CS3640

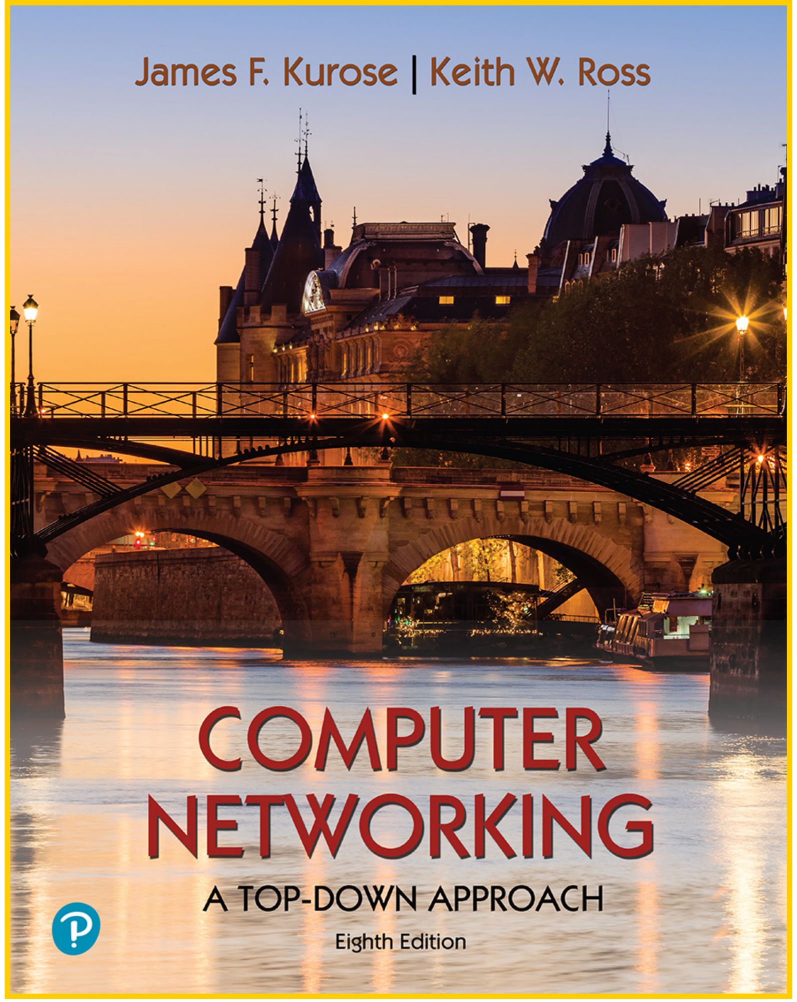
Link Layer (2): MAC Protocols

Prof. Supreeth Shastri
Computer Science
The University of Iowa

Lecture goals

continued exploration of MAC protocols and link layer technologies

- *Channel Partitioning Protocols (2)*
- *Random Access Protocols (3)*
- *Taking Turns Protocols (2)*
- *Real-world example*



Chapter 6.3

Multiple Access Protocols

sharing a single broadcast channel amongst multiple nodes

1. Channel Partitioning Protocols

divide channel into small pieces (e.g., time slots, frequency), and allocate each piece to one node

2. Random Access Protocols

do not divide the channel, and allow nodes to transmit at any time, but detect/ recover from collisions

3. Taking-turns Protocols

nodes take turn to send frames; this dynamism achieves balance between the first two class of protocols

Characterizing MAC Protocols

- MAC protocols are distributed algorithms that determine how nodes share a channel, *i.e., determine when nodes transmit*
- all communications about channel sharing must use channel itself, *i.e, no out-of-band channel for coordination*

An ideal MAC protocol

For a broadcast channel with a rate R bits/sec, it should be/enable the following:

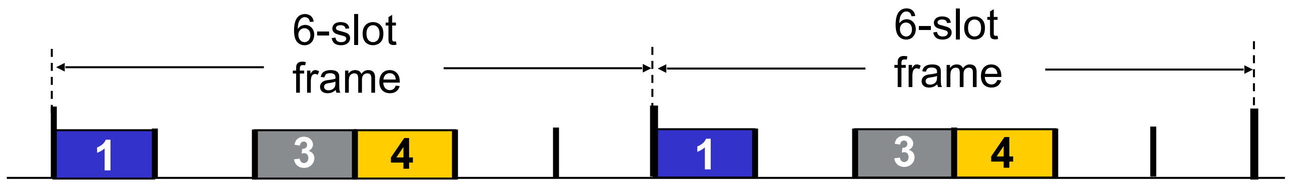
1. if only one node wants to transmit, it can send at rate R
2. if M nodes want to transmit, each can send at average rate R/M
3. be fully decentralized i.e., no centralized controller, no sync of clocks amongst nodes
4. be simple

Channel Partitioning Protocols

Channel Partitioning Protocols

TDMA: Time Division Multiple Access

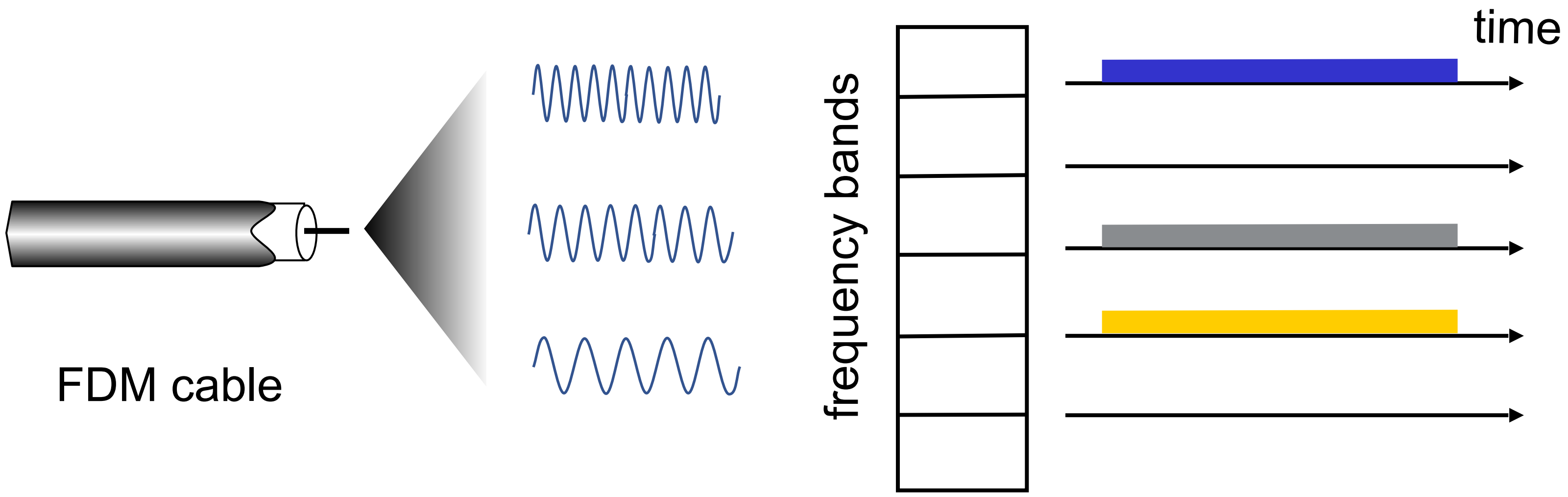
- access to channel in "rounds"
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



Channel Partitioning Protocols

FDMA: Frequency Division Multiple Access

- channel spectrum is divided into frequency bands
- each station assigned a particular frequency band
- unused transmission time in a frequency band goes idle/waste
- example: 6-station LAN, 1,3,4 have packet to send, and bands 2,5,6 are idle



Random Access Protocols

Random Access Protocols

- When node has packet to send,
  - ➡ it transmits at full channel data rate R
  - ➡ no *a priori* coordination among nodes
- If two or more nodes transmit simultaneously: collision
- Random Access Protocols specify
  - ➡ how to detect collisions
  - ➡ how to recover from collisions
- Examples of random access protocols
  - ➡ ALOHA, slotted ALOHA (impolite speakers)
  - ➡ CSMA, CSMA/CD (more polite speakers)

Slotted ALOHA

Channel assumptions

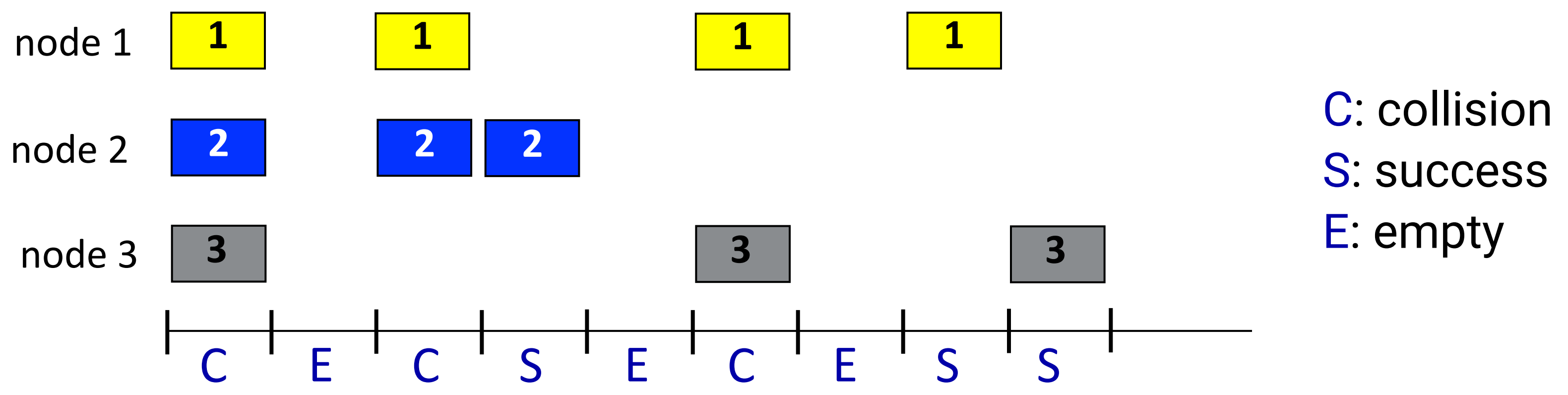
- all frames are of same size
- time is divided into equal length slots *(e.g., time to transmit 1 frame)*
- clocks at all nodes are synchronized
- nodes start to transmit only at the beginning of a slot
- if two or more nodes transmit in slot, all nodes detect collision

Protocol operation

- When a node has a new frame, it transmits it in next slot
- If no collision is detected: node can send any new frame in next slot
- If collision is detected: node retransmits the frame in each subsequent slot with probability **p** until success

randomization – *why*?

Slotted ALOHA



Pros

- single active node can continuously transmit at full rate of channel
- highly decentralized
- simple

Cons

- a collision wastes the full slot
- leaves idle slots even when nodes have data to transmit
- clock synchronization

Slotted ALOHA

Efficiency: long run fraction of slots that transmit data successfully (assuming many nodes, and all of them have many frames to send)

Suppose ***N*** *nodes with many frames to send, each transmits in slot with probability* ***p***

- prob that given node has success in a slot = p(1-p)N-1
- prob that any node has a success = Np(1-p)N-1
- take limit of Np*(1-p*)N-1 as N goes to infinity, max efficiency = 1/e = .37

At best, only 37% of the slots do useful work!

Carrier Sense Multiple Access (CSMA)

Simple CSMA: listen before transmit

- if channel sensed to be idle: *transmit entire frame*
- if channel sensed to be busy: *defer transmission*

human analogy: don't interrupt while others are talking!

CSMA/CD: CSMA with collision detection

- collisions detected within short time
- colliding transmissions are immediately aborted, thus reducing channel wastage

human analogy: a polite conversationalist

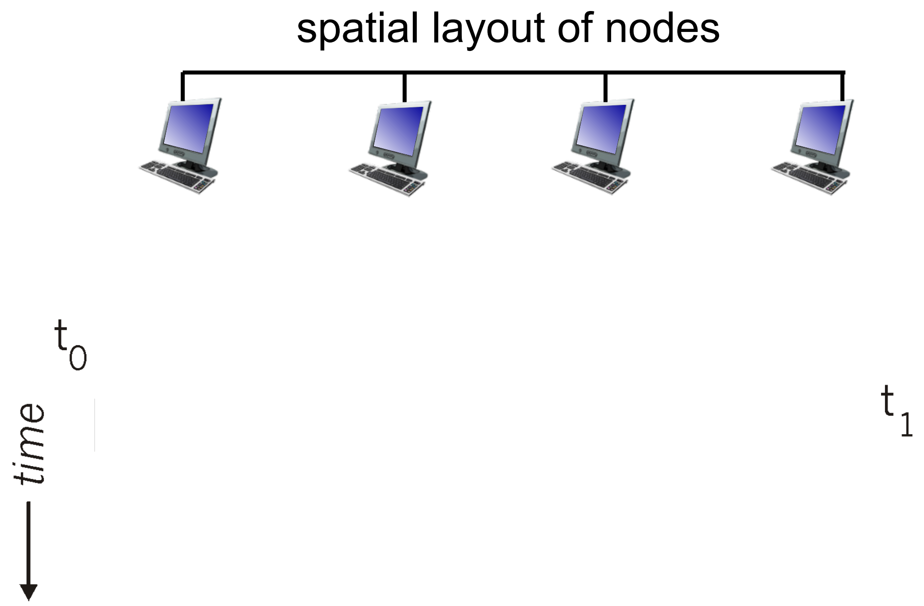
Simple CSMA

Collisions can still occur with carrier sensing:

- propagation delay means two nodes may not hear each other's just-started transmission

When there is a collision: entire packet transmission time is wasted

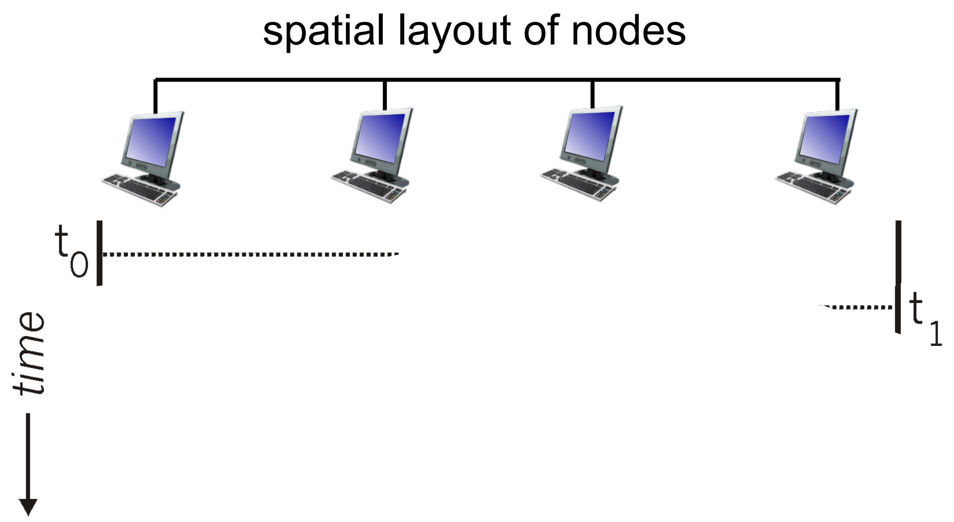
- distance and propagation delay play role in in determining collision probability



CSMA/CD

CSMA/CD reduces the amount of time wasted in collisions

- transmission aborted on collision detection



Ethernet CSMA/CD algorithm

1. Ethernet receives datagram from network layer, creates a frame
2. If Ethernet senses channel:

   if idle: start frame transmission.

   if busy: wait until channel idle, then transmit
3. If entire frame transmitted without collision - done!
4. If another transmission detected while sending: abort, send jam signal
5. After aborting, enter binary exponential backoff
   - after m^{th} collision, chooses K at random from $\{0,1,2, \ldots, 2^m - 1\}$. Ethernet waits K slots, returns to Step 2
   - more collisions for the same frame: longer backoff interval

Taking-Turns Protocols

Taking Turns MAC Protocols

Channel Partitioning MAC Protocols

- inefficient at low load: if only one node is active, 1/N bandwidth is allocated to it!
- allows sharing the channel *efficiently and fairly* at high load

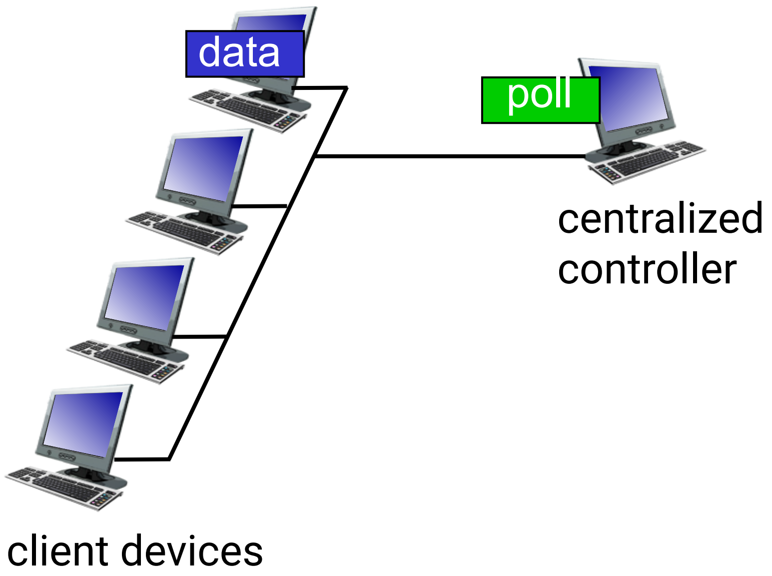
Random Access MAC Protocols

- *efficient at low load*: a single node can fully utilize channel
- experiences overhead at high load due to collisions

Taking Turns Protocols *look for best of both worlds!*

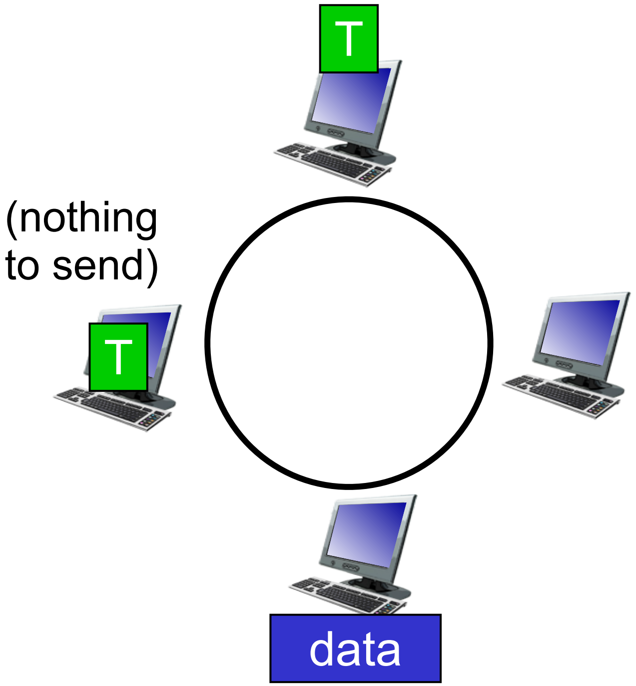
Polling Protocols

- a controller "invites" other nodes to transmit in turn
- typically used with "dumb" devices
- suffers from polling overhead, latency, and single point of failure (controller)



Token Passing Protocols

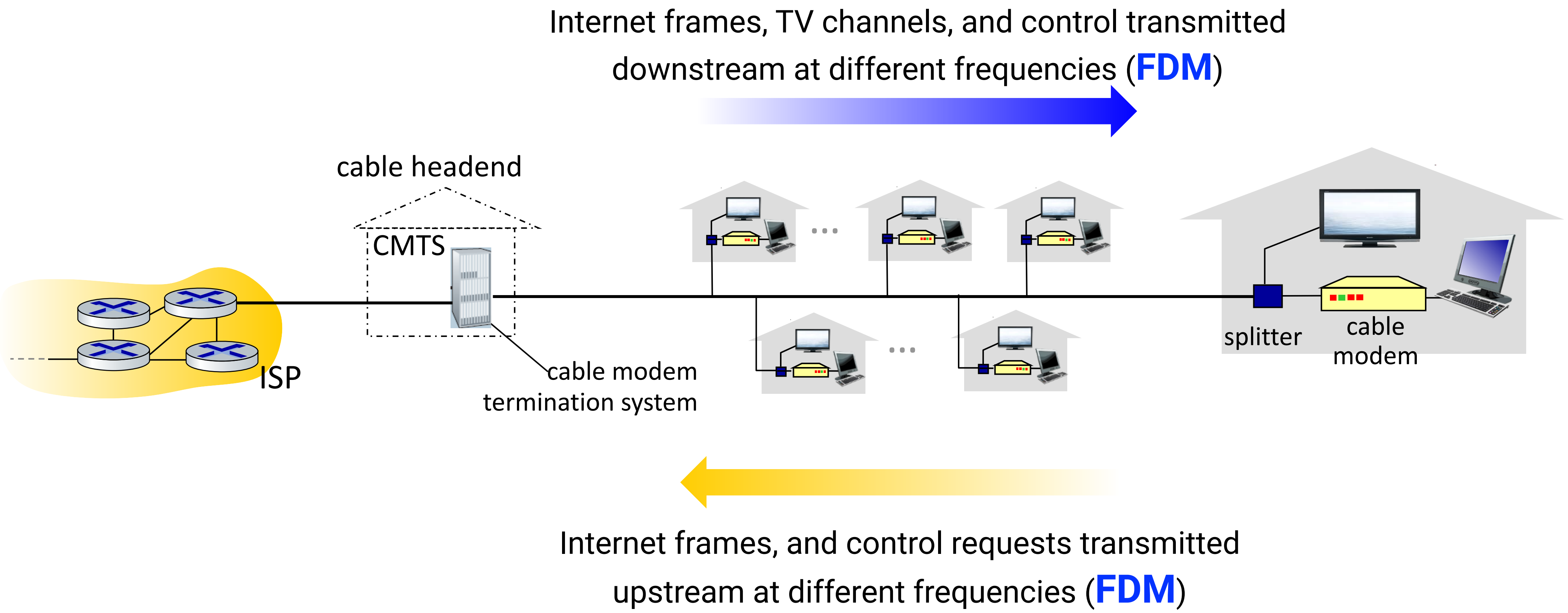
- control token message explicitly passed from one node to next, sequentially
- transmit while holding token
- suffers from token overhead, latency, and single point of failure (token)

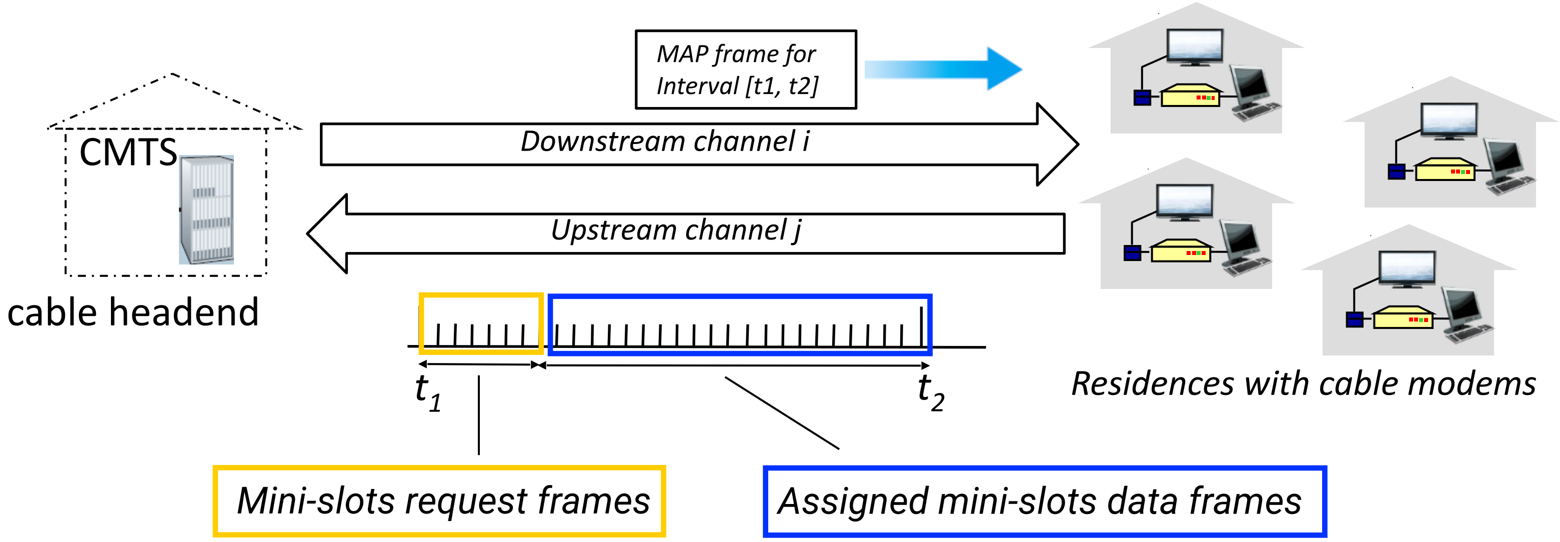


Protocols in Practice

Example: Cable Access Network

DOCSIS: Data Over Cable Service Interface Specification





Upstream frequency channels are further divided into mini-slots using **TDM**

- Mini-slot-data frames: to send data upstream; could only be used after CMTS grants permission
- Mini-slot-request frames: modems send request for a desired mini-slot in the next interval
- However, mini-slot-request frames are not preassigned; so modems employ **random access** to select one or more slots, and use **binary backoff** upon inferring collision
- CMTS sends a MAP frame in downstream to assigns upstream mini-slots for the next iteration

Spot Quiz (ICON)