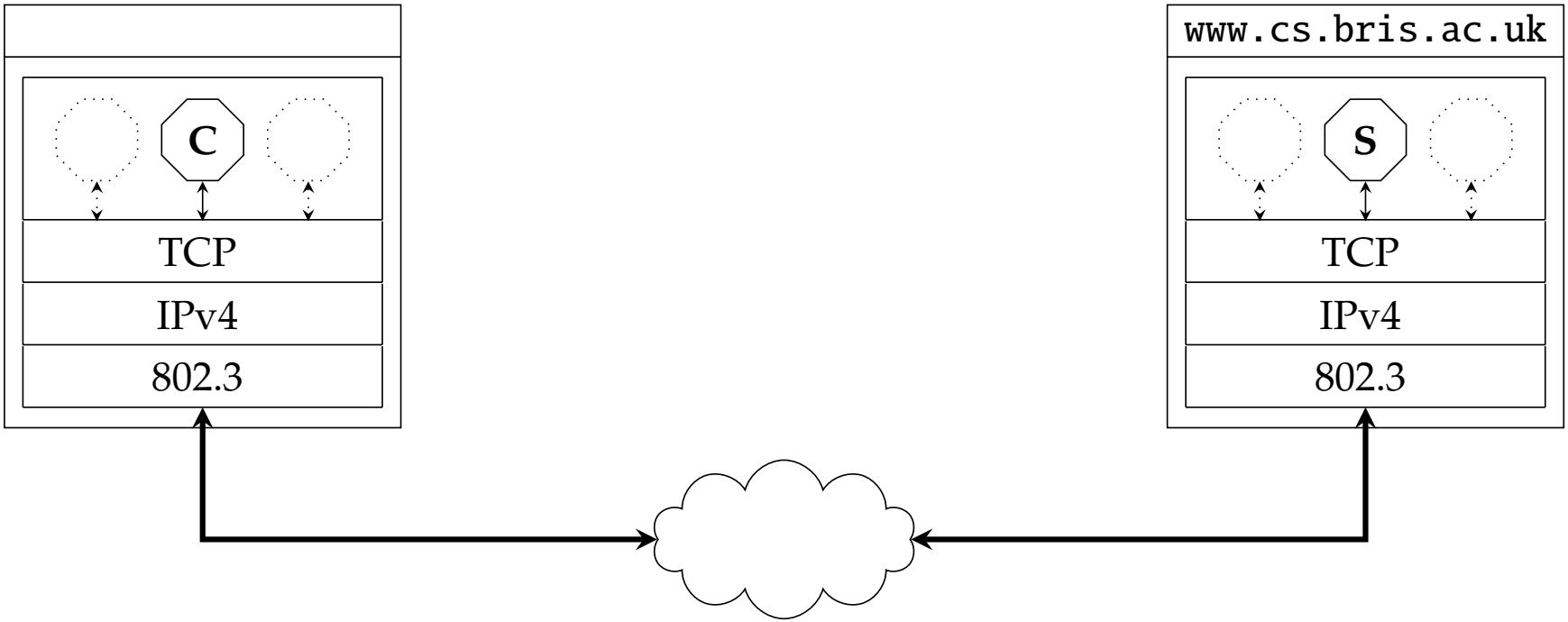
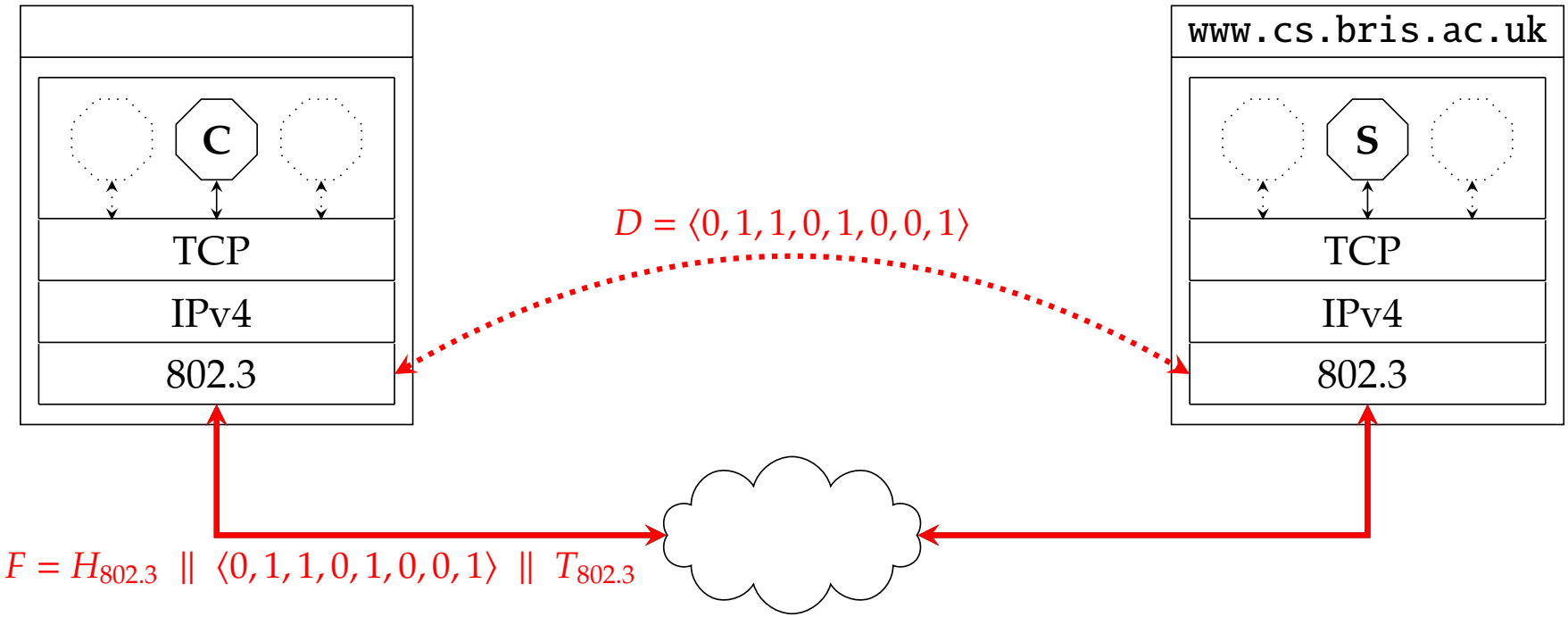
- Goal: investigate the **link layer** e.g.,
  1. addressing,
  2. framing,
  3. multiple access protocols, and
  4. examples implementations

  st. we can transmit (structured) **frames** between two end-points using a (shared) physical connection.



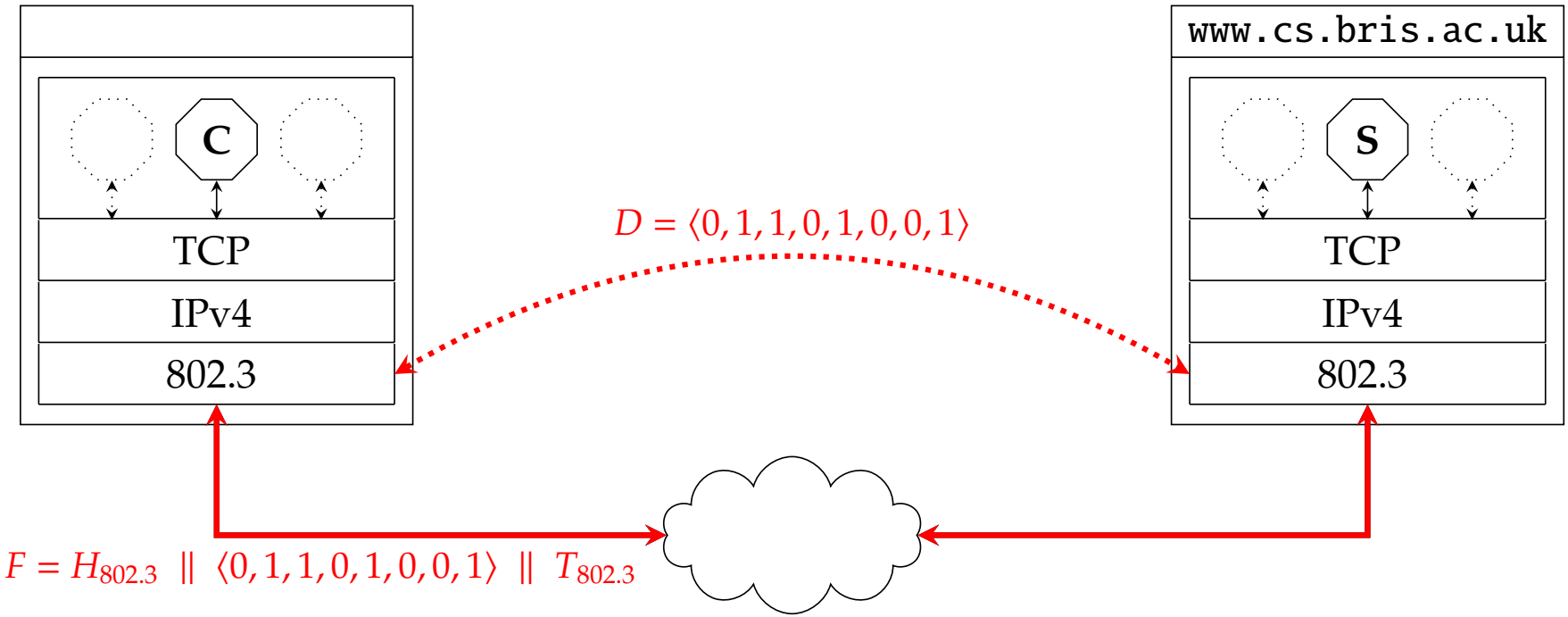
- Goal: investigate the **link layer** e.g.,
  1. addressing,
  2. framing,
  3. multiple access protocols, and
  4. examples implementations

  st. we can transmit (structured) **frames** between two end-points using a (shared) physical connection.

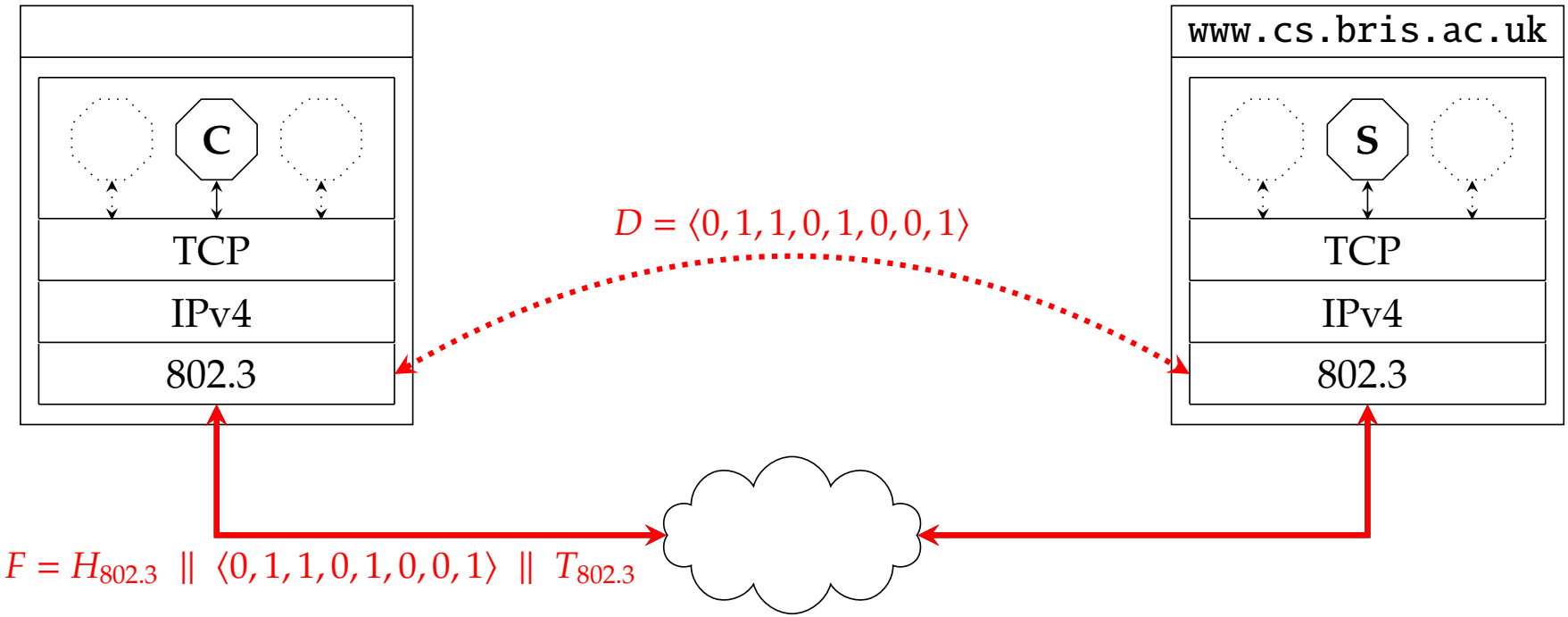


- Goal: investigate the **link layer** e.g.,
  1. addressing,
  2. framing,
  3. multiple access protocols, and
  4. examples implementations

  st. we can transmit (structured) **frames** between two end-points using a (shared) physical connection.

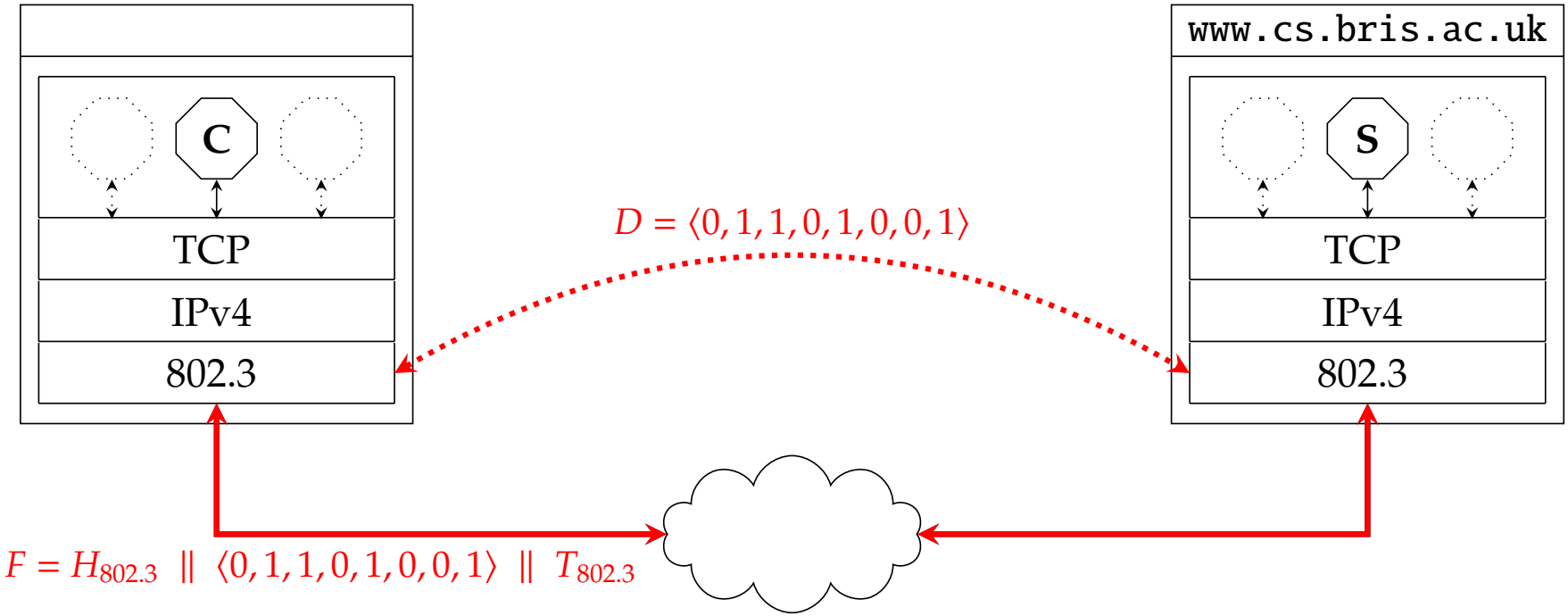


- In theory (per the OSI model): the link layer is composed of
  1. a higher-level **Logical Link Control (LLC)** sub-layer (e.g., 802.2) which interfaces with higher-level protocols, and
  2. a lower-level **Media Access Control (MAC)** sub-layer (e.g., 802.3) which manages the physical communication medium.

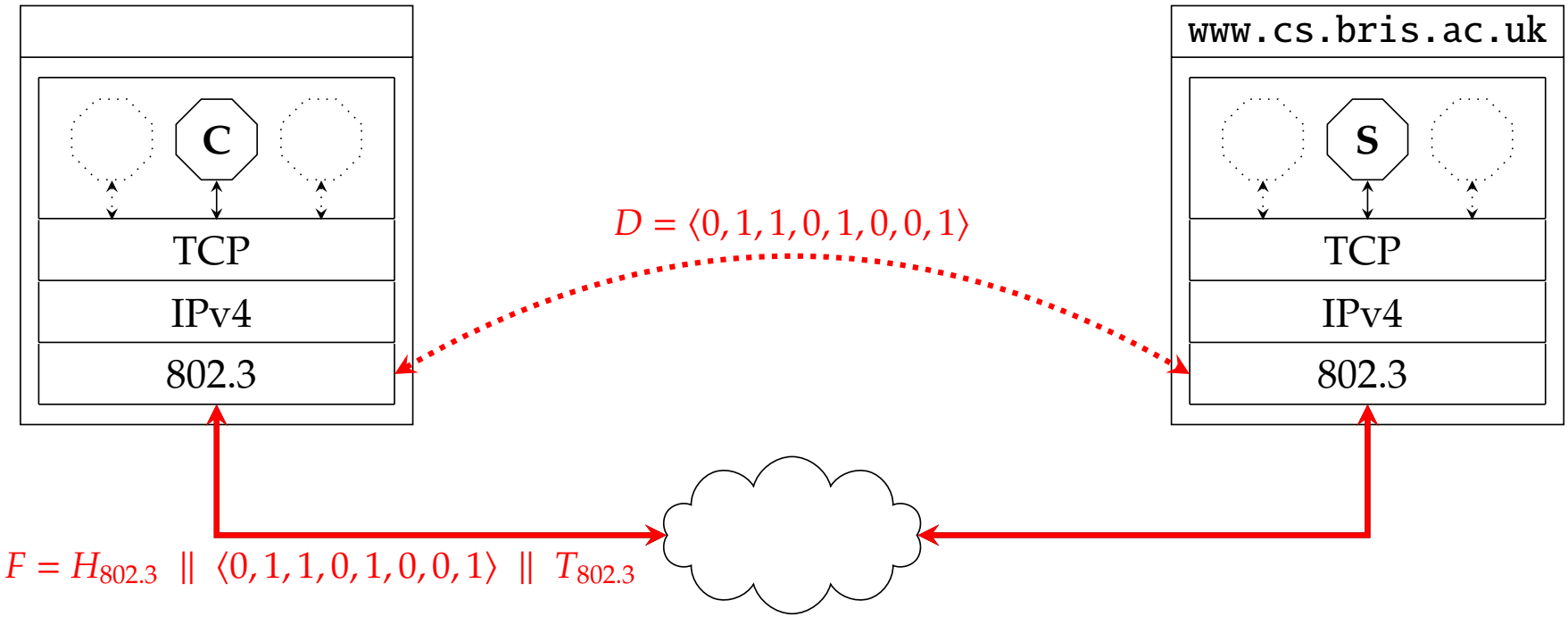


- In practice (per the Internet model [5, Section 2]):
  - separation of LLC and MAC sub-layers is less prescriptive, plus
  - if we only consider IPv4 and TCP above the link layer, the LLC sub-layer becomes very lightweight

  so we focus on the MAC sub-layer.



- In theory: the MAC sub-layer supports *either*
  1. point-to-point connections meaning dedicated access, *or*
  2. multi-point connections meaning shared access.

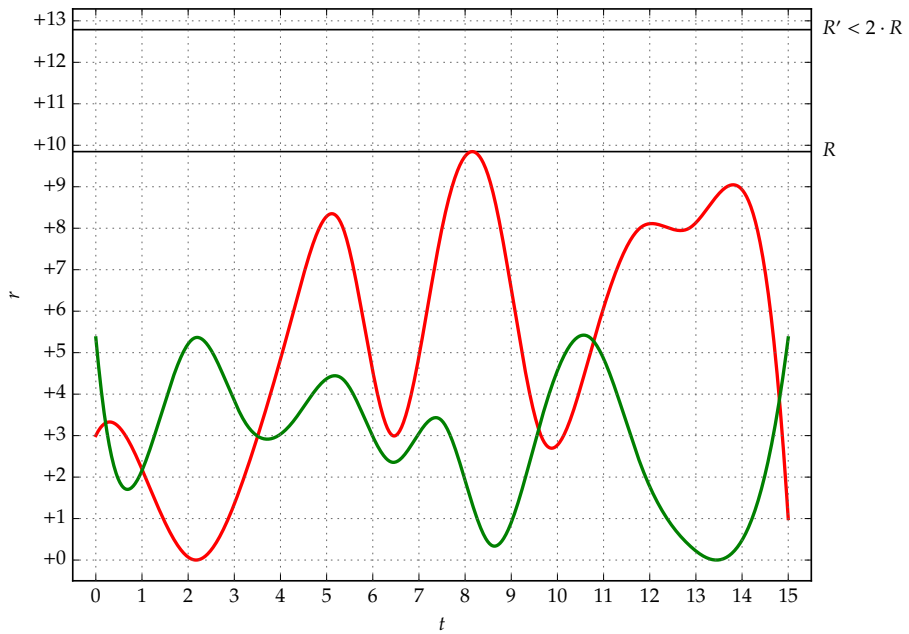


- In practice: point-to-point connections
  - prevents contention (i.e., require no management), *but*
  - imply a fixed topology which is hard to scale,
  - cannot support genuine broadcast transmission, plus
  - are likely to be under-utilised

so we focus on managing **multiple access** to a multi-point connection.

Concepts (1)

- Consider two hosts $\mathcal{H}_i$ and $\mathcal{H}_j$ communicating at some rate r:

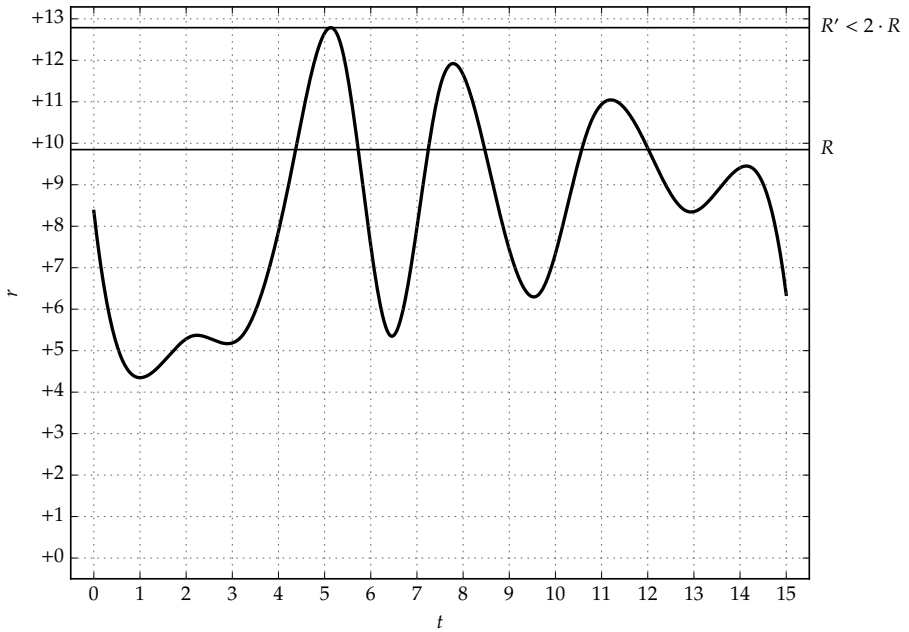


- Problem: how can we support shared access to a connection?
- Solution #1: use **static multiplexing**, i.e.,
  - assume a connection whose available bandwidth is $R > r$ for all t,
  - share that bandwidth, e.g., by using TDM or FDM

noting this is ineffective if demand is "bursty".

Concepts (1)

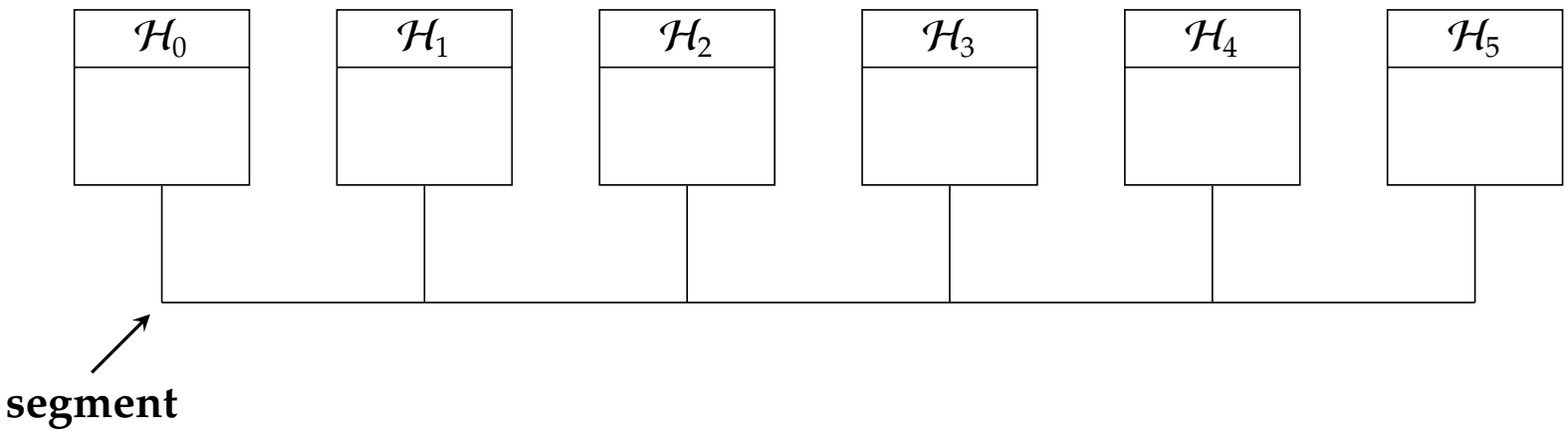
- Consider two hosts $\mathcal{H}_i$ and $\mathcal{H}_j$ communicating at some rate r:



- Problem: how can we support shared access to a connection?
- Solution #2: use **statistical multiplexing**, i.e.,
  - assume a connection whose available bandwidth is $R' < 2 \cdot R$,
  - enforce a **multiple access protocol** that considers the *combined* rate at a given t, thus
  - *dynamically* sharing the bandwidth based on *demand*.

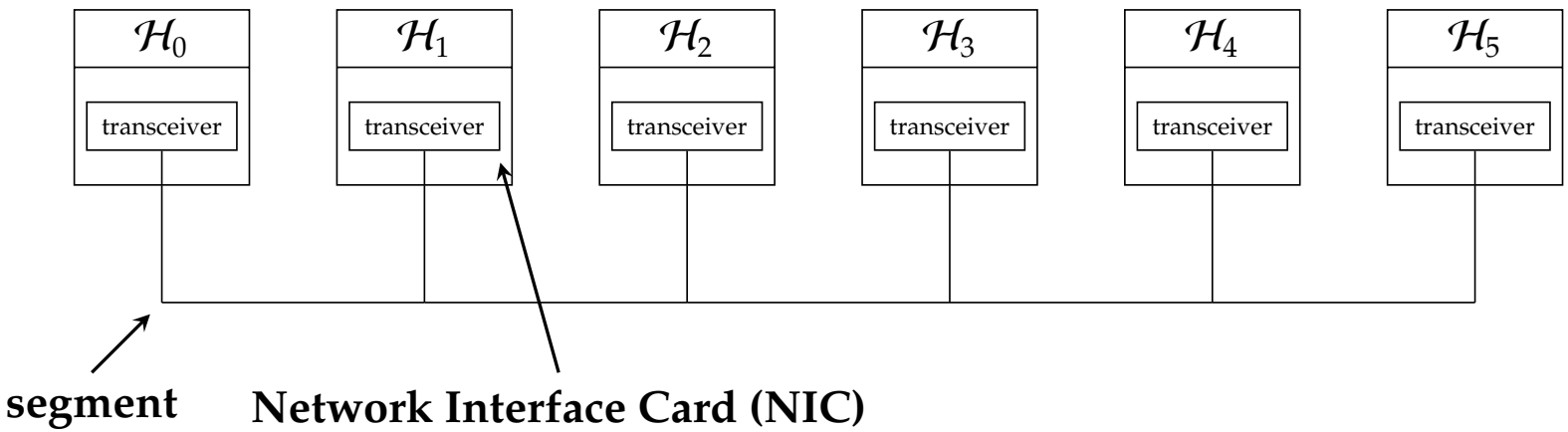
Definition

A network **segment** is a portion of said network, typically (but not necessarily, depending on the network type) understood as a portion on which attached hosts are physically connected.



Definition

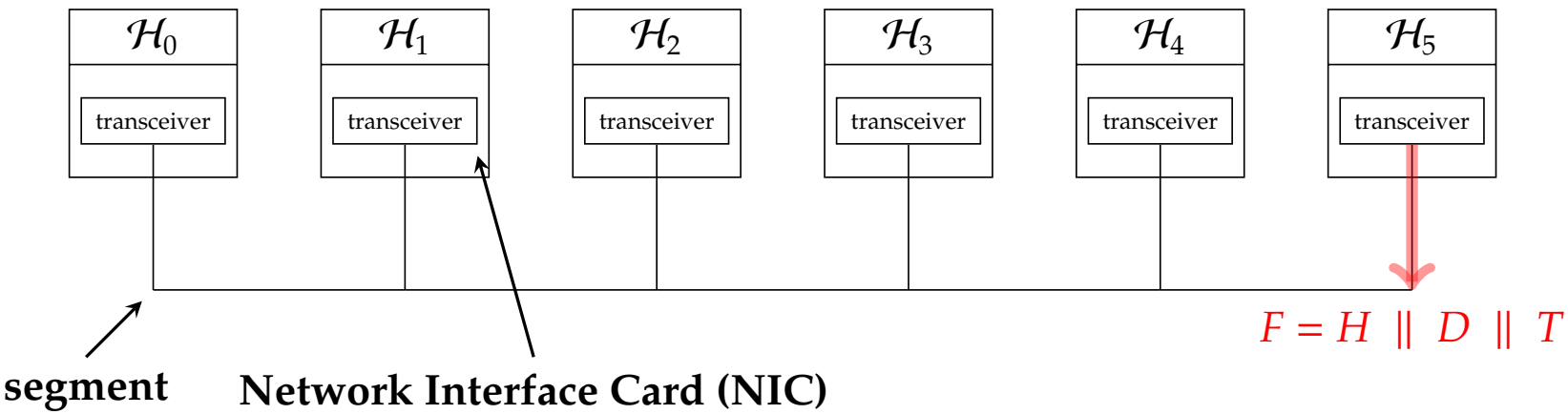
Each host is connected to a network segment via a transceiver, which we normally term a **Network Interface Card (NIC)**.



Definition

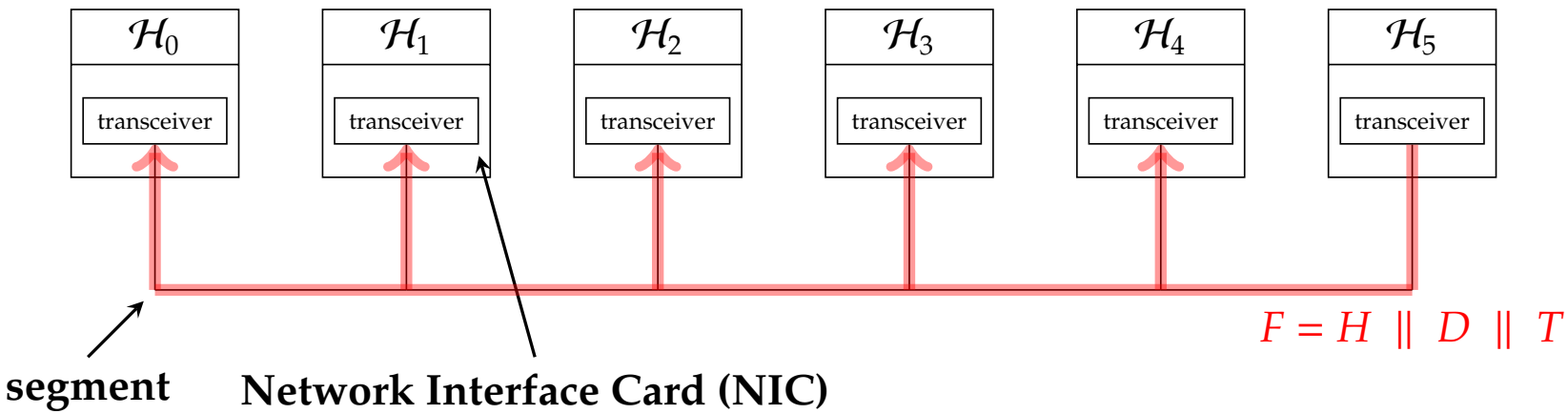
When a host wants to transmit some payload, it encapsulates it in a **frame** then presented to the physical layer: the act of **framing**

1. deliniate the start and end of the frame st. it can be parsed from an unstructured bit-sequence, and
2. adds relevant control information.



Definition

A **broadcast domain** is a logical (sub-)division of a network st. all hosts within it can communicate with each other through use of broadcast transmission.

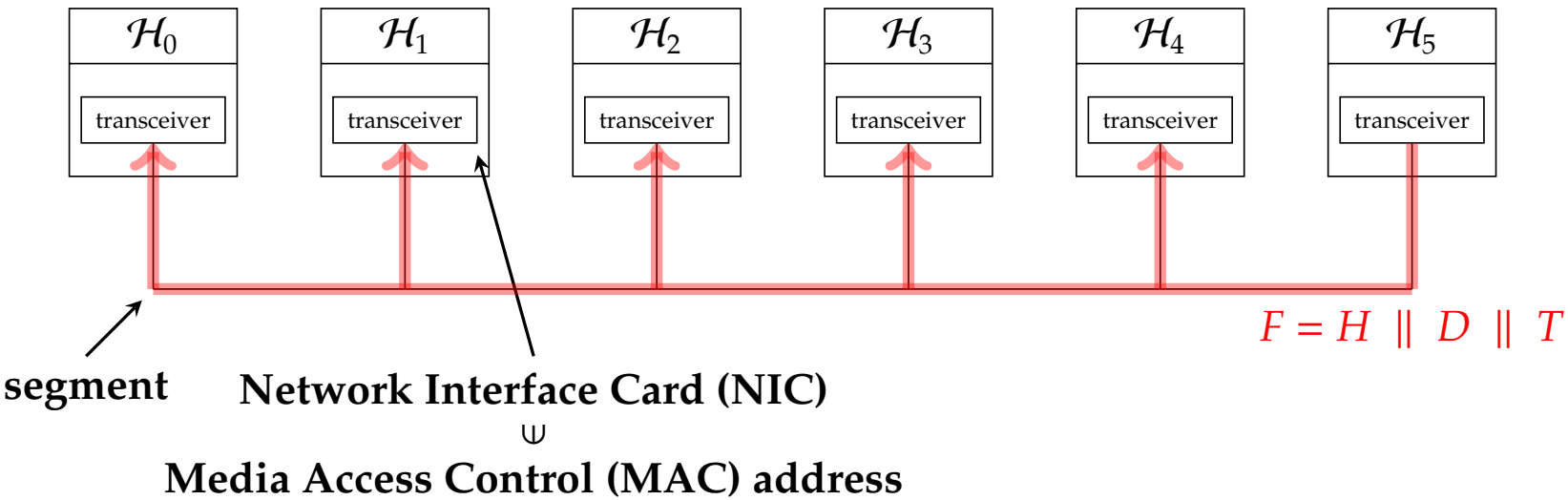


Definition

To disambiguate broadcast transmissions, each host is identified by a **Media Access Control (MAC) address**:

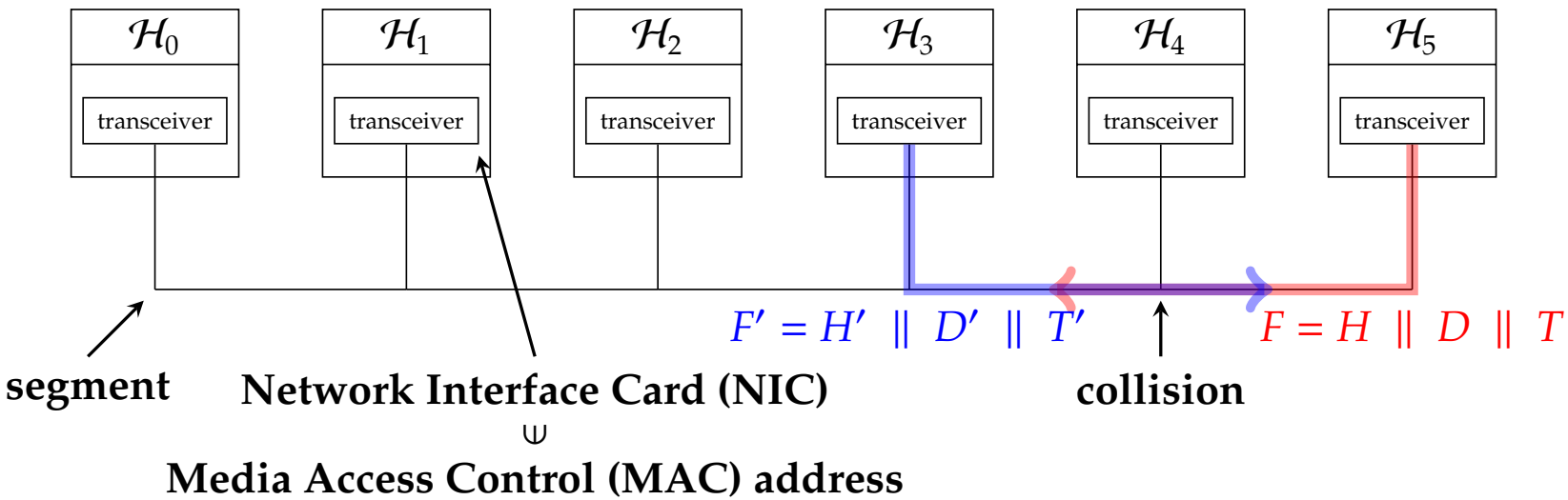
- typically hard-coded into NIC,
- needs to be unique wrt. broadcast domain,
- formed from an **Organisationally Unique Identifier (OUI)** and a NIC-specific identifier.

which form source/destination fields in the frame header. There is typically a reserved **broadcast address** allowing *all* hosts to receive a transmitted frame.



Definition

A **collision domain** is a logical (sub)division of a multiple access network st. transmission by one host may **collide** (or interfere) with a (simultaneous) transmission by another. In general, collision domains are smaller than (i.e., contained within) an associated broadcast domain.

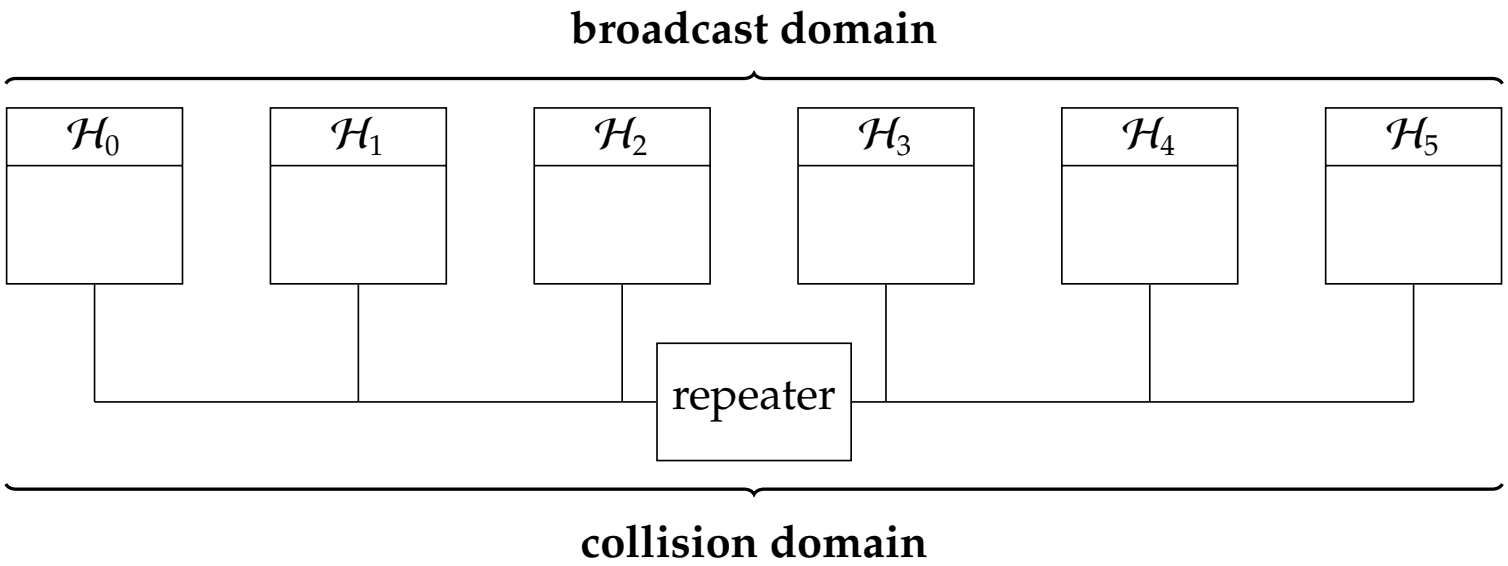


Definition

Using a suitable component (or **network appliance**), e.g.,

1. a **repeater**,
2. a **bridge**,
3. a **hub**, or
4. a **switch**

one can connect multiple separate networks, or network segments, into a single aggregate network. These components a) operate in different layers, and so b) have somewhat differing capabilities and purposes.

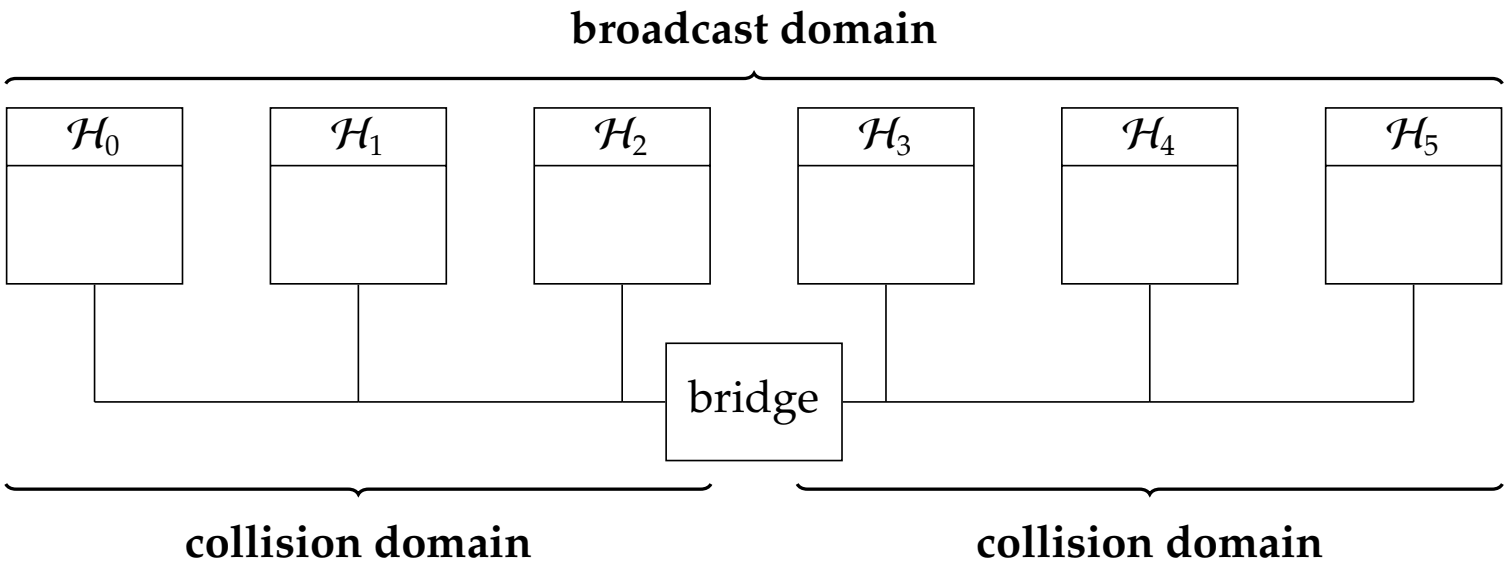


Definition

Using a suitable component (or **network appliance**), e.g.,

1. a **repeater**,
2. a **bridge**,
3. a **hub**, or
4. a **switch**

one can connect multiple separate networks, or network segments, into a single aggregate network. These components a) operate in different layers, and so b) have somewhat differing capabilities and purposes.

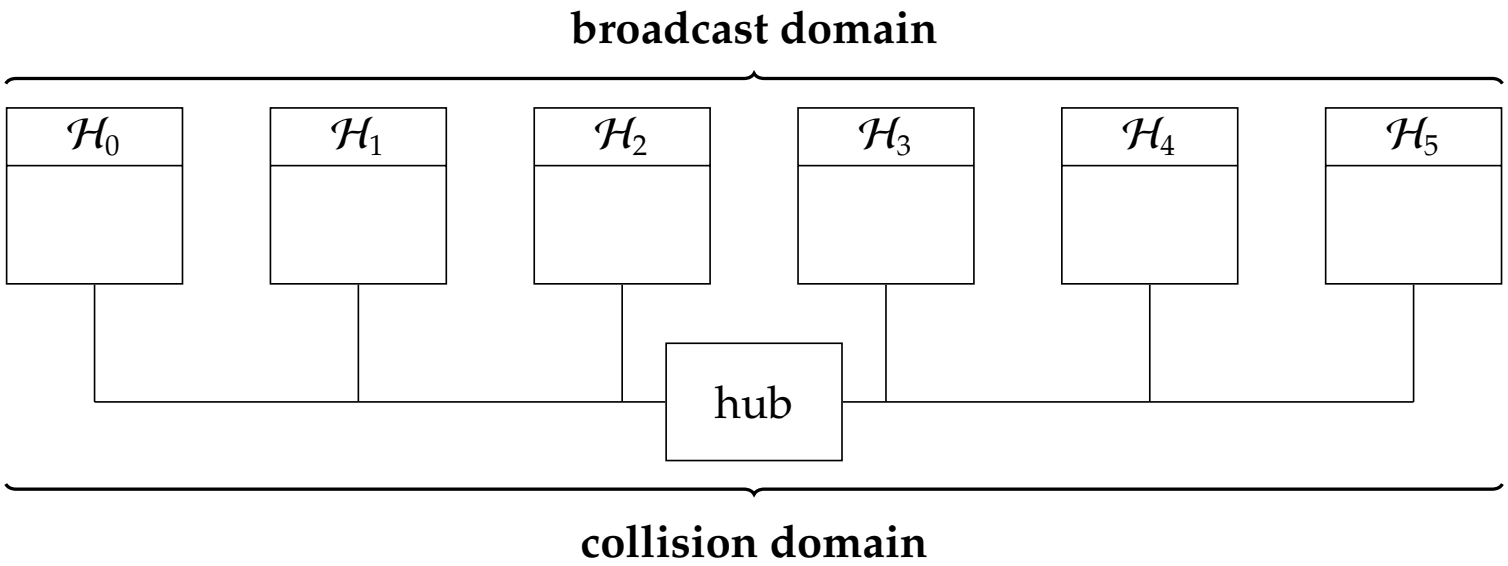


Definition

Using a suitable component (or **network appliance**), e.g.,

1. a **repeater**,
2. a **bridge**,
3. a **hub**, or
4. a **switch**

one can connect multiple separate networks, or network segments, into a single aggregate network. These components a) operate in different layers, and so b) have somewhat differing capabilities and purposes.

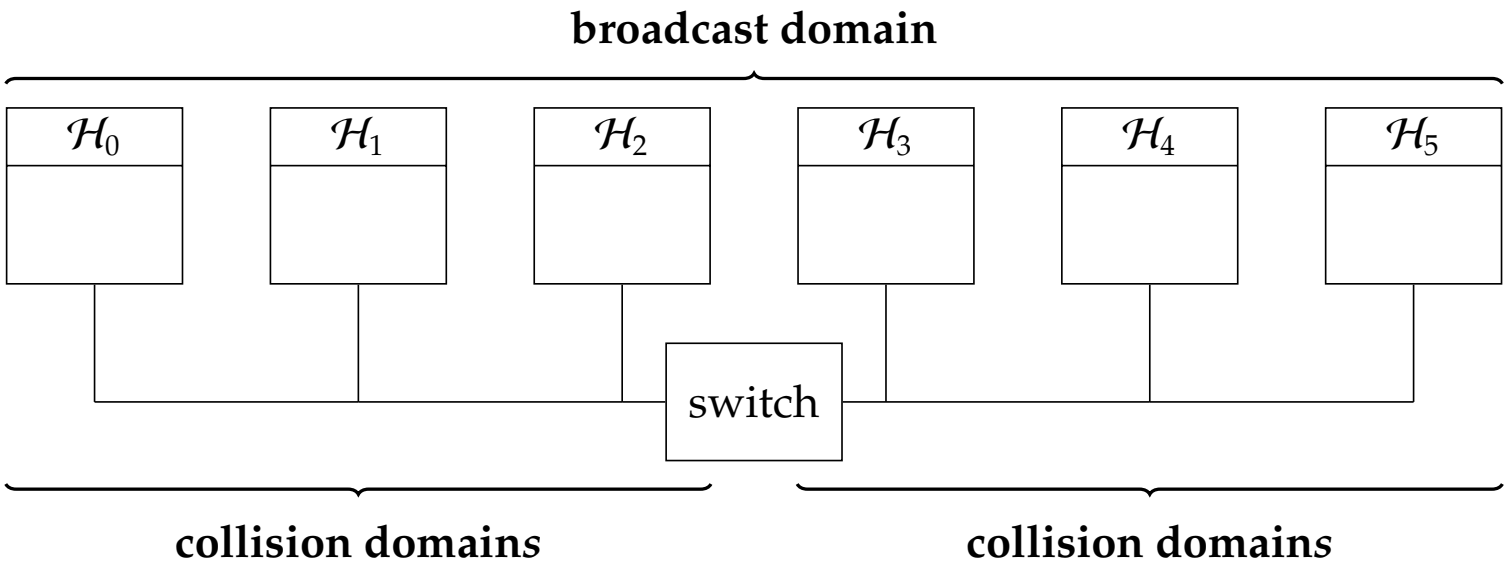


Definition

Using a suitable component (or **network appliance**), e.g.,

1. a **repeater**,
2. a **bridge**,
3. a **hub**, or
4. a **switch**

one can connect multiple separate networks, or network segments, into a single aggregate network. These components a) operate in different layers, and so b) have somewhat differing capabilities and purposes.



802.3 (1)

- 802.3 is based on a number of more general principles.
- In 802-like terminology:
  1. uses a p-persistent **Carrier Sense Multiple Access (CSMA)** protocol,
  2. reduces the cost of collisions via **Collision Detection (CD)**,
  3. reduces the probability of collision synchronisation via **Binary Exponential Back-off (BEB)**

  which is probably clear as mud!

- 802.3 is based on a number of more general principles.

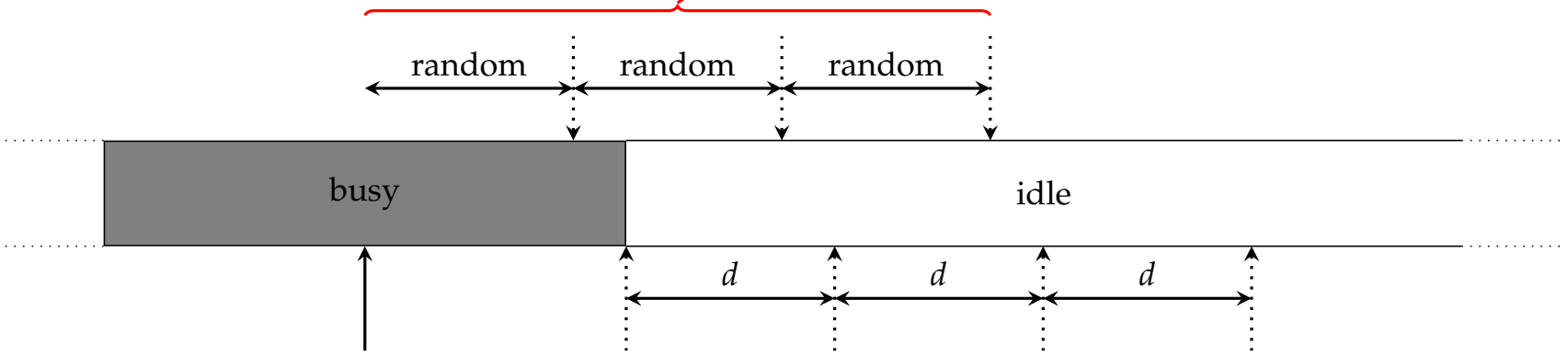
- In English:
  1. senses the medium, and transmits with probability p if idle,
  2. stops transmitting when a collision is detected,
  3. waits upto $2^m \cdot d$ time units when the m-th successive collision is detected

  which *hopefully* makes more sense.

802.3 (2) – CSMA/CD

0-persistent CSMA:

1. if idle, transmit immediately,
2. if busy, wait for random period then retry,
3. if collision, back-off then retry.



1-persistent CSMA:

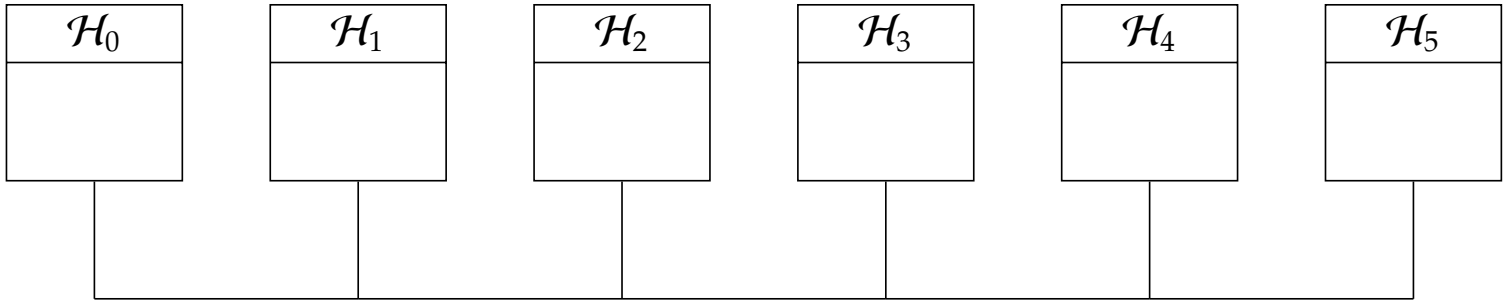
1. if idle, transmit immediately,
2. if busy, wait until idle.
3. if collision, back-off then retry.

***p*-persistent CSMA**:

1. if idle,
   - with probability p transmit immediately, or
   - with probability $1 - p$ wait for d time units then retry
2. if busy, wait until idle.
3. if collision, back-off then retry.

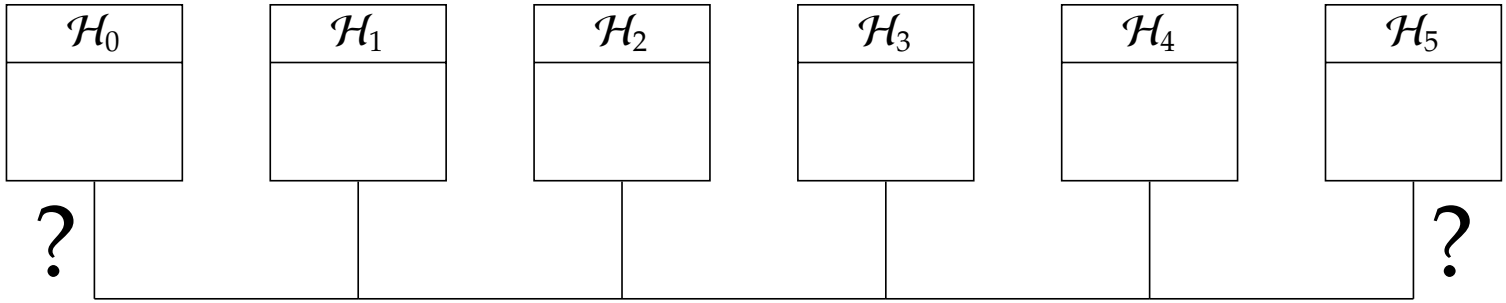
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



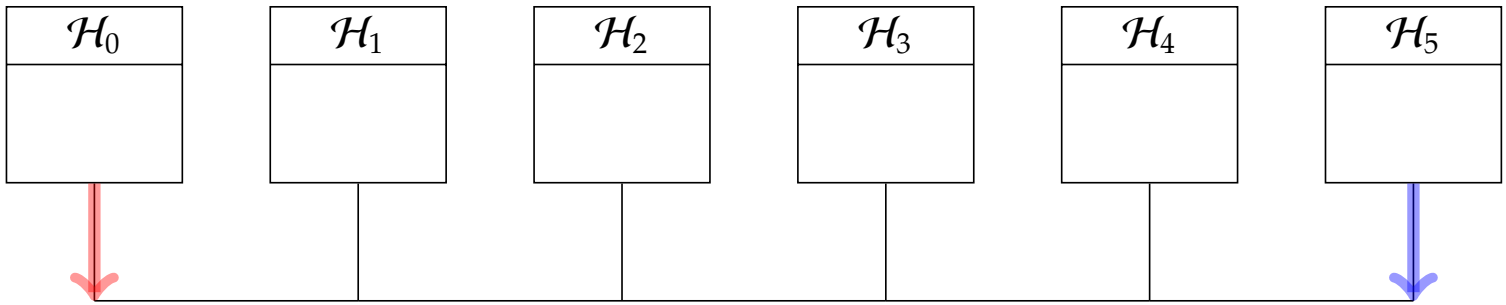
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



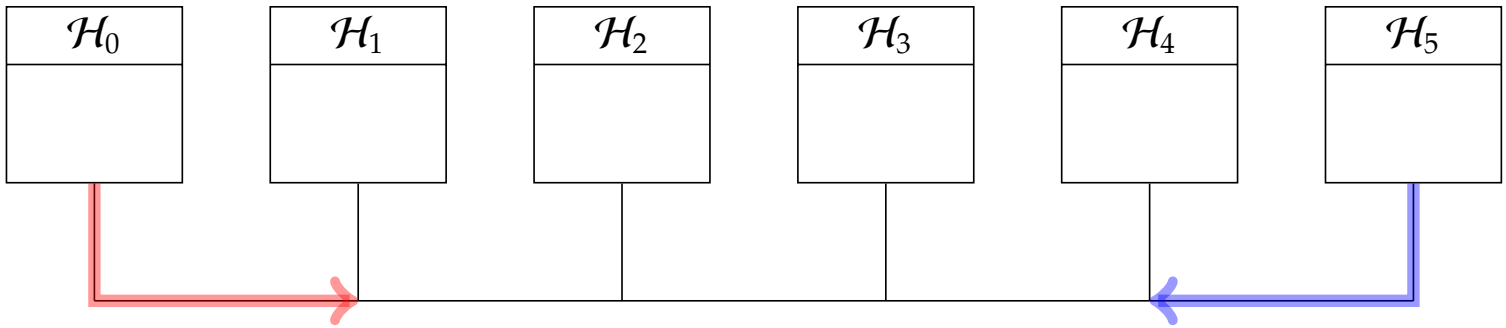
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



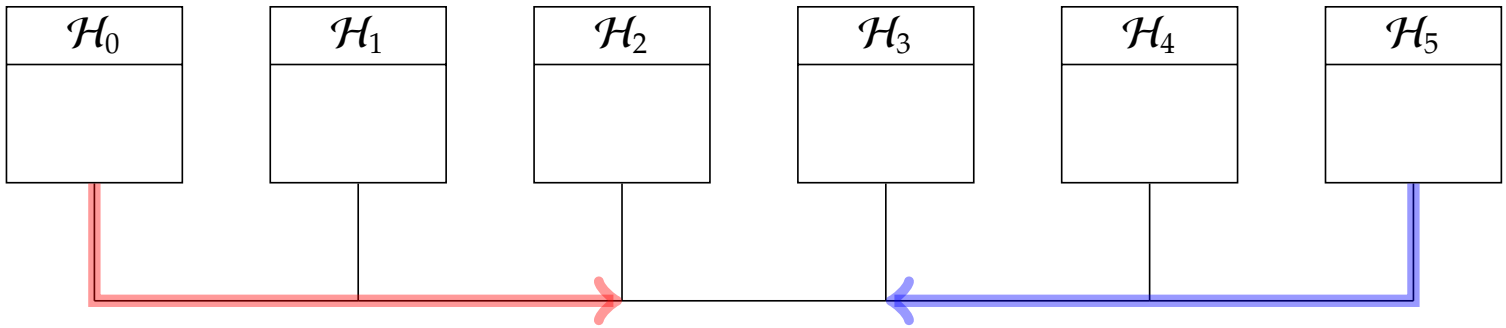
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



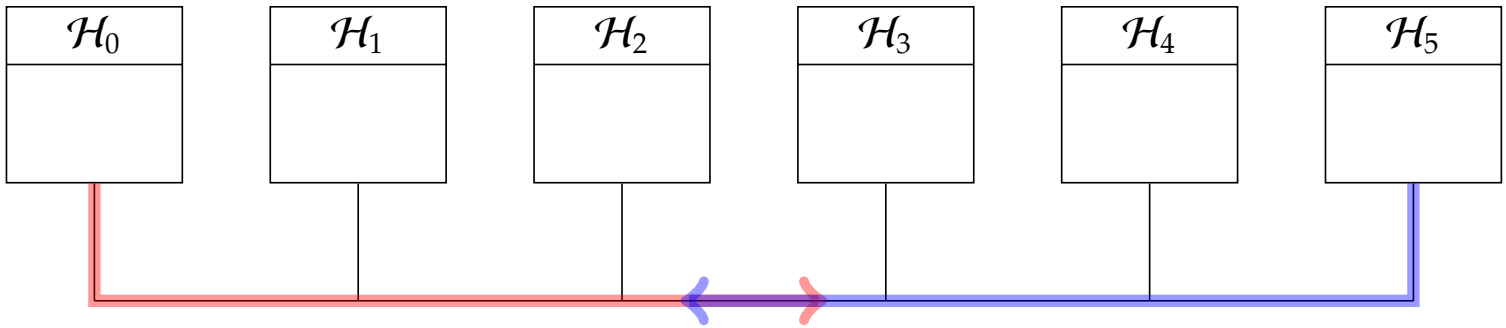
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



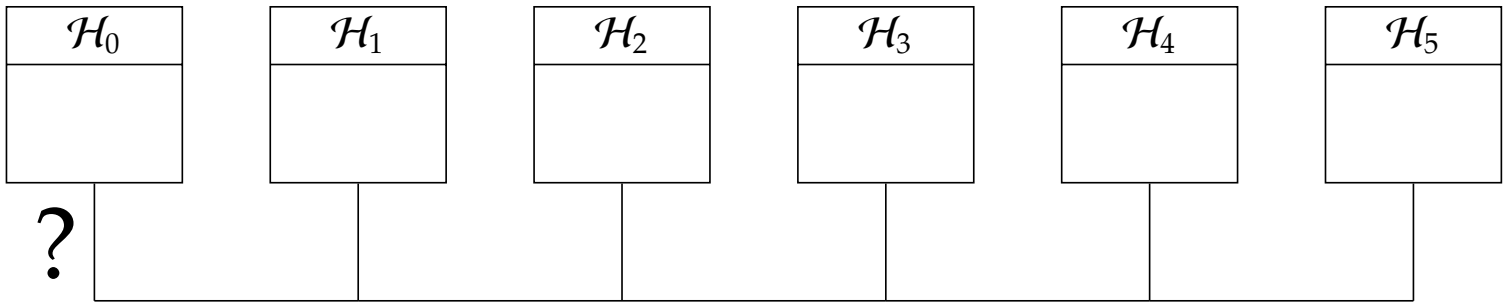
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



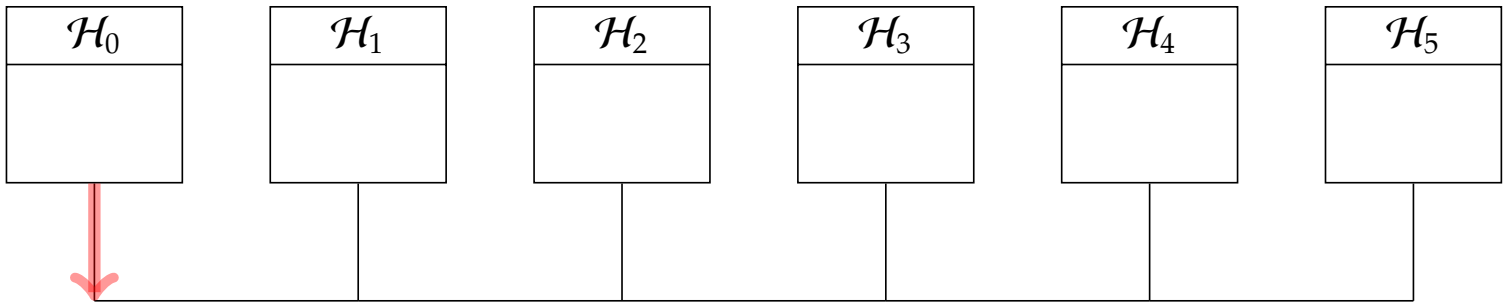
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



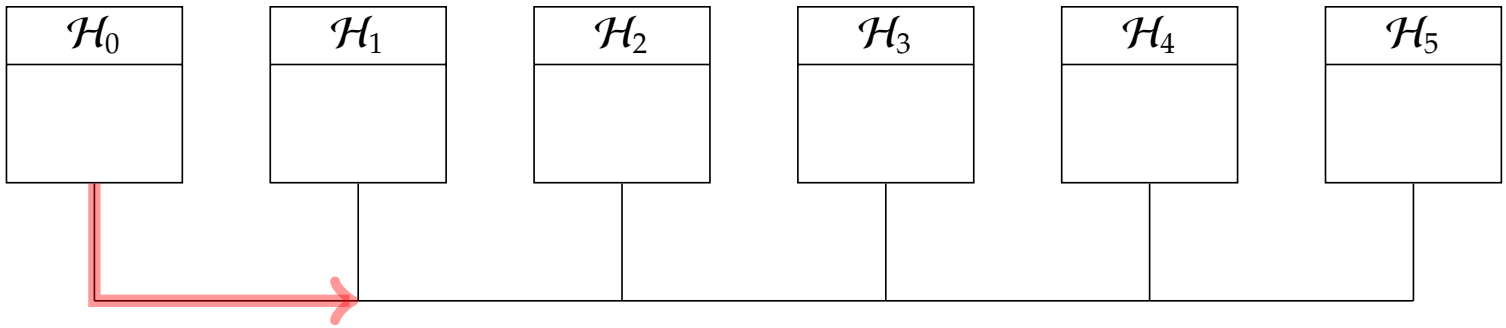
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



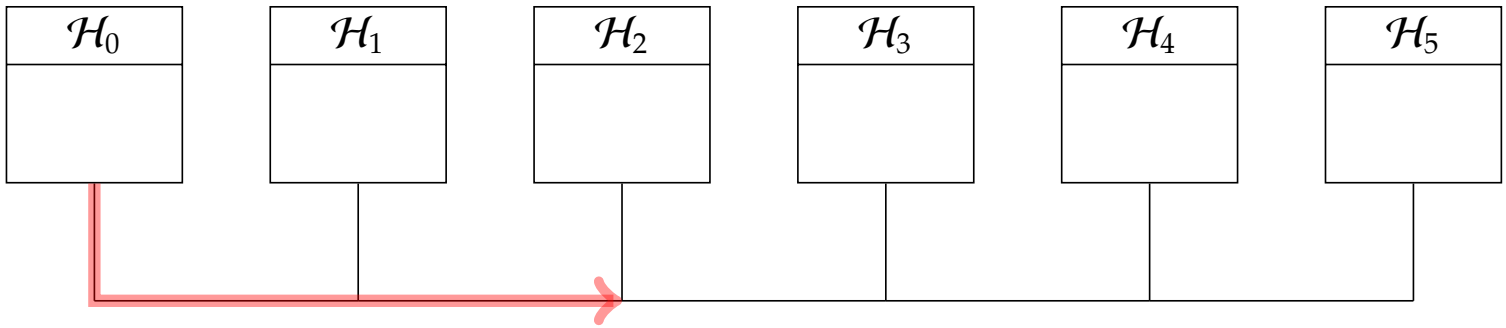
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



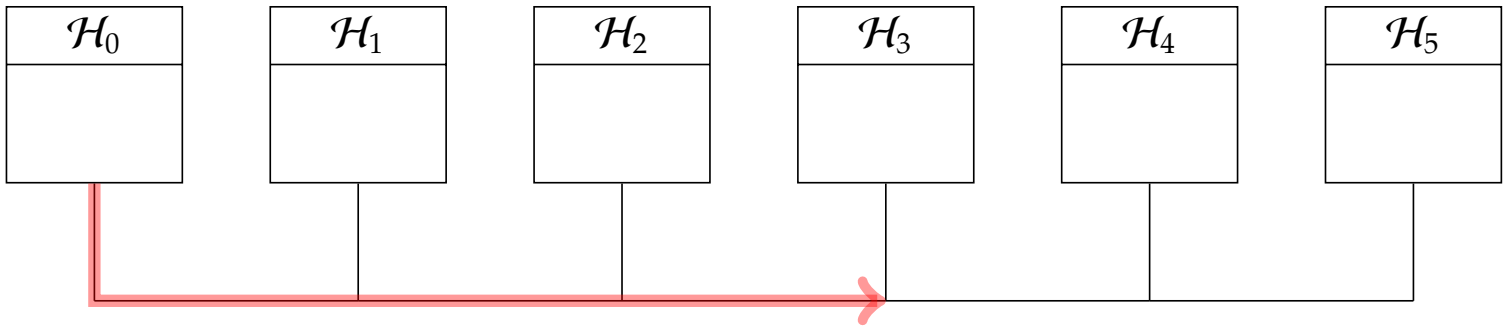
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



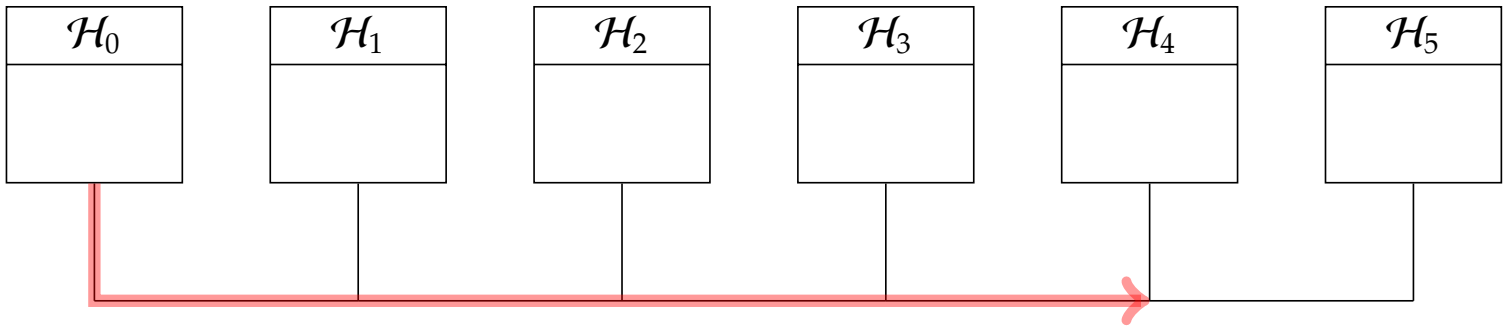
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



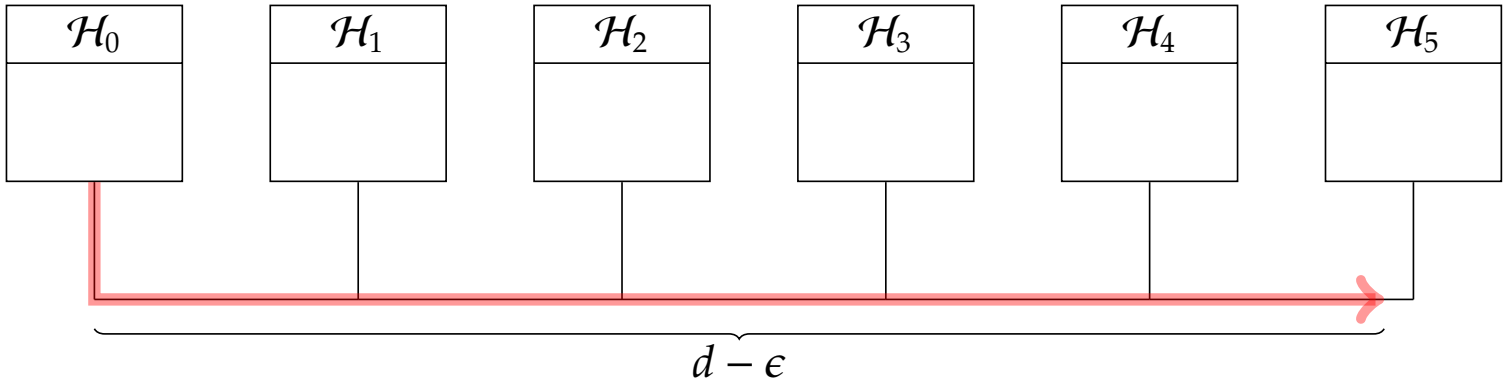
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



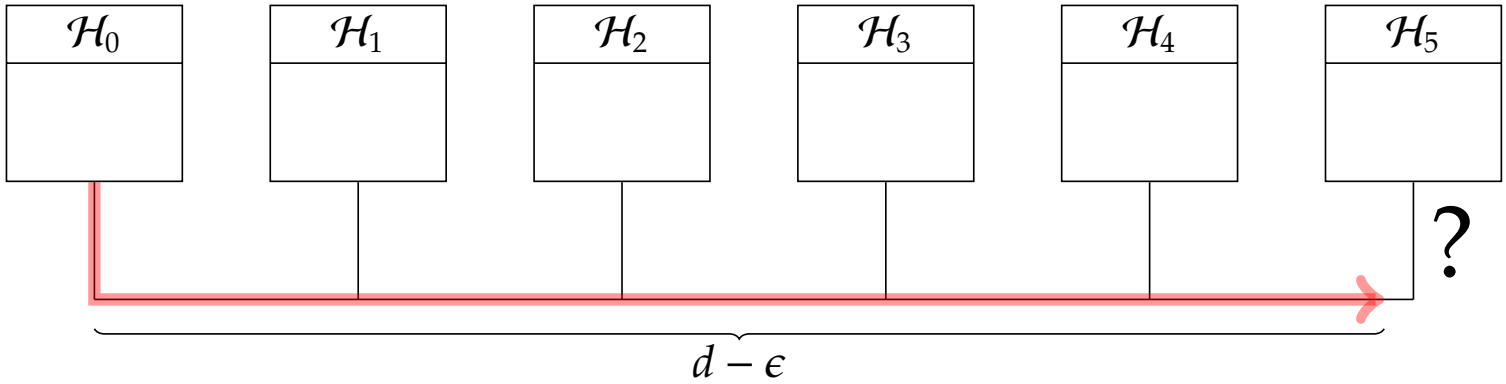
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



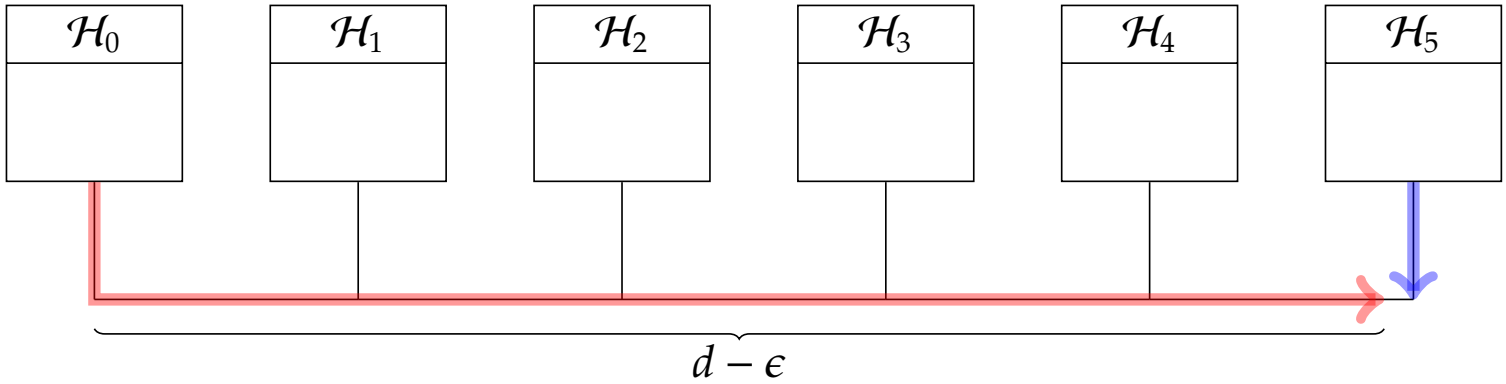
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



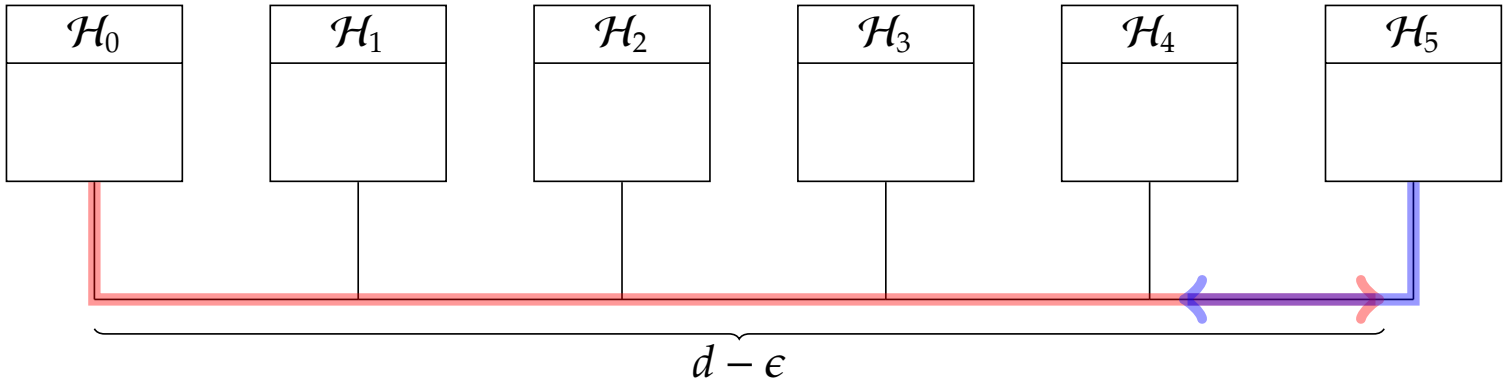
802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions:



802.3 (3) – CSMA/CD

- CSMA reduces but does not *eliminate* collisions ...
- ... and, even then, the addition of collision detection via CSMA/CD

CSMA/CD:

1. if idle, follow appropriate CSMA persistence strategy,
2. if busy, follow appropriate CSMA persistence strategy,
3. if collision,
   - abort transmission,
   - transmit **jam signal**,
   - back-off then retry.

focuses on optimisation rather than elimination:

1. all hosts are guaranteed to detect each collision, so there is no need for acknowledgements, and
2. hosts can stop transmitting as soon as a collision is detected, so the medium is idle again sooner.

802.3 (4) – BEB

- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.

802.3 (4) – BEB

- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.
- Eh?! Basically,

$$\begin{aligned} 0 \text{ collisions} &\rightsquigarrow \text{guess } n = 1 \Rightarrow \text{transmit immediately} \\ &\Rightarrow \text{transmission probability is } \tfrac{1}{1} \end{aligned}$$

then give up after ~ 16 attempts.

802.3 (4) – BEB

- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.
- Eh?! Basically,

$$1 \text{ collisions} \quad \rightsquigarrow \quad \text{guess } n = 2 \quad \Rightarrow \quad \text{select } r \xleftarrow{\$} \{0, \ldots, 2^1 - 1\}, \text{ wait for } r \cdot d \text{ time units}$$
$$\Rightarrow \quad \text{transmission probability is } \frac{1}{2}$$

then give up after ~ 16 attempts.

802.3 (4) – BEB

- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.
- Eh?! Basically,

$$\text{2 collisions} \quad \rightsquigarrow \quad \text{guess } n = 4 \quad \Rightarrow \quad \text{select } r \xleftarrow{\$} \{0, \ldots, 2^2 - 1\}, \text{ wait for } r \cdot d \text{ time units}$$

$$\Rightarrow \quad \text{transmission probability is } \frac{1}{4}$$

then give up after ~ 16 attempts.

802.3 (4) – BEB

- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.
- Eh?! Basically,

$$\text{3 collisions} \quad \rightsquigarrow \quad \text{guess } n = 8 \quad \Rightarrow \quad \text{select } r \xleftarrow{\$} \{0, \ldots, 2^3 - 1\}, \text{ wait for } r \cdot d \text{ time units}$$

$$\Rightarrow \quad \text{transmission probability is } \frac{1}{8}$$

then give up after ~ 16 attempts.

802.3 (4) – BEB

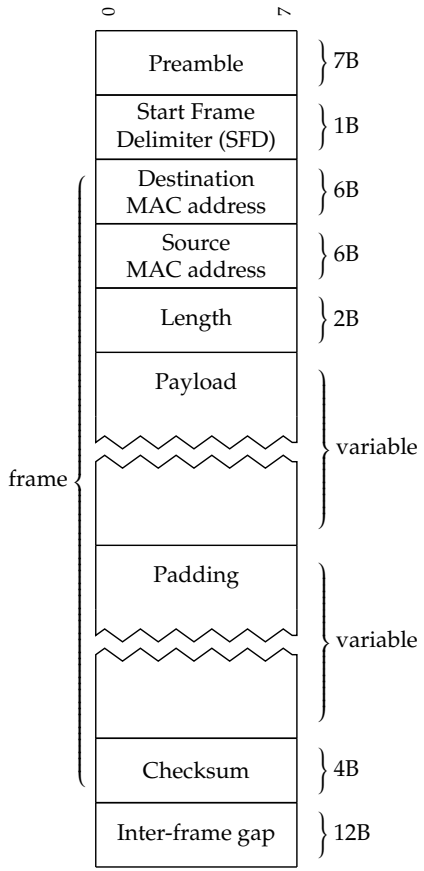
- Problem:
  - n hosts share access to the medium, and we ideally want 1 to transmit,
  - if each host transmits with probability $\frac{1}{n}$, then on average this satisfies the requirement ...
  - ... but we don't necessarily know what n is!
- Solution: BEB
  - estimate n, assuming more collisions implies larger n,
  - cater for low- and high-loads by starting optimistically, but allowing transmission probability to shrink quickly

  by (randomly) waiting upto $2^m \cdot d$ time units after m collisions.
- Eh?! Basically,

$$m \text{ collisions} \rightsquigarrow \text{guess } n = 2^m \Rightarrow \text{select } r \xleftarrow{\$} \{0, \ldots, 2^m - 1\}, \text{ wait for } r \cdot d \text{ time units}$$
$$\Rightarrow \text{transmission probability is } \frac{1}{2^m} \simeq \frac{1}{n}$$

then give up after ~ 16 attempts.

Data Structure (**802.3 frame**)

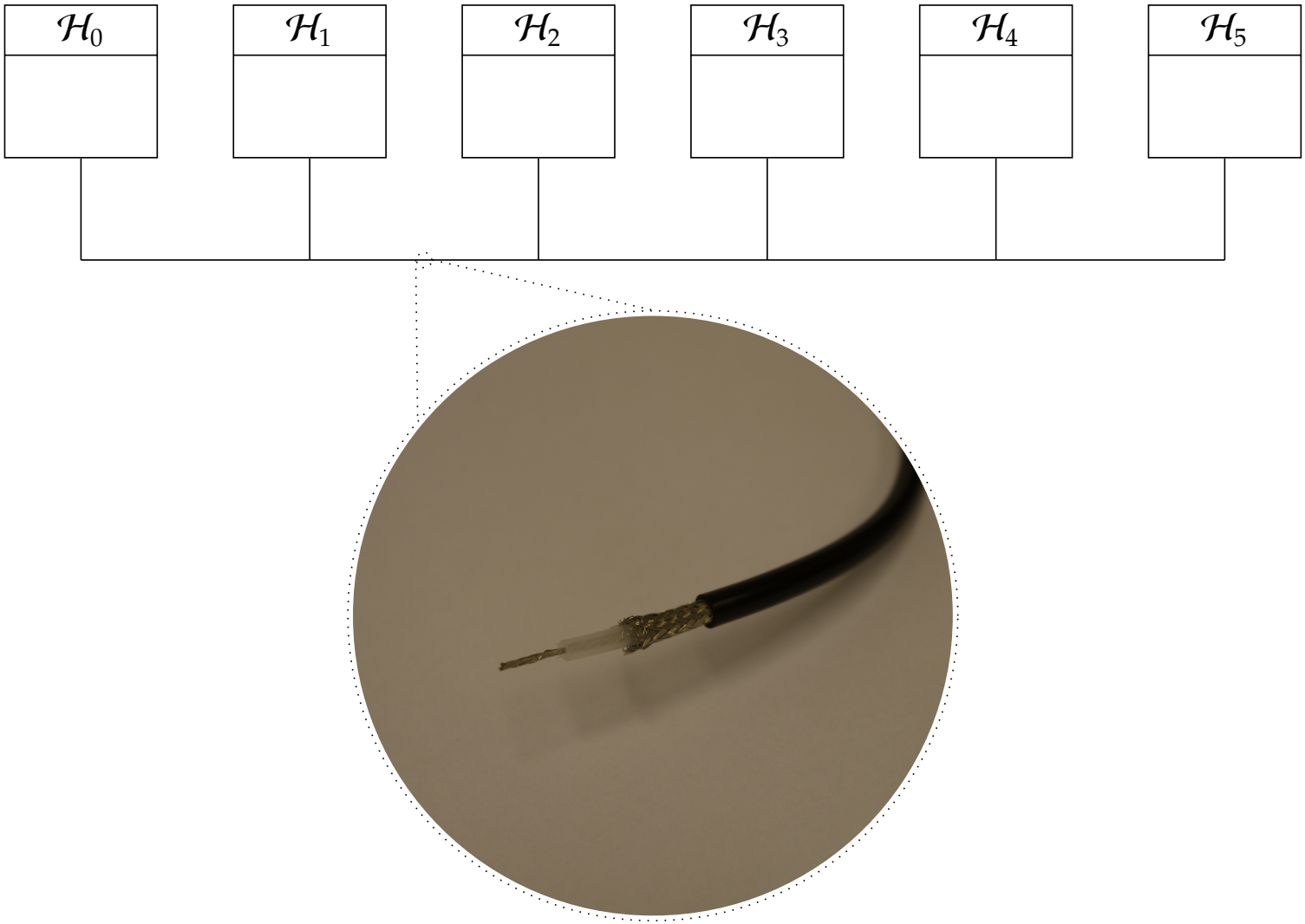


The data structure includes:

- A destination MAC address.
- A source MAC address.
- A payload length (measured in 8-bit octets).
- The payload.
- Any padding required to ensure the frame satisfies the minimum length required.
- A 32-bit checksum (on the entire frame) used to detect errors.

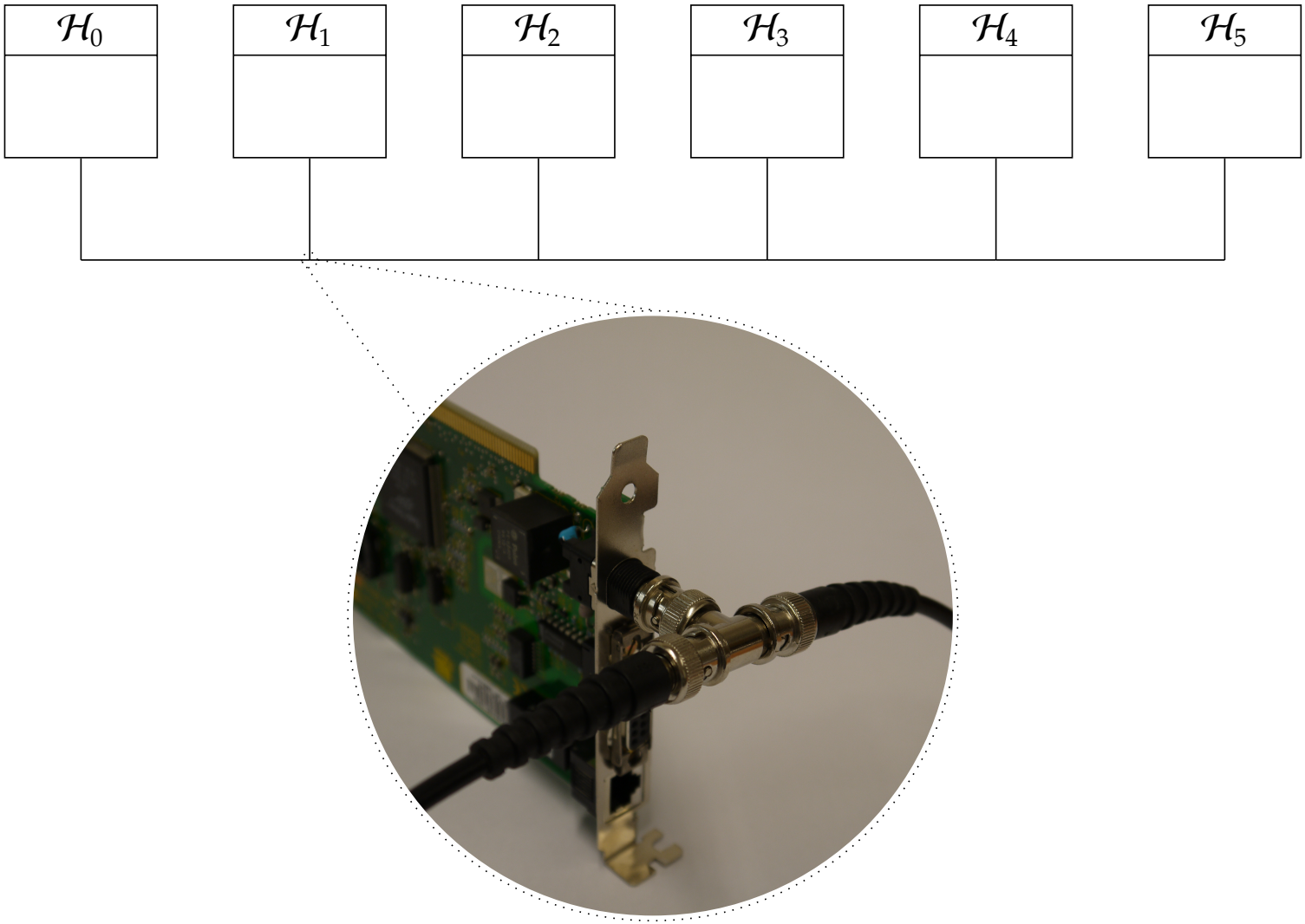
802.3 (6) – “classic” Ethernet

- Basically, “classic” Ethernet = 1-persistent CSMA/CD with BEB:



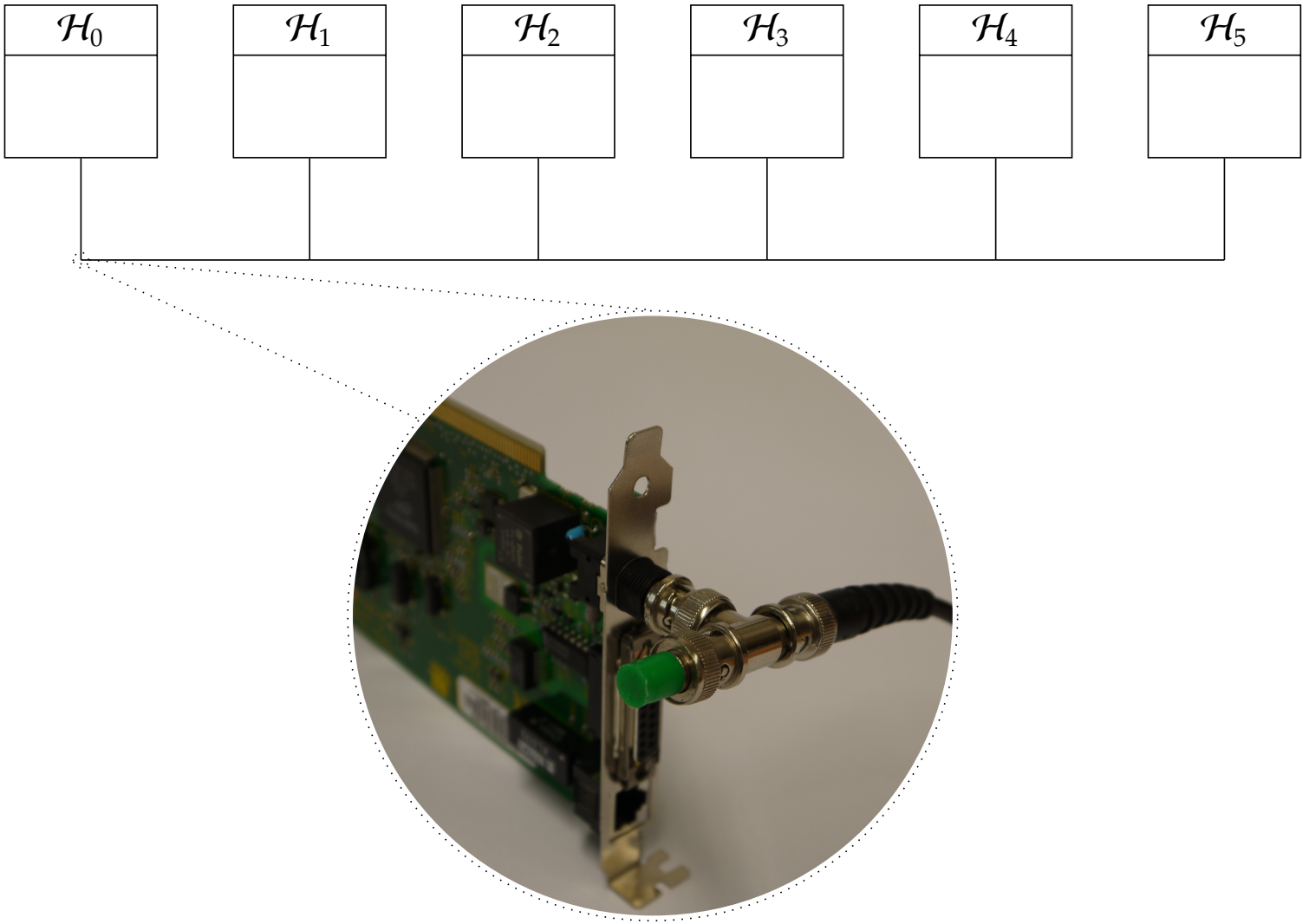
802.3 (6) – “classic” Ethernet

- Basically, “classic” Ethernet = 1-persistent CSMA/CD with BEB:



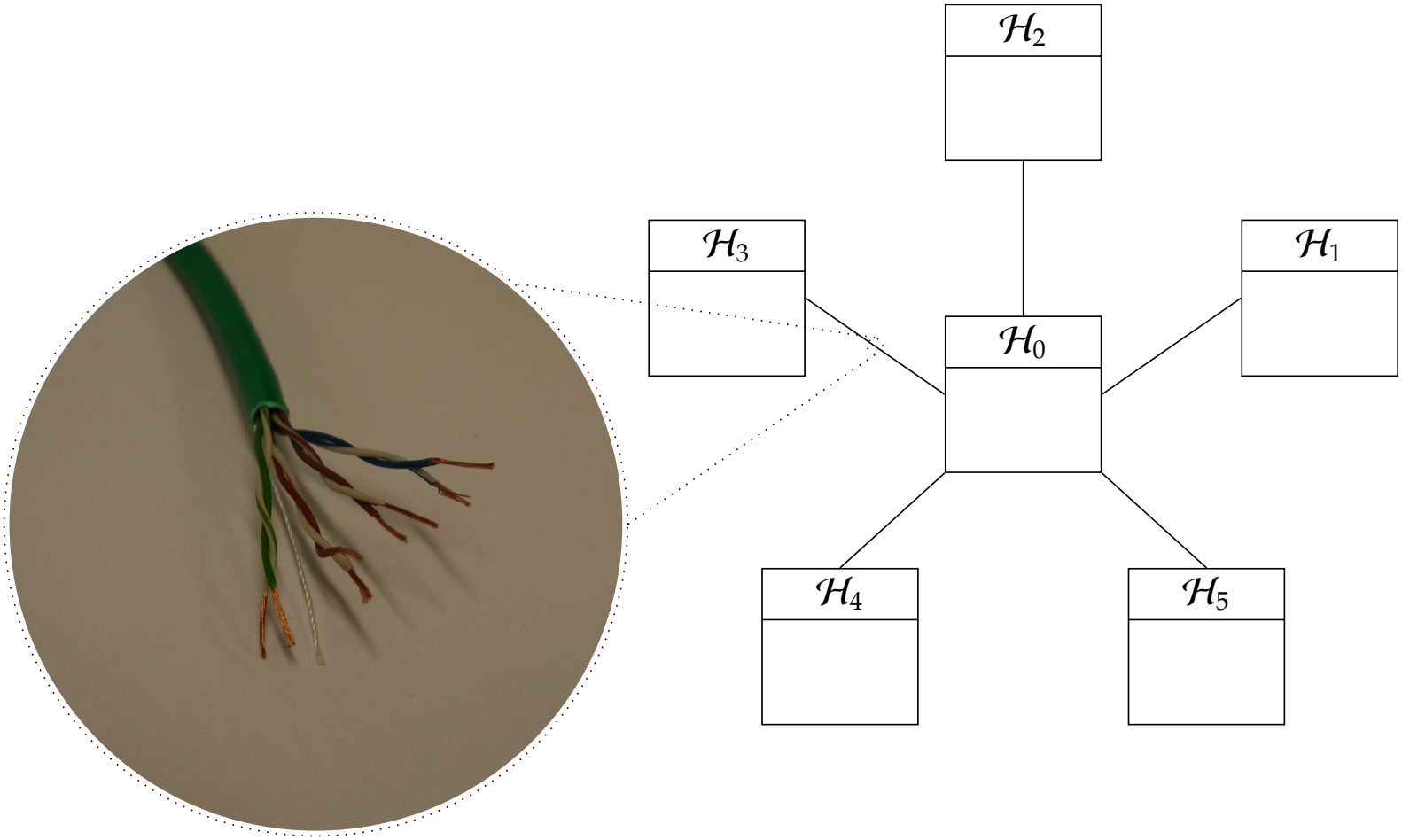
802.3 (6) – “classic” Ethernet

- Basically, “classic” Ethernet = 1-persistent CSMA/CD with BEB:



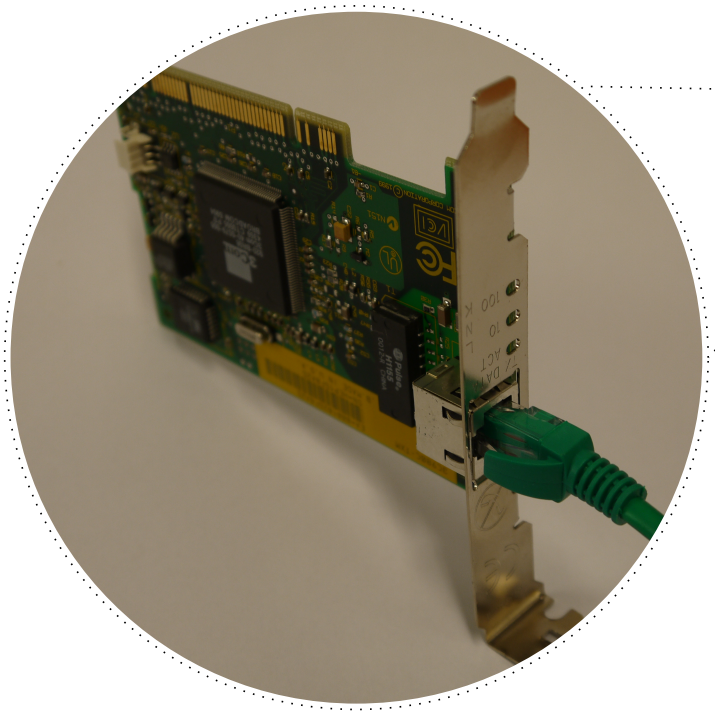
802.3 (7) – “modern” Ethernet

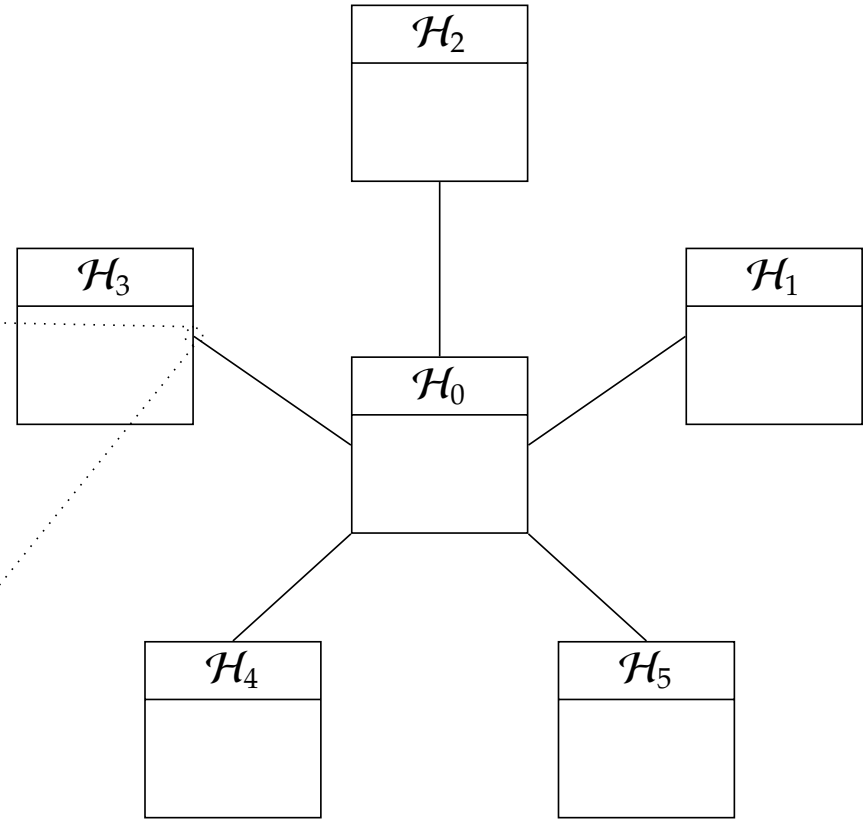
- Use of “modern” Ethernet differs significantly:



802.3 (7) – “modern” Ethernet

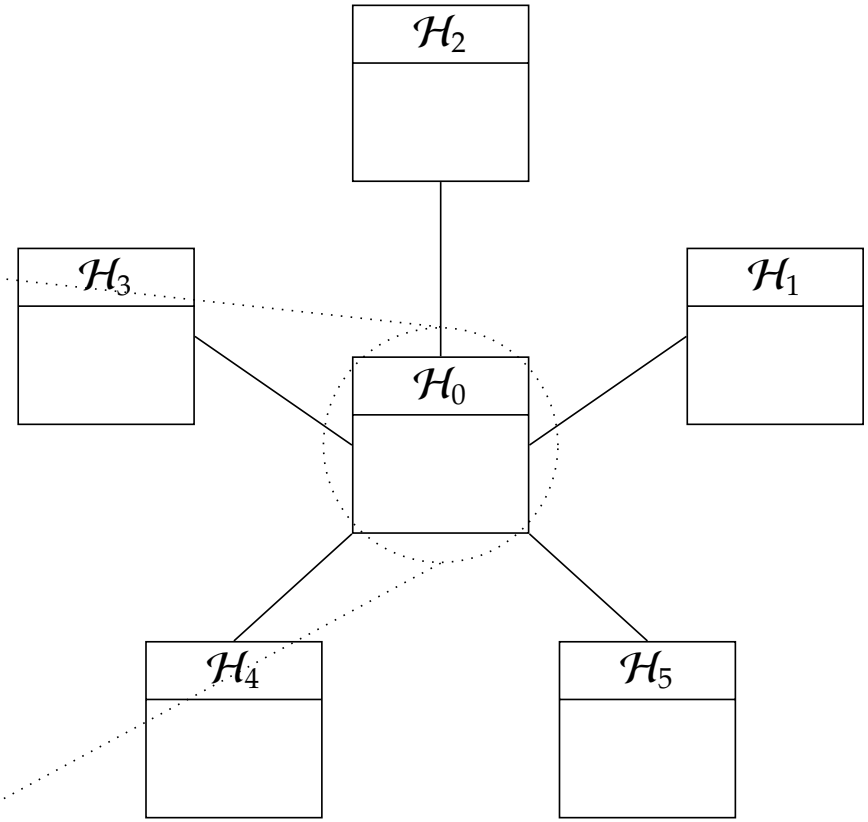
- Use of “modern” Ethernet differs significantly:





802.3 (7) – “modern” Ethernet

- Use of “modern” Ethernet differs significantly:



Conclusions

- Take away point: we now have a **Local Area Network (LAN)**.
  - topological complexity is reduced using a shared medium,
  - this limits locality of connections (i.e., *local* area), and
  - we need a protocol to control access to the (shared) medium, but
  - once a host has access, the medium looks like a point-to-point connection.

Additional Reading

- *Wikipedia: Link layer*. URL: http://en.wikipedia.org/wiki/Link_layer.
- W. Stallings. "Chapter 16: Local area network overview". In: *Data and Computer Communications*. 9th ed. Pearson, 2010.
- W. Stallings. "Chapter 17: Ethernet". In: *Data and Computer Communications*. 9th ed. Pearson, 2010.
- R. Rom and M. Sidi. *Multiple Access Protocols: Performance and Analysis*. Springer, 2011.

References

[1] *Wikipedia: Link layer*. URL: http://en.wikipedia.org/wiki/Link_layer (see p. 54).

[2] R. Rom and M. Sidi. *Multiple Access Protocols: Performance and Analysis*. Springer, 2011 (see p. 54).

[3] W. Stallings. "Chapter 16: Local area network overview". In: *Data and Computer Communications*. 9th ed. Pearson, 2010 (see p. 54).

[4] W. Stallings. "Chapter 17: Ethernet". In: *Data and Computer Communications*. 9th ed. Pearson, 2010 (see p. 54).

[5] R. Braden. *Requirements for Internet hosts – Communication Layers*. Internet Engineering Task Force (IETF) Request for Comments (RFC) 1122. 1989. URL: http://tools.ietf.org/html/rfc1122 (see p. 5).

[6] N. Abramson. "The ALOHA system: Another alternative for computer communications". In: *Fall Joint Computer Conference (AFIPS)*. 1970, pp. 281–285.