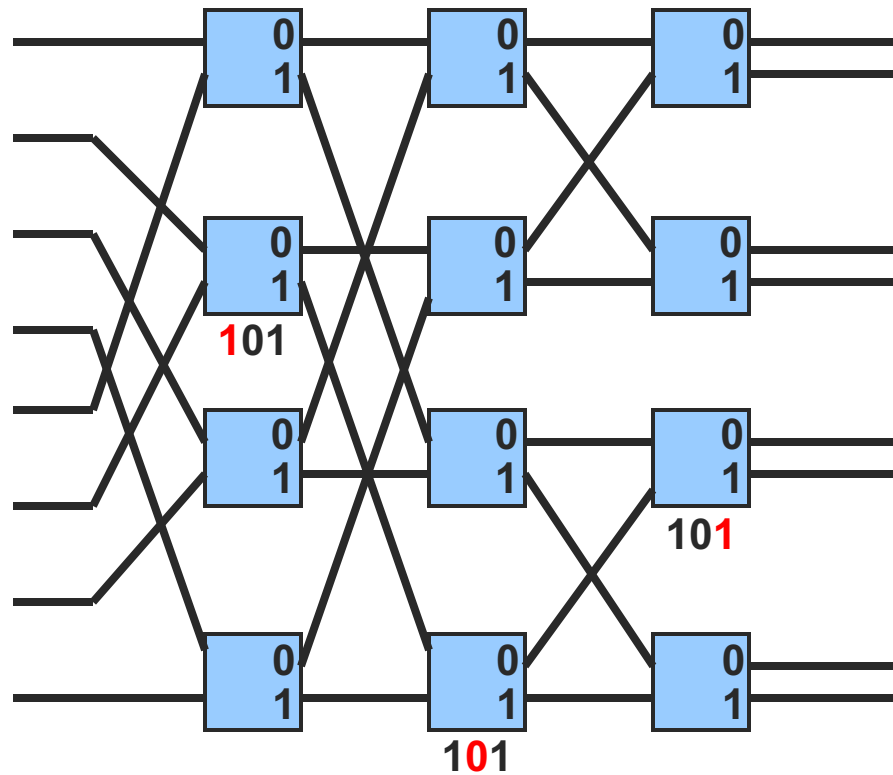
# [Banyan Network]

- A network of 2x2 switches
  - Each element routes to output 0 or 1 based on packet header
  - A switch at stage  $i$  looks at bit  $i$  in the header



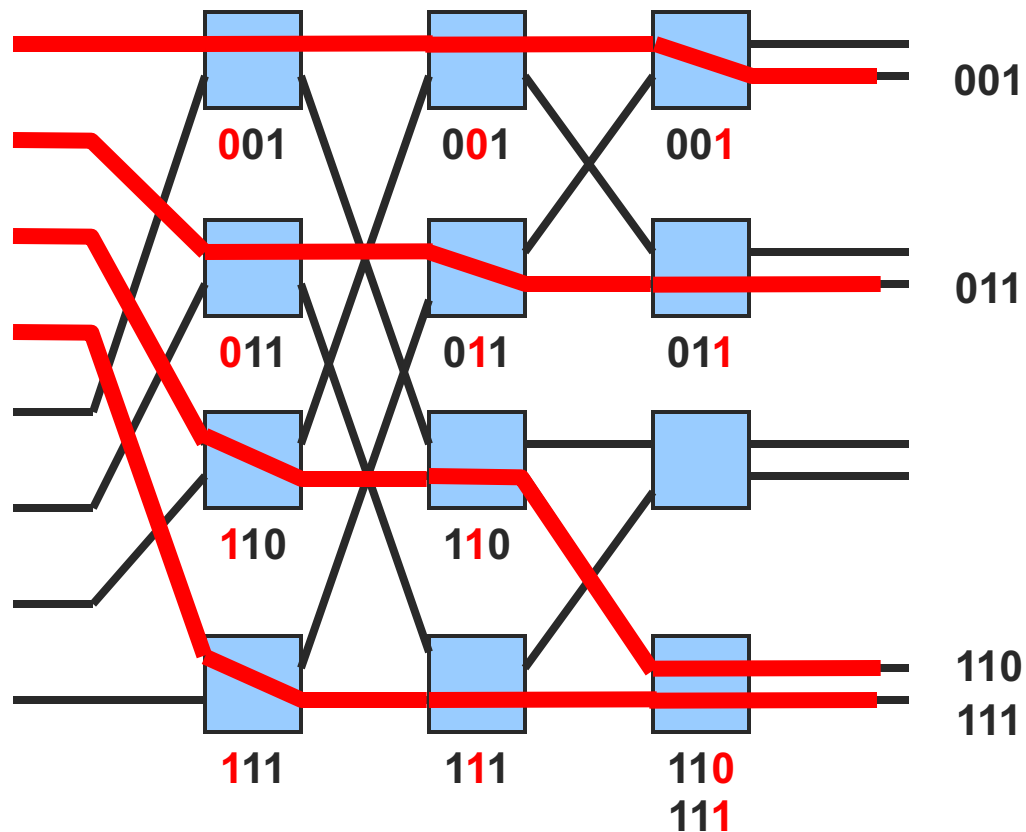


# [Banyan Network





# [ Banyan Network



# [Banyan Network]

## ■ Perfect Shuffle

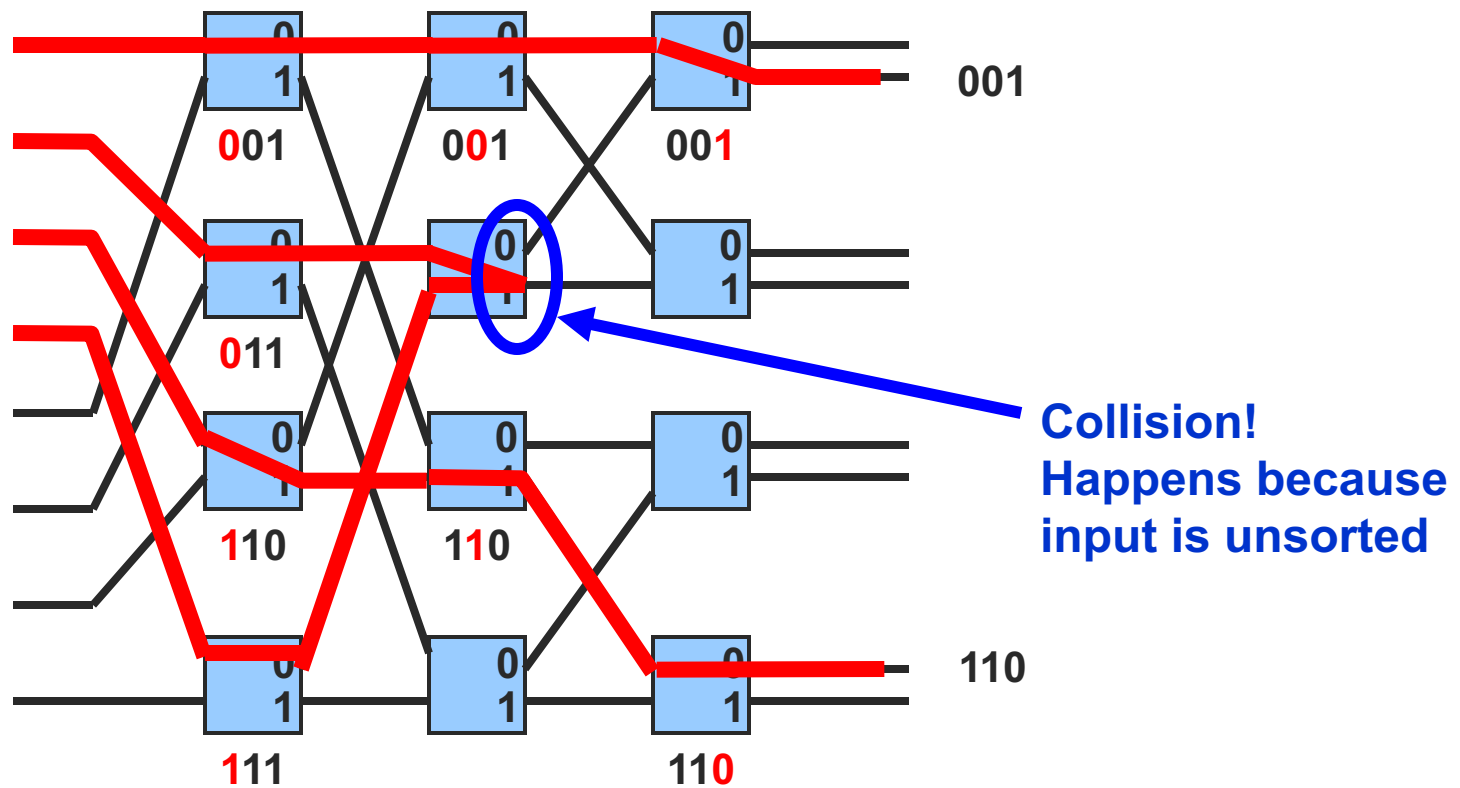
- N inputs requires  $\log_2 N$  stages of  $N/2$  switching elements
- Complexity on order of  $N \log_2 N$

## ■ Collisions

- If two packets arrive at the same switch destined for the same output port, a collision will occur
- If all packets are sorted in ascending order upon arrival to a banyan network, no collisions will occur!



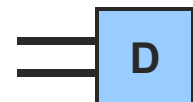
# Collision in a Banyan Network



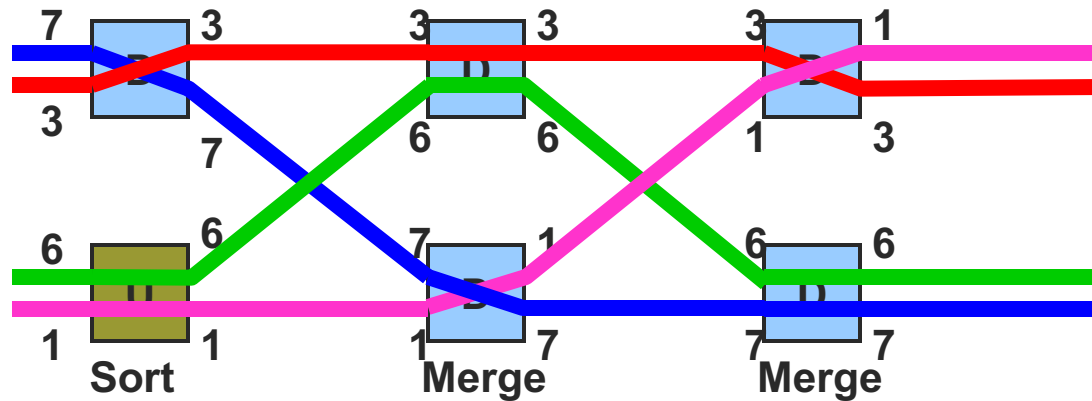


# [ Batcher Network ]

- Performs merge sort
- A network of 2x2 switches
  - Each element routes to output 0 or 1 based on packet header
  - A switch at stage  $i$  looks at the whole header
  - Two types of switches
    - Up switch
      - Sends higher number to top output (0)
    - Down switch
      - Sends higher number to bottom output (1)

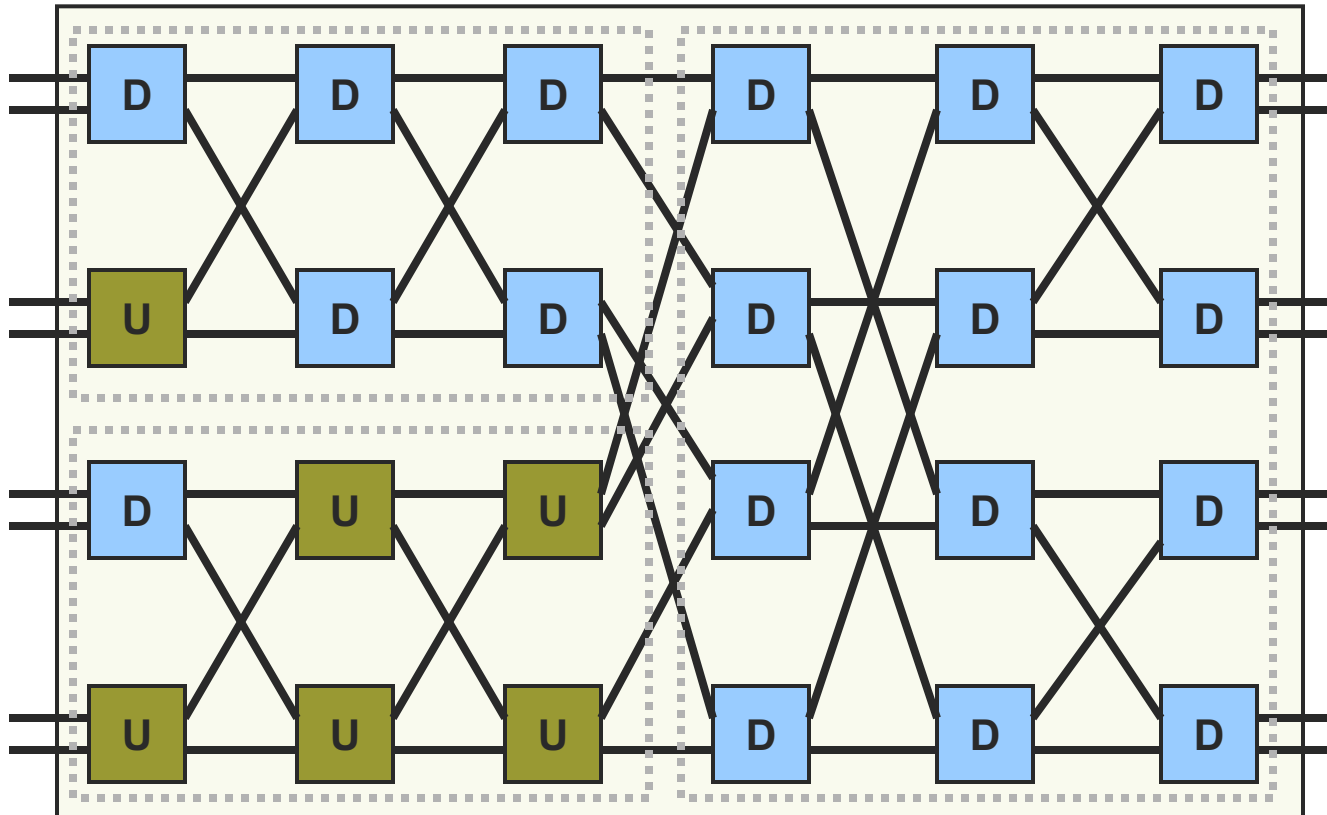


# [Batcher Network



# Batcher Network

Sort inputs 0 – 3 in ascending order



Merge 0 – 3  
with 4 – 7

Sort inputs 4 – 7 in descending order

# [Batcher Network]

## ■ How it really works

- Merger is presented with a pair of sorted lists, one in ascending order, one in descending order
- First stage of merger sends packets to the correct half of the network
- Second stage sends them to the correct quarter

## ■ Size

- $N/2$  switches per stage
- $\log_2 N \times (1 + \log_2 N)/2$  stages
- Complexity =  $N \log_2^2 N$



# [Batcher-Banyan Network]

## ■ Idea

- Attach a batcher network back-to-back with a banyan network
- Arbitrary unique permutations can be routed without contention



# [ Sunshine Switch ]

- Sunshine Switch
  - Like a knockout switch
    - Can handle up to  $L$  packets per output port
  - Recirculates overflow packets
    - If more than  $L$  packets arrive for any output port in one cycle



# [ Sunshine Switch ]

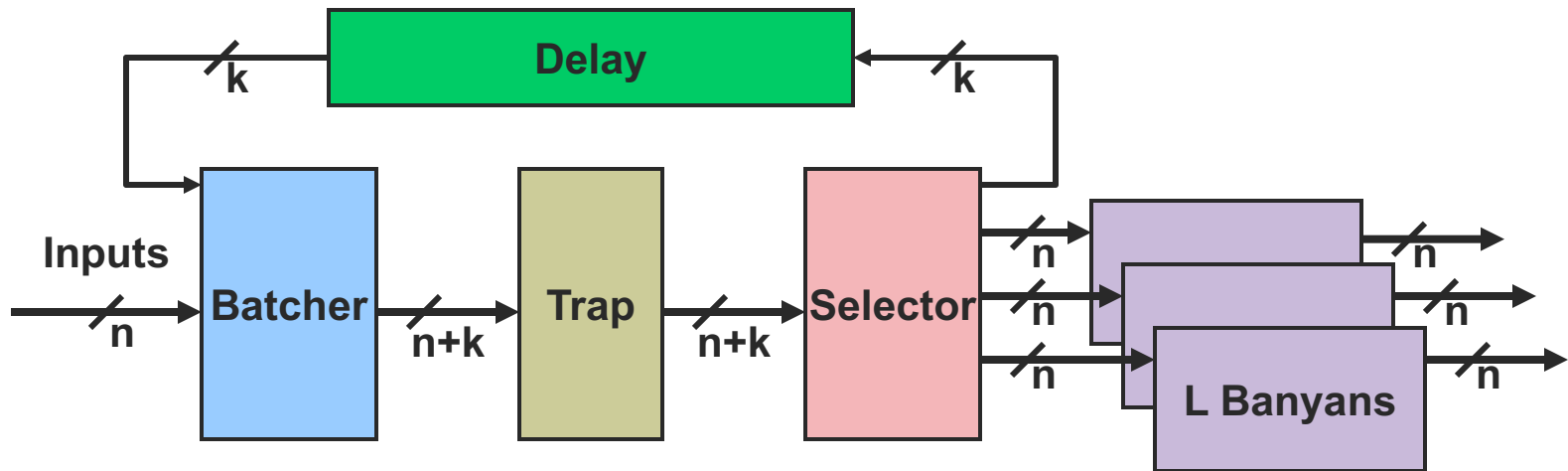
- Elements

- Multiple Banyan networks
  - Enables multiple packets per output port
- Delay Box
  - Excess (K) packets are recirculated and resubmitted to the switch
- Batcher network
  - N new packets
  - K delayed packets
- Trap
  - Identifies packets destined for banyan
  - Identifies excess packets
- Selector
  - Routes multiple packets for same output on separate banyans





# [ Sunshine Switch ]





# [ Sunshine Switch ]

- Can packets circulate for ever?
  - Priority bit is used to favor older packets
  - Priority bit also ensure packet order is preserved through the switch

