Contd with direct link between two hosts

Now, the link is shared by other nodes.

We are limited by the physical medium, such that if we put something into the channel at another time as another host, the information may collide and be lost.

So, we can draw up several hosts, maybe connected by a single wire to each computer, or a central wire with connections to each host.

Whatever the setup, we still need to send info from a to b.

A and b have a direct medium, but there are other machines there.

We can think of the other machines as passing the connection along, giving us a direct physical link.

The problem is that if you’re sending something, another machine may be sending one at the same time, causing colliding of the signals and loss.

So we need something to coordinate this information exchange.

We can call these mechanisms protocols

Medium Access Protocols. > solves Medium Access Problem.

Question is one of coordination.

There are several solutions, all called Medium Access Protocols, aka MACs.

Solution 1: we set a master node in the network, which decides which node will communicate when. If a node wishes to communicate, it must ask the master node. Alternatively, it assigns specific time intervals to the various nodes.

Problem is thinking of a master node, as we require a decentralised solution. If the master node goes down, then the entire network goes down.

So this will not work.

Some MACs are used in Ethernet/LAN connections. There are three commonly used, broad protocol approaches, which solve the problem differently.

Approach 1: Channel Partitioning Protocol

Approach 2: Random Access Protocol (most important)

Approach 3: Taking Turn Protocol (rename)

Approach 2:

Methods: Slotted ALOHA//Pure ALOHA//CSMA/CD (Ethernet Protocol) (carrier sense multiple access/collision detection)

For the moment, we aren’t looking at the addresses of the nodes.

Note that: any message by node n\_i is received by all other nodes. The next step is what the nodes do with this. Since this is meant for node n\_j, the node n\_i should have the address of node n\_j.

Slotted ALOHA:

Aloha is a project name for a network. It was one of the first networks. This protocol was on the ALOHA network, which was a radio network. The University of Hawaii was the host, where there was one node, and there were several islands and hosts far away, communicating over the medium of radio networks.

There were two frequencies used at that time, one was when the communication was being BROADCAST, (when one node sends something all other receive it).

Now the central node wants to know if the other nodes have received the message, so all other nodes were sending back an acknowledgement. To prevent collision, the other nodes were allowed to use a different specific frequency.

The problem is if two nodes were sending a response at the same time, which could cause two radio waves to collide and result in loss of information.

Thus the question of establishing a proper protocol to prevent this came up.

Robinson/Branson from the uni worked on this problem, and this is his protocol. Pure ALOHA.

Slotted ALOHA is a hypothetical protocol.

Assumption 1: there is a master node. Its activity is unique.

Let there be many nodes. We have nodes from n\_1 to n\_i.

If the network begins at t\_0, after that the network time will split into different slots. The protocol says that every time a node has a frame to transmit, if at that time the node is in between slots, it should wait until the beginning of the next slot, and at that time it should begin to transmit. Ie, the node can only transmit at the beginning of any slot.

This requires the nodes to be on the same clock, which the master node will ensure.

Say n\_1 has a frame to send, and begins to transmit at t\_0. When it transmits, we can show the signal as spreading to other nodes in the medium.

The time slots are set such that they are sufficient for the nodes to transmit any frame which the node is sending, and for the other nodes to receive said frame.

Assumption 2: Every node is sending a frame of equal size, and the frame size is always of L bits. So if the networks is M bits per second, it is sufficient to send a frame of L bits.

If no one else is sending a frame at this time, all nodes will receive the frame easily.

Next step: if when n\_1 is sending the message, another node n\_3 is sending a message. This would result in a collision. The nodes have the capability to detect a collision (this is left to the hardware), so the nodes realise they need to re-transmit.

Say T=5ms, and both n\_1 and n\_3 transmit at the start of the slot. They find out about the collision at 3ms. The nodes will continue to transmit and finish the transmission. They know they have to re-transmit, with the slot at t\_0+t. So one node has to back off, in a decentralised method.

They decide via a probability decision. Take a p, such that 0<p<1. (a coin flip)

So they decide that n\_1 and n\_3 will transmit with probability p, so both may transmit, one may transmit, or neither may transmit.

Problem: If there are more nodes in a system, this leads to more collisions. Time gets wasted if neither transmits, or there are repeated collisions.

The efficiency of the system is the fraction of slots which are successful against total number of slots. (ie when there is no collision, and one node transmits)

Now, what is the probability that any one specific slot is successful? P[One slot is successful] = P[n\_1 (and) !n\_2 (and) !n\_3…] + P[!n\_1 (and) n\_2 (and) !n\_3…] + … = np(1-p)^{n-1}

Now, we want this probability to be as high as possible.

Theta(p)=np(1-p)^(n-1)

Log theta(p)= log n + log p + (n-1)log(1-p)

D log theta(p)/d p = 0 + 1/p + n-1/1-p – 1/p – n-1/1-p

D log theta (p)/d p = 0

1/p – n-1/1-p = 0

p= 1/n

this gives us the highest efficiency.