Pure Aloha vs Slotted Aloha

Pure Aloha	Slotted Aloha
Any station can transmit the data at any time.	Any station can transmit the data at the beginning of any time slot.
The time is continuous and not globally synchronized.	The time is discrete and globally synchronized.
Vulnerable time in which collision may occur = $2 \times T_{Fr}$	Vulnerable time in which collision may occur = T_{Fr}
Probability of successful transmission of data packet= $G \times e^{-2G}$	Probability of successful transmission of data packet= $G \times e^{-G}$
Maximum efficiency = 18.4% (Occurs at G = 1/2)	Maximum efficiency = 36.8% (Occurs at G = 1)
Main advantage: Simplicity in implementation.	Main advantage: It reduces the number of collisions to half and doubles the efficiency of pure aloha.

IEEE 802.11 – Collision Avoidance

- 802.11 addresses these two problems with an algorithm called Multiple Access with Collision Avoidance (MACA).

Key Idea

- Sender and receiver exchange control frames with each other before the sender actually transmits any data.
- This exchange informs all nearby nodes that a transmission is about to begin.
- Sender transmits a Request to Send (RTS) frame to the receiver.
  - The RTS frame includes a field that indicates how long the sender wants to hold the medium. Length of the data frame to be transmitted.
- Receiver replies with a Clear to Send (CTS) frame
  - This frame echoes this length field back to the sender

IEEE 802.11 – Collision Avoidance

- The idea of using ACK in MACA is Proposed in MACAW: MACA for Wireless LANs.
- Receiver sends an ACK to the sender after successfully receiving a frame.
- All nodes must wait for this ACK before trying to transmit.
- If two or more nodes detect an idle link and try to transmit an RTS frame at the same time
  - Their RTS frame will collide with each other
- 802.11 does not support collision detection
  - So the senders realize the collision has happened when they do not receive the CTS frame after a period of time
  - In this case, they each wait a random amount of time before trying again.
  - The amount of time a given node delays is defined by the same exponential backoff algorithm used on the Ethernet.

Classes of IPv4 address

Address Class	1st Octet range in decimal	1st Octet bits (Blue Dots do not change)	Network (N) and Host (H) Portion	Default mask (Decimal)	Number of possible networks and hosts per network
A	0–127	00000000 - 01111111	N.H.H.H	255.0.0.0	128 Nets (2^7) 16,777,214 hosts (2^{24}–2)
B	128–191	10000000 - 10111111	N.N.H.H	255.255.0.0	16,384 Nets (2^{14}) 65,534 hosts (2^{16}–2)
C	192–223	11000000 - 11011111	N.N.N.H	255.255.255.0	2,09,150 Nets (2^{21}) 254 hosts (2^8–2)
D	224–239	11100000 - 11101111	NA (Multicast)	-	-
E	240–255	11110000 - 11111111	NA (Experimental)	-	-

CSMA/CD

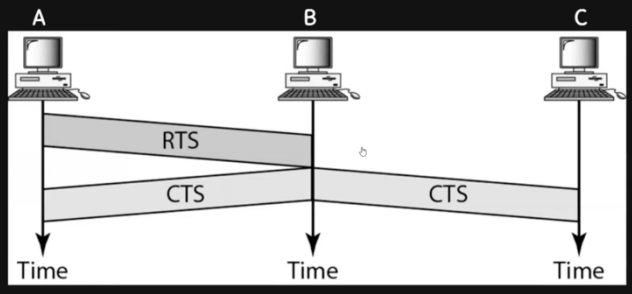
- If two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately.
- Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected.
- Quickly terminating damaged frames saves time and bandwidth.
- This protocol, known as CSMA/CD (CSMA with Collision Detection) is widely used on LANs in the MAC sublayer.
- Access method used by Ethernet: CSMA/CD.

CSMA/CA

- Carrier-sense multiple access with collision avoidance (CSMA/CA) is a network multiple access method in which carrier sensing is used, but nodes attempt to avoid collisions by beginning transmission only after the channel is sensed to be "idle".
- It is particularly important for wireless networks, where the collision detection of the alternative CSMA/CD is not possible due to wireless transmitters desensing their receivers during packet transmission.
- CSMA/CA is unreliable due to the hidden node problem and exposed terminal problem.
- Solution: RTS/CTS exchange.
- CSMA/CA is a protocol that operates in the Data Link Layer (Layer 2) of the OSI model.

- An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a device (for example, a computer or a router) to the Internet.

Use of RTS/CTS handshaking

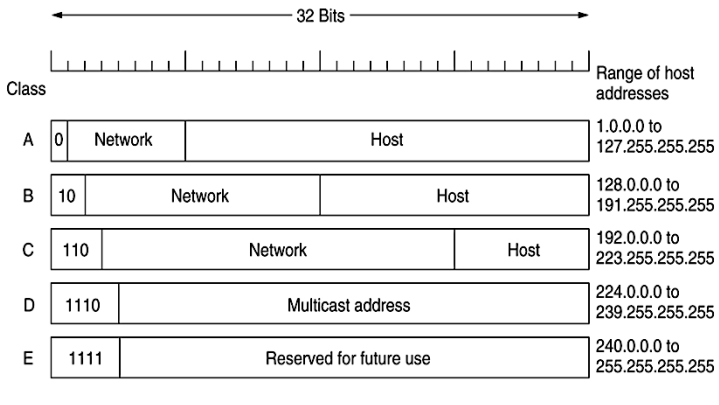


Station C doesn't hear RTS from A, but it does hear CTS from B, so it knows something is up.

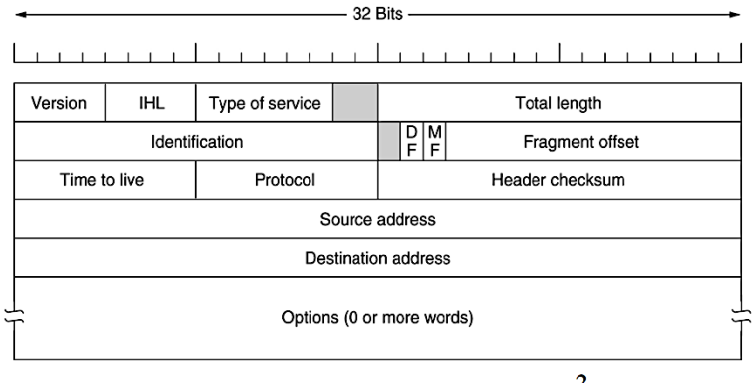
Subnet mask (Slash notation)

Class	Subnet Mask (in Decimal)	Subnet Mask (in Binary)	Slash Notation
A	255.0.0.0	11111111.00000000.00000000.00000000	/8
B	255.255.0.0	11111111.11111111.00000000.00000000	/16
C	255.255.255.0	11111111.11111111.11111111.00000000	/24

• Address classes



• IPv4



b) CSMA/CD is typically more efficient than slotted ALOHA for a few reasons:
- CSMA/CD has nodes stop transmitting immediately upon CD; slotted ALOHA always transmits the whole frame
- CSMA/CD nodes refrain from transmitting until they sense the channel is idle; slotted ALOHA simply transmits during the first slot occurring. Thus CSMA/CD has fewer collisions, especially during long transmissions
- CSMA/CD using binary exponential backoff to determine how long to wait before retransmitting a collision; slotted ALOHA simply retransmits with probability p in each slot. This makes CSMA/CD have fewer collisions when the channel is very busy overall.

These are some of the reasons (among others) that Ethernet uses CSMA/CD over slotted ALOHA and are reasons that CSMA/CD is more efficient than slotted ALOHA.

---

5. An IP packet (with no IP options) has the size of 3000 bytes. The packet has the source IP address of 142.104.81.63 and the destination IP address of 216.58.218.163.

   (a) After the packet has been forwarded through a link with the maximum transmission unit (MTU) of 1500 bytes (i.e., the maximum IP packet size supported by the link is 1500 bytes), please write down the "IP Header Length (IHL)", "Total length", "Fragment offset" and "More Fragment (MF)" of all IP fragments of the original 3000-byte IP packet. [3]

   (b) A router received this packet has the following forwarding table. Which output interface the router should choose to send out the above fragments? [2]

Destination network	Netmask	Output interface
142.104.64.0	19	1
142.104.80.0	20	2
216.58.216.0	21	3
216.58.220.0	22	4
...	...	...

since there's no mechanism to detect collisions.

a) Split 20 header + 2980 data into
- 20 header + 1480 data, offset 0, MF=1
- 20 header + 1480 data, offset 1480, MF=1
- 20 header + 20 data, offset 2960, MF=0

So

Fragment A:
IHL = 5
Total Length = 1500
Fragment offset = 0
MF = 1

Fragment B:
IHL = 5
Total Length = 1500
Fragment offset = 1480
MF = 1

Fragment C:
IHL = 5
Total Length = 40
Fragment offset = 2960
MF = 0

b) destination: 216.58.11011010.163
masks: interface 3: 216.58.11011000.0/21
interface 4: 216.58.11011100.0/22
The destination IP address matches the subnet mask for interface 3 so this is where the router should send the above fragments.

In a communication network with a shared medium assume all packets have the same size of 1000 bytes and the transmission rate is 8 MBPS. The propagation time between nodes within the network is 100 NS. Use an example to show how Collision Detection (CD) improves the efficiency in local area networks.

Let's consider a simplified example to demonstrate how Collision Detection (CD) improves the efficiency in local area networks (LANs). Suppose we have a LAN with a shared medium where multiple nodes are connected. Each node wants to transmit packets across the network. For this example, let's assume there are two nodes: Node A and Node B.

Without Collision Detection (CD):
- Node A and Node B both want to transmit packets at the same time.
- They start transmitting simultaneously
- As a result, their packets collide in the shared medium.
- Both nodes detect the collision after a certain time (based on the propagation delay) and initiate the retransmission process.
- This leads to wastage of bandwidth and time.

With Collision Detection (CD):
- Node A and Node B both want to transmit packets at the same time.
- They start transmitting simultaneously, assuming the medium is free.
- However, due to the Collision Detection mechanism:
- Node A and Node B can detect the collision almost immediately after it occurs.
- They stop transmitting as soon as they detect the collision.
- Each node waits for a random backoff time before attempting to retransmit.
- By stopping transmission upon collision detection, the wasted bandwidth and time are minimized.
- Eventually, Node A and Node B reattempt transmission at different times, reducing the likelihood of collision.

In summary, Collision Detection improves efficiency in local area networks by minimizing the

2. An IP packet has the following information in its header arrives at a WLAN:

...	length	ID	fragflag	offset	...
	5000	x	0	0	

Since the maximum transmission unit (MTU) of the WLAN is 2308 bytes, the packet will be fragmented into how many packets? What will be the length, ID, fragflag and offset values in their IP headers?

Looking at the packet provided, we can see that the total size of the IP packet header and data is 5000 bytes. The IP header typically takes up 20 bytes for IPv4. For each fragment, except the last one, there will also be an additional 20 bytes for the IP header. The rest of the space is used for the actual data payload.

Data payload size per fragment = MTU - IP header size = 2308 - 20 = 2288 bytes.
Number of fragments required = Total length / Data payload size per fragment
= 5000 / 2288 = 2.19

Because fragments cannot be divided into smaller parts, we require three fragments to encompass the entire packet. The initial two fragments will contain 2308 bytes each (comprising the IP header and data payload), while the final fragment will consist of the remaining 424 bytes, comprising only the data payload.

The ID field assists the receiving host in recognizing which packet a newly received fragment belongs to. Consequently, all fragments belonging to the same packet will share the same ID value. The fragflag is necessary to determine when all fragments of a packet have been received. Hence, all fragments except the final one have this bit set to 1.

The offset value indicates where a particular fragment starts within the current packet. These values increase in chunks of 8 bytes. So, the offset for the initial fragment is 0. For the second fragment, it's calculated as 2288 divided by 8, which equals 286. Likewise, for the third fragment, it's calculated as 4576 divided by 8, resulting in 572. Here's how the three data segments look visually:

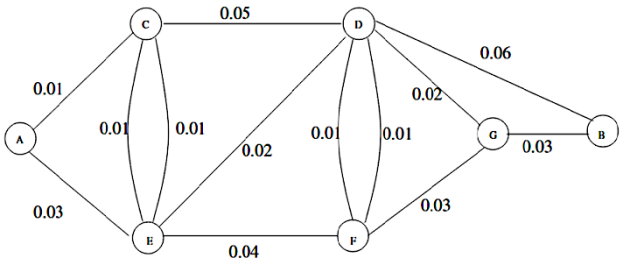
impact of collisions and reducing the wastage of bandwidth and time. It allows

...	length	ID	fragflag	offset	...
	2308	x	1	0	

...	length	ID	fragflag	offset	...
	2308	x	1	286	

...	length	ID	fragflag	offset	...
	424	x	0	572	

3. The number shown in the following figure is the probability of the link failing. It is assumed that links fail independently of each other.



(a) **Find the most reliable path from A to B, i.e., the path for which the probability that all links stay intact is maximal. [Hint: for link i with failing probability $p_i << 1$ and link j with failing probability $p_j << 1$, Pr{fail of the path through link i and link j} $= 1 - (1 - p_i)(1 - p_j) = p_i + p_j - p_i p_j \approx p_i + p_j$]**

Iterations	A	B	C	D	E	F	G
Initially	(0, A)	(∞, .)	(0.01, A)	(∞, .)	(0.03, A)	(∞, .)	(∞, .)
1	(0, A)	(∞, .)	(0.01, A)	(0.06, C)	(0.02, C)	(∞, .)	(∞, .)
2	(0, A)	(∞, .)	(0.01, A)	(0.04, E)	(0.02, C)	(0.06, E)	(∞, .)
3	(0, A)	(0.1, D)	(0.01, A)	(0.04, E)	(0.02, C)	(0.05, D)	(0.06, D)
4	(0, A)	(0.1, D)	(0.01, A)	(0.04, E)	(0.02, C)	(0.05, D)	(0.06, D)
5	(0, A)	(0.09, G)	(0.01, A)	(0.04, E)	(0.02, C)	(0.05, D)	(0.06, D)
6	(0, A)	(0.09, G)	(0.01, A)	(0.04, E)	(0.02, C)	(0.05, D)	(0.06, D)

The most reliable path from A to B is the path through A→C→E→D→G→B. The probability of the link failing through this path is 0.09.

(b) Find the second most reliable path from A to B which does not share any link belonging to the path found in (a).

In this case, the most reliable path would be through A→E→F→D→B. The probability of the link failing through this path is 0.14.

accordingly, leading to better network performance and throughput.

Question 3: An 8-bit byte with binary value 10101100 is to be encoded using Hamming code. What is the binary value after encoding? [Hint: use 4 check bits]

Solution.

Given, the binary value is 10101100, which is an 8-bit binary value. We know 8-bit sequence can contain 4 check bits (2^0=0, 2^1=2, 2^2=4, 2^3=8). We can also check if the number of check bits (r) and data bits (m) follows the relation mentioned in the textbook as:

$m+r+1 \leq 2^r$
or, $8+4+1 \leq 2^4$
or, $13 \leq 16$ *which satisfies the condition for the number of checkbits = number of parity bits.*

Bit Position	1	2	3	4	5	6	7	8	9	10	11	12
Binary Before Encoding	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100

From the Binary Values before encoding, we get this table:

Seq.	Binary Table				Check Bit Positions	XOR
1	0	0	0	1	Check bit positions for 1: 3, 5, 7, 9, 11	0
2	0	0	1	0	Check bit positions for 2: 3, 6, 7, 10, 11	1
3	0	0	1	1		
4	0	1	0	0	Check bit positions for 4: 5, 6, 7, 12	0
5	0	1	0	1		
6	0	1	1	0		
7	0	1	1	1		
8	1	0	0	0	Check bit positions for 8: 9, 10, 11, 12	1
9	1	0	0	1		
10	1	0	1	0		
11	1	0	1	1		
12	1	1	0	0		

From this stipulation, we have:

Bit Position	1	2	3	4	5	6	7	8	9	10	11	12
Binary Before Encoding	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
Data Bits			1		0	1	0		1	1	0	0
Check Bits	0	1		1				0				
Binary After Encoding	0	1	1	1	0	1	0	0	1	1	0	0

nodes to detect collisions promptly and adjust their transmission behavior