# DELAY ANALYSIS OF A WINDOW TREE CONFLICT RESOLUTION ALGORITHM IN A LOCAL AREA NETWORK ENVIRONMENT*

*George C. Polyzos and Mart L. Molle*

Department of Computer Science
UNIVERSITY OF TORONTO
Toronto, Canada M5S 1A4

*Abstract* — **Expressions are found for the throughput and delay performance of a Tree Conflict Resolution Algorithm that is used in a Local Area Network with carrier sensing (and possibly also collision detection). We assume that Massey's constant size window algorithm is used to control access to the channel, and that the resulting conflicts (if any) are resolved using a Capetanakis-like preorder traversal tree algorithm with $d$-ary splitting. We develop and solve functional equations for various performance metrics of the system and apply the "Moving Server" technique to calculate the main component of the delay. Our results compare very favorably with those for CSMA protocols, which are commonly used in Local Area Networks that support sensing.**

## 1. INTRODUCTION

Many Local Area Networks (LANs) can be modeled as a collection of nearby stations (i.e., host computers, work stations, servers, gateways, etc.) connected to a common broadcast channel through which they can exchange messages ([1,2]). In a broadcast channel, each message sent by one station will be received by every other station. However, if the arrivals of two or more messages overlap in time at a single receiver, then we say that a *collision* has occurred, and assume that none of those messages is received correctly. In the case of LANs, it is usually assumed that the data rate of the channel is very high, but the channel can only span a limited distance. The limited distance assumption is a key feature in distinguishing LANs from other similar systems because it offers the possibility of improving the scheduling efficiency of the protocol through the use of *carrier sensing* (where we assume that by monitoring the channel a quiescent station can determine whether or not the channel is currently idle) and possibly also *collision detection* (where we assume that an active station can transmit and monitor the channel at the same time to determine whether or not it is participating in a collision). Of course, perfect scheduling is still impossible because of the finite signal propagation rate between the stations: the status information that one station can infer about the rest using carrier sensing and collision detection is always slightly out of date.

Access to the common channel is controlled by a distributed algorithm known as a random access protocol, such as the well-known CSMA protocols ([3,4]). Here access to the channel is governed by simple rules (such as "transmit as soon as the channel is clear" in the case of 1-Persistent CSMA ([3]) and whenever a collision occurs, each of the affected messages is rescheduled for a later time according to some random "backoff" algorithm. In the case of Ethernet ([1]), for example, the backoff delay is chosen uniformly between 0 and $2^{\min\{k,8\}}$ times the worst-case propagation time following the $k$th collision for this message.

The throughput and delay performance of various CSMA protocols has been studied extensively in the literature, typically by assuming either an infinite population model in which the combined traffic "in equilibrium", due to new messages and scheduled retransmissions, is represented by a Poisson process (e.g., [3,4]), or an unbuffered station model in which the state of the network is characterized by the number of stations that are ready to transmit a message (e.g., [5]). However, in the absence of dynamic control procedures, it is well known ([6,7]) that CSMA protocols may be unstable and, in particular, that results based on the "strong" Poisson assumption (in which the total traffic on the channel *including retransmissions* can be viewed as a Poisson process) should be regarded with skepticism, especially when the average number of retransmissions per message is significant.

Instead of a CSMA protocol, we assume that a Tree Conflict Resolution Algorithm (TCRA) is used to control access to the common channel. The operation of TCRAs is quite different from CSMA protocols because they resolve conflicts algorithmically, when they occur, through the use of group testing ([8]) instead of just asking the stations to go away for a while and try again. And as a consequence, TCRAs are provably stable for arrival rates below some predetermined value. Each 'step' in the execution of a TCRA only depends on knowing its current state and observing the feedback (i.e., 'idle', 'success' or 'collision' — available to all stations through carrier sensing and/or collision detection) that resulted from the previous 'step'. TCRAs have been studied extensively in the literature under the assumption that the 'cost' of each step (i.e., the durations of empty slots, successful transmissions, and collisions) is the same ([9,10,11,12]).

Although the idea of applying a TCRA to a Local Area Network is not new (e.g., ([13,12,14]), the novelty in our work is in finding an analytical expression for the expected delay. Furthermore, we permit the set of conflicting messages to be partitioned into $d$ groups, $d \geq 2$, following a collision rather than the usual case where $d = 2$ is assumed.* This generalization is important in the LAN environment: because of carrier sensing, the 'cost' (in terms

* This work was supported by the Natural Sciences and Engineering Research Council of Canada under research grant A5517.



* Mathys and Flajolet in [15] have considered $d$-ary splitting in the equal length slot case.

of wasted channel time) of an idle slot will always be less than a collision, and perhaps many times less if there is no collision detection. Thus, it may take less time to endure many (short) idle slots than to risk a further (long) collision. And, indeed, it can be shown ([14]) that if a collision lasts $b$ times as long as an idle slot, then the optimal number of groups satisfies

$$d^* \simeq 1+\sqrt{b}$$

## 2. THE MODEL

We make most of the usual assumptions about the environment, except from the features that come as a result of the availability of carrier sensing and/or collision detection (CS/CD). There is a single error free synchronous broadcast channel. The packets are of fixed length. Stations have the ability to recognize silence (no transmission) and/or collisions on the channel in an early stage and terminate the time slot within an a priori known time interval. The time to detect the absence of transmission, through carrier sensing, is usually called a *minislot* — while the time needed for the transmission of a packet is called a slot. We take the minislot to be our basic time unit, assuming that the slot and the time to detect and recover from a collision can be expressed as integral numbers of minislots. We denote by $a$ the ratio of the time needed for a successful transmission (slot) to the carrier sensing time (minislot) and by $b$ the ratio of the time spent in a collision to the carrier sensing time.† Our definitions for $a$ and $b$ as positive integers do not agree with most of the literature, where they are usually expressed as fractions of a message transmission time. However, these definitions are more natural for our study which is based on discrete time systems. Although our time unit is, throughout the analysis, the minislot, it is more convenient to normalize our final delay results in units of packet transmission time, as we do in all our plots.

The Conflict Resolution Algorithm we consider is the well known $d$-ary (preorder traversal) Tree Conflict Resolution Algorithm (referred to, in the binary case, as the Capetanakis Collision Resolution Algorithm by Massey in [12]):

> If a conflict arises, the active set is partitioned into $d$ subsets according to an addressing scheme. All conflicts in the first subset, if any, are resolved first, recursively. Then, those of the second, and then the third, etc., up to the $d$th subset, are resolved (see also figure 2 for an example in the binary case).

The partitioning can be on the basis of random "coin tosses" or packet arrival times (random addressing), or on the basis of station addresses (deterministic addressing). It is known from the work of Capetanakis ([10]) that in the finite population case, a deterministic addressing scheme performs better than a random one. In the infinite population model, however, the addressing schemes are equivalent from the point of view of throughput or epoch length statistics, but the use of the time of arrival to resolve collisions leads to a FIFO system with lower variance. This method, however, introduces correlation between different parts of the delay, as evaluated in this study, complicating the analysis; we thus avoid it and instead we assume that the addressing scheme is based on "coin tosses".

† The usual channel models for tree algorithms assume "zero" propagation time and/or *common destination*, while most CSMA studies assume any other station as destination is possible. We have decided to follow the first approach so that our results can be reduced (setting $a = 1$, $b = 1$) to the standard case for tree algorithms. Hence, to be fair in our comparisons with other CSMA studies, we incorporate one extra minislot in our packet transmission time. That is, if the ratio of packet transmission time to carrier sensing time is $r$, we set $a = r + 1$.

To complete a random access system, in addition to a conflict resolution algorithm, we require also a channel access algorithm to specify when packets may join a conflict resolution algorithm. We adopt a simple and efficient channel access algorithm proposed by Massey ([12]), the simplified Window Algorithm (see figure 1):

> Assume that at time $t$ (measured in minislots), the conflict resolution algorithm has just completed the transmission(s) of all packets that arrived before $\tau$. (Initially, we set $t \leftarrow 0$, $\tau \leftarrow 0$.) Then at time $t^+ = \max\{ t , \tau+w \}$, the channel access algorithm permits all packets that arrived in the window $( \tau , \tau+w ]$ to be transmitted. After all of these packets have been transmitted successfully (possibly with the aid of the conflict resolution algorithm) using $l$ minislots, $l \geq 1$, we repeat the process after setting $t \leftarrow t^+ + l$, $\tau \leftarrow \tau + w$.

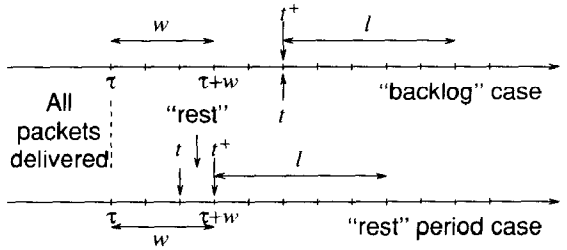


Figure 1: The Window Algorithm: previous epoch ends at $t$, new one begins at $t^+ = \max\{ t , \tau+w \}$.

Notice that the window size, $w$, is assumed *constant*. This is a crucial assumption for the analysis and has as a consequence the existence of "rest" periods — time periods where it would have been safe for the channel access algorithm to permit the start of a new epoch but such an epoch is not started. There is, however, an obvious improvement in delay if the channel access algorithm does not impose "rest" periods, whenever there is no backlog from previous windows, but instead starts an epoch from a "small" window. This second channel access algorithm is called the (non-simplified) Window Algorithm. The two channel access algorithms do have the same maximum throughput ([12]) and their difference in delay at light loads is less than one window, which is less than a packet transmission time for most cases. Obviously, the main reason for our choice of the simplified window algorithm is analytic tractability.

To simplify the description and concentrate on the CS/CD features, we assume a large number of independent unbuffered stations, modeled as an infinite population. The finite non-homogeneous population case without CS/CD has been presented in [16]. The extension of this study to such an environment is straightforward. Since we concentrate on the infinite population model, we assume that the arrival process is Poisson with rate $\lambda$ packets per packet transmission time (or $\lambda/a$ packets per minislot).

### 2.1. An Example

We refer to figure 2. The time axis is (mini)slotted and furthermore divided in windows of length 5 (minislots). The time it takes to serve a window, resolving all conflicts, is called an *epoch*. At the beginning of an epoch all stations may transmit. If at most one transmits, there are no conflicts and the epoch ends there, the window having been served. If more than one station transmits, a conflict arises. At the next slot only stations that "toss" zero are permitted to transmit. If no conflicts arise, at the next slot permission is granted to stations that "tossed" one, otherwise the algorithm continues recursively splitting the set of the original con-

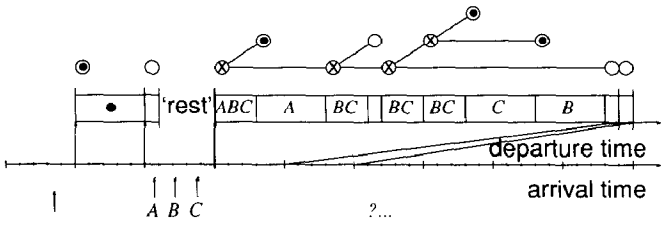


Figure 2: The Channel Access and Conflict Resolution Algorithms ($a = 5, b = 3, d = 2, w = 5$).

tenders in progressively smaller subsets until no collision arises. Eventually, all $d$ subtrees of the initial collision will have the opportunity to resolve their conflicts.

## 3. EPOCH LENGTH

The epoch length, the time required to serve a window resolving all conflicts, if any, is denoted by the random variable $l$. The first performance measure we can obtain is $Q(x,z)$, the generating function of epoch lengths for a window with expected number of packet arrivals $x$, defined by

$$Q(x,z) \triangleq \sum_{n=0}^{\infty} \Pr\{ l = n \mid x \}\, z^n$$

from which we can derive all the desired statistics of $l$. This generating function can be obtained by solving the following functional equation

$$Q(x,z) = z^b\, Q^d(x/d,z) + (z - z^{b+d})\, e^{-x} + (z^a - z^{b+a+d-1})\, x\, e^{-x} \quad (1)$$

The proof of equation (1) is given in Appendix A. The intuitive justification, however, of this functional equation is very easy. The epoch length, in the case of an epoch starting with a collision, is the sum of the lengths of the initial collision and the subepochs of all the $d$ subtrees, the sum being represented by the first term of equation (1), a product in the transform domain. Since the splitting is fair and the population is considered infinite all subepochs are statistically indistinguishable and their generating functions equal, with expected number of packets in each subepoch $1/d$ of the number of the original contenders. However, if there is no collision to begin with, an adjustment is needed. The second term of equation (1) describes the situation in the case of an idle window and the third in the case of a single packet arrival. From the true transform of the epoch, $z$ in the case of an empty window and $z^a$ in the case of a window with a single packet, we subtract the result implied by the first term of the equation, $z^{b+d}$ and $z^{b+a+d-1}$ respectively, and weight these terms by the probabilities of the respective events.

The solution of this functional equation can be obtained in the form of a power series

$$Q(x,z) = \sum_{n=0}^{\infty} q_n(x)\, z^n \quad (2)$$

where the $q_n(x)$ can be determined by substituting (2) in (1) and equating the coefficients of the same powers of $z$. This method was presented in [17] where the distribution of the delay has been obtained in the case of equal slot lengths (i.e., no carrier sensing or collision detection) and binary splitting (which is optimal for this case).

To obtain an expression for the expected packet delay, we will need expressions for the first two moments of the epoch length. The mean epoch length can be obtained as

$$L(x) = \frac{\partial}{\partial z}\Big[ Q(x,z) \Big]_{z=1} \quad (3)$$

Differentiating (1) with respect to $z$, setting $z = 1$, and using (3) we get the following functional equation for the expected epoch length.

$$L(x) = b + d\, L(x/d) - (b+d-1)(1+x)\, e^{-x} \quad (4)$$

The solution of this functional equation in the form of a series is

$$L(x) = \sum_{n=0}^{\infty} \alpha_n\, x^n$$

with

$$\alpha_0 = 1\ ,\ \alpha_1 = a - 1\ ,\ \alpha_n = (-1)^n\, \frac{(b+d-1)(n-1)}{n!\,(1-d^{1-n})}\ ,\ n>1. \quad (5)$$

The second moment can be obtained as

$$H(x) = \frac{\partial^2}{\partial z^2}\Big[ Q(x,z) \Big]_{z=1} + L(x)\,. \quad (6)$$

We can then obtain the following functional equation for the second moment

$$H(x) = b^2 + 2bd\, L(x/d) + d\, H(x/d) + d(d-1)\, L^2(x/d) - (b+d-1)\,[\,(2a+b+d-1)\,x + (b+d+1)\,]\, e^{-x} \quad (7)$$

with solution

$$H(x) = \sum_{n=0}^{\infty} \beta_n\, x^n$$

where

$$\beta_0 = 1\ ,\ \beta_1 = a - 1\ ,$$

$$\beta_n = \alpha_n \left[ (b+d-1) + \frac{2n}{n-1}\,(a-1) \right] + \frac{d^{1-n}}{1-d^{1-n}} \left[ 2b\, \alpha_n + (d-1) \sum_{i=0}^{n} \alpha_i\, \alpha_{n-i} \right]\ ,\ n>1. \quad (8)$$

It should perhaps be emphasized that the expansions of both $L(x)$ and $H(x)$ are convergent alternating series and thus the error of the computation can be made arbitrarily small.

## 4. THROUGHPUT AND STABILITY

Since we deal with an infinite population model, and the conflict resolution algorithm guarantees that by the end of an epoch all packets involved in the original collision have been successfully transmitted, average throughput and input rate, $\lambda$, coincide, as long as the system is stable.

To find the maximum stable throughput of the protocol, we have to consider the conditions under which there is no long term backlog in the system. Intuitively it is easy to justify the following condition for stability:

$$L(w\,\lambda/a) < w\,. \quad (9)$$

This condition essentially states that for the system to be stable, the mean service time for a window, $L(\lambda w / a)$, must be less than the interarrival time between successive windows, $w$. The proof of this stability condition will be obvious after the development of a queueing model for the system, in the next section, from which the delay is obtained.

The above inequality contains two parameters, one explicit, $w$, and one implicit, $d$, that can be tuned to obtain the maximum stability region. This region of course depends on the other parameters of the system as well.

| Maximum Stable Throughput $(w,d)$ | | | |
|---|---|---|---|
| $a$ | $b=a$ | $b=a/2$ | $b=2$ |
| 1 | **.428** (3,2) | | |
| 2 | **.493** (4,3) | **.584** (4,2) | **.493** (4,3) |
| 4 | **.593** (4,3) | **.661** (5,3) | **.661** (5,3) |
| 10 | **.698** (6,4) | **.761** (8,4) | **.830** (10,3) |
| 20 | **.764** (8,6) | **.822** (10,4) | **.907** (19,3) |
| 40 | **.820** (11,8) | **.866** (14,6) | **.951** (36,3) |
| 100 | **.877** (16,11) | **.910** (22,8) | **.979** (89,3) |
| 400 | **.934** (30,21) | **.952** (42,16) | **.994** (349,3) |
| 1000 | **.957** (48,33) | **.969** (64,24) | **.997** (870,3) |

Table 1: Maximum Stable Throughput and the corresponding values of the optimal window size, $w$, and the optimal degree of the tree, $d$, (in parentheses) for some combinations of the packet length, $a$, and collision length, $b$, in minislots.

## 5. DELAY

In this section we obtain expressions for the delay experienced by a packet from the moment of its arrival to a station up to the completion of its successful reception at the destination. To arrive at such expressions we split the total delay, denoted by the random variable $t$, in three independent parts, $t_0$, $t_1$, and $t_2$, that require different techniques to be evaluated. Of course, $t = t_0 + t_1 + t_2$. We define the following random variables:

$t_0$: The time from the appearance of the packet at the station up to the end of the current window (i.e., the first window boundary).

$t_1$: The time from the end of the window of arrival of the packet until the beginning of the epoch in which the packet is transmitted (and also successfully received, since the conflict resolution algorithm guarantees the resolution of all conflicts).

$t_2$: The time spent in the epoch of transmission from the moment of the initial collision, if any, up to the successful transmission of the packet, inclusive.

The initial delay, $t_0$, is merely the residual life of the current window at the moment of the packet arrival. Since we have an infinite population with Poisson arrivals, $t_0$ is distributed uniformly in $[0,w)$. The expectation of the initial delay is thus

$$T_0 \triangleq E\{t_0\} = \frac{w}{2}. \qquad (10)$$

Since, however, we deal with discrete time models for all other measures, it is convenient to have a discrete version of the distribution of $t_0$. The discrete uniform distribution over $[0,w)$ may therefore be used, providing the generating function for $t_0$, seen as a discrete random variable

$$\Theta_w(z) = \frac{1}{w}\sum_{n=0}^{w-1} z^n . \qquad (11)$$

Unfortunately the analysis for the second and third components of the delay is not as obvious. We apply the "Moving Server" technique to obtain $t_1$, and solve a functional equation to obtain $t_2$.

### 5.1. The Lag Between the Departure and the Arrival Time Axes

We have defined $t_1$ to be the time from the end of the window of arrival of the packet until the beginning of the epoch generated by this window. This time is clearly the waiting time in a discrete time queueing system with arrivals at the end of each window and service times equal to the corresponding epoch lengths. Since there is an arrival at the end of each window, the system can be described as a discrete time D/G/1 queue with interarrival time $w$ and service time distribution given by the generating function of epoch lengths, $Q(\lambda w/a)$. The distribution of the waiting time for such a system is known ([18]) but to compute it one needs to find the $w$ complex roots of $Q(\lambda w/a) - z^w$ (which is usually a high order polynomial or a series (!) in the infinite population case) that lie inside (or on) the unit circle. We have applied this method in [16]. Here we avoid this tedious approach and instead we apply the "Moving Server" technique ([19]) to obtain the distribution of $t_1$.

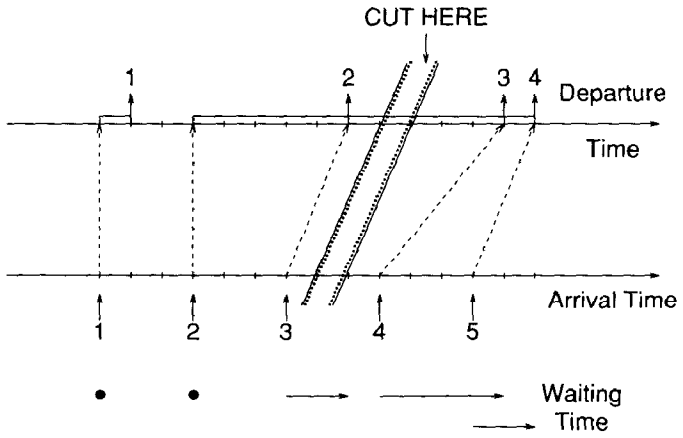


Figure 3: The Transformation to obtain a discrete time M/G/1 model from a D/G/1 queue.

The main idea is to transform the discrete time D/G/1 queueing system to an M/G/1 system which is much easier to solve. Since we only need the (waiting) time in queue (and not also the time in service) to compute $t_1$, we can take advantage of the following transformation, as illustrated in figure 3. Notice that if we "shorten" both arrival and service time axes by removing one minislot from the interarrival time between customers $i$ and $i+1$ and from the service time for customer $i$, *the waiting times for every customer remain exactly the same.* Therefore, since every service time and interarrival time is at least 1 minislot long, we can apply this transformation everywhere. The result is a new system with service times $\tilde{l} = l - 1$, and constant interarrival times $\tilde{w} = w - 1$. However, some service times (i.e., those corresponding to empty windows) are now zero after the transformation, so we can "erase" them if we like. In this case we find that the remaining customers exhibit geometric interarrival times (in units of $\tilde{w}$) and independent service times. The question is now whether this system is any easier to analyze than the original one. In some interesting cases, the transformed system can be analyzed as a discrete time M/G/1 queue. The only condition is a restriction on $l$

$$(l-1) \bmod (w-1) = 0 . \qquad (12)$$

An example of this transformation applied to a D/G/1 system, for which the above condition holds, and the resulting M/G/1 system is shown in figure 4.

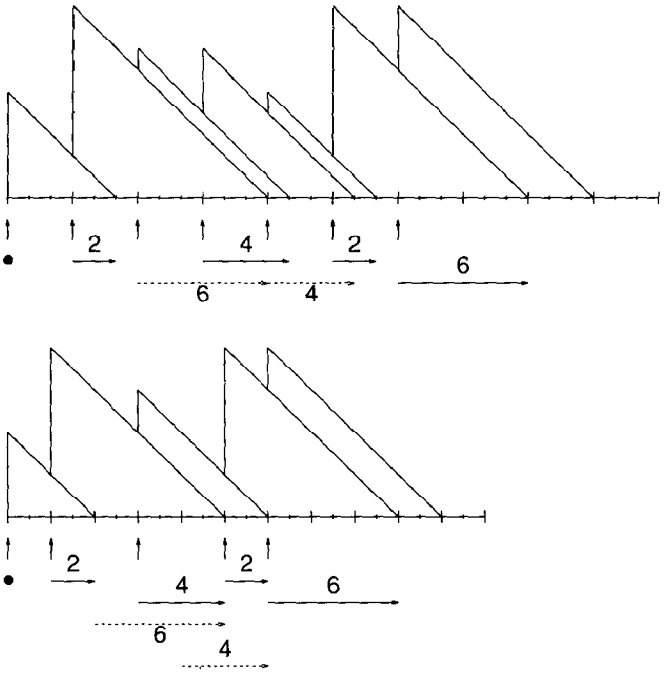


Figure 4: The Transformation of the D/G/1 Model to an M/G/1 Model. Busy Periods and Waiting Times for the original and the transformed systems. Arrivals #3 and #5 of the original system have been "erased" since their service time was just 1 time unit (minislot).

The properties of $d$-ary trees guarantee that the condition for the equivalence of the D/G/1 and the (resulting from the transformation) M/G/1 model (i.e., condition (12)) is satisfied when

$$(a-1) \bmod (w-1) = 0 \quad \text{and} \quad (b+d-1) \bmod (w-1) = 0 . \qquad (13)$$

This condition is sufficient but not necessary. It is obtained through the following argument. If there are $m$ collision slots in an epoch, corresponding to internal nodes of the $d$-ary tree, there should be $(a-1)m+1$ terminal nodes, corresponding to $n$ successful slots and $(d-1)m+1-n$ idle slots. We have then the following equation

$$l = m\,b + n\,a + (d-1)m + 1 - n \quad \text{or} \quad l - 1 = m\,(b+d-1) + n\,(a-1)$$

with $m$ and $n$ integers. This equation together with requirement (12) lead to condition (13).

Condition (13) is not very easy to satisfy. There are, however, combinations of parameters that do satisfy it. One of the most interesting examples is the case with no carrier sensing or collision detection, i.e., $a = 1$ and $b = 1$, for which the optimal parameter choices are $d = 2$ and $w = 3$ (which ironically is easy to solve explicitly by the D/G/1 queue method). Another interesting combination is: $a = 11$, $b = 11$, $d = 5$, and $w = 6$ where $d$ and $b$ are the optimal values leading to maximum throughput of 0.705. Of interest are also suboptimal combinations which differ so little from the optimal that they are practically equivalent; for example the combination: $a = 106$, $b = 106$, $d = 15$, and $w = 16$ leads to maximum throughput equal to 0.880, which is only 0.032 % below the optimal figure. Also, for $a = 101$ and $b = 101$, which corresponds to examples given in the literature ([5,4]), choosing $d = 10$ and $w = 11$ satisfies equation (13) and gives maximum throughput 0.870, just 0.88 % below the 0.878 optimal figure obtained through $d = 11$ and $w = 16$. Of more practical importance is, however, the fact that the results obtained through this method, although not exact for all cases, are very good estimates as has been determined through simulation (for the mean delay — see figure 6).

We proceed now to the calculation of statistics of $t_1$. The distribution of the delay for a discrete time M/G/1 queue is given by

$$\Omega(\zeta) = \frac{[\,1 - p\,B'(1)\,]\,(\zeta-1)}{\zeta - 1 - p\,[\,B(\zeta) - 1\,]} \qquad (14)$$

where $B(\zeta)$ is the (transform of the) distribution of the service time and $p$ is the probability of an arrival. $B'(\zeta)$ denotes the derivative with respect to $\zeta$. In our case $p = 1 - e^{-x}$ where $x = \lambda w/a$ and

$$B(\zeta) = \left[\, Q^{(\bar{1})}(x,z)\, z^{-1} \right]_{z^{w-1} = \zeta} . \qquad (15)$$

$Q^{(\bar{1})}(x,z)$ is the conditional generating function of epoch lengths for non-empty windows. This last equation is nothing more than the application of the transformation we have described, and is meaningful when and only when condition (13) is satisfied. The final result for the distribution of $t_1$ is obtained by changing the time unit to the original one, the minislot, by

$$W(z) = \Omega(\zeta^{w-1}) . \qquad (16)$$

The expectation of the waiting time can be obtained from the discrete time M/G/1 queue as

$$\bar{W} = \frac{\rho}{p}\,\frac{\rho\,(1+C^2) - p}{2\,(1-\rho)} \qquad (17)$$

where $\rho$ is the utilization factor given by

$$\rho = \frac{L(\lambda w/a) - 1}{w - 1} \qquad (18)$$

and $C^2$ is the squared coefficient of variation of the service time. $C^2$ can be expressed in terms of the first two moments of the epoch length as

$$C^2 = \frac{(1-e^{-x})\,[\,H(x) - e^{-x}\,] - [\,L(x) - e^{-x}\,]^2}{[\,L(x) - 1\,]^2} \qquad (19)$$

while $p = 1 - e^{-x}$. The final result, in minislots, is

$$T_1 = (w-1)\,\bar{W} . \qquad (20)$$

### 5.2. The Delay in the Epoch of Transmission

In the same way we obtained a functional equation for $Q(x,z)$ we can obtain such an equation for the generating function of the delay in the epoch of transmission, $G(x,z)$, based on the recursive nature of the conflict resolution algorithm. The equation is

$$G(x,z) = z^b\,G(x/d,z)\,\frac{1}{d}\,\frac{Q^d(x/d,z) - 1}{Q(x/d,z) - 1} + \left[\, z^a - \frac{1}{d}\,z^{b+a}\,\frac{z^d - 1}{z - 1} \right] e^{-x} \qquad (21)$$

and is obtained in Appendix B.

Differentiating equation (21) we obtain the following functional equation for $T_2$, the expectation of $t_2$

$$T_2(x) = T_2(x/d) + \frac{d-1}{2}\,L(x/d) + b - \left[\, b + \frac{d-1}{2} \right] e^{-x} \qquad (22)$$

with solution

$$T_2(x) = \sum_{n=0}^{\infty} \theta_n \, x^n$$

where

$$\theta_0 = a \quad , \quad \theta_1 = -\frac{1}{2}\left[\frac{a-1}{d} + \frac{2b+d-1}{d-1}\right] \quad ,$$

$$\theta_n = \frac{(d-1)\,d^{-n}\,\alpha_n}{2\,(1-d^{-n})} - \frac{(2b+d-1)\,(-1)^n}{n!\,2\,(1-d^{-n})} \quad , n>1 \, . \quad (23)$$

### 5.3. The Total Delay

In the previous sections we have obtained the generating functions for the components of the delay. Notice that these components of the delay are independent random variables since

- $t_0$: depends only on the arrival process in the current window
- $t_1$: depends only on traffic generated before the window of arrival of the (tagged) packet and thus does not depend on $t_0$ or $t_2$, and
- $t_2$: depends only on other packets generated (independently) during the window of arrival of the packet and does not depend on the lag of the algorithm ($t_1$), or the exact arrival moment of the packet in the window ($t_0$), since the participants are determined as soon as the window completes and random "coin tosses" are used for conflict resolution.

The final result, the generating function of the total delay can then be obtained as the product of the generating functions of the three components of the delay:

$$\Gamma_w(\lambda,z) = \Theta_w(z) \;\; W(z) \;\; G(\lambda w/a, z) \, .$$

The mean packet delay is, of course,

$$T = T_0 + T_1 + T_2$$

where the terms are taken, respectively, from equations (10), (20), and (23).

It is now easy to prove that equation (9) is the stability condition. Such a system is said to be stable if the expectation of the delay is finite. In our case we have only to investigate $T_1$, since $T_0$ is bounded and the finiteness of $T_2$ is guaranteed by the convergence of the series solution (equation (23)).† But $T_1$ is finite, and the discrete time queue is stable, whenever $\rho < 1$, which is equivalent to condition (9).

## 6. RESULTS

Figure 5 presents exact throughput-delay curves obtained analytically for some values of the parameters $a$ and $b$ for which the optimal schemes are solvable exactly. The curve for the case with no carrier sensing or collision detection ($a=1, b=1$) is identical to the curve obtained with a discrete time D/G/1 model for $T_1$ (which in this case is easily solvable). In figure 6 exact results are shown again but not for the optimal values of $d$ and $w$.

In figure 7 we plot analytical and simulation results to verify the accuracy of the method when the results are not exact. We have particularly tried to validate the curve with $a = 100$ and $b = 2$ since these are common values for the parameters and condition (13) is far from being satisfied. At the time when the simulation was developed we were considering only binary splitting. This is the reason why simulation results are shown only for the case $d = 2$. The agreement of analysis and simulation is very good, as seen in figure 7. In figure 8 we plot throughput-delay curves for the optimal, under the described protocol, schemes for some interesting values of the carrier sensing and collision detection time parameters. Figures 9 and 10 provide a comparison of throughput-delay performance with previous results for other protocols.

† The series converges since it is alternating and the sequence of its terms, for any given $x$, has limit zero.

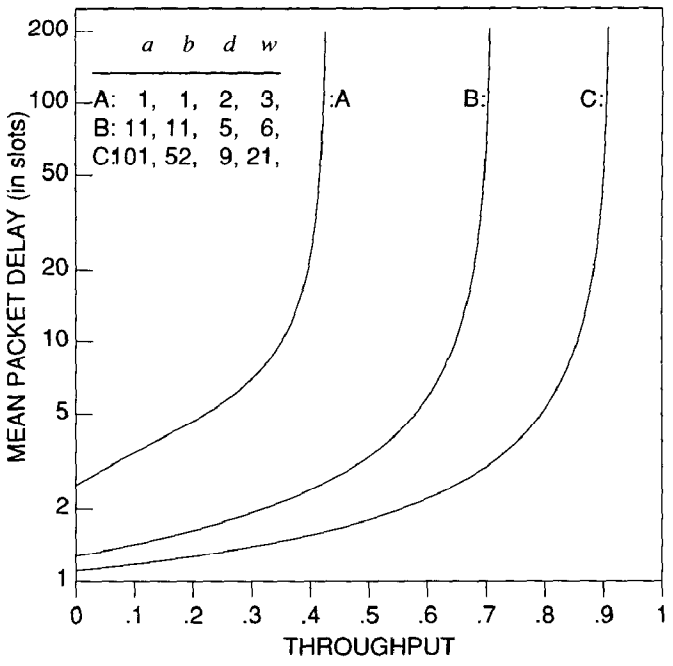


Figure 5: Throughput-Delay Curves. Exact Analytical Results for Optimal Schemes.

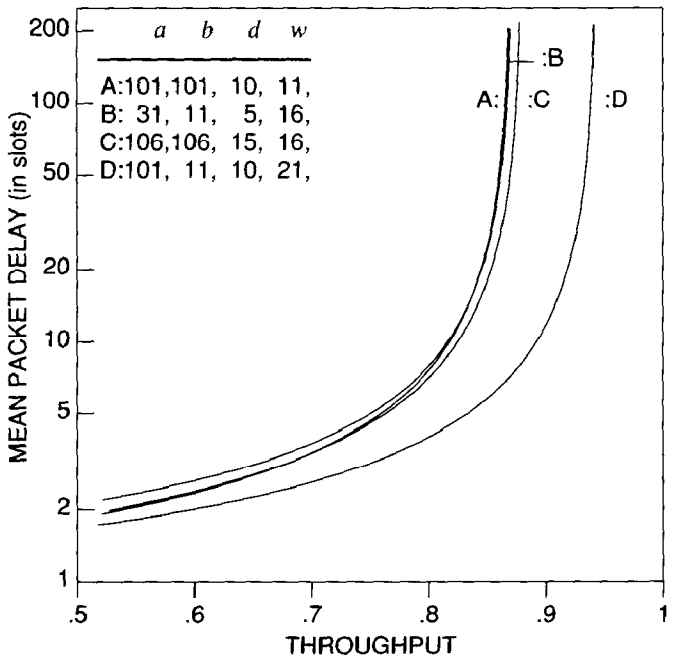


Figure 6: Throughput-Delay Curves. Exact Analytical Results for Suboptimal Schemes.

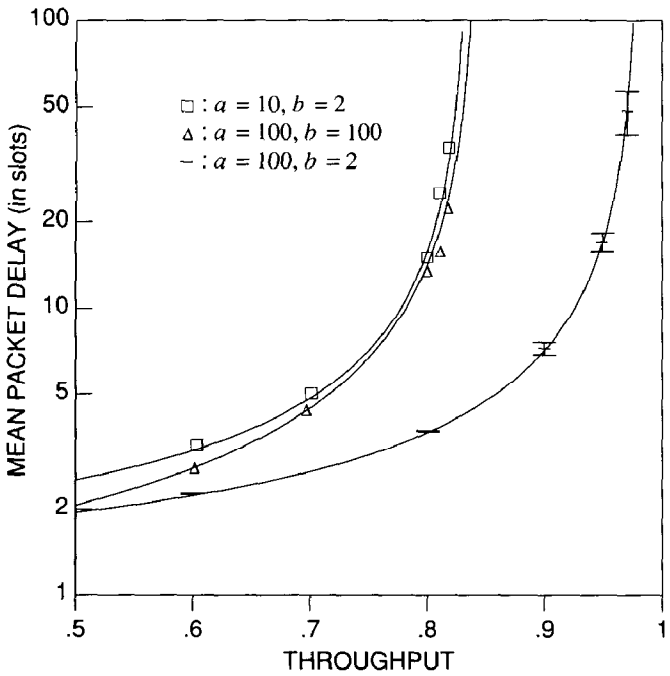


Figure 7: Comparison of Analytical with Simulation Results when condition (13) is violated. 95 % Confidence Intervals on the $a = 100, b = 2$ Curve and (single run) Point Estimates on the $a = 10, b = 2$ (□) and the $a = 100, b = 100$ (△) Curves. All these results use binary splitting ($d = 2$) and the optimal window size which is $w = 83$, 10, and 12 respectively.

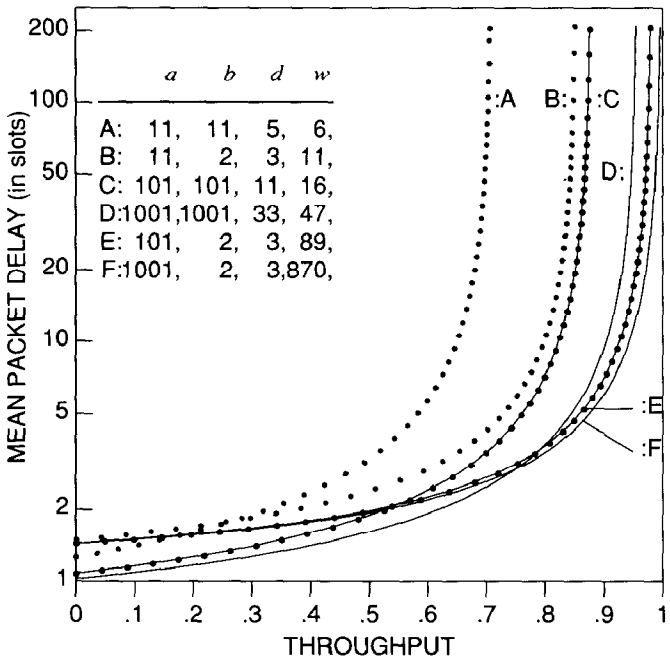


Figure 8: Throughput-Delay Curves for Optimal Schemes. Only curve "A" is exact.

## 7. CONCLUSIONS

We have presented a method to obtain the throughput-delay performance curve for Windowed Tree Conflict Resolution Algorithms in a Local Area Network environment where carrier sensing and/or collision detection might be available. We used a standard preorder traversal tree algorithm for conflict resolution and a constant size window algorithm for channel access. Although more efficient algorithms can be used (e.g. "level skipping" and the "Part-and-Try" algorithm), their use is not so important in this case because the cost of some wasted slots is much lower when carrier sensing and/or collision detection is possible, while the complexity of the more efficient algorithms is much higher for both implementation and analysis.

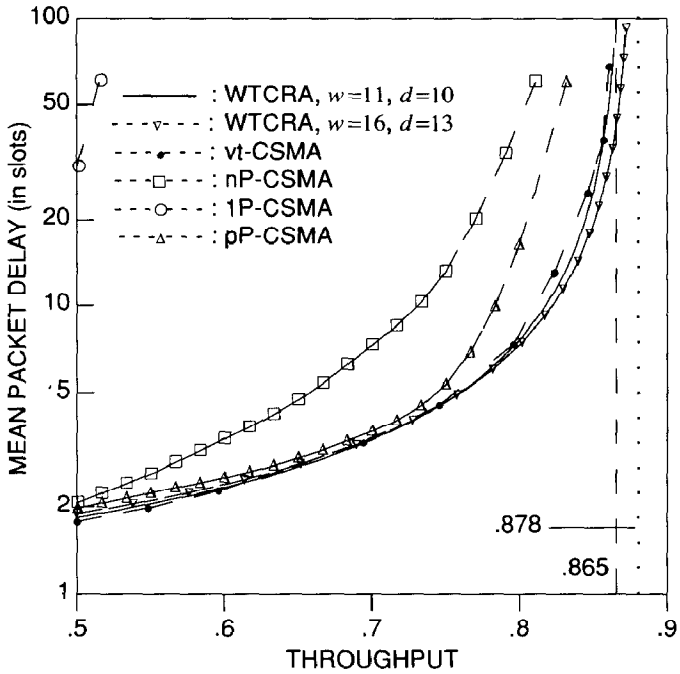


Figure 9: Comparison with previous results for CSMA ($a = b = 101$). Non-Persistent, 1-Persistent, and optimal p-Persistent CSMA curves are reproduced from [3], the virtual time CSMA curve comes from [4].

A modification that comes at no cost for implementation but cannot be handled directly by our analysis is the avoidance of the "rest periods" through the use of a variable window channel access algorithm. However, the importance of this modification is diminished in the LAN environment, since the optimal window sizes obtained for most cases are smaller than the packet transmission time (some times much more smaller) and the same is true for "rest periods". The main consequence of the use of the "rest period" algorithm is that with an empty channel a packet has to wait on the average for half a window before attempting transmission.

The main part of the delay for the system is obtained through the use of the "Moving Server" technique. The technique provides exact results if a condition on the lengths of idle, successful, and collision slots, and the window size (condition (13)) is met. In particular it gives exact results in the case of a common slot size (no carrier sensing or collision detection). If, however, the condition is not met, although the results cannot be regarded as exact, it has been found through simulation that they are very accurate.

### References

[1]. R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," *Communications of the ACM*, vol. 19, no. 7, July 1976.

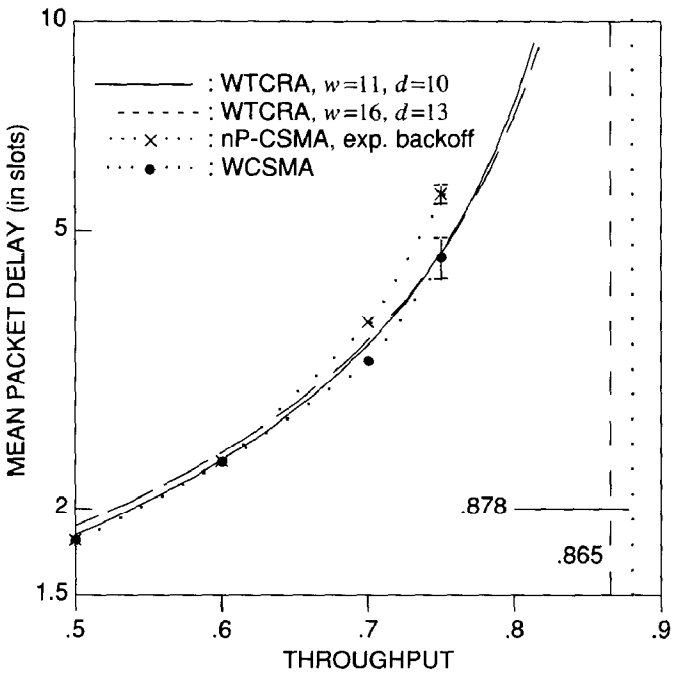


Figure 10: Comparison with previous results ($a = b = 101$). Simulation results for non-Persistent CSMA with adaptive exponential backoff and the Window CSMA (Gallager's algorithm ([11]) in a LAN environment with binary splitting) are taken from [13].


[2]. R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, "Advances in Packet Radio Technology," *Proceedings of the IEEE*, vol. 66, pp. 1468-1496, November 1978.

[3]. L. Kleinrock and F. A. Tobagi, "Packet Switching in Radio Channels: Part I — Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics," *IEEE Transactions on Communications*, vol. COM-23, no. 12, pp. 1400-1416, December 1975.

[4]. M. L. Molle and L. Kleinrock, "Virtual Time CSMA: Why Two Clocks are Better than One.," *IEEE Transactions on Communications*, vol. COM-33, no. 9, September 1985.

[5]. F. A. Tobagi, "Distributions of Packet Delay and Interdeparture Time in Slotted ALOHA and Carrier Sense Multiple Access," *Journal of the ACM*, vol. 29, no. 4, pp. 907-927, October 1982.

[6]. F. A. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part IV — Stability Considerations and Dynamic Control in Carrier Sense Multiple Access," *IEEE Transactions on Communications*, vol. COM-25, pp. 1103-1119, October 1977.

[7]. J. S. Meditch and C. A. Lea, "Stability and Optimization of the CSMA and CSMA/CD Channels," *IEEE Transactions on Communications*, vol. COM-31, no. 6, pp. 763-774, June 1983.

[8]. T. Berger, N. Mehravari, D. Towsley, and J. Wolf, "Random Multiple-Access Communication and Group Testing," *IEEE Transaction on Communications*, vol. COM-32, no. 7, pp. 769-779, July 1984.

[9]. J. I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Transactions on Information Theory*, vol. IT-25, pp. 505-515, September 1979.

[10]. J. I. Capetanakis, "Generalized TDMA: The Multi-Accessing Tree Protocol," *IEEE Transactions on Communications*, vol. COM-27, no. 10, pp. 1476-1484, October 1979.

[11]. R. G. Gallager, "Conflict Resolution in Random Access Broadcast Networks," *Proceedings of the AFOSR Workshop in Communication Theory and Applications*, pp. 74-76, Provincetown, Mass., Sept. 17-20, 1978.

[12]. J. L. Massey, "Collision-Resolution Algorithms and Random-Access Communications," in *Multi-User Communications*, ed. G. Longo, Springer-Verlag, New York, 1981. (preprint: UCLA Report no. UCLA-ENG-8016, April 1980).

[13]. D. Towsley and G. Venkatesh, "Window Random Access Protocols for Local Computer Networks," *IEEE Transactions on Computers*, vol. C-31, no. 8, pp. 715-722, August 1982.

[14]. M.L. Molle, "Asynchronous Multiple Access Tree Algorithms," *ACM SIGCOMM '83 Symposium on Communications Architectures and Protocols*, Austin, TX, March 1983. (Preprint: Technical Report CSRG-145, University of Toronto)

[15]. P. Mathys and P. Flajolet, "Q-ary Collision Resolution Algorithms in Random-Access Systems with Free or Blocked Channel Access," *IEEE Transaction on Information Theory*, vol. IT-31, no. 2, March 1985.

[16]. G. C. Polyzos, M. L. Molle, and A. N. Venetsanopoulos, "Delay Analysis of Tree Conflict Resolution Algorithms: The Non-Homogeneous Case," *Proc. IEEE Globecom '85*, pp. 1504-1509, New Orleans, LA, December 1985.

[17]. G. C. Polyzos and M. L. Molle, "The Delay Distribution of Tree Conflict Resolution Algorithms Using Constant Size Window Access," *1986 IEEE International Symposium on Information Theory*, Ann Arbor, Michigan, October 6-9, 1986.

[18]. E. R. Berlekamp and R. J. McEliece, "Average-Case Optimized Decoders," *1983 NATO Advanced Study Institute on the Impact of Processing Techniques on Communications*, Chateau de Bonas, France, July 11-22, 1983.

[19]. M. L. Molle, "Modelling the Delay in 'Moving Server' Random Access Protocols," in *Current Advances in Distributed Computing and Communications*, ed. Y. Yemini, Computer Science Press, 1987.

## APPENDIX A

In this Appendix we obtain equation (1), the functional equation for the generating function of epoch length. We first prove a recursive relation between the generating functions of epoch length at levels $j$ and $j-1$ of a finite homogeneous population system with Bernoulli arrivals with probability $\pi$ per station per window and then we get infinite population results by taking the limit of the expressions as the population size increases without bound but the total traffic remains constant.

We define the following events:

$$i : \text{idle} \quad , \quad s : \text{success} \quad , \quad c : \text{collision} \quad , \quad * : i \text{ or } s \text{ or } c \quad , \text{ and } \bar{e} : \text{the complement of } e \quad ,$$

and the probabilities

$$\Phi_j(\pi) \triangleq \Pr\{s \mid j\} = (1-\pi)^{2^j} \quad , \quad \Psi_j(\pi) \triangleq \Pr\{\textit{success} \mid j\} = 2^j\pi(1-\pi)^{2^j-1} \quad , \text{ and } \quad \bar{F}_j(\pi) \triangleq \Pr\{\bar{c} \mid j\} = 1 - \Phi_j(\pi) - \Psi_j(\pi) \, .$$

The relationship between the generating functions at levels $j$ and $j-1$ can be obtained through an exhaustive enumeration of possible events at level $j-1$.

| Conditional generating function × Pr{ Event } | | Event @ level $j$ | Event @ level $j-1$ |
|---|---|---|---|
| $Q_j(\pi,z) =$ | $z\,\Phi_j(\pi)$ | $i$ | |
| | $+ z^a\,\Psi_j(\pi)$ | $s$ | |
| | | $c$ | $\exists\, c$: |
| | $+ z^b\, Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^{d-1}$ | | $c,*,\cdots,*$ |
| | $+ z^b\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^{d-2}$ | | $\bar{c},c,*,\cdots,*$ |
| | $+ z^b\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^2\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^{d-3}$ | | $\bar{c},\bar{c},c,*,\cdots,*$ |
| | $\cdots$ | | |
| | $+ z^b\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^{d-1}\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)$ | | $\bar{c},\cdots,\bar{c},c$ |
| | | | $\sim\exists\, c$: |
| | $+ z^b \sum_{n=2}^{d} \binom{d}{n} [z^a\Psi_{j-1}(\pi)]^n\,[z\,\Phi_{j-1}(\pi)]^{d-n}$ | | $n \times s$ and $(d-n) \times i$ |

(A1)

Collecting terms we get

$$Q_j(\pi,z) = z\,\Phi_j(\pi) + z^a\,\Psi_j(\pi) + z^b\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\sum_{n=0}^{d-1}[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^n\,[Q_{j-1}(\pi,z)]^{d-1-n}$$

$$+ z^b \sum_{n=2}^{d} \binom{d}{n} [z^a\Psi_{j-1}(\pi)]^n\,[z\,\Phi_{j-1}(\pi)]^{d-n} \tag{A2}$$

Recognizing, however, that

$$Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi) = Q_{j-1}(\pi,z) - z\,\Phi_{j-1}(\pi) - z^a\,\Psi_{j-1}(\pi) \tag{A3}$$

substituting in equation (A2) and summing we get

$$Q_j(\pi,z) = z\,\Phi_j(\pi) + z^a\,\Psi_j(\pi) + z^b\,[Q_{j-1}(\pi,z)-z\,\Phi_{j-1}(\pi)-z^a\Psi_{j-1}(\pi)]\,\frac{[Q_{j-1}(\pi,z)]^d - [z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^d}{[Q_{j-1}(\pi,z)-z\,\Phi_{j-1}(\pi)-z^a\Psi_{j-1}(\pi)]}$$

$$+ z^b\left[\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^d - [z\,\Phi_{j-1}(\pi)]^d - d\;z^a\Psi_{j-1}(\pi)\,[z\,\Phi_{j-1}(\pi)]^{d-1}\right] .$$

But

$$\Phi_j(\pi) = [\Phi_{j-1}(\pi)]^d \quad \text{and} \quad \Psi_j(\pi) = d\;\Psi_{j-1}(\pi)\,[\Phi_{j-1}(\pi)]^{d-1}$$

thus

$$Q_j(\pi,z) = (z-z^d)\,\Phi_j(\pi) + (z^a-z^{b+a+d-1})\,\Psi_j(\pi) + z^b\,[Q_{j-1}(\pi,z)]^d - z^b\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^d + z^b\,[z\,\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^d$$

and finally we get the recursion

$$Q_j(\pi,z) = z^b\,Q_{j-1}^d(\pi,z) + (z - z^{b+d})\,\Phi_j(\pi) + (z^a - z^{b+a+d-1})\,\Psi_j(\pi) \, . \tag{A4}$$

The infinite population Poisson model is the limit of the finite population Bernoulli model as the population tends to infinity with the total traffic remaining constant, say $x$ packets per window. It is then natural to define the (infinite population) generating function of epoch length as

$$Q(x,z) \triangleq \lim_{M(j)\pi = x \atop j\to\infty} Q_j(\pi,z) \tag{A5}$$

where $M \triangleq M(j) \triangleq d^j$ is the population size at level $j$. It then follows that

$$\lim_{M(j)\pi = x \atop j\to\infty} Q_{j-1}(\pi,z) = \lim_{M(j-1)\pi = x/d \atop j-1\to\infty} Q_{j-1}(\pi,z) = Q(x/d,z) \, . \tag{A6}$$

We also need the following two limits:

$$\lim_{M(j)\pi = x \atop j\to\infty} \Phi_j(\pi) = \lim_{M\pi = x \atop M\to\infty} (1-\pi)^M = \lim_{M\to\infty} \left(1 - \frac{x}{M}\right)^M = e^{-x} \tag{A7}$$

and

$$\lim_{\substack{M\to\infty \\ M(j)\pi = x}} \Psi_j(\pi) = \lim_{\substack{M\to\infty \\ M\pi = x}} M\,\pi\,(1-\pi)^{M-1} = \lim_{M\to\infty} x\,\left(1 - \frac{x}{M}\right)^M = x\,e^{-x}\,. \tag{A8}$$

Taking the limit of both sides of equation (A4) and substituting (A5-A8) we obtain

$$Q(x,z) = z^b\,Q^d(x/d,z) + (z^a - z^{b+d})\,e^{-x} + (z^a - z^{b+a+d-1})\,x\,e^{-x}$$

which is equation (1).

□

## APPENDIX B

Here we obtain equation (21), the functional equation for the generating function of the delay of a packet in its epoch of transmission. We follow the approach of Appendix A, developing a recursive relation between the generating functions at levels $j$ and $j-1$ of a finite homogeneous population system. In this case, however, it is given that there is at least one transmission in the epoch (the tagged packet), and the recursion involves not only $G_j(\pi,z)$, the generating function of the delay in the epoch of transmission, but $Q_j(\pi,z)$ as well. We define a modified $\Phi$ function,

$$\Phi'_j(\pi) = \Pr\{\text{ no transmission except the tagged packet in this subtree }\} = (1-\pi)^{2^j-1}\,.$$

Also, we denote by $A_{j-1}$ the result of the "coin toss" for the tagged packet in the case of a collision at level $j$. Thus, $G_j^{(A_{j-1}=n)}(\pi,z)$ is the generating function of $t_2$ under the condition that if there is a collision at level $j$, the packet will choose the $n$th subtree.

| $G_j^{(A_{j-1}=n)}(\pi,z) =$ Conditional generating function × Pr{ Event } | Event @ level $j$ | Event @ level $j-1$ |
|---|---|---|
| $z\,\Phi'_j(\pi)$ | $s$ | |
| | $c$ | $\exists\, c$ @ $n^{th}$ tree: |
| $+\ z^b\,G_{j-1}^{(c)}(\pi,z)\,\bar{\Phi}'_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^n$ | | $*,*,\cdots,*,c,*,\cdots,*$ |
| | | $\sim\exists\, c$ @ $n^{th}$ tree: |
| $+\ z^b\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,z^a\,\Phi'_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^{n-1}$ | | $c,*,\cdots,*,s,*,\cdots,*$ |
| $+\ z^b\,[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,z^a\,\Phi'_{j-1}(\pi)\,[Q_{j-1}(\pi,z)]^{n-2}$ | | $\bar{c},c,*,\cdots,*,s,*,\cdots,*$ |
| $\cdots$ | | |
| $+\ z^b\,[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^{n-1}\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,z^a\,\Phi'_{j-1}(\pi)$ | | $\bar{c},\cdots,\bar{c},c,s,*,\cdots,*$ |
| $+\ z^b\,[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,\Phi_{j-1}(\pi)$ | | $\bar{c},\cdots,\bar{c},s,c,*,\cdots,*$ |
| $\cdots$ | | |
| $+\ z^b\,[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,\Phi_{j-1}(\pi)[F_{j-1}(\pi)]^{d-2-n}$ | | $\bar{c},\cdots,\bar{c},s,\bar{c},\cdots,\bar{c},c$ |
| | | $\sim\exists c$: |
| $z^b\,[z\Phi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,[\,[F_{j-1}(\pi)]^{d-1-n} - [\Phi_{j-1}(\pi)]^{d-1-n}\,]$ | | $0\times s$ before tagged |
| $z^b\,\binom{n}{m}\,[z^a\Psi_{j-1}(\pi)]^m\,[z\Phi_{j-1}(\pi)]^{n-m}\,z^a\,\Phi'_{j-1}(\pi)\,[F_{j-1}(\pi)]^{d-1-n}$ | | $m\times s$ before tagged $m=1,\cdots,n$ |

(B1)

Collecting terms we get

$$\begin{aligned} G_j^{(A_{j-1}=n)}(\pi,z) = {} & z^a\,\Phi'_j(\pi) + z^b\,G_{j-1}^{(c)}(\pi,z)\,\bar{\Phi}'(\pi)\,[Q_{j-1}(\pi,z)]^n \\ & + z^b\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,z^a\,\Phi'_{j-1}(\pi)\sum_{m=0}^{n-1}[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^m\,[Q_{j-1}(\pi,z)]^{n-1-m} \\ & + z^b\,[z\Phi_{j-1}(\pi)+z^a\Psi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,[\,1-[F_{j-1}(\pi)]^{d-1-n}\,] \\ & + z^b\,[z\Phi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,[\,[F_{j-1}(\pi)]^{d-1-n} - [\Phi_{j-1}(\pi)]^{d-1-n}\,] \\ & + z^b\,z^a\,\Phi'_{j-1}(\pi)\,[F_{j-1}(\pi)]^{d-1-n}\sum_{m=1}^{n}\binom{n}{m}\,[z^a\Psi_{j-1}(\pi)]^m\,[z\Phi_{j-1}(\pi)]^{n-m} \end{aligned} \tag{B2}$$

Recognizing, however, that

$$Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi) = Q_{j-1}(\pi,z) - z\,\Phi_{j-1}(\pi) - z^a\,\Psi_{j-1}(\pi) \quad \text{and} \quad G_j^{(c)}(\pi,z)\,\bar{\Phi}'_j(\pi) = G_j(\pi,z) - z^a\Phi'_j(\pi) \tag{B3}$$

substituting in equation (B2) and summing we get

$$G_j^{(A_{j-1}=n)}(\pi,z) = z^a\,\Phi'_j(\pi) + z^b\,[\,G_{j-1}(\pi,z) - z^a\,\Phi'_{j-1}(\pi)\,]\,[Q_{j-1}(\pi,z)]^n$$
$$+ z^b\,Q_{j-1}^{(c)}(\pi,z)\,\bar{F}_{j-1}(\pi)\,z^a\,\Phi'_{j-1}(\pi)\frac{[Q_{j-1}(\pi,z)]^n - [z\,\Phi_{j-1}(\pi)+z^a\,\Psi_{j-1}(\pi)]^n}{Q_{j-1}(\pi,z) - [z\,\Phi_{j-1}(\pi)+z^a\,\Psi_{j-1}(\pi)]}$$
$$+ z^b\,[z\,\Phi_{j-1}(\pi)+z^a\,\Psi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,[\,1-[F_{j-1}(\pi)]^{d-1-n}\,]$$
$$+ z^b\,[z\,\Phi_{j-1}(\pi)]^n\,z^a\,\Phi'_{j-1}(\pi)\,[\,[F_{j-1}(\pi)]^{d-1-n} - [\Phi_{j-1}(\pi)]^{d-1-n}\,]$$
$$+ z^b\,z^a\,\Phi'_{j-1}(\pi)\,[F_{j-1}(\pi)]^{d-1-n}\,[\,[z\,\Phi_{j-1}(\pi)+z^a\,\Psi_{j-1}(\pi)]^n - [z\,\Phi_{j-1}(\pi)]^n\,] \qquad \text{(B4)}$$

But

$$\Phi'_j(\pi) = [\Phi_{j-1}(\pi)]^{d-1}\,\Phi'_{j-1}(\pi)$$

thus

$$G_j^{(A_{j-1}=n)}(\pi,z) = (\,z^a - z^{b+a+n}\,)\,\Phi'_j(\pi) + z^b\,G_{j-1}(\pi,z)\,[Q_{j-1}(\pi,z)]^n\ .$$

Now, since the tagged packet chooses fairly one of the $d$ subtrees,

$$G_j(\pi,z) = \frac{1}{d}\sum_{n=0}^{d-1} G_j^{(A_{j-1}=n)}(\pi,z)$$
$$= z^b\,G_{j-1}(\pi,z)\,\frac{1}{d}\,\frac{[Q_{j-1}(\pi,z)]^d - 1}{Q_{j-1}(\pi,z) - 1} + \Phi'_j(\pi)\left[\,z - z^b\,\frac{1}{d}\,\frac{z^{ad}-1}{z^a-1}\,\right] \qquad \text{(B5)}$$

We define the (infinite population) generating function of the delay in the epoch of transmission as

$$G(x,z) \triangleq \lim_{M(j)p=x} G_j(\pi,z) \qquad \text{(B6)}$$

where $M \triangleq M(j) \triangleq d^j$ is the population size at level $j$. Again

$$\lim_{M(j)\pi=x} G_{j-1}(\pi,z) = \lim_{M(j-1)\pi=x/d} G_{j-1}(\pi,z) = G(x/d,z) \qquad \text{(B7)}$$

and

$$\lim_{M(j)\pi=x} \Phi'_j(\pi) = \lim_{\substack{M\to\infty \\ M\pi=x}} (1-\pi)^{M-1} = \lim_{M\to\infty}\,(\,1-\frac{x}{M}\,)^{M-1} = e^{-x}\ . \qquad \text{(B8)}$$

Taking the limit of both sides of equation (B5) and substituting (B6-A8) we obtain

$$G(x,d) = z^b\,G(x,z)\,\frac{1}{d}\,\frac{[Q(x,z)]^d - 1}{Q(x,z) - 1} + e^{-x}\left[\,z - z^b\,\frac{1}{d}\,\frac{z^{ad}-1}{z^a-1}\,\right]$$

which is equation (21).

□