Some Utilization Analyses for ALOHA and CSMA Protocols

Norman Matloff
University of California at Davis

May 18, 2000

1 Contention-Based LANs

In **contention-based local-area networks** we have all stations on a common channel, such as a cable or a specific radio frequency, and the stations contend with each other for access to that channel. Only one station can transmit at a time, so if two or more stations attempt to use the channel at the same time (a **collision**), all of their frames will be garbled, and some mechanisms must be devised to (a) detection the problem and (b) to arrange for retransmission.

Here we will be interested in the efficiencies of such networks.

2 Notation and Assumptions

Let S denote the mean number of "original" frames generated at all nodes of a contention-based network (ALOHA, CSMA, etc.) per unit time. By "original" we mean that we are not including retransmitted frames. Then let G denote the mean number of all frames, original or retries, generated per unit time.

Scale time so that a frame takes one unit of time to transmit. Then since we can transmit at most one frame per unit of time, we must have $S \leq 1$. Accordingly, S is the utilization of the channel, that is the proportion of the time that the channel is carrying data which reach their destination intact. (Think about what happens over a time interval of, say, length 10000. The channel will be busy sending frames (not counting retransmits) for $10000 \times S \times 1$ of this time 10000, thus S proportion of that time.)

We will be interested in deriving S as a function of G, and in acquiring some qualitative insight from that function.

Note that $S = GP_0$, where P_0 is the probability that no other frame will be transmitted during the time a given frame, which we will call the **reference frame**, is being sent. (Of course, this includes the case in which a frame overlaps with the reference frame by even a small amount; this still is a collision, and thus still requires retransmission.) So, our task now is to express P_0 in terms of G.

We will assume, as is common in such analyses, that the number of frames transmitted has a Poisson distribution: The probability that k frames are transmitted during interval of length s is

$$\frac{(Gs)^k e^{-Gs}}{k!}, k = 0, 1, 2, ... \tag{1}$$

This also means that the times between successive frames has an exponential distribution with mean 1/G. (It can be proven that this is a property of Poisson processes—event counts have a Poisson distribution if and only if the interevent times are exponentially distributed.)

3 ALOHA

ALOHA was developed at the University of Hawaii by Norman Abramson and others. It consisted of a radio link between stations on several islands.

The protocol was simple: A station would transmit whenever it had data to send. If it were unlucky enough that some other station has data to send around that time (whether earlier or later), and the transmission time intervals of the stations overlap, then of course the destination station would never receive it properly, and thus never send an ACK. The sending station would then timeout, and retransmit.

3.1 Utilization Analysis: Ordinary ALOHA

Let t_0 denote the time our reference frame begins transmission. The transmission will be successful if and only if there are no other transmissions which begin during $(t_0 - 1, t_0 + 1)$. Setting s = 2 (since the interval length is 2) and k = 0 in our Poisson formula,[1] we have $P_0 = e^{-2G}$. So,

$$S = Ge^{-2G}. \tag{2}$$

Setting $\frac{dS}{dG} = 0$, we find that S is maximum when G = 0.5, and that at that time $S = \frac{1}{2e}$. (Note that G is not under our control, so the specific value of G which maximizes S is not so important. We are simply interested in knowing how large S can get, in this case 1/(2e).)

In other words, the best-case utilization of ALOHA is about 18%, rather poor. This of course was the later motivation for ALOHA refinements and other methods.

3.2 Utilization Analysis: Slotted ALOHA

If you take another look at the analysis of ALOHA above, it is clear that ALOHA suffers from the very wide "window of vulnerability" $(t_0 - 1, t_0 + 1)$. The whole idea of slotted ALOHA is to make this window narrower, thus improving performance.

Here transmissions are allowed to occur only at integer time points, 1, 2, 3, ... A frame will be transmitted at t_0 if and only if it became "ready" during $(t_0 - 1, t_0)$ (either it originated during that time, or became ready for retransmission after failing in the past). The same analysis as we have above then yields

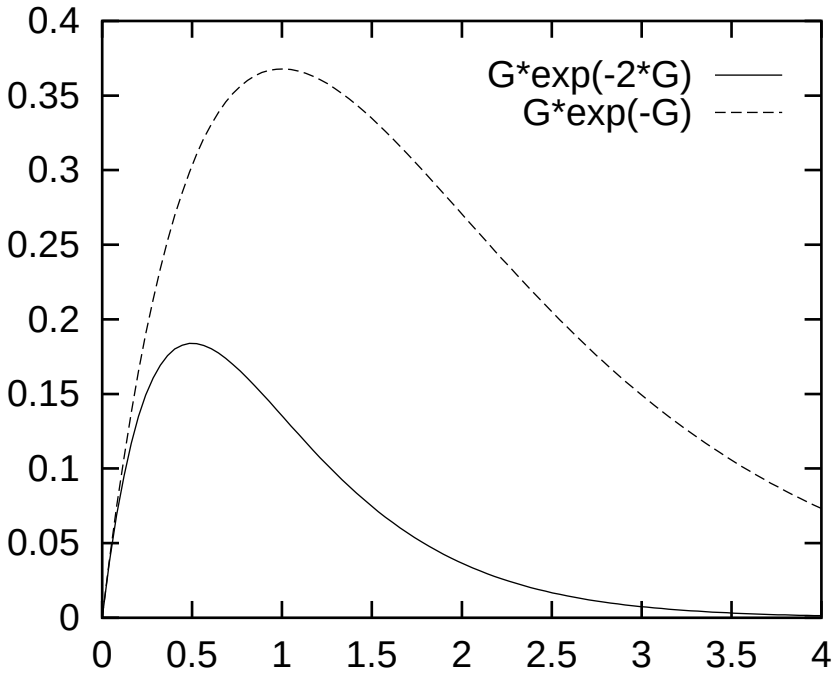
$$S = Ge^{-G}. \tag{3}$$

[1] You may wonder why we do not set k = 1. The point is that we are given that there is a transmission at t_0, so we are actually interested in the conditional distribution of the times of other frames sent, given a transmission at t_0. But those other frames still follow the Poisson process described above. This is a subtle point, whose careful derivation would be well beyond the scope of our course, but can be at least seen in outline form by recalling that the times between successive frames has an exponential distribution. Since the exponential distribution is "memoryless," the chances of having a frame in, say, $(t_0, t_0 + 1)$ is the same, whether or not we have the knowledge that a frame transmission began at t_0.

This gives a maximum utilization of 1/e, double that of nonslotted ALOHA, about 37%.

3.3 Comparison

The relations between S and G for ALOHA and slotted ALOHA are plotted in this figure:



Note that both curves fall off quickly once we pass their peaks. The more load we place on the channel, the less actually gets through. However, slotted ALOHA is clearly the superior performer.

4 CSMA

The Carrier Sense Multiple Access (CSMA) protocol, or more precisely its refinement, Carrier Sense Multiple Access/Collision Detect (CSMA/CD), was developed by Robert Metcalfe at Xerox Corp. (who later founded 3-Com) and others. It is commonly known as Ethernet, and is in very wide usage. When it first became popular, it used as its medium coaxial cable, though these days more complex arrangements are common.

The idea of CSMA is "listen before talk." Before attempting to transmit, a station, say X, will sense the cable for a carrier signal, which, if present, signifies that some other station, say Y, is sending. In such a case, station X reschedules a future retransmission, by generating a random wait time.

This is called **nonpersistent** CSMA. A variation is **1-persistent** CSMA, in which station X continues to listen to the line, and then sends immediately after Y finishes. The problem with 1-persistent CSMA is that a third station, say Z, also wants to send while Y is sending; in this case, X and Z would collide right after Y is done.

Thus a common variation is **p-persistent** CSMA, in which stations X and Z would generate random num-

bers, and transmit right after Y is done only with probability p. If the random number tells a station to wait, it waits—"backs off"—for random amount of time and then tries again. The advantage is that the probability X and Z collide in the above scenario is only p^2. On the other hand, if, say, only Z had been waiting but not X, then Z might back off unnecessarily.

Even if a station with a message to send checks the line and "hears" nothing, a collision may still occur, because another station may be currently transmitting but due to propagation delay its frame may not have arrived yet at the first station. For that reason, the CD mechanism was added to CSMA: While a station is sending, it continues to monitor the line, to check whether its frame is on the line intact. If a collision occurs, both stations will detect this and cease transmission. So, a collision will be detected much earlier than it would if we merely just wait for an ACK (or lack of one), as in the ALOHA case.

4.1 Utilization Analysis: Nonpersistent CSMA

(Adapted from *Modeling and Analysis of Computer Communications Networks*, by Jeremiah Hayes, pub. by Plenum, 1984.)

The line will be idle for a while, then busy for a while (whether with successful transmission or a collision), then idle, then busy, and so on. Let B and I denote random variables representing the lengths of the busy and idle times, respectively. The mean length of a busy/idle cycle will be

$$E(B) + E(I) \tag{4}$$

During a busy/idle cycle, let T be the time spent successfully sending a message. In each busy/idle cycle, there will be either no successful transmissions (the busy period had a collision) or exactly one successful transmission.[2] By definition of busy period, there will be at least one station with something to send. Consider the station which sends first, and let t_0 denote the time it starts, so that it sends during $(t_0, t_0 + 1)$. (Note also that t_0 is the time this busy period starts.) The probability no other station collides with it is, in the same manner we saw for ALOHA,

$$e^{-G\alpha} \tag{5}$$

So,

$$E(T) = 1 \cdot e^{-G\alpha} + 0 \cdot (1 - e^{-G\alpha}) = e^{-G\alpha} \tag{6}$$

The utilization of the line is

$$u = \frac{E(T)}{E(B) + E(I)} \tag{7}$$

Let α denote the ratio of end-to-end propagation delay in the cable to the frame transmission time. Due to our time scaling, α is also the end-to-end propagation delay.

[2]This is where we use the fact that we have nonpersistent CSMA. With persistent CSMA, a second successful transmission could follow right on the heels of a previous one, if the second station sees the first busy and then starts sending right after the first one finishes.

To derive E(B), note that if there is no collision, then B will equal the frame transmit time plus the propagation delay, which on average will be

$$B = 1 + \frac{3}{4}\alpha \tag{8}$$

assuming the stations are uniformly spread out along the length of the cable. On the other hand, if there is a collision, then B will equal

$$B = 1 + \frac{3}{4}\alpha + D \tag{9}$$

where D is defined as follows: Again some station will be the first to start, at time t_0. Then D is the amount of time later that the last station to send in this busy period starts, that is the last station starts at time $t_0 + D$.[3]

We need to find E(B), so we need E(D). To this end, note that we must have $D \leq \alpha$; recall that this is how the collisions occur in the first place—a station thinks the line is free but has not received a transmission in progress yet, due to propagation delay. Now draw a number line showing t_0, $t_0 + D$, $t_0 + x$ and $t_0 + \alpha$, in that order, and you will see why $D \leq x$ if and only if there are no transmissions during the time interval $(t_0 + x, t_0 + \alpha)$. The probability of that event is

$$e^{-G(\alpha - x)} \tag{10}$$

So,

$$P(D \leq x) = e^{-G(\alpha - x)} \tag{11}$$

Thus, the probability density function of D is

$$\frac{d}{dx} P(D \leq x) = \frac{d}{dx} e^{-G(\alpha - x)} = G e^{-G(\alpha - x)} \tag{12}$$

for $0 < x < \alpha$.

Thus

$$E(D) = \int_0^\alpha x \cdot G e^{-G(\alpha - x)} = \alpha - \frac{1}{G}(1 - e^{-G\alpha}) \tag{13}$$

So,

$$E(B) = 1 + 0.75\alpha + \alpha - \frac{1}{G}(1 - e^{-G\alpha}) \tag{14}$$

Finally the quantity E(I) is easily determined as follows. I is the time until the first transmission following a certain time ($t_0 + 1 + \alpha$), so it has an exponential distribution with mean 1/G.

Thus,

[3] Note that here we are using the fact that this is CSMA, not CSMA/CD.

$$u = \frac{E(T)}{E(B) + E(I)} = \frac{Ge^{-G\alpha}}{G(1 + 1.75\alpha) + e^{-G\alpha}} \tag{15}$$

For very small α, we have $u \approx G/(G+1)$, suggesting that CSMA can be very efficient.

4.2 Refinements of Ethernet

Originally the 10 megabits per second transmission speed of Ethernet seemed sufficient. (The token rings at the time sent at the rate of 4 megabits per second.) However, with the advent of much faster protocols such as FDDI, and most importantly the tremendous growth in network applications and need for speed, the original Ethernet speed is considered slow today.

A newer technology is Fast Ethernet, which uses a regular CSMA/CD protocol but increases speed to 100 megabits per second via the following modifications:

- Transmission is on three lines instead of one.
- 8B6T coding is used instead of Machester.
- The clock rate is 25 MHz instead of 20 MHz.

Further refinements have recently led to Gigabit Ethernet, with speed 1000 megabits per second.

Moreover, instead of the older single-cable Ethernet topology, Ethernet hubs and switches have become popular. We will discuss this in another handout later.

5 Simulation Analyses

Many network protocol analyses are mathematically intractable, so that simulation must be used instead. Following is an example of how this is done.

```
 1
 2
 3     /*  Sample simulation program:  ALOHA protocol, with a "p-persistent"
 4         feature added.
 5
 6         There are NNodes network nodes which transmit on the same channel.
 7         Time is slotted, i.e. transmission can begin only at integer times.
 8         If more than one station attempts transmission during a given slot,
 9         they will "collide," corrupting each others' messages, and they
10         must try again.  To avoid repeated collisions, a station which has
11         a frame to be transmitted will do so only with probability P.  Its
12         interface hardware generates a random number between 0 and 1, and
13         transmits only if the number is less than P; otherwise it waits
14         until the next slot and repeats the process.
```

```
15
16          We say a station is ACTIVE if it has something to send; otherwise
17          it is IDLE.  An IDLE station will become active in any given slot
18          with probability NewMsgProb.
19
20          We are interested in the long-run average message delay.   Our
21          approximation to "long-run" will be 10,000 time slots. */
22
23
24      #include <stdlib.h>  /* needed for RAND_MAX */
25
26
27      #define IDLE 0
28      #define ACTIVE 1
29      #define NSLOTS 10000
30      #define MAXNODES 100
31
32
33      int NNodes, /* number of nodes in the network */
34          State[MAXNODES], /* current node states, IDLE or ACTIVE */
35          Delay[MAXNODES], /* delays so far to message, if any, at each node *
36          NMsg, /* number of successfully transmitted messages so far */
37          SumDelay; /* overall total delays accumulated so far */
38
39
40      float P, /* probability that an ACTIVE node will send */
41            NewMsgProb; /* probability that an IDLE node will become ACTIVE */
42
43
44      /* the function Rnd(Prob) simulates a random event of probability
45         Prob, with the return value 1 meaning the event occurred and 0
46         meaning that it did not occur */
47
48      int Rnd(Prob)
49         float Prob;
50
51      {  return((rand() < Prob * RAND_MAX));  }
52
53
54      Init(argc,argv)
55         int argc; char **argv;
56
57      {  int Node;
58
59         sscanf(argv[1],"%f",&NewMsgProb);
```

```
60          sscanf(argv[2],"%f",&P);
61          sscanf(argv[3],"%d",&NNodes);
62
63          SumDelay = 0;
64          NMsg = 0;
65          for (Node = 0; Node < NNodes; Node++)
66             State[Node] = IDLE;
67       }
68
69
70       main(argc,argv)
71          int argc; char **argv;
72
73       {  int Slot,NTry,Node,TryNode;
74
75          Init(argc,argv);
76
77
78          /* simulate the system for NSLOTS time periods */
79          for (Slot = 0; Slot < NSLOTS; Slot++)  {
80             NTry = 0;
81             /* for each node, check whether it has changed from IDLE to
82                ACTIVE, and if ACTIVE (from just now or beforehand) check
83                whether it will attempt to transmit */
84             for (Node = 0; Node < NNodes; Node++)  {
85                if (State[Node] == IDLE && Rnd(NewMsgProb))  {
86                   State[Node] = ACTIVE;
87                   Delay[Node] = 0;
88                }
89                if (State[Node] == ACTIVE)  {
90                   Delay[Node]++;
91                   /* decide whether to transmit */
92                   if (Rnd(P))  {
93                      NTry++;
94                      TryNode = Node;
95                   }
96                }
97             }
98             /* a successful transmission will occur if exactly one node
99                attempted transmission */
100            if (NTry == 1)  {
101               NMsg++;
102               SumDelay += Delay[TryNode];
103               State[TryNode] = IDLE;
104            }
```

```
105         }
106
107         if (NMsg > 0)
108            printf("long-run average delay = %f\n",SumDelay/((float) NMsg));
109         else
110            printf("NMsg = 0\n");
111      }
```