1 GSW… Multiple Access Protocols

Put a lot of people in the same room, give them all a piece of information, and ask them to communicate this information to everyone else in the room. Don't allow anyone to move closer to anyone else, so they can't just go up to everyone in turn, and whisper in their ear. What happens?

There might be an awkward silence for a short time, while everyone works out the best way to do this. At some point, though, someone will start talking. If only one person starts talking, then everyone else will probably not interrupt them; after all there's no point in starting to talk if someone is already talking. For reasons we'll talk about later, this is known as *carrier sense*. The sole speaker will then be able to get their message across to everyone else.

However, if two people start to talk at the same time, there will be what is known as a *collision*. The people closest to each of the speakers might be able to tune-out the more distant speaker and just listen to the person closest to them (this is known as the *capture effect*), but most people will not be able to understand anything either of them says.

What will probably happen next is that the speakers will notice that someone else is speaking at the same time, and they'll stop speaking. This is known as *collision detection*. After a short pause, some number of people (hopefully just one) will start to speak.

Once again, if only one person starts to speak, that's fine. If more than one starts to speak, there will be another collision. If there are only a small number of people in the room, then it won't take long before all of them manage to find a gap to start speaking in, and everyone gets their message across to everyone else.

But what if there are hundreds of people in the room? The chances of only one person deciding to start talking at any time are pretty remote. They'll need to organise themselves, and agree on a set of rules that will allow them all to get their message across to everyone else with the minimum number of collisions. This is exactly the problem that multiple access protocols try and solve.

Multiple access protocols are near the bottom of the multi-layer stack of protocols: in the ISO OSI model these protocols are found in the Data Link Layer. This layer has no idea what any of the bits in the message mean, only that it needs to get access to the medium[1], and send the message[2]. The problem is when and how: every node needs to avoid every other node trying to access the medium at the same time.

In this chapter, I'll talk about the issues and performance metrics used to evaluate multiple access protocols in general, and illustrate them with two extreme cases: unreliable ALOHA and round-robin. I'll talk about other schemes in the following chapters.

[1] In this sense, the 'medium' is the shared 'thing' that everyone needs to have sole access to, in order to communicate. In the room full of people wanting to talk, it's the pressure of the air: that's what carries the sound. In a local area network, it might a specific range of radio frequencies in the air (for wireless LANs), or the voltages on a copper cable, or the intensity of light. Whatever is shared by everyone, and used to transmit all the information *through*.

[2] Imagine the message you gave to everyone in the room was in code. No-one was told what their message means, only that they had to speak it exactly correctly, and write down all the other messages exactly correctly too. They were then told to take the messages they heard to their boss, who would be able to decode the message. That's pretty much what happens with protocol stacks.

1.1 Classifying Multiple-Access Schemes

There are a huge number of multiple access schemes (they probably number into the thousands by now). Fortunately, not many of them are actually used.

There are two distinct types of multiple access schemes: contention-based schemes, and contention-free schemes. The example described above (of all those people in the room trying to talk at the same time and interrupting each other) was a contention-based scheme. Any scheme in which it is impossible for two nodes to transmit at the same time is a contention-free scheme.

As a simple example of a contention-free scheme: suppose that in this room, there was one orange. Someone has the orange. Everyone who wants to speak puts up their hand. If the person with the orange wants to speak, they can do so. When they finish, or if they don't want to speak, they pass the orange in the direction of the next person with their hand up. Eventually, everyone will be able to speak. (This is known as a *token-passing* scheme, the orange is the *token*.) Clearly, since there is only one orange, it is impossible for two people to try and talk at the same time.

The problem with contention-free schemes is that it if there are a large number of people in the room, but only a few want to talk, it be a long time between being given a message to transmit, and being able to start talking. You have to wait for the token to arrive. The advantage is much less wasted time due to collisions.

In general, contention-free schemes are more complex, and give better performance when the nodes have a lot of messages to transmit (they prevent collisions), and/or there aren't very many nodes (no-one has to wait very long before they can start transmitting). On the other hand, contention-based schemes are simpler, and give better performance when there aren't very many messages to transmit and/or there are a very large number of nodes.

1.2 Division Multiplexing and Multiple Access

However, before I go much further, I should mention something about multiple access protocols in the context of radio systems[3]. Unlike almost all local area networks, copper and fibre-based systems, there are three distinct types of multiplexing used in radio systems. (Local area networks, copper and fibre-based systems typically only use one of them.)

These three types of multiplexing are known as time-division multiple access (TDMA), frequency-division multiple access (FDMA) and code-division multiple access (CDMA). Briefly, TDMA consists of dividing up the radio spectrum into small units of time (known as *timeslots* or sometimes just *slots*), and allowing different people to transmit in different slots. All you need to do is organise who is allowed to transmit in which slot. It's similar to the example above, where only one person can talk at the same time: they just have to agree on the order in which they talk.

Frequency-division multiplexing consists of dividing up the radio spectrum into small units of frequency, and then assigning one of these frequencies to each user (exactly how each user finds out which frequency they should transmit on is the job of the multiple access protocol).

[3] Most of the interesting media access schemes being developed at the moment are for radio systems.

Code-division multiple access is a technique by which everyone transmits at the same time on the same frequency. You might think the result would be a complete mess, but due to the use of *spreading codes*, each receiver can work out what everyone is saying. There's much more about how this works in the chapters on wireless systems. This chapter is part of the section about the Internet, and just about all multiple access protocols on the Internet are time-division multiple access, so for now I won't say any more about FDMA or CDMA protocols.

Another issue it's worth clearing up before we get into the details of multiple access protocols is the difference between a *multiple access scheme*, and *multiplexing*.

Briefly, a multiple access scheme, such as TDMA, is a set of rules that enables more than one node to transmit information without too much interference from the other nodes in the same system. A multiplexing scheme, such as time-division multiplexing (TDM) is a method by which a single node separates out its transmissions to several other nodes.

Consider a mobile phone system. All the people with mobile phones use a multiple-access system (such as TDMA) to transmit their information to the base station. On the other hand, the base station uses a multiplexing scheme (such as TDM) to transmit the information back to the mobile phones. The key point is that the base station has one transmitter that is used to send information to multiple receivers; whereas the mobile phones are multiple transmitters sending information to one receiver[4].

(Sorry if that's labouring the point a bit, but this does seem to cause a lot of confusion. Back to the Internet, and the particular case of local area networks.)

Most of the terminals in a communications network do not send and receive data all the time, so it doesn't make sense for them to have a dedicated series of timeslots or a dedicated frequency just for themselves. Any such protocol would avoid all collisions, but would restrict all users to transmit at a maximum of $1/N$ of the possible capacity of the network (where N is the number of nodes in the network). With a lot of nodes, all of which are not transmitting very often (the usual case), that's a huge waste of network capacity[5]. Instead, they often share a transmission medium (e.g. a wire, or a bit of radio spectrum) with their neighbours. The set of nodes sharing a transmission medium belong to the same *local area network* (LAN).

That's not a perfect definition of a LAN, but it's not too bad either. A better definition is that a local area network is a single *broadcast domain* (a set of nodes that receive a broadcast frame). In other words, any frame[6] sent to a broadcast address is received by every node in the same local area network as the sender, but no other nodes[7].

[4] It doesn't have to be multiple transmitters talking to a single receiver. Multiple transmitters talking to multiple receivers would still count as a multiple-access system. The important bit is that there is more than one node transmitting, so collisions are possible. With just one transmitter, you can't get collisions.

[5] Even if this does happen quite a lot of the time in wired systems. For example, most modern wired Ethernet systems have a separate cable running from the hub/switch to each node that only carries information for that node.

[6] These protocols operate at the data link layer, so I'll refer to the groups of bits travelling around together as frames.

[7] Like most rules in protocol engineering, there are some exceptions to this, but it's not a bad general guideline. Don't be too surprised if you come across some broadcast frames that go outside their LAN (see for, example, the DHCP request frames).

The multiple access protocol is responsible for determining which users on the LAN can talk, and when, while attempting to minimise the number of collisions, and ensuring that everyone has a fair share of the available network capacity.

1.3 Some Performance Metrics

The ideal multiple access protocol would ensure that someone was receiving useful information all the time. Unfortunately that's impossible, some of the network capacity will always either be wasted during collisions, or be required to send and receive data-link layer messages organising the nodes so that they don't try and transmit at the same time. To evaluate and compare different multiple access schemes, we'll need a few metrics.

1.3.1 Offered Traffic

The *offered traffic* is the amount of traffic the next higher level is trying to send over the shared medium; this would be equal to the throughput of the network if the network had an infinite network capacity. Offered traffic is usually measured in bits/sec or packets/sec.

1.3.2 Network Capacity

The *network capacity* is the maximum amount of traffic that could be sent across the network if one transmitter was transmitting continuously. This is the throughput that would occur if the access scheme was perfect, and the offered traffic was at least equal to the network capacity, so the transmitting nodes never ran out of packets to send. Network capacity is also measured in bits/sec or packets/sec, and usually a fixed number for any given network.

1.3.3 Throughput

The throughput is the amount of data received by the nodes on the network (as before, I'm defining this in terms of data received rather than data transmitted, so that any frames lost due to collisions and other errors don't count as part of the throughput). Again, it's measured in bits/sec or packets/sec, and is a function of the offered traffic. Obviously the throughput can never be greater than the offered traffic.

1.3.4 Utilisation

Just like for a point-to-point link, the utilisation of a network is the ratio of the throughput to the network capacity. Utilisation is dimensionless, but a function of the offered traffic. An ideal multiple access protocol would achieve a utilisation of 100% when the offered traffic was equal to or greater than the network capacity.

1.3.5 Efficiency

The efficiency of a multiple access scheme is the ratio of the throughput to the offered traffic; it's a measure of how much of the offered traffic actually gets through to its destination. Again, this is dimensionless, and a function of the offered traffic. For offered traffic levels less than the network capacity, an ideal multiple access protocol would have an efficiency of 100%.

1.3.6 Fairness and Priority

It would be very easy to design a protocol that could provide 100% utilisation when the offered traffic was very high. All you do is pick one node, and allow it to transmit all the time. Any such protocol would, of course, rather inconvenience everyone else (to put it mildly). A good multiple access protocol should attempt to ensure that every user gets their fair share of the network capacity.

There's no universally accepted definition of fairness, but the most common is one suggested by Jain[8]. If the i^{th} user manages to transmit x_i bytes of information, then a fairness index can be defined as:

$$\text{fairness} = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{\left(n\sum_{i=1}^{n} x_i^2\right)} \tag{0.1}$$

When all of the n users have the same share of the network capacity this fairness index is equal to one. If all the x_i are not equal, then the fairness reduces. The minimum possible value of fairness is zero, and this occurs when there are an infinite number of users, only a finite number of which are ever allowed to transmit.

The problem with this fairness index is that it assumes that the ideal condition is that all users should be able to transmit the same amount of information, irrespective of what that information is. For modern networks that carry emails, voice, video and just about everything else, and offer different qualities of service to different users, sometimes at different prices, things are no longer that simple.

Many multiple access protocols deal with the problem of carrying different types of information by assigning *priorities* to different frames. In this case, an ideal multiple access scheme should provide equal access to users with the same priority of message to send, but more capacity to those with more important information.

1.3.7 Delays and Latency

Some multiple access schemes introduce significant delays. For example, suppose there was a central node for the network, and this central node went around each user in turn, asking if they had anything they wanted to transmit. (This scheme is called *polling*, and it's a contention-free scheme.) This scheme works very well if everyone has something to transmit: it avoids any possibility of collisions, and the network is kept full of useful information most of the time[9]. However, if there are a very large number of nodes, and not many of them have any information to transmit, a node can wait for a long time before its turn comes round.

(As noted before: these additional delays are a common problem for these contention-free multiple access schemes that attempt to eliminate collisions. On the other hand, contention-based multiple access schemes that allow nodes to decide for themselves when they transmit don't have the same problem with delays, but they do run the risk of collisions.)

[8] Jain, R., Chiu, D.M., and Hawe, W. "*A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems*", DEC Research Report TR-301, September 1984

[9] Only most of the time: the rest of the time it's full of short messages to and from the controller, asking if anyone has anything to transmit.

1.3.8 Ideal and Realistic Multiple Access Schemes

The most common way to plot the performance of a multiple-access protocol is in terms of the plot of throughput against offered traffic. For an ideal protocol, such a plot would look like this:

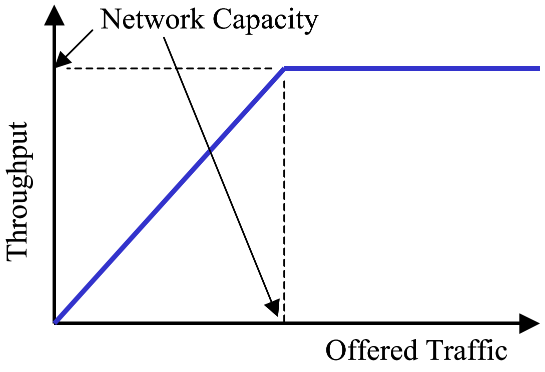

Figure 1-1 - Performance of an Ideal Multiple Access Protocol

This is the best that any network access protocol can do. It accepts all offered traffic up to the capacity of the network, and then continues to provide this maximum level of service no matter how much more offered traffic is presented to the network (at this point it is said to be *overloaded*). No access protocol can achieve a graph like this, although some are rather better than others at approaching this ideal.

The performance of real multiple access protocols never quite attains this ideal. Contention-based protocols (which allow every user to decide individually when to start to transmit) always have the possibility of collisions, and the probability of a collision increases with the number of nodes with some new information to send. A more typical graph for a real contention-based multiple access protocol would look something like:

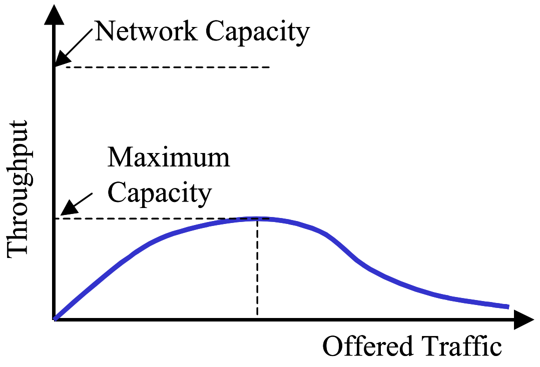

Figure 1-2 - Performance of a Representative Contention-Based Multiple Access Protocol

Note that there is a value of offered traffic which gives a maximum capacity; with any more offered traffic the throughput actually reduces, as the network spends most of its time suffering from collisions. Also note that at very low levels of offered traffic, the throughput is almost equivalent to the ideal case shown above, and the efficiency is therefore reasonably high. Just about any multiple access protocol will do provided the offered traffic does not approach the network capacity[10].

Figure 1-3 below shows the performance of a typical contention-free multiple-access protocol. In this case there is no drop off in performance as the offered traffic increases past the network capacity, since there are no collisions. The disadvantage is that contention-free protocols are more complex, usually require one node to take charge of the network (albeit temporarily in some cases) and suffer from greater latency.

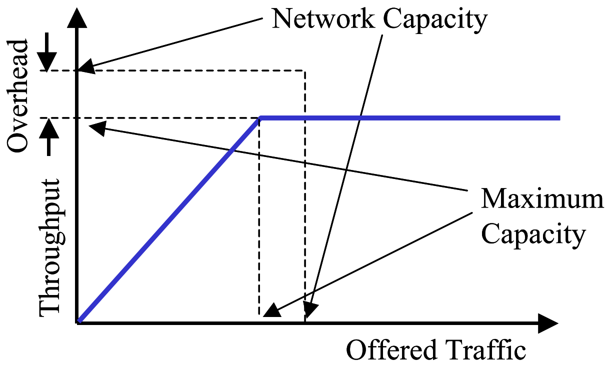

Figure 1-3 - Performance of a Representative Contention-Free Multiple Access Protocol

The difference between the network capacity and the maximum throughput is known as the *overhead*. This is the part of the network capacity used for sending round the control frames that tell the different nodes when they are allowed to transmit.

1.3.9 Stability

Finally, there is the issue of stability in the case of some contention-based multiple access protocols. If a protocol subject to collisions is a reliable protocol, or at least attempts to re-transmit any frame that it detects has collided with another frame, there is a real danger that the entire protocol could become unstable.

One simple way to understand how this can happen is to consider a protocol that, if it detects that a collision has happened, waits for a fixed period of time (for example 1 ms) and then tries to transmit the frame again. Obviously, this is a stupid idea, since if there is such a collision, and both transmitting nodes detect the collision, then they will both wait 1 ms, attempt to re-transmit their frames, and there will be another collision[11]. Neither of the two nodes will ever get their frames successfully transmitted; however they will continue to try. This reduces the time available for all the other nodes to transmit, increasing the probability of collision between other nodes. And so on, until all the available network capacity is being used to try and re-transmit colliding frames.

[10] This is true for a wide range of different networking problems. Many of them can be solved by just ‘throwing some more capacity’ at the problem. It’s only when the offered traffic starts approaching the network capacity that things start getting really interesting.

[11] This should be obvious, although I know of at least one chip manufacturer who designed an Ethernet chip that did exactly this.

Any protocol that attempts to detect collisions and re-transmit frames that have collided must be carefully designed to prevent this happening.

1.4 Some Example MACs

Analysis of the performance of multiple-access schemes is beyond the scope of this chapter: see later in the book for those. However, it might be useful to illustrate the variety of multiple access schemes available with a description of a few important examples.

1.4.1 ALOHA

The very first multiple-access data communications network in the world was developed by Norm Abramson and his team at the University of Hawaii in 1970 to allow communication between the computer systems on the different islands. The system they developed was called ALOHA. It's very simple: if you have a message to send, you just send it. That's it. You don't bother to check to see if anyone else is talking at the time, you just talk. Obviously, occasionally two people talk at once, there is a *collision*, and neither message gets through.

The original version of ALOHA used a logical star configuration, with a central hub receiving all the frames, and then transmitting them back out on a different frequency. If a node did not receive back its own frame, it assumed there had been a collision, and re-transmitted it some time later.

It's difficult to imagine a simpler multiple access protocol about the only possibility is the unreliable version of ALOHA, in which collisions are not detected at all (assuming any sort of reliable service was required, re-transmissions would have to be handled by a higher layer protocol). It is very easy to imagine more efficient schemes, however if the offered traffic is always well below the network capacity so that collisions are rare, there may not be a need for any additional complexity, and ALOHA can, and often does, work just fine.

1.4.2 Carrier-Sense Multiple Access

A slight refinement of ALOHA can dramatically increase the performance. Instead of just transmitting as soon as you have any information to transmit, you listen first (*sensing* the radio *carrier* frequency), to see if anyone else is already transmitting. If so, you wait for them to finish. This is known as *carrier-sense multiple access*, or *CSMA*. There are still collisions, but these now only happen when two nodes start to transmit at about[12] the same time. It's very popular, and used today in many wireless LANs.

1.4.3 Carrier-Sense Multiple Access with Collision Detection

A further refinement of ALOHA: not only do you not interrupt anyone already transmitting, but once you have started transmitting, you listen to see if anyone else has started to transmit at the same time. If so, and a collision has happened, you immediately stop transmitting, and

[12] "about" the same time means that the time between the first node starting to transmit and the second node starting to transmit is less than the time required to get a signal from the first node to the second node (equal to the distance between the nodes divided by the speed of light for radio waves). If all the nodes are close together, so the other nodes find out very quickly when one node starts to transmit, CSMA can be a very efficient multiple-access scheme.

don't bother to complete sending the frame (well, there's no point, there's been a collision and the receiver will not be able to receive it correctly anyway).

This is significantly more efficient than CSMA for networks where long frames are used, however it requires that a node is able to detect another transmission while it is transmitting itself. For radio, this is very hard to do.

1.4.4 Carrier-Sense Multiple Access with Collision Avoidance

This terms encompasses several refinements to CSMA that aim to prevent any two nodes attempting to transmit at the same time, and therefore avoid any collisions. Probably the most common is the RTS/CTS scheme, whereby a node wishing to transmit a frame transmits a short request-to-send (RTS) control frame first, asking for permission to send the frame. The destination then replies with a short clear-to-send (CTS) frame granting that permission. All other nodes that hear either the RTS or CTS frames then know that the frame is about to be transmitted, and don't interrupt.

Of course, two RTS frames can still collide, but since these are very short frames, there isn't much precious time lost in these collisions.

1.4.5 Polling

If there is one *master* node in the network (perhaps a router or access point to the rest of the Internet), then this node can go round all the other nodes in turn, asking if they have something to transmit (known as *polling*). If they do, they can then transmit it, and then tell the master node that they've finished. This is clearly a contention-free scheme, since there are never any collisions, but equally clearly, there is some additional delay: a node can't just start to transmit whenever it likes, it has to wait to be asked. There are also additional complications when a new node wants to join in.

Many variations on this scheme have been proposed, including providing some method for nodes to signal to the master that they want to transmit a frame, or tell the master that they will not want to transmit for a while, so the master doesn't waste time continually asking them. Further refinements allow nodes to reserve transmit time in advance. This can be a particular advantage for some types of traffic, for example digitised speech, which produces a new frame's worth of information at regular intervals. This is only possible where there is a central intelligent node (the master) controlling which node is allowed to transmit at what time.

1.4.6 Token-Passing

A variation on polling in which there is no master node. In token passing, there is a small control frame called a *token*. If a node has a frame to transmit, it has to wait for the token to arrive. When it does, it can transmit its frame, and then it must transmit the token onto the next node. And so on. This is another contention-free multiple access scheme, and suffers from the additional delay of all such schemes.

Token passing requires the nodes to be put in a well-defined order so nodes know where to send the token once they have finished with it. Organising this order as nodes continually arrive and leave the network, together with making sure the protocol can recover when the token frame is corrupted by noise and lost, can be a complex task.

1.5 Questions

1) Define the terms “efficiency”, “offered traffic”, “utilisation" and "network capacity" of a multiple access network, and hence derive equations relating them.

2) A network is built connecting the islands of the Maldives, which are randomly spaced over a distance of 600 km in the Indian Ocean. The frame size is chosen to be 1500 bytes (the maximum size of an Ethernet frame), and the transmission speed is 1 MBit/s. Two possible multiple access schemes are being considered: token-passing and CSMA. Which is likely to be the better choice? Is there a better alternative?

3) Imagine a polled network working quite happily with five nodes, then another two nodes are switched on, and want to join in. Suggest how these new nodes could make themselves known to the master node, and join the network.

4) A polling scheme has 100 nodes, and allows each of them to transmit for 10 ms each second. At small levels of offered traffic, what is the average delay of frames being transmitted across this network? At high levels of delay, if the time required to transmit a packet is uniformly distributed between a t_{packet} of 1 ms and a t_{packet} of 10 ms, what is the utilisation achieved by this protocol?