Ethernet Performance

Adam M. Costello ⟨amc@cs.berkeley.edu⟩

1999-Sep-29-Wed

Efficiency When several stations on an Ethernet have data to send, there are contention periods during which collisions happen and no data is successfully transmitted. We define the efficiency of Ethernet as the percentage of time spent doing frame transmission. When only one station has data to send, there is no contention, frames are sent back-to-back, and the efficiency is 100%.

If large frames are sent, they dwarf the contention periods between them, so the efficiency is near 100%. More interesting is the efficiency when small frames are sent. Many analyses were performed during the 1970s and 80s, all seeming to show that Ethernet is quite inefficient for small frames (the most often quoted figure is 37%), but in 1988 in the paper *Measured Capacity of an Ethernet: Myths and Reality* Boggs, Mogul, and Kent presented measurements in which a real Ethernet exhibited high efficiency for small frames (85–90%). The simplified models used by the analyses had failed to accurately reflect the performance of the true protocol. Two example analyses appear at the end of these notes.

Channel capture effect It was not until 1994 that an explanation was found for the high efficiency. In the paper *A New Binary Logarithmic Arbitration Method for Ethernet*, Mart Molle described the *channel capture effect* (which was independently discovered that same year by Brian Whetton and others, who called it the "packet starvation effect"), which is a consequence of the binary exponential backoff algorithm and was never accounted for in any of the analyses. When a station wins the contention period and succeeds in transmitting a frame, it resets its collision counter to zero, and if it has more data to send, it tries again immediately. The other stations still have nonzero collision counters and random backoff timers, so the winning station is more likely to win the next contention period as well. The longer a station holds the channel, the more collisions experienced by everyone else, the longer their backoff timers, and the more likely that the winning station will continue to hold the channel until it has sent all its data. (If the winner never runs out of data, other stations will eventually get a second chance after their collision counters reach 16 and they give up, moving on to their next frame with a collision counter of 0.)

Although the channel capture effect causes large delays, it does have the advantage of reducing the number of contention periods, because a station that has captured the channel gets to send many frames back-to-back. This explains how Ethernet can be so efficient even for small frames.

BLAM The paper goes on to propose an alternative to the binary exponential backoff (BEB) protocol, called the binary logarithmic arbitration method (BLAM), which retains the high efficiency while eliminating the large delays. Stations deliberately let the winner hold the channel for a bounded burst period, after which the winner voluntarily gives everyone else a chance to use it. The winner is no more likely to win the next contention period than anyone else, because *everyone* resets their collision counter whenever *anyone* succeeds in transmitting a frame. BLAM also reduces the length of the contention periods by having *everyone* increment their collision counters whenever *anyone* experiences a collision. This way, when there are n contenders, only about $\log_2 n$ collisions must happen to make everyone's backoff timers large enough. In BEB, where only the stations involved in the collision increment their counters, the average amount of time until they increment again doubles, and so the length of the contention period grows as the sum of 2^i for $0 \le i < \log_2 n$, which is $O(n)$.

BLAM was in the process of being standardized in 1995 and 1996, but interest fizzled out because more and more Ethernets are switched rather than shared. In switched Ethernets there are no collisions, so there is no use for either BEB or BLAM.

Switched Ethernet Ethernet switches are designed to be drop-in replacements for hubs, and not to require changes to the hardware of the end hosts. That means that the stations attached to the switch are still performing collision detection, even though there are never any collisions. Therefore, even though replacing the hub by a switch allows disjoint pairs of stations to communicate simultaneously without colliding, it does not allow two stations to send to each other simultaneously, because each station will interpret the incoming data as a collision with its own outgoing data.

This is unfortunate, since stations using twisted-pair cables have always used two separate pairs of wires to connect to hubs, one for incoming data and one for outgoing data, so in principle they should be able to send and receive and the same time when connected to a switch. Some newer network interfaces support the *full duplex* option, which basically detects that it is connected to a full duplex switch and disables collision detection.

Faster Ethernets For a long time "Ethernet" meant 10 Mbps, but now a 100 Mbps version is becoming common, and a 1 Gbps version has been developed and should appear on the market in the coming months (see http://www.gigabit-ethernet.org/).

Efficiency analyses Here are two example analyses of Ethernet efficiency. For the first we use a very simplified model in which stations try to send frames only at integer multiples of the *slot time*, which is the total time required to send a frame. For minimal-sized frames, this is not an entirely ludicrous approximation, because every instant is not far from a slot boundary. We approximate the behavior of the random backoff timers as a random process that injects frames at the optimal average rate, and is equally likely to inject a frame in one slot as any other. The efficiency is the fraction of slots in which a successful transmission occurs, which is the probability that exactly one frame is injected into a given slot (obviously there is no transmission if zero frames are injected, and there is a collision if more than one frame is injected). From statistics we know that the number of frames injected into a slot has a Poisson distribution with some mean λ, so the probability that the number of frames in a slot is exactly 1 is $\lambda e^{-\lambda}$. By setting the derivative equal to zero, we find that this probability is maximized when $\lambda = 1/e$, in which case the probability (which is the efficiency) is also $1/e \doteq 37\%$.

In the next analysis we allow stations to transmit at any time. When a real Ethernet station decides it wants to transmit a frame, there is a delay G called the *interframe gap* before it actually starts transmitting, to give the previous transmitter time to switch back to listen mode. During the first G_s of the gap the station is *sensing* the channel to make sure it is idle. If the channel stays idle for the whole G_s period, the station commits to sending a frame, and does nothing for the remainder of the gap ($G_c = G - G_s$). For simplicity, we assume that all frames have the same duration T, and that every pair of stations is separated by a one-way delay of D (a star topology). We also neglect the *persistence* of the real Ethernet MAC protocol—we assume that if the check for idleness fails, the transmission attempt aborts, whereas in reality the station would continuously monitor the channel until it stayed idle for a period of G_s. (Persistence is very difficult to model.) We again approximate the behavior of the random backoff timers as a Poisson process, this time one that is equally likely to attempt to send a packet at any time, and makes attempts at the optimal average rate r.

After a station commits to sending a frame, it doesn't start sending until G_c has elapsed, and the signal doesn't reach other stations until another D has elapsed. Therefore an attempt during a contention period is successful iff there are no other attempts made within $G_c + D = \delta$ either before or after it. (Actually, an attempt can be successful even if that condition is not satisfied, if the competing attempts are nullified because they encounter a non-idle channel. We are neglecting to account for that.) From statistics we know that the probability of having no other attempts in those intervals is $e^{-2\delta r}$. Therefore, the average rate of successful attempts is $re^{-2\delta r}$, so the average waiting time after a successful transmission until the next successful transmission is the reciprocal of that rate, $\frac{1}{r}e^{2\delta r}$. By setting the derivative with respect to r equal to zero, we find that the average waiting time is minimized when $\frac{1}{r} = 2\delta$, in which case the average waiting time is $2e\delta$.

If frames were sent back-to-back, each would require $G + T + D = F$. So the efficiency is

$$\eta = \frac{F}{F + 2e\delta}$$

For 10 Mbps Ethernet $G = 96$ bit times, the minimum $T = 576$ bit times (this is the worst-case traffic load), and the maximum $D = 232$ bit times (this is the worst-case topology). G_s can be anywhere between 0 and 64 bit times, but let's assume the implementation uses the best performing value, 64. So we have $F = 904$ bit times, and $\delta = 264$ bit times, yielding $\eta \doteq 39\%$.

Remember that these results are completely bogus, because they do not account for the channel capture effect, which has a profound impact on efficiency.