COMP90007 Internet Technology

Week6

Yiran (Scott) Ruan

Email: yrrua@unimelb.edu.au

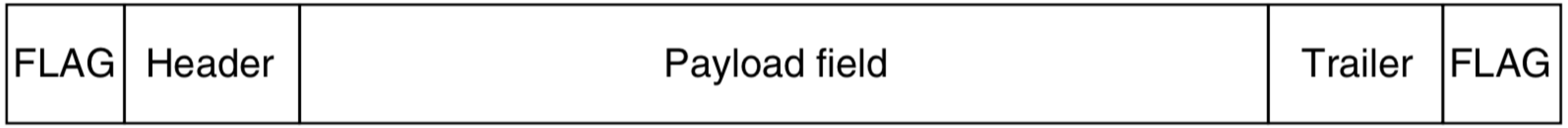
GitHub: https://yiranruan.github.io

Question 1

- A channel has a bit rate of 4 kbps and a propagation delay of 20 ms. For what range of **frame** sizes does **stop-and-wait** give an efficiency of at least 50 percent?

Frame

- A frame is a digital data transmission unit that we use in the data link layer.



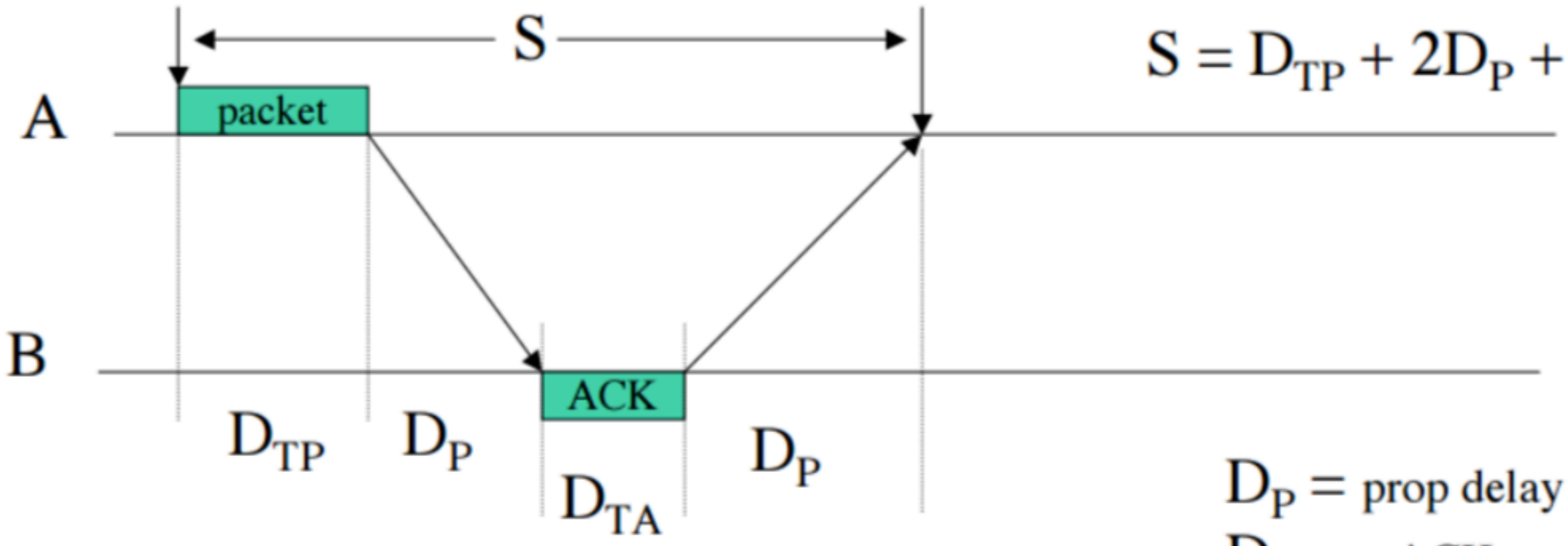
An overview of Frame

Stop-and-wait

- Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called **stop-and-wait**.

Solution 1

Efficiency (no errors) = D_{TP}/S



$$S = D_{TP} + 2D_P + D_{TA}$$

D_P = prop delay

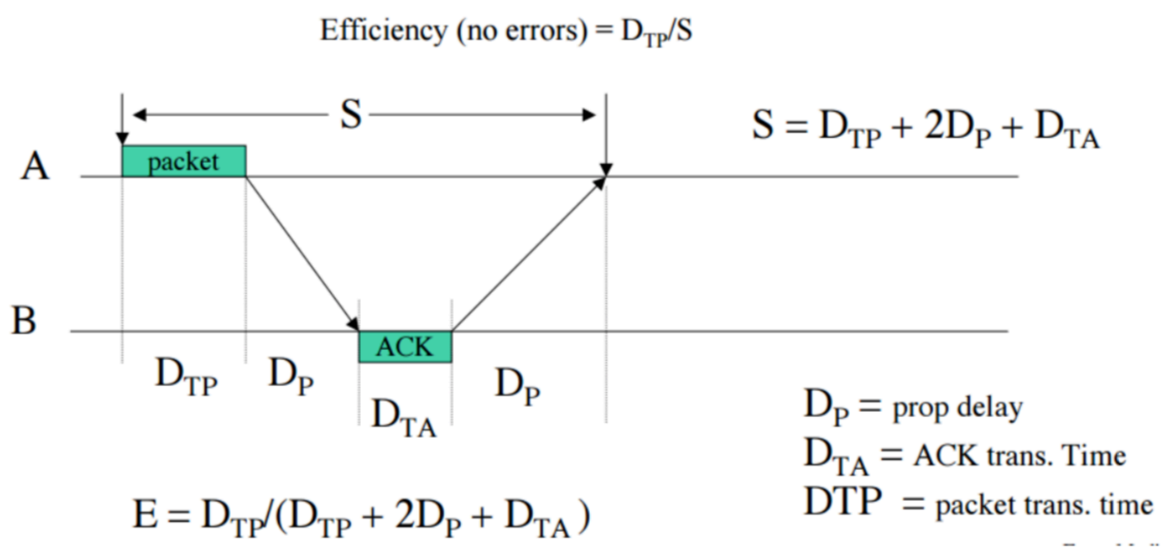
D_{TA} = ACK trans. Time

DTP = packet trans. time

$$E = D_{TP}/(D_{TP} + 2D_P + D_{TA})$$

Solution 1

- Efficiency will be 50% when the time to transmit the frame equals the round trip propagation delay.
- At a transmission rate of 4 kbps, 40 ms will transfer 160 bits. For frame sizes greater than 160 bits, stop-and-wait is reasonably efficient.



Efficiency (no errors) = D_{TP}/S
S
$S = D_{TP} + 2D_P + D_{TA}$
A
packet
B
ACK
D_{TP}
D_P
D_{TA}
D_P
$E = D_{TP}/(D_{TP} + 2D_P + D_{TA})$
D_P = prop delay
D_{TA} = ACK trans. Time
DTP = packet trans. time

Question 2

- Using the polynomial code method, compute the CRC for the frame: 1101011111 having a generator polynomial G(x) as $x^4 + x + 1$.

CRC (Cyclic Redundancy Check)

- Polynomial code
- Polynomial codes are based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only.
- A *k*-bit frame is regarded as the coefficient list for a polynomial with *k* terms, ranging from $x^{k-1}$to x^0. --> degree $k-1$
- The high-order (leftmost) bit is the coefficient of x^{k-1}, the next bit is the coefficient of x^{k-2}, and so on.

Interesting arithmetic

- Polynomial arithmetic is done **modulo 2**, according to the rules of algebraic field theory. It does not have carries for addition or borrows for subtraction.
- Both addition and subtraction are identical to **exclusive OR (XOR)**.

$$
\begin{array}{r}
10011011 \\
+\ 11001010 \\
\hline
01010001
\end{array}
\qquad
\begin{array}{r}
00110011 \\
+\ 11001101 \\
\hline
11111110
\end{array}
\qquad
\begin{array}{r}
11110000 \\
-\ 10100110 \\
\hline
01010110
\end{array}
\qquad
\begin{array}{r}
01010101 \\
-\ 10101111 \\
\hline
11111010
\end{array}
$$

G(x)

- When the polynomial code method is employed, the sender and receiver must agree upon a **generator polynomial**, $G(x)$, in advance.
- To compute the CRC for some frame with m bits corresponding to the polynomial $M(x)$, <u>*the frame must be **longer** than the generator polynomial.*</u>

Checksum

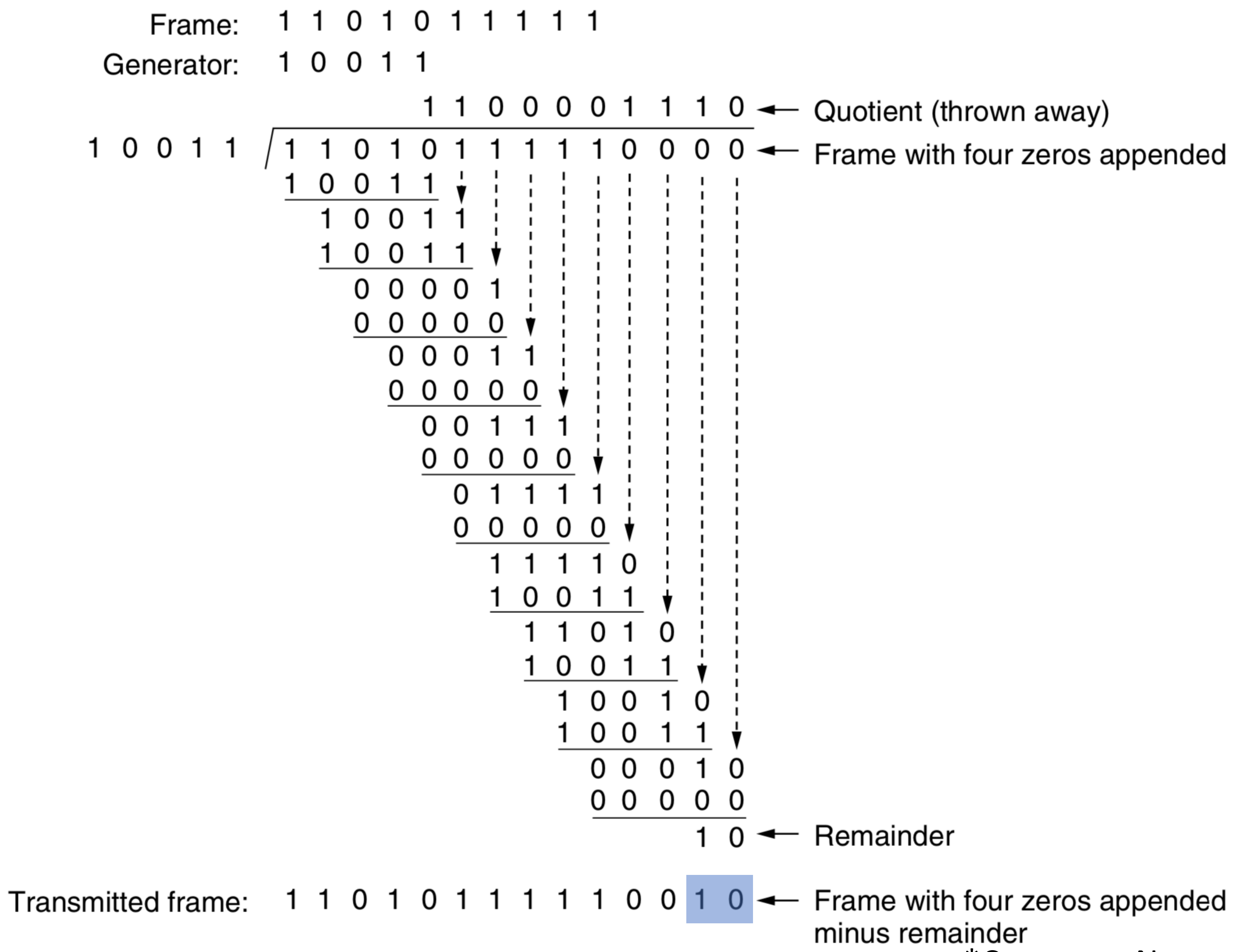
- The idea is to append a CRC to the end of the frame in such a way that the polynomial represented by the checksummed frame is *divisible by G(x).*
- When the receiver gets the checksummed frame, it tries dividing it by $G(x)$.
    - If there is a **remainder**, there has been a transmission error.

Algorithm😨

- 1. Let r be the degree of $G(x)$. Append r zero bits to the low-order end of the frame so it now contains $m + r$ bits and corresponds to the polynomial $\mathrm{x}^r M(x)$.
- 2. Divide the bit string corresponding to $G(x)$ into the bit string corresponding to $\mathrm{x}^r M(x)$ using modulo 2 division.
- 3. Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to $\mathrm{x}^r M(x)$ using modulo 2 subtraction. The result is the checksummed frame to be transmitted. Call its polynomial $T(x)$.

Let’s do it

- $G(x) = x^4 + x + 1$
- What we know from G(X)
  - 1. degree = 4
    - `1101011111` ---> append 4 zeros in the frame ---> `1101011111`**`0000`**
  - 2. $G(x) = 1 * x^4 + 0 * x^3 + 0 * x^2 + 1 * x^1 + 1 * x^0$
    - Get each coefficient ---> Generator: **`10011`**



*Computer Network Page 213 and 214

Question 3

- Consider the delay of pure ALOHA versus slotted ALOHA at low load. Which one is less? Explain your answer.

Pure ALOHA

- Basic idea: Allow users to transmit **whenever** they have data to be sent.
- carrier sense
- In the ALOHA system, after each station has sent its frame to the central computer, this computer **rebroadcasts** the frame to all of the stations. A sending station can thus listen for the broadcast from the hub to see if its frame has gotten through. In other systems, such as wired LANs, the sender might be able to listen for collisions while transmitting.

Pure ALOHA

- If the frame was destroyed, sender just waits a **random** amount of time and sends it again.
  - The waiting time must be random or the same frames will collide over and over, in lockstep.
- Systems in which multiple users share a common channel in a way that can lead to conflicts are known as **contention** systems.

Collisions in pure ALOHA

- Whenever two frames try to occupy the channel at *the same time*, there will be a **collision** and both will be garbled.
- If the first bit of a new frame **overlap**s with just the last bit of a frame that has almost finished, both frames will be totally destroyed and both will have to be retransmitted later.

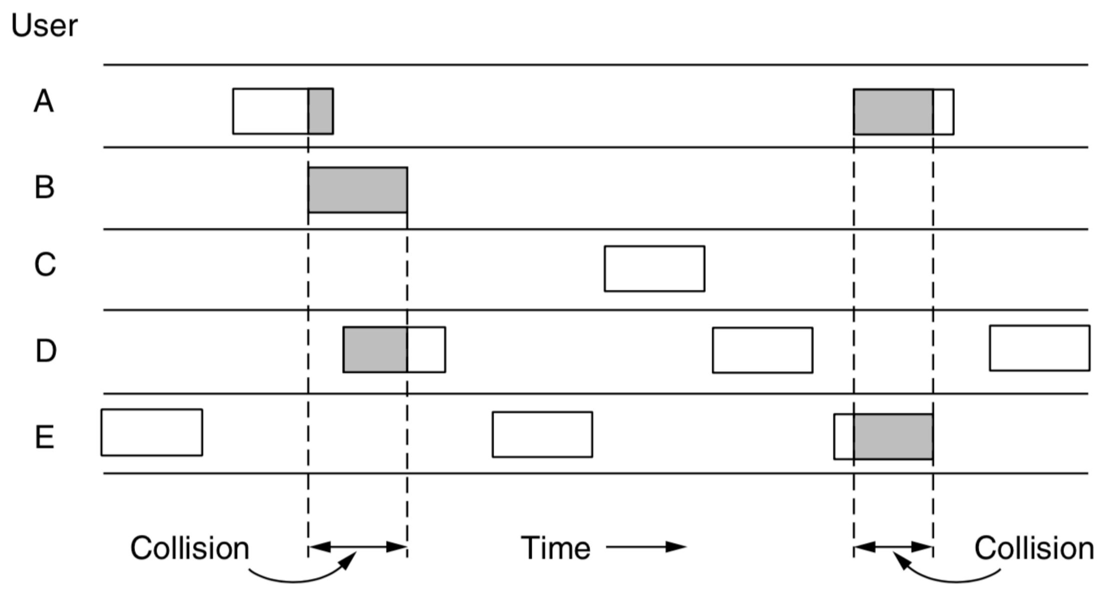


Figure 4-1. In pure ALOHA, frames are transmitted at completely arbitrary times.

Slotted ALOHA

- Double the capacity of an ALOHA system.
- Divide time into discrete intervals called **slots**, each interval corresponding to one frame.
- A station is not permitted to send whenever the user types a line. Instead, it is required to wait for the beginning of the next slot.

Efficiency comparison

- Some of you may have seen….
- Pure ALOHA: $S = Ge^{-2G}$
  - The maximum throughput occurs at *G* = 0.5, with *S* = 1/2*e*, which is about 0.184.
- Slotted ALOHA: $S = Ge^{-G}$
  - slotted ALOHA peaks at *G* = 1, with a throughput of *S* = 1/*e* or about 0.368
- What do we need to know from those result….

Efficiency comparison

- Only one sentence:

  The throughput of slotted ALOHA is twice that of pure ALOHA.

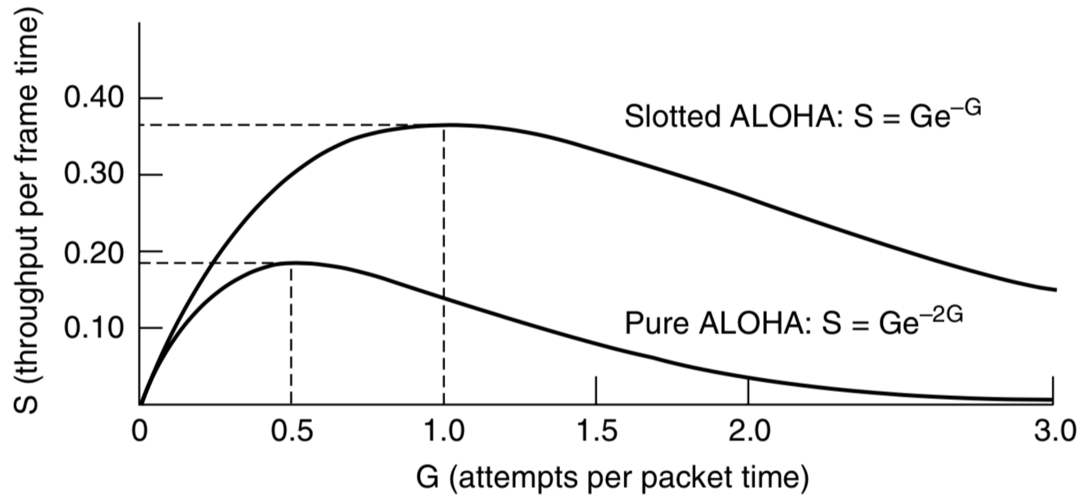


Figure 4-3. Throughput versus offered traffic for ALOHA systems.

Question 3

- Consider the delay of pure ALOHA versus slotted ALOHA at low load. Which one is less? Explain your answer.

Solution 3

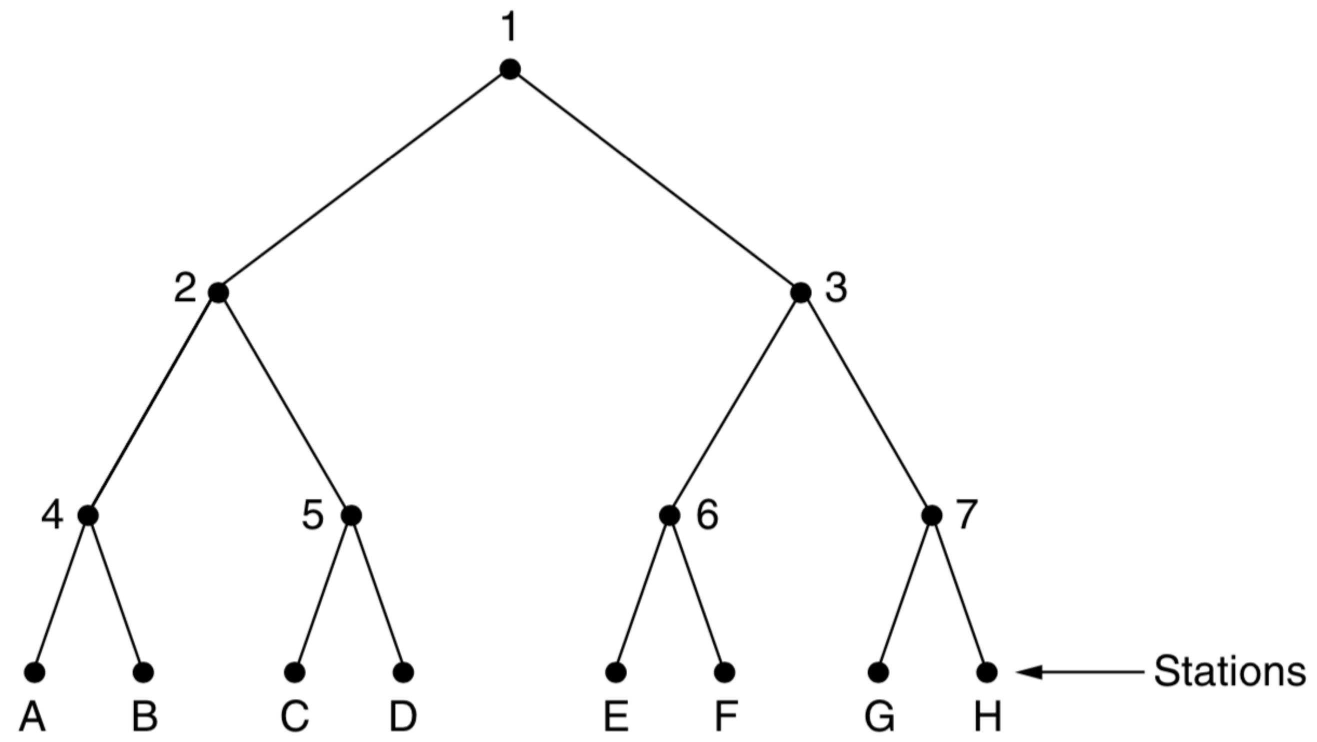
- With slotted ALOHA, it has to wait for the next slot. This introduces half a slot time of delay. With pure ALOHA, transmission can start instantly. At low load with minimal collisions, pure ALOHA will have less delay.
- However, at higher loads, there is more probability for collisions in pure ALOHA compared to slotted ALOHA. This is because frames can collide in midway. By enforcing synchronisation, slotted ALOHA is able to achieve much greater efficiency.

Question 4

- Eight stations, numbered 1 through 8, are contending for the use of a shared channel by using the adaptive tree walk protocol. If all the stations whose addresses are prime numbers suddenly became ready at once, how many slots are needed to resolve the contention?

Adaptive tree walk protocol

- think of the stations as the leaves of a binary tree

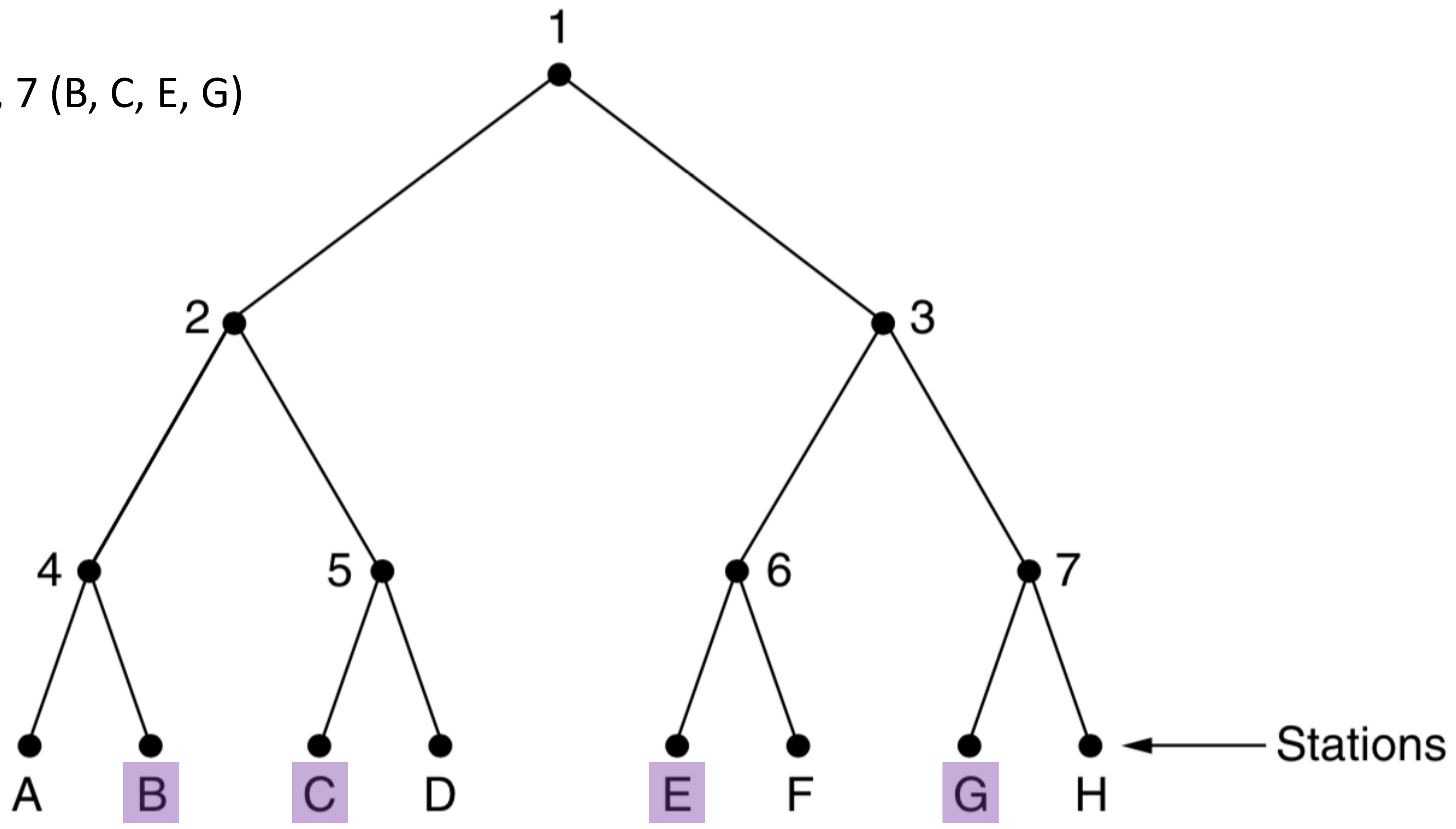


How it works

In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4’s turn during slot 2.

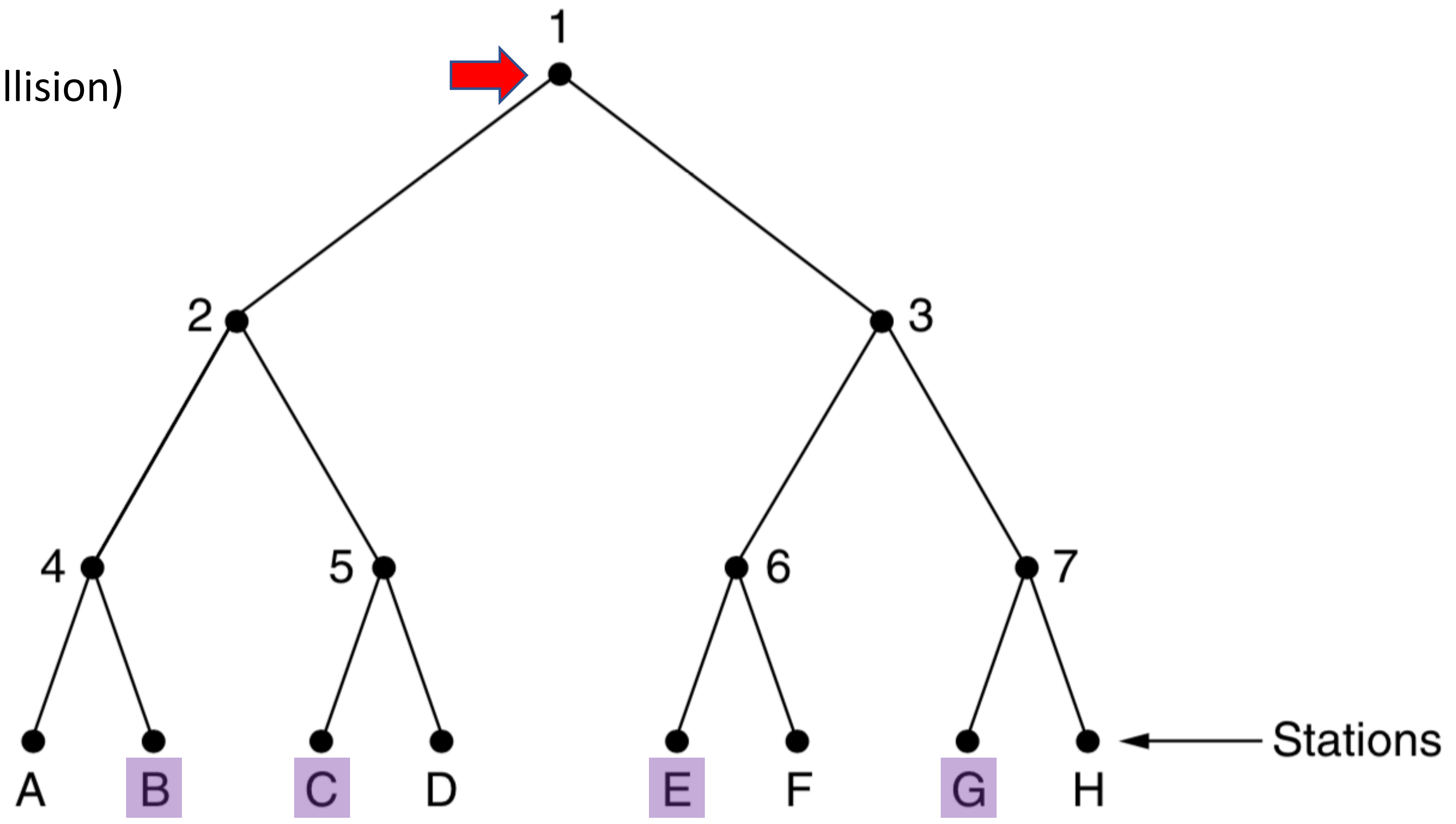
Solution 4

Prime number: 2, 3, 5, 7 (B, C, E, G)



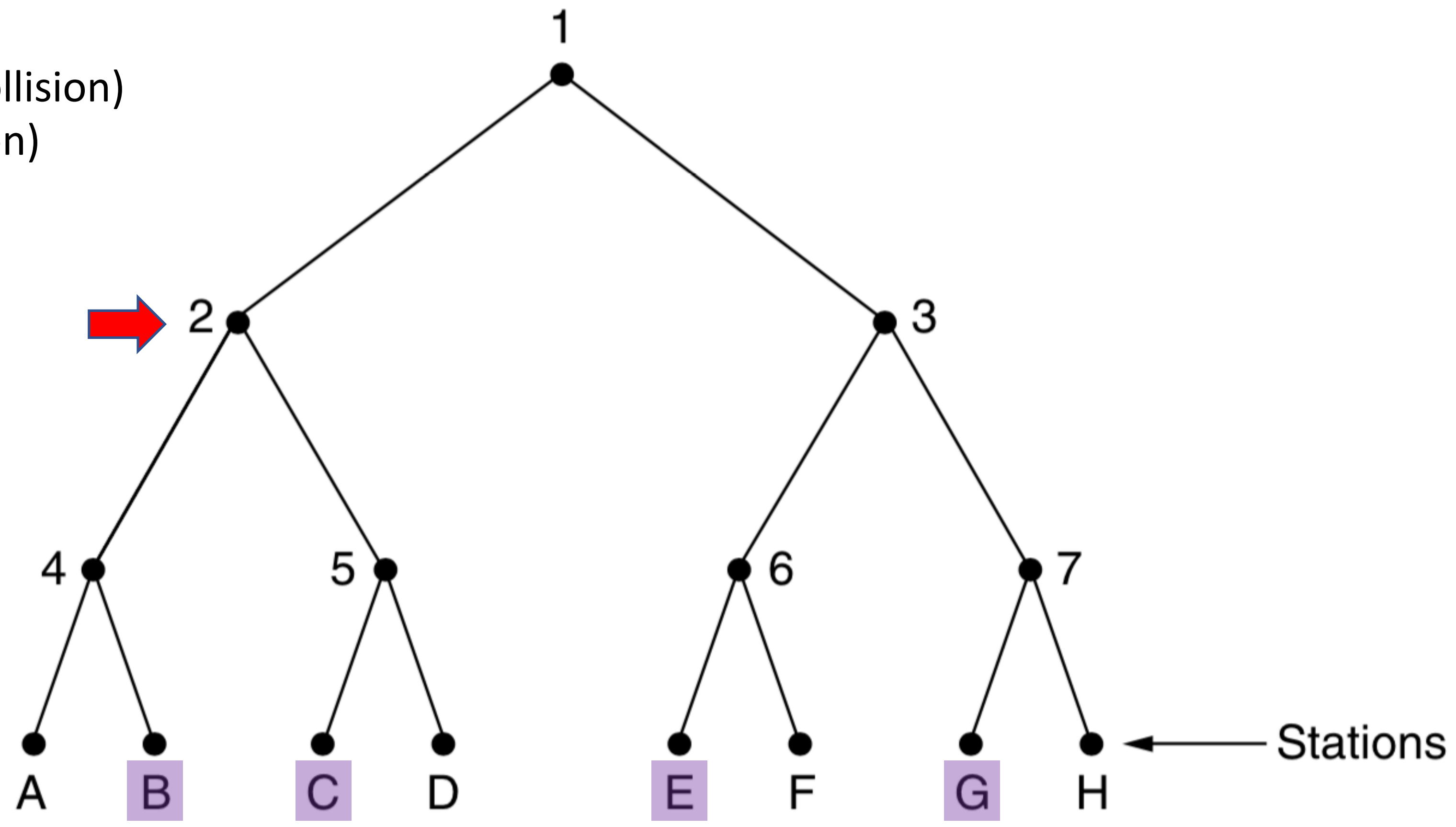
Solution 4

slot 1: B, C, E, G(collision)



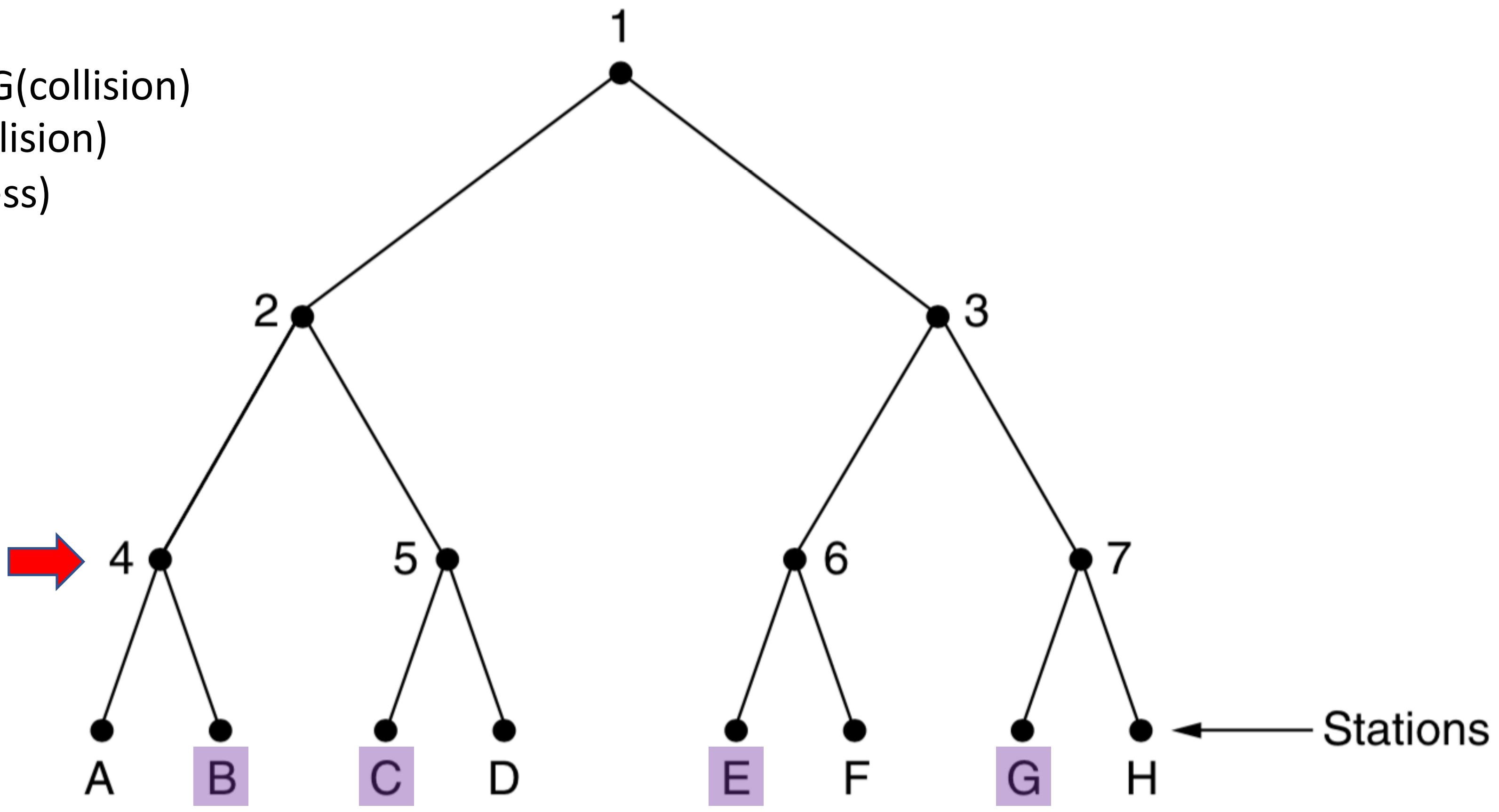
Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)



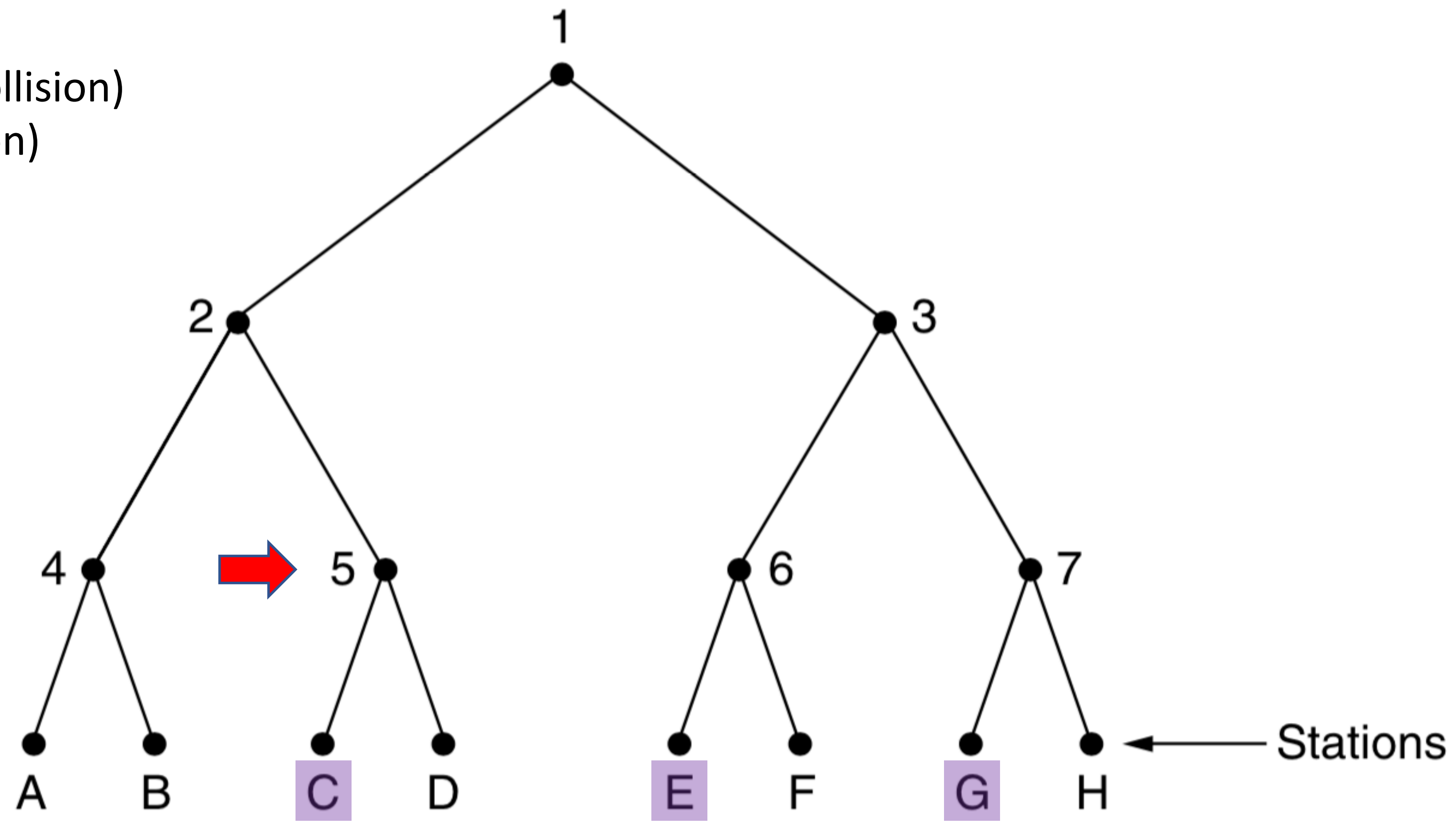
Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)



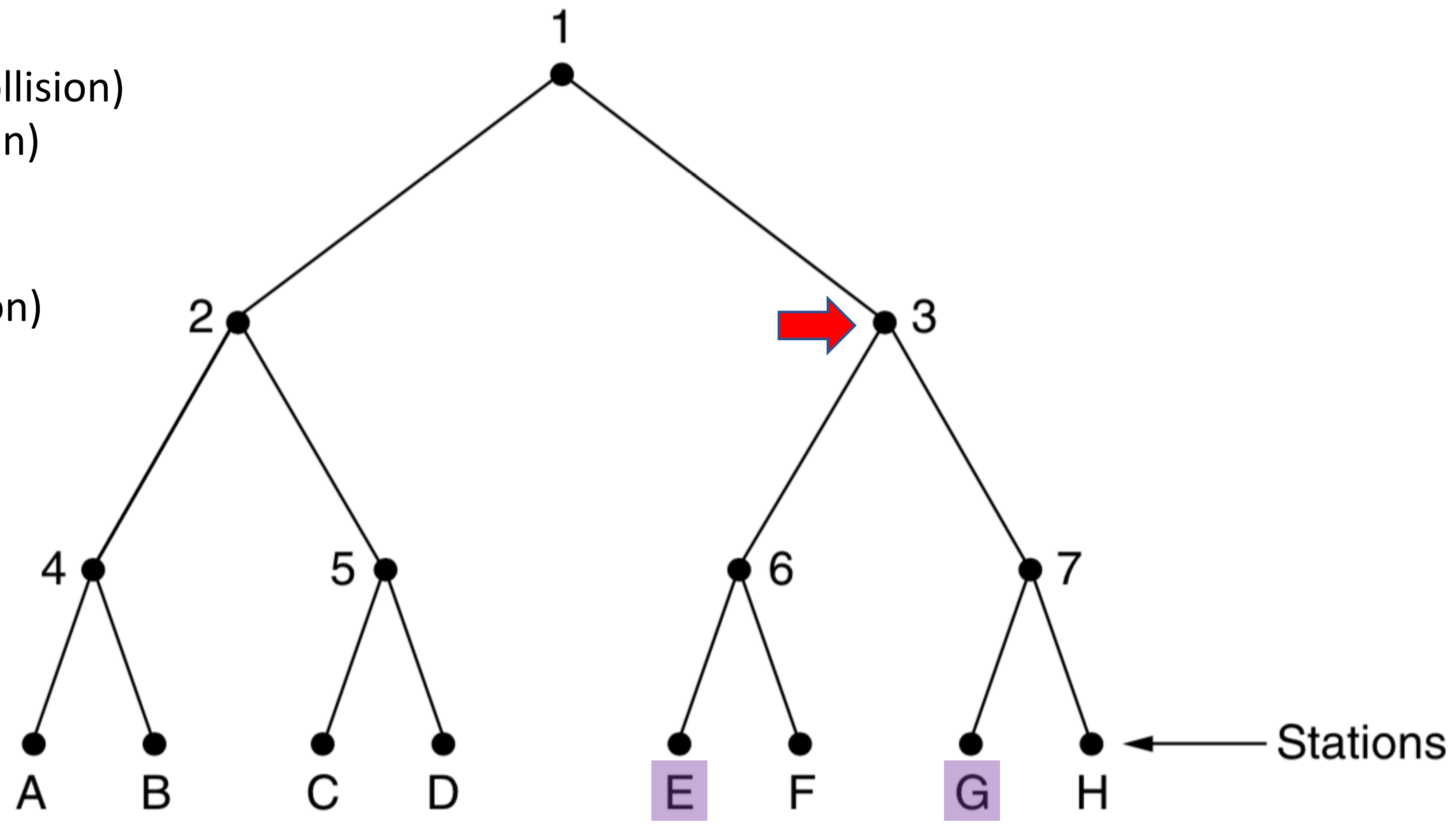
Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)



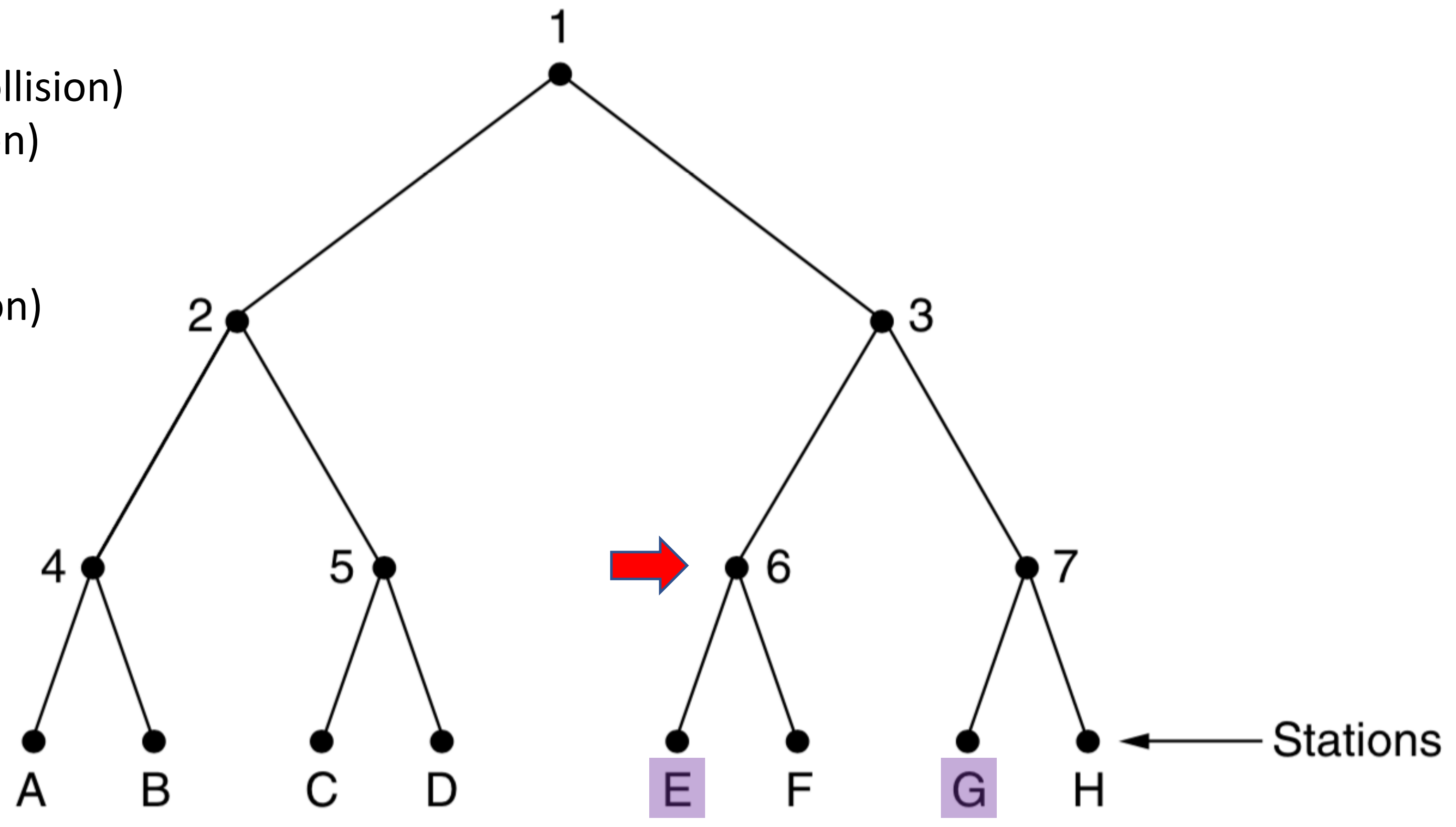
Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)
slot 5:	E, G (collision)



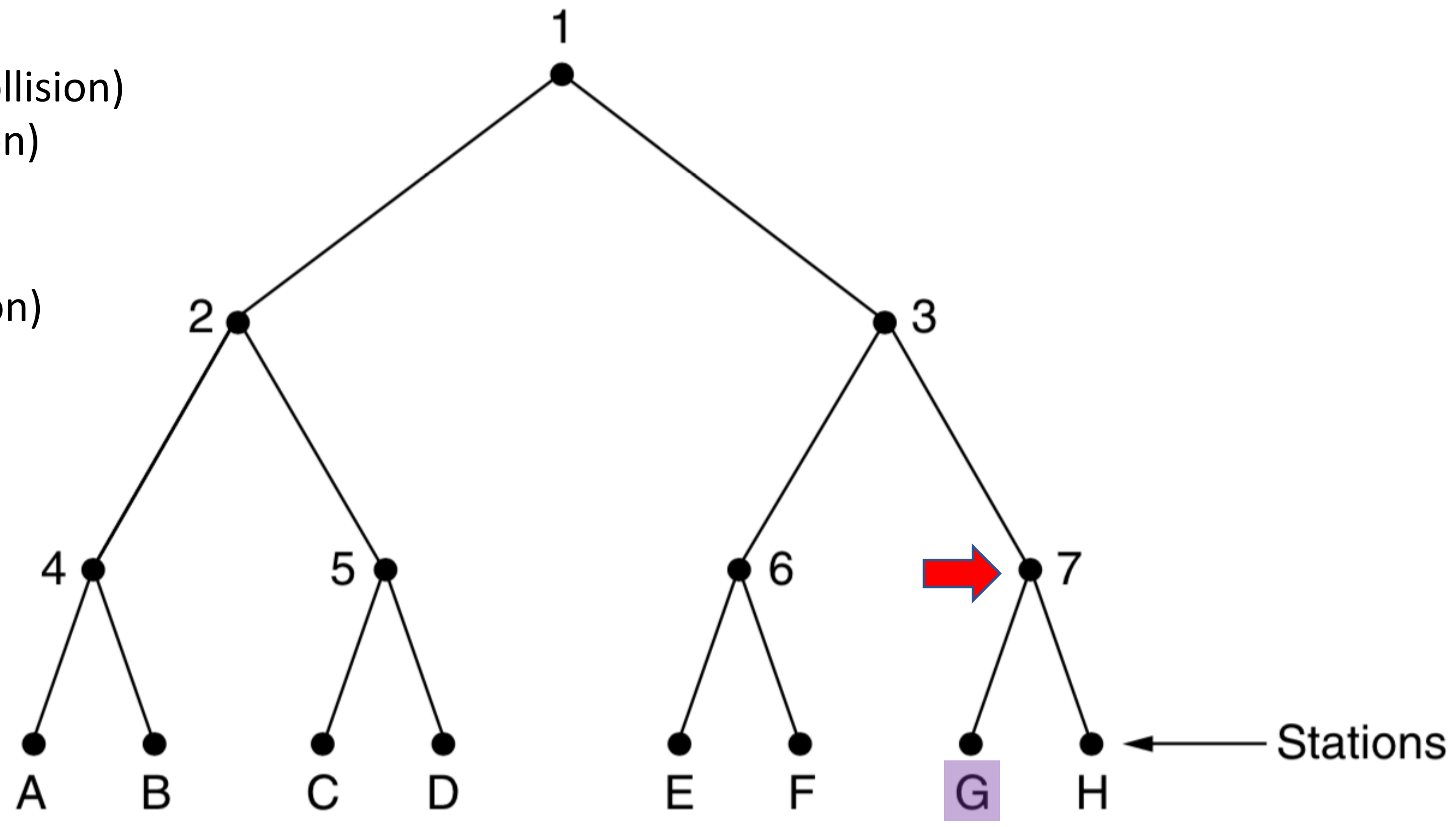
Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)
slot 5:	E, G (collision)
slot 6:	E (success)



Solution 4

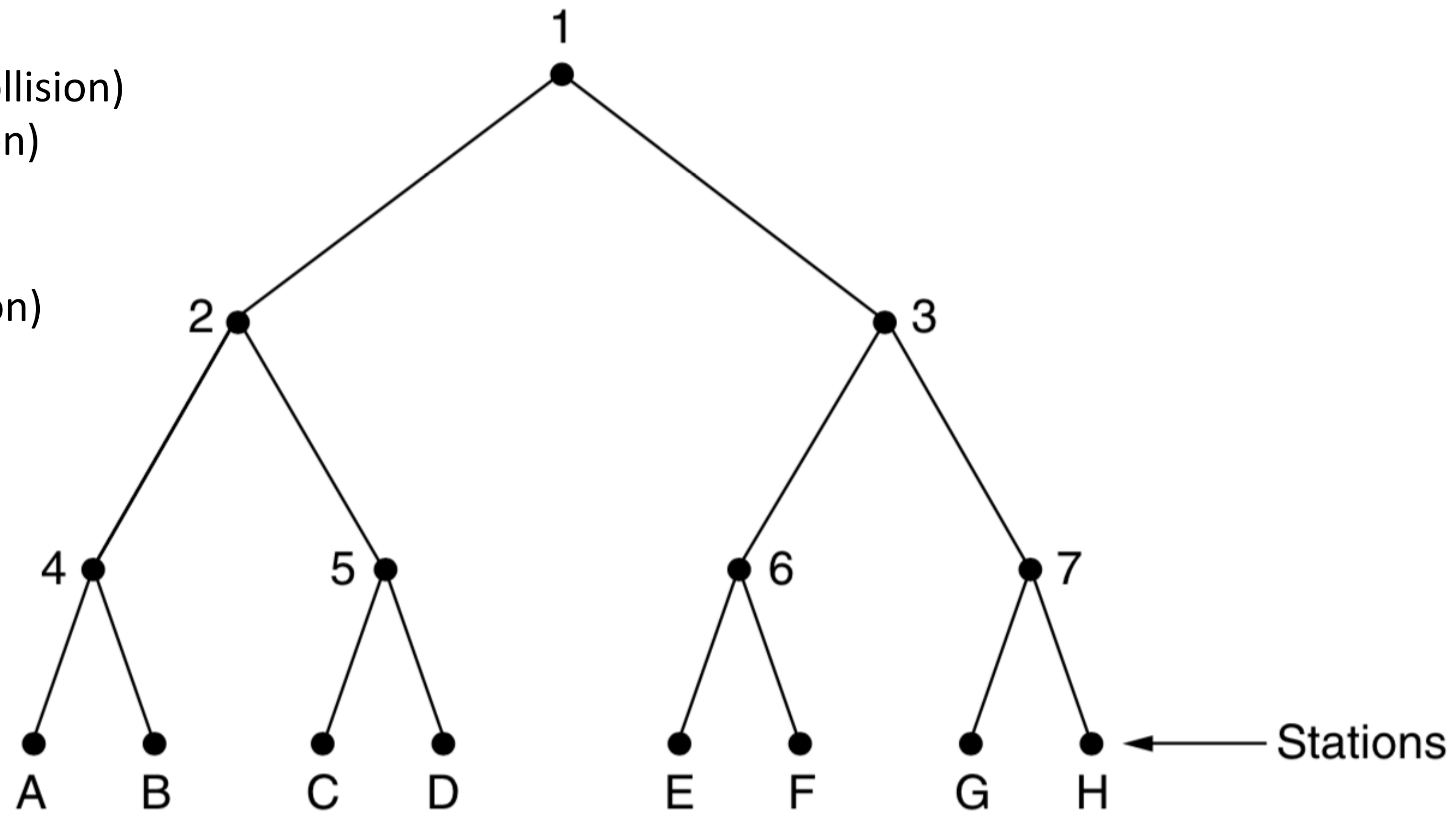
slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)
slot 5:	E, G (collision)
slot 6:	E (success)
slot 7:	G (success)



Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)
slot 5:	E, G (collision)
slot 6:	E (success)
slot 7:	G (success)

END

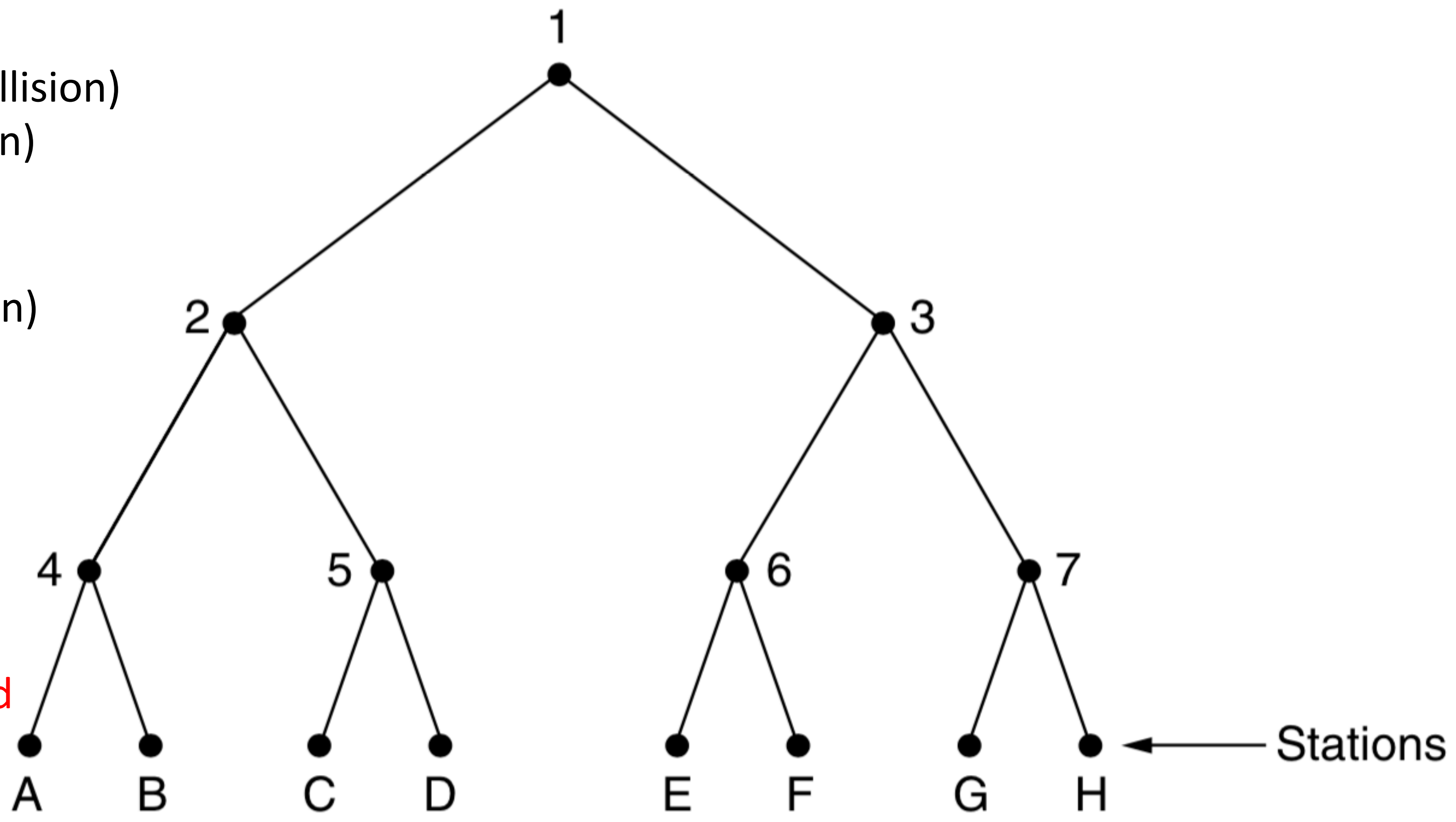


Solution 4

slot 1:	B, C, E, G(collision)
slot 2:	B, C (collision)
slot 3:	B (success)
slot 4:	C (success)
slot 5:	E, G (collision)
slot 6:	E (success)
slot 7:	G (success)

END

The heavier the load,
The farther down the
Tree the search should
be



Question 5

- Convert the IP address

```
11000001,01010010,11010010,00001111
```

- to dotted decimal notation.

IPv4

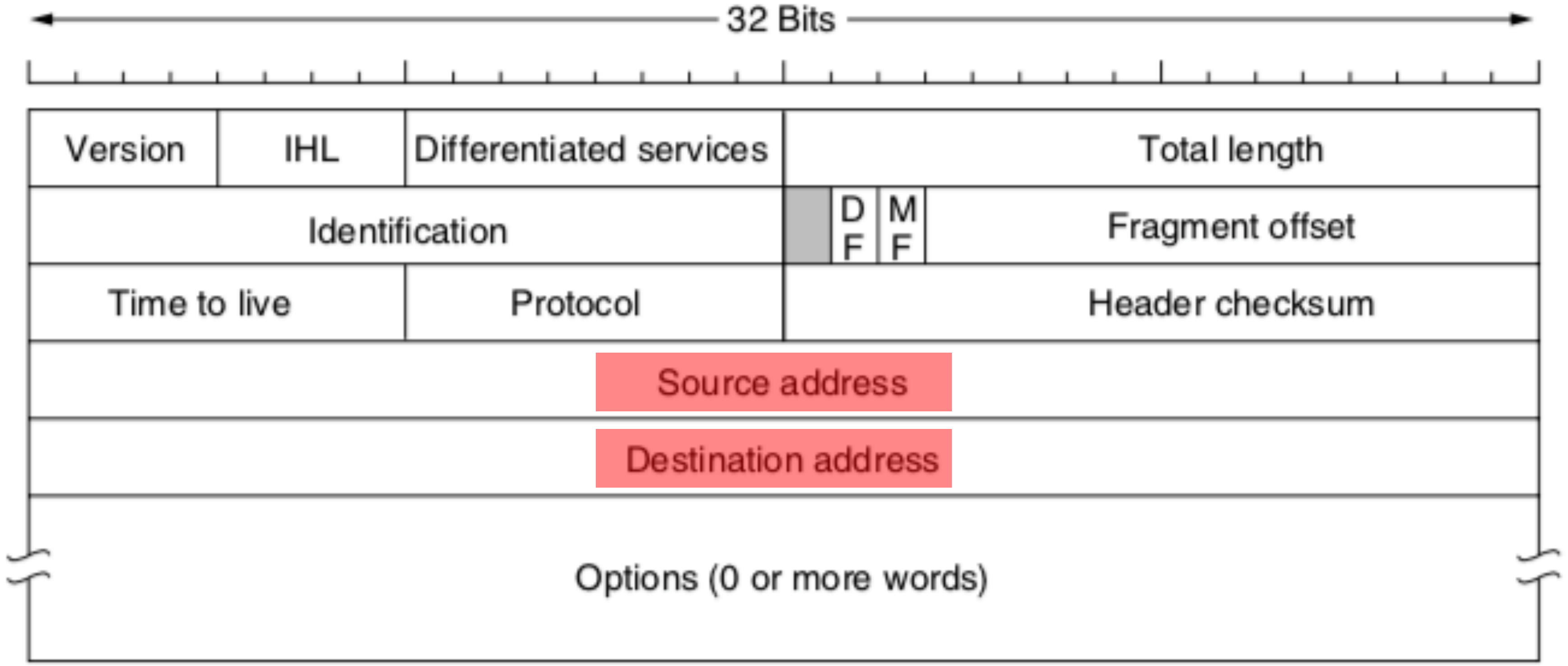


Figure 5-46. The IPv4 (Internet Protocol) header.

IP Address

- 32 bits
- It is important to note that an IP address does not actually refer to a host. It really refers to a **network interface.**
  - if a host is on two networks, it must have two IP addresses.
  - in practice, most hosts are on one network and thus have one IP address.
  - 127.0.0.1

IP address

- IP address: Two portion
  - Network
    - the **same value** for all hosts on a single network
    - Corresponds to a contiguous block of IP address space (Prefix)
  - Host

Prefix

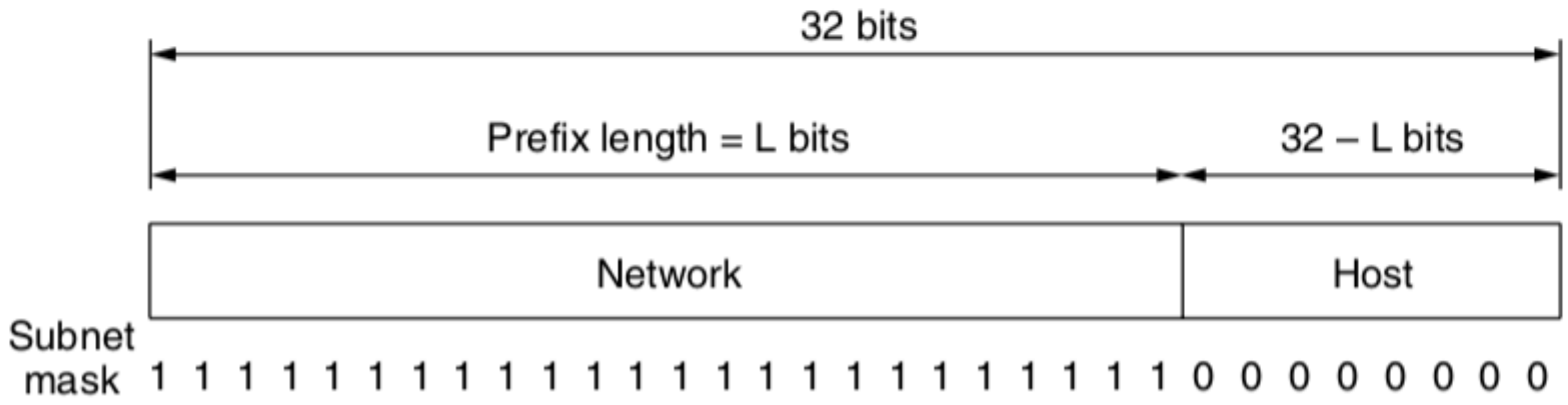


Figure 5-48. An IP prefix and a subnet mask.

Represent of IP address

- Binary :

  `10000000.11010000.00000010.10010111`
- Decimal: `128.208.2.151`
  - **Dotted decimal notation**
  - In this format, each of the 4 bytes is written in decimal, from 0 to 255

Prefix

- Prefixes are written by giving the lowest IP address in the block and the size of the block.
- The size is determined by the number of bits in the network portion; the remaining bits in the host portion can vary.
- the size must be a power of two
- For example
  - 128.208.2.151
  - if the prefix contains 2^8 addresses and so leaves 24 bits for the network portion, it is written as 128.208.2.0/24

Subnet mask

- The length of the prefix corresponds to a binary mask of 1s in the network portion.
- It can be ANDed with the IP address to extract only the network portion.

Convert (B -> D)

- 10000000 -> 2^7 -> 128
- 01000000 -> 2^6 -> 64
- 00100000 -> 2^5 -> 32
- 00010000 -> 2^4 -> 16
- 00001000 -> 2^3 -> 8
- 00000100 -> 2^2 -> 4
- 00000010 -> 2^1 -> 2
- 00000001 -> 2^0 -> 1

Solution 2

- 11000001
- -> 10000000 + 01000000 + 00000001
- -> 128 + 64 + 1
- -> 193
- 01010010,11010010,00001111
- -> 01000000 + 00010000 + 00000010
- -> 64 + 16 + 2
- -> 82

Solution 2

- 11010010
- -> 10000000 + 01000000 + 00010000 + 00000010
- -> 128 + 64 + 16 + 2
- -> 210
- 00001111
- -> 00001000 + 00000100 + 00000010 + 00000001
- -> 8 + 4 + 2 + 1
- -> 15

Solution 2

- 11000001,01010010,11010010,00001111
- -> 193.82.210.15

Question 6

- Convert the IP address 240.68.10.10 to binary format

Use the following key:

- 10000000 -> 2^7 -> 128
- 01000000 -> 2^6 -> 64
- 00100000 -> 2^5 -> 32
- 00010000 -> 2^4 -> 16
- 00001000 -> 2^3 -> 8
- 00000100 -> 2^2 -> 4
- 00000010 -> 2^1 -> 2
- 00000001 -> 2^0 -> 1

Solution 6

- 240
- -> 128 + 64 + 32 + 16
- -> 1000000 + 01000000 + 00100000 + 00010000
- -> 11110000
- 68
- -> 64 + 4
- -> 01000000 + 00000100
- -> 01000100

Solution 6

- 10
- -> 8 + 2
- -> 00001000 + 00000010
- -> 00001010

0

Solution 6

- 240.68.10.10
- -> 11110000 . 01000100 . 00001010 . 00001010