Some Conjectures on the Behavior of Acknowledgment-Based Transmission Control of Random Access Communication Channels

Scott Shenker
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
(Shenker@Xerox.COM)

Abstract: *A class of acknowledgment-based transmission control algorithms is considered. In the finite population case, we claim that algorithms based on backoff functions which increase faster than linearly but slower than exponentially are stable up to full channel capacity, whereas sublinear, exponential, and superexponential algorithms are not. In addition, comments are made about the nature of the quasistationary behavior in the infinite population case, and about how systems interpolate between the finite and infinite number of station cases. The treatment presented here is nonrigorous, consisting of approximate analytic arguments confirmed by detailed numerical simulations.*

Introduction

One way of networking computers is to connect them to a common communication channel and allow them to access the channel whenever they have a message to transmit. The obvious drawback in such schemes is that several stations may transmit simultaneously, creating a *collision* and necessitating retransmission of the garbled messages. The key to efficient random access networks is effective retransmission control algorithms. While the algorithms are exceedingly simple in nature, their behavior has been quite hard to divine[1-13]. In this paper, we use a simplified model of a random access channel, introduced by Goodman et. al: [6], to study a family of transmission control algorithms that resemble the binary exponential backoff of Ethernet [11]. The model is synchronous, in that time is discretized into slots, and it is assumed that the duration of all messages is exactly one slot. Furthermore, the retransmission algorithm is a random process with the following restrictions after a collision, the probability that the message will retransmit on each successive slot is constant until retransmission occurs (yielding a geometric distribution of resend times). This probability of retransmission will depend only on the number of collisions, or backoffs, that particular message has previously experienced.

More specifically, consider n workstations attached to a common bus, with each workstation having a local storage queue. Since the resend probability of a message depends only on the number of its collisions, the entire state of the network can be specified by the number of messages in each station's queue and the number of collisions (or backoffs) the topmost message at each station has had. We will denote these two sets of quantities by $\{q_i\}$ and $\{b_i\}$ respectively. We can characterize the level of activity on the network, the average number of new messages generated per time slot, by a global source strength r. For the purposes of simplicity, we will assume that this load is spread evenly among the stations. The retransmission algorithm is described by a function p(b), the probability of resending a message that has had b collisions. Since messages that have just arrived at the top of a queue will transmit immediately, we always set $p(0) = 1$. The average time until transmission is given by what we will call the backoff function $f(b)$, with $f(b) = [p(b)]^{-1}$. During each time interval, the following happens:

- Each workstation increments q_i by one with probability r/n, representing the generation of new messages.
- Each station i having a nonempty queue ($q_i > 0$) transmits its topmost message with probability $p(b_i)$.
- If only one station transmitted, then the transmission is successful and that station decrements q_i by one and sets b_i to zero. If more than one station transmitted, then there is a collision and the transmitting stations leave q_i unchanged and increase b_i by one.

The behavior of a retransmission or backoff algorithm can be characterized by $Q_n(r)$, defined to be the total average queue length of a system with load r and n stations. $Q_n(r)$ will become infinite at some value of r, called $r_c(n)$. Clearly $r_c(n) \leq 1$, since the transmission rate can never exceed the channel capacity of one. The main focus in this paper is not on quantitative aspects of $Q_n(r)$ nor on the exact value of $r_c(n)$. These properties will depend on the details of the system, so quantitative results from simplified models such as the one considered here will not apply to the more general situation. Instead, we will focus on the qualitative question of whether $r_c(n) = 1$, or $r_c(n) = 0$, or $0 < r_c(n) < 1$. We will identify those functions $p(b)$ that have $r_c(n) = 1$ and thereby allow the channel to operate at full capacity. Given the qualitative nature of the issue, it is hoped that the results derived from this simple model are relevant to more realistic systems.

In the previous description of the model, we tacitly assumed that there was a finite population of stations. The limit $n \to \infty$ yields the infinite population case where new messages are generated via a Poisson source with strength r, and stations have no local queue (since no station ever generates more than one message). Here, the state of the system is completely specified by the set $\{n_b\}$, where n_b is the number of messages that have been backed off b times. This case has quite different behavior than the finite population case (it has, in fact, $r_c = 0$ for all backoff algorithms in the class we are considering) and will be discussed separately.

For qualitative questions about asymptotic stability, which is what we will be discussing, it appears that the only relevant property of the algorithm is the asymptotic behavior of $p(b)$ for large b. We will consider three classes of functions (normalized so that $p(0) = 1$):

Algebraic:	$p(b) = (1 + b)^{-z}$ for some constant $z > 0$.
Exponential:	$p(b) = a^{-b}$ for some constant $a > 1$.
Superexponential:	$p(b) = a^{(1-a^b)}$ for some constant $a > 1$.

The commercially implemented Ethernet transmission control algorithm utilizes binary exponential backoff ($a = 2$), but there are substantial differences (packet dropping after a certain number of backoffs, uniform distribution of resend times, etc.) between the commercial implementation and the algorithms presented here. Algebraic backoffs were first discussed in [6]. Another similar algorithm that is commonly referred to in the literature is the Aloha algorithm[1], where $p(0) = 1$ and $p(b) = \varepsilon < 1$ for all $b > 0$.

Before proceeding, a quick word about terminology is in order. An overloaded system is *transient* or *unstable*, in that eventually the system will tend toward an infinite total queue. An underloaded system, one with asymptotically finite queues, is *recurrent* or *stable*. We often encounter situations where a system is asymptotically transient, but for long times appears to be recurrent. These cases will be referred to as *quasistationary* or *metastable* states. The rigorous formulation of the concept of metastability in this context is not yet available, but it will be assumed that there is indeed a well defined quasistationary total queue length $\underline{Q}(r)$.

In the first section we will consider the finite population case. Approximate analytic arguments will be made for the optimal stability ($r_c(n) = 1$) of algebraic backoff algorithms with $z>1$, followed by supporting numerical evidence. Recently, D. Aldous and R. Fresnedo have formulated similar conjectures in a rigorous fashion and have made substantial progress toward a proof. This treatment also predicts that $0<r_c(n)<1$ for sublinear algebraic ($z<1$) and exponential algorithms, and $r_c(n)=0$ for superexponential algorithms. These results hold for _any_ finite number of stations. The second section will describe the quasistationary states of the infinite station case, where there is a state that has (in the quasistationary sense) full throughput (equal to r) but infinite queue. We conclude with conjectures on how systems interpolate between the finite and infinite number of station cases. In particular, we claim that in the limit of large n, the superlinear ($z>1$) algebraic and the exponential algorithms have stable behavior for r below some finite critical value, which is somewhat counterintuitive given the fact that the Poisson model is never stable.

Finite Population of Stations

General Theory

For the finite number of stations case, it has been shown by Goodman et. al.[6] that $r_c(n)>0$ for binary exponential backoffs. Numerical simulations in the same paper with $n=2$ suggested that the linear ($z=1$) algorithm is stable up to full channel capacity ($r_c(2)=1$), whereas the binary exponential backoff has $r_c(2)<1$. Furthermore, it is known that the Aloha algorithm, where $p(b)$ is constant for $b>0$, has $0<r_c(n)<1$[13]. These results are consistent with the *folk wisdom* that an efficient backoff algorithm should back off fast enough to avoid collisions but not so fast as to waste too much time. In this section we will try to make this notion more precise.

Under a heavy load ($r \approx 1$) the system exhibits an oscillatory behavior consisting of either one or two phases. The first phase is the *dumping* phase. One station, the dumping station, has a large queue with its backoff counter set at zero, and is sending messages every time step. The other stations, the nondumping stations, have increasing queues with large backoff counters. Whenever one of the nondumping stations attempts to transmit, its message will collide with a message from the dumping station. Since the backoff counter at the dumping station is low (usually only one), and the backoff counter of the nondumping station is high (rendering the probability of resending low), the dumping station's message will usually *win* the collision. By *winning* we mean that it will resend its message before the other station, thereby having a successful transmission, resetting its backoff counter to zero, and regaining use of the channel. Thus, the backoff counters and the queues of the nondumping stations increase while the queue of the dumping station decreases. If the dumping station is able to exhaust its queue before losing a collision, we enter the *idle* phase. The dumping station, now having an empty queue, is sending out newly generated messages at a rate of r/n and otherwise is idle. The other stations are not sending, and will not send for a time proportional to $p(b_i)^{-1}$. Eventually, one of the nondumping stations starts sending and we are back in a dumping phase, except we now have a different station in the role of the dumper. There is one phase with very high throughput (close to channel capacity) and another phase with a low throughput of r/n. The relative durations of these phases will determine the system's asymptotic throughput. If the original dumping station does not exhaust its queue before relinquishing the channel, that is if it loses a collision while dumping, then there is no idle phase and the system goes straight from one dumping phase to another.

If $p(b)$ decreases rapidly, then the long queues will have a chance to dump completely. If $p(b)$ decreases slowly, then the dumping phase will be cut short. For instance, in the Aloha case the probability that the dumping queue will lose a collision is constant, independent of the backoff counter of the nondumping station, so that the average number of messages it dumps before it loses a collision and relinquishes the channel is bounded above. To get a more precise estimate of how fast one must backoff in order to have a finite probability of dumping an essentially infinite queue completely, we must compute several quantities. First let us consider a station attempting to transmit a newly generated message, which we artificially block (i.e., we ensure, by fiat, that each of its attempted transmissions results in a collision) for a period of time t. We are interested in the value of the backoff counter after this time t. An approximation to this value, call it B(t), is given by the largest integer satisfying the following inequality (for a more exact analysis, see Kelly[8] and Goodman et. al.[6]).

$$t \geq \sum_{c=0}^{B(t)-1} [p(c)]^{-1}$$

In the three cases under consideration, the function has the following asymptotic behavior (where ~ denotes that the ratio of the two quantities is finite and bounded away from zero as the implied parameter, t in this case, approaches its limit; ≈ will be used to denote approximate equality without any guarantee that the approximation is asymptotically correct within a multiplicative constant):

Algebraic:	$B(t) \sim t^{1/(z+1)}$
Exponential:	$B(t) \sim \log(t)$
Superexponential:	$B(t) \sim \log(\log(t))$

Next, consider a quantity δ(q), defined to be the duration of the dumping phase given that the dumping station starts with a queue of length q, and has no new messages generated during the dumping. To approximate δ(q), first consider the two station case when an old message with backoff counter b-1 collides with a new message having a backoff counter of zero. Define W(b) as the probability that the older message, now having backoff counter b, wins the collision. If we ignore the case where the messages collide again, then W(b) can be approximated as

$$W(b) \approx \sum_{m=0}^{\infty} p(b)[1-p(b)]^m [1-p(1)]^{m+1} = \frac{p(b)[1-p(1)]}{p(b)+p(1)-p(1)p(b)}$$

Returning to the n station case, and making the further approximation that each of the n-1 nondumping stations experience B(q) collisions with the dumping station's messages during the dumping phase, then

$$\delta(q) \approx \sum_{d=1}^{B(q)-1} [p(d)]^{-1} \prod_{c=1}^{d} [1-W(c)]^{(n-1)}$$

Since $p(b) \to 0$ as $b \to \infty$, the product can be approximated by an exponentiated sum

$$\prod_{c=1}^{d} [1-W(c)]^{(n-1)} \approx e^{-\frac{(n-1)(1-p(1))}{p(1)} \sum_{c=1}^{d} p(c)}$$

If p(b) is summable, then δ(q) diverges as $q \to \infty$. Otherwise, δ(q) reaches some finite limit as $q \to \infty$ (to be precise, there is a more complicated condition on how fast the sum of p(b) must diverge in order for δ(q) to reach a finite limit). Thus, we have the following behaviors as $q \to \infty$:

Algebraic:	$z<1$	$\delta(q) \to$ constant
	$z>1$	$\delta(q) \sim q$
Exponential:		$\delta(q) \sim q$
Superexponential:		$\delta(q) \sim q$

Thus, algorithms that backoff faster than linearly have a finite chance to completely dump an essentially infinite queue. Algorithms that backoff slower than linearly will never be able to dump an infinitely long queue. The linear case, z = 1, is rather more delicate, with the behavior of δ(q) depending on the number of stations n and on the details of the function p(b).

The duration of the idle phase is closely linked to the function B(t). If we block n-1 stations for time t, the delay before one of them attempts a transmission is given roughly by $D(t) \approx [(n-1)p(B(t))]^{-1}$. The asymptotic behaviors are:

Algebraic:	$D(t) \sim t^{z/(z+1)}$
Exponential:	$D(t) \sim t$
Superexponential:	$D(t) \gg t$ ($= t^a$)

A rough estimate of the asymptotic throughput can be obtained by calculating the time the system spends in its various phases. Consider a system starting with the dumping station with queue q and backoff at zero. Then, the time that is spent in the dumping phase is just $T_d \approx \delta(q/(1-r/n))$. The extra multiplicative factor $(1-r/n)^{-1}$ is due to the messages generated during the dumping phase. The time spent resolving collisions that occur during the dumping phase is merely proportional to the backoff counter on the nondumping stations; $T_c \approx B(T_d)$. The duration of the idle phase, if indeed we are in a case where there is an idle phase, is given by $T_i \approx D(T_d)$.

In the case of a sublinear algebraic backoff algorithm, there is no idle phase and the throughput of the system for a cycle starting off with a queue length of q is given by

$$E(q)=\frac{T_d-T_c}{T_d}$$

The throughput of the system for a cycle which has an idle phase is:

$$E(q)=\frac{T_d-T_c+(r/n)T_i}{T_d+T_i}$$

One can write a differential equation for the dynamical behavior of the queue length,

$$\frac{dq}{dt}=r-E(q)$$

that, while a drastic oversimplification of the process, seems to capture the essential ingredients. Define Q(r) to be the limit of q(t) as $t \to \infty$.

Plugging the various expressions (not just the asymptotic large q values) into these two throughput formulae, we find that for the algebraic and exponential cases, the function E(q) is monotonic in q. For $r < E(\infty)$, Q(r) will be finite and the system is stable. For $r > E(\infty)$, q(t) will increase linearly in time and the system is unstable. Thus, when E(q) is monotonic, $r_c = E(\infty)$. The superexponential E(q) is not monotonic, and will be discussed later.

The asymptotic throughput for large q is

Algebraic:	$z<1$	$E(q) \to \text{constant} < 1$
	$z>1$	$1-E(q) \sim q^{-1/(z+1)}$
Exponential:	$E(q) \to \text{constant} < 1$	
Superexponential:	$E(q) \to 0$	

In the algebraic case, with $z<1$, E(q) has a maximum value less than one. The reason for the less than optimal throughput is that the algorithm does not back off quickly enough. A very long queue never gets to dump completely; it is always eventually cut off by one of the nondumping stations *winning* one of the collisions in spite of its higher backoff count. When the average number of messages a station can dump before losing a collision is bounded above, the time spent resolving collisions becomes a finite fraction of the dumping time, so the maximal throughput is always less than optimal. For $z>1$, this problem disappears, and the throughput asymptotically approaches 1 as q increases, yielding $r_c(n)=1$. If one plugs in the expression for E(q) into the differential equation, one finds that

$$q(t) \sim t^{\left(\frac{z+1}{z+2}\right)} \qquad for\ r=1$$

and

$$Q(r) \sim (1-r)^{-(z+1)} \quad for\ (1-r) \ll 1$$

These power laws, as will be discussed in the numerical results section, provide an important test of the validity of the preceding calculations.

The linear case is special and the above approximations no longer give exact asymptotic results. However, the qualitative nature of the results may still apply. If we define a family of functions $p(b)=(2+(b-1)/\kappa)^{-1}$ for $b>0$ and $p(0)=0$, then the treatment here indicates that there will be a function $\kappa_c(n)$ such that for $\kappa < \kappa_c(n)$ we have $r_c(n)=1$. The above approximation yields the function $\kappa_c(n) \approx 2/(n-1)$.

In the exponential case, as in the algebraic case with $z<1$, the system has $r_c(n)<1$. When the algorithm backs off this quickly, the idle time becomes a finite fraction of the dumping time, thereby limiting the throughput to be less than one.

In the superexponential case the asymptotic throughput goes to zero for large q. Thus, $r_c(n)=0$. Here, the idle time dominates the dumping time. However, it is to be expected that this lack of stability may be apparent only after long times; the throughput goes to zero only for large q. If one does a more careful calculation of the throughput in this case, one finds a unimodal function and, as in Aloha-like schemes with a Poisson source, there is a metastable solution for small enough r. The system will settle into a quasistationary state until there is a large enough fluctuation to knock the system into the unstable solution.

Numerical Results

The preceding results were obtained with rather cavalier approximations. Numerical simulations, however, support the conclusions. The model simulated was as described in the introduction. For the algebraic case, Tables 1 and 2 give the values of $Q_2(r)$ for $z = 0.5$ and $z = 2.0$; they clearly support the prediction that sublinear algorithms have $0 < r_c(2) < 1$ whereas superlinear algorithms have $r_c(2) = 1$. Similar data was obtained for higher values of n. Figure 1 contains a plot of $\log(Q_2(r))$ vs. $\log(1-r)$ for $z = 2.0$; fitting to a power law yields $Q(r) \sim (1-r)^{-y}$ where $y = 2.89$, in reasonable agreement with the theoretical result of $y = (z+1) = 3$. Figure 2 shows a plot of $\ln(q(t))$ vs. $\ln(t)$ for a two station system with algebraic backoff with $z = 2$; when fit to a power law we find $q(t) \sim t^x$ where $x = 0.737$, in agreement with the prediction of $x = (z+1)/(z+2) = 0.75$. The discrepancies between the theoretical and numerical values of the exponents are smaller than the uncertainties in the numerical values themselves. Similar supporting results were found for $n = 2$, $z = 2.5$ and for $n = 3$, $z = 2.0$. These exponents are of limited value in and of themselves, but they provide a signature for the dynamics of the system: a signature in the sense that it reflects the details of the dynamics, not just some qualitative macroscopic quantity. The fact that the experimental and theoretical signatures are in agreement gives us more confidence in the arguments presented here and in the claim of optimal throughput in the superlinear algebraic case. This is especially crucial to note since simulations for large n with nearly full loads are not feasible (equilibration for $n > 5$ and $r > .99$ was not possible on the time scales, $\approx 10^7$, simulated here). The simulations by Goodman et. al. [6] that gave the original evidence for the optimal stability of algebraic backoffs involved the case $z = 1$ with $n = 2$ and a constant $\kappa = 1$. Further simulations performed here were consistent with the prediction of the existence of $\kappa_c(n)$. However, overly long equlibration times prevented a quantitative test of the approximate result $\kappa_c(n) \simeq 2/(n-1)$.

Simulations of the exponential algorithms, both here (see Table 3) and in previous publications[6], have indicated that $0 < r_c(n) < 1$. The nature of the dynamics of exponential algorithms in the heavily loaded regime is noticeably different than the dynamics of algebraic algorithms with $z < 1$. In the sublinear algebraic case, the inefficiency is caused by the fact that long queues never completely dump. There are no long idle periods and the average queue length reaches equilibrium reasonably quickly. In the exponential case, the duration of both the idle and dumping phases are proportional to the queue size, so the system reaches equilibrium very slowly.

Simulations of the superexponential case were consistent with the prediction that $r_c = 0$, and indicated the presence of metastable states with full throughput and finite queues.

Infinite Population of Stations

General Theory

Since it is well known that a throughput of full channel capacity is impossible for the Poisson source multiple access channel, the interesting question here is whether or not the system is stable at all. Aldous[2] has answered this question in a recent preprint which proves that the binary exponential algorithm is always unstable (in fact, he shows that the asymptotic throughput is zero); it appears that this proof can be simply modified to apply to any backoff scheme in the class considered in this paper. However, these are asymptotic results, and simulations have indicated that there are long-lived metastable states[6]. It is the purpose of the following discussion to illuminate the nature of these metastable states, and to relate them to the behavior of the finite station case in the limit of the number of stations becoming large. To do this, I will start by reviewing a mean value analysis of the Poisson case that was first done by Hajek (readers wishing more details should consult the original[7]).

Let us assume that the environment of message transmissions that a particular message sees is largely decorrelated from its own history, and then characterize the equilibrium behavior of the system by $\{m_b\}$ where these numbers are the *average* values of the numbers $\{n_b\}$. We assume that for each b there is a Poisson source of strength m_b generating transmissions (we set $m_0 = r$). The total traffic on the net is then given by a Poisson source of strength $S = \Sigma m_b p(b)$. If we focus attention on some "test message" which is attempting to transmit, we see that the probabilities of a successful transmission and a collision are just e^{-S} and $(1-e^{-S})$ respectively. To find the equilibrium properties, we look for the steady state solution of the following dynamical equation

$$m_b(t+1) = m_b(t) - m_b(t)p(b) + \{1 - e^{-S}\}m_{b-1}(t)p(b-1)$$

which is just $m_b = m_{b-1}(p(b-1)/p(b))(1-e^{-S}) = r(1-e^{-S})^b/p(b)$. There is always the transient solution $S = \infty$, with $m_b = r/p(b)$, which represents the completely jammed case. This was the basis for Hajek's conjecture, which Aldous has proven, that the system is transient for all r. There are other solutions, the metastable or quasistationary solutions, which have finite S. As Hajek noted, the steady state solution, when combined with the expression for S in terms of the m_b's, yields the Poisson source equation $Se^{-S} = r$, which has solutions for $r \leqq e^{-1}$. Thus, metastable solutions exist only when $r \leqq e^{-1}$. This result, in the mean value analysis, applies to any algorithm in the class under consideration. These metastable solutions always have throughput equal to r. The total queue length is given by $\underline{Q} = \Sigma m(b) = r\Sigma(1-e^{-S})^b/p(b)$. When the backoff function is algebraic this is always finite (for finite S), and in the superexponential case it is always infinite. For exponential backoffs, this is finite only if $(1-e^{-S})$ is less than a^{-1}. The only novel point here is that there are some metastable solutions that have infinite queues but, rather surprisingly, still have throughput equal to r.

Note that for the algebraic case, and the exponential case with $a < a_c = (1-e^{-1})^{-1} \approx 1.52$, the value of the queue at the metastable threshold, $\underline{Q}(e^{-1})$, is finite. This means that one can operate near the threshold without incurring large queues. In the other cases, Q(r) becomes infinite below threshold. $\underline{Q}$ is finite at the threshold in those systems which do not exhibit the infinite queue, full throughput metastable state.

Numerical Results

The Poisson model was simulated for each of the classes of backoff functions and the results are in general agreement with the treatment above. For example, for each backoff function there is a value of the load above which the system would quickly overload (with throughput going to zero and total queues becoming infinite). Well below this threshold, the system would appear to have full throughput for the duration of the simulation (which were typically on the order of 10^7 iterations). This critical value of the load is not, as I have described it here, a precisely defined quantity; possibilities for a more precise definition will be discussed later. In this metastable regime, the Hajek prediction of $m_b = r(1-e^{-S})^b/p(b)$ is qualitatively, but not quantitatively, correct. However, the predictions for the fraction of time slots that are idle, successful transmissions, and collisions, which are given by e^{-S}, Se^{-S}, and $1-(1+S)e^{-S}$, were in good agreement with the simulation results (see the last line in Table 4; the agreement depended more on p(1) being small than it did on the asymptotic nature of p(b)). This tends to validate the approximation that the messages can be treated independently.

More specifically, for the algebraic backoff functions, simulations were run for $z = 0.5$ and $z = 2.0$. Below the threshold, the system appeared stable with finite queue and full throughput. The exponential case with $a = 10$ and $r = 0.05$ exhibited metastable behavior with finite queues. For $r = 0.2$, the queue length diverged while still maintaining full throughput, verifying the prediction above of this unusual state (see Table 5). In the superexponential case, below threshold the metastable state always had diverging queue lengths and full throughput. Also, simulations were consistent with the prediction, in certain cases, of a discontinuity in $\underline{Q}(r)$ at the metastable threshold (see the last line in Table 6, where the queues are quite small even when r is within 10% of threshold).

Discussion

The conjectures of the first section of this paper give a relatively complete classification scheme for algorithms with a finite population of stations. The sweeping applicability of Aldous' result and Hajek's analysis leave little else to be understood about the Poisson case. These two sets of results are very different, so one is left with the issue of how the results for finite n cross over into the results for infinite

n. Since all of the algorithms under consideration are unstable in the Poisson case, are they unstable in the large but finite n case? What is the behavior of $Q_n(r)$ and $r_c(n)$ as $n\to\infty$? There is no mystery for superexponential algorithms, where $r_c(n)=0$ and $Q_n(r)=\infty$. For $n\to\infty$, simulations indicate that sublinear algebraic algorithms have $r_c(n)\sim 1/n$ and $Q_n(r)\to\infty$. However, the simulations suggest that for superlinear algebraic and exponential algorithms there is a threshold R_c such that in the limit of $n\to\infty$ (see Table 6):

$$Q_n(r)\to\underline{Q}(r), \qquad r<R_c$$
$$Q_n(r)\to\infty, \qquad R_c<r$$

For the exponential case we expect $r_c(n)\to R_c$. This will clearly not be true in the $z>1$ algebraic case since $r_c(n)=1$ for all n. For both the superlinear algebraic and exponential algorithms, the dynamics of the system in the finite n case (as measured by the queue length, the number of attempted transmissions per time slot, and the fraction of time slots that are idle, successful transmissions, and collisions) appears to converge to the metastable Poisson dynamics when n becomes large (see Table 4). Therefore, this quantity R_c can be considered the aforementioned threshold for the Poisson model; when $r<R_c$, the Poisson model exhibits the "normal" finite queue metastable behavior. Note that the algorithms that still have stability in the $n\to\infty$ limit are those that, in the finite n case, have the ability to dump long queues. This allows them to recover from the fluctuations that drive the Poisson model unstable.

There are two aspects to the large n results here. The more theoretical one is that for the superlinear algebraic and exponential algorithms, the limit $n\to\infty$ provides precise definitions of Poisson metastable quantities, such as the queue $\underline{Q}(r)$ and threshold R_c. The more practical point is that both the superlinear algebraic and exponential backoff algorithms remain stable in the limit of large n, even though the Poisson model is never stable. There has been much interest in algorithms that are stable in the Poisson model, not just for purely intellectual reasons, but also because it was tacitly assumed that Poisson behavior was a good indicator of large n behavior. That has been confirmed here, but in a different sense; Poisson metastability, not just stability, is sometimes a good predictor for large n stability. When $r<R_c$, systems with large but finite n will have behavior resembling that of the Poisson metastable state, but, for the cases we have outlined here, the finite system will be truly stable. Whatever paradox this poses can be resolved by noting that to describe the limit of large n behavior, one takes the $t\to\infty$ limit first, then the $n\to\infty$ limit; the asymptotic behavior of the Poisson model is described by taking the limits in reverse order. The difference in stability between large but finite n and the Poisson model merely reflects the fact that these two limits do not commute.

Acknowledgments:

I would like to thank A. Greenberg for several fruitful discussions, and D. Aldous and R. Fresnedo for freely sharing their results and insights prior to publication.

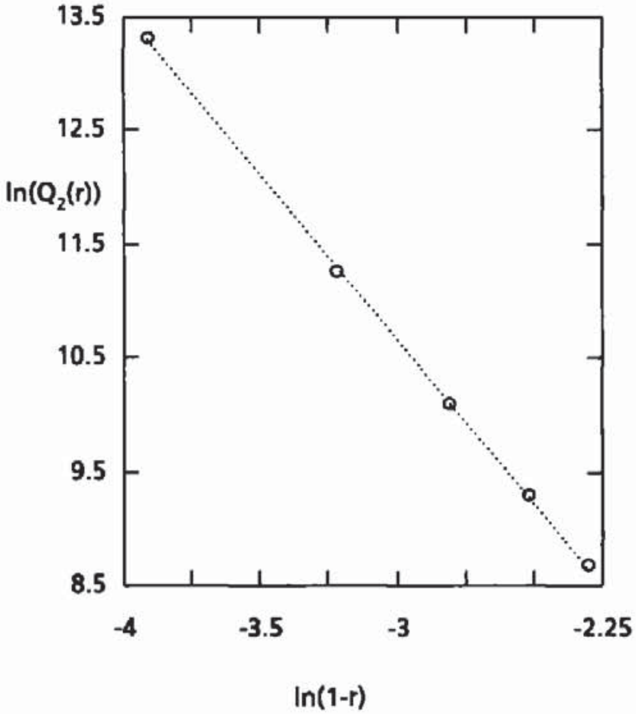

Figure 1: $\ln(Q_2(r))$ vs $\ln(1-r)$ for $z = 2.0$ in an algebraic backoff function. The slope of the least-squares fit line is -2.89.

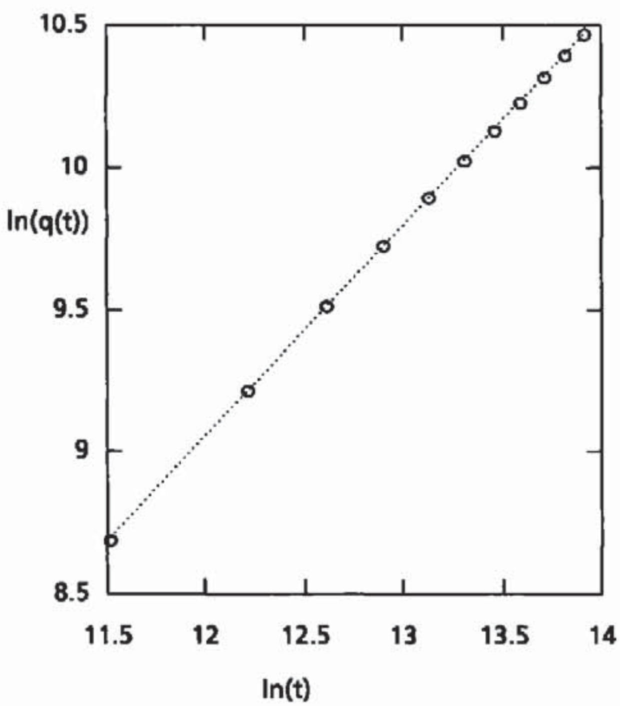

Figure 2: $\ln(q(t))$ vs $\ln(t)$ for an algebraic backoff function with $n = 2$, $z = 2.0$. The slope of the least-squares fit line is 0.737.

r	$Q_2(r)$
0	0
.05	.0045
.1	.02
.15	.052
.2	.11
.25	.211
.3	.386
.35	.708
.4	1.34
.45	2.79
.5	7.27
.55	28.65
.57	57.67
.59	133
.60	225
.61	459
.62	1939

Table 1: $Q_2(r)$ for algebraic backoff with $z = 0.5$. For $r > .63$, the value of Q kept increasing with time, indicating that $.62 < r_c(2) < .63$. The statistical uncertainties in the data are less that 10%.

r	$Q_2(r)$
0	0
.1	.044
.2	.31
.3	1.4
.4	6.5
.5	26
.6	79
.65	134
.7	241
.75	420
.8	785
.85	1957
.9	5921
.92	10987
.94	24542
.96	78413
.98	617020

Table 2: $Q_2(r)$ for algebraic backoff with $z = 2.0$. The statistical uncertainties in the data are less than 10%.

r	$Q_2(r)$
0	0
.05	.0058
.1	.028
.15	.082
.2	.2
.25	.5
.3	1.2
.35	4.2
.4	24
.425	184

Table 3: Values of $Q_2(r)$ for exponential backoff with $a = 2.0$. The statistical uncertainties of the values are quite large, due to the extreme fluctuating nature of the dynamics. The value of $r_c(2)$ is difficult to estimate, but the system is clearly unstable for $r > .6$.

n	$Q_n(.2)$	S	Idle	Suc. Trans.	Col- lisions
2	.31	.227	.796	.200	.014
3	.42	.239	.781	.200	.019
5	.50	.249	.776	.200	.024
10	.55	.261	.771	.200	.029
20	.56	.266	.768	.200	.032
30	.55	.269	.768	.200	.032
100	.54	.271	.767	.200	.033
Poisson	.54	.275	.765	.200	.035

Table 4: Properties of algebraic backoff with $z = 2.0$ for various values of n. The last row contains the Poisson results. For the averages in the columns, S denotes the total number of attempted transmissions per time slot, and the next three columns give the fraction of time slots which are idle, successful transmissions, and collisions. The statistical uncertainty of $Q_n(.2)$ is less than 5%, the other quantities have uncertainties closer to 1%. The simulations were run for 10^6 iterations.

Q	S	Idle	Suc. Trans.	Col- lisions
103.8	.2625	.767	.200	.033

Table 5: Properties of Poisson exponential backoff with $a = 10.0$ and $r = 0.2$. Q denotes total queue length, S denotes the total number of attempted transmissions, and the next three columns give the fraction of time slots which are idle, successful transmissions, and collisions. The simulation was run for 10^7 iterations. For longer times the total queue increased but the other numbers were unchanged.

n	$Q_n(.1)$	$Q_n(.2)$	$Q_n(.3)$	$Q_n(.4)$
2	.044	.31	1.4	6.55
3	.056	.42	2.4	14.3
5	.069	.50	3.1	25.7
10	.073	.55	3.7	53
20	.076	.56	3.5	130
30	.077	.55	3.5	428
Poisson	.076	.54	3.5	∞

Table 6: Values of $Q_n(r)$ for several values of r with varying n for algebraic backoff with $z=2.0$. Further simulations showed that $R_c \approx 0.33$. The statistical uncertainty of the values is less than 10%.

References:

[1] Abramson, N., "Packet Switching with Satellites", *AFIPS Conference Proceeding* NCC 42 (June 1973) 695-703.

[2] Aldous, D., "Ultimate instability of exponential backoff protocol for acknowledgment-based transmission control of random access communication channels" preprint 1986.

[3] Aldous, D., Fresnedo, R., Private Communication

[4] Capetanakis, J. I., "Tree algorithms for packet broadcast channels", *IEEE Transactions on Information Theory*, IT-25, 5 (Sept. 1979) 505-515.

[5] Gallager, R. G., "Conflict resolution in random access broadcast networks", *Proceedings of the AFOSR Workshop in Communication Theory and Applications*, Provincetown, Mass. (Sept. 1978), 74-76.

[6] Goodman, J., Greenberg, A. G., Madras, N., March, P., "On the stability of Ethernet", Proceedings of the Seventeenth Annual ACM Symposium on the Theory of Computing, Providence, Rhode Island (May 1985), 379-387.

[7] Hajek, B., "Acknowledgment-based random access transmission control: an equilibrium analysis", *IEEE International Conference on Communications*, Philadelphia, Penn. (June 1982), 1C1.1-1C1.7.

[8] Kelly, F. P., "Stochastic models of computer communication systems", *Journal of the Royal Statistical Society (B)* 47, to appear.

[9] Kleinrock, L., Tobagi, F. A., "Packet Switching in radio channels: Part I - Carrier sense multiple access modes and their throughput-delay characteristics", *IEEE Transactions on Communications*, COM-23 (Dec. 1975) 1400-1416.

[10] Lam, S. S., "A carrier sense multiple access protocol for local networks", *Computer Networks*, 4 (Feb. 1980), 21-32.

[11] Metcalfe, R. M., Boggs, D. R., "Ethernet: Distributed packet switching for local computer networks", *Communications of the ACM*, 19 (July 1976), 395-404.

[12] Tobagi, F. A., Hunt, V. B., "Performance analysis of carrier sense multiple access with collision detection", *Computer Networks*, 4 (Oct. 1980) 243-249.

[13] Tsybakov, B. S., Mikhailov, V. A., "Ergodicity of a slotted Aloha system", *Problems of Information Transmission*, 15 (April 1980).