Data packet loss in a queue with limited buffer space

Problem presented by

José Gil

Motorola Research

Problem statement

When channels with limited buffer space are shared by multiple bursty data sources then some fraction of submitted packets is lost when the buffer becomes full. Feedback signals enable the sources to reduce their burst sizes when the buffer becomes congested. On the other hand, if there is plenty of free space in the buffer then burst sizes are allowed to increase. Such systems may be designed with a variety of queueing disciplines and packet discard policies. The Study Group was asked to analyse the packet-loss process for packet submission using either first-in-first-out or weighted fair queueing, and for packet discard using either drop-tail or random early discard policies.

Study Group contributors

David Allwright (Smith Institute)
Paul Dellar (Imperial College London)
Robert Leese (Smith Institute)
Malwina Luczak (London School of Economics)
Steven Noble (Brunel University)
Domingo Salazar (University of Oxford)

Report prepared by

David Allwright (Smith Institute)

1 Introduction[1]

Outline: A system that occurs in several contexts in computer and data networks has the generic form shown in Figure 1.

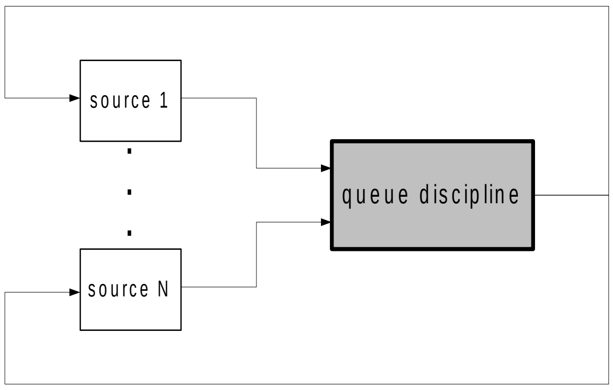

Figure 1: Diagram of the system under consideration.

There are N sources sharing a single channel, and their access to this channel is administered by a resource management system which has a limited buffer space. The sources transmit packets in bursts, and in some circumstances (*e.g.* if the buffer is full) some of the packets may be lost. The source receives an explicit feedback signal from the system reporting the packet loss ratio for the burst. The size of the next burst is determined by the source as a function of this packet loss ratio. The time between sending a burst of packets and receiving the feedback packet loss signal is called the round trip time. It is variable due to queuing within the resource management system and channel variability. The elements of the resource management system are a service discipline that schedules the packets, a buffer system that stores the packets waiting for service, and a packet discard policy that discards incoming packets depending on the buffer occupancy. The problem is to find the steady state distribution of the packet loss ratios, and other information about the statistics of the distribution, *e.g.* the variance of the packet loss distribution.

1.1 Sources

We shall be thinking of the number of sources N as being large: sometimes it may be about 60–100, but in other circumstances up to 1,000. In practice the sources may be of various kinds, and may for instance be part of some networking software. However, for the purposes of this problem the sources are to be treated as essentially described by the way they respond to packet loss.

There are two cases to be studied:

[1]This section combines the problem description circulated before the Study Group, the presentation made on the opening day, and additional information provided later during the week.

(1) Case 1: each source switches ON and OFF randomly, with transition rates $q_{n,\mathrm{ON}}$ and $q_{n,\mathrm{OFF}}$, say, which depend on the source index $n = 1, 2, \ldots, N$. The sources switch ON and OFF independently of each other, *i.e.* there is no 'clumping'.

(2) Case 2: each source is always ON.

When a source is ON, it submits a sequence of bursts to the queuing system. Each burst consists of a number of packets. (Internally, a packet may be of 576–1,500 bytes, and is divided up into blocks of 20–50 bytes, but we do not consider that structure here, thinking instead of the packet as a single entity. In GPRS, a packet may typically take about one second to transmit.) If the source is labelled X we shall denote the size of the j-th burst (*i.e.* the number of packets submitted) by X_j^s. Some of those packets may be discarded by the resource management system, so the number of packets actually transmitted may be less, and we denote it by X_j. When they have been successfully transmitted by the channel, a feedback signal is sent to source X that X_j packets were successfully transmitted, so the packet loss ratio is

$$p = 1 - \frac{X_j}{X_j^s}. \tag{1}$$

The number of packets submitted in the next burst is then

$$X_{j+1}^s = \min\left(f(1 - X_j^s/X_{j+1}^s), 2X_j\right), \tag{2}$$

where

$$f(p) = \frac{b}{\sqrt{(b^2-1)p^V + 1}}, \quad \text{rounded to an integer.} \tag{3}$$

If there is no packet loss, $p = 0$ and $f(p) = b$, which is the maximum number of packets in a burst, typically taken in the range 10–64. As the packet loss p increases, $f(p)$ decreases: this is the control system by which the sources temper the demand they make on the system in accordance with the response they obtain. When all packets are lost, $p = 1$ and $f(p) = 1$. The parameter V (which will take different values V_n for different sources) controls how much the source responds to packet loss: a source with a high value of V still prefers to submit larger bursts even when p falls, but a source with low V submits smaller bursts more readily. In other words, larger values of V represent more aggressive behaviour, and smaller values more moderate. (This is different from other kinds of aggressive behaviour in congested networks, *e.g.* sending *multiple* bursts to try to grab more channels — in our case aggressive sources do not do this, but simply continue to maintain the submitted burst size.) The typical range of V is $0.5 \leq V \leq 2$, and one could initially consider a situation where sources were of two different types, N_a sources with $V = V_a$, and N_b sources with $V = V_b$. The graphs in Figure 2 show the function $f(p)$ for $b = 10$, with $V = 0.5$ or $V = 2$, plotting both the continuous function and its value rounded to the nearest integer. It is perhaps worth noting that there is

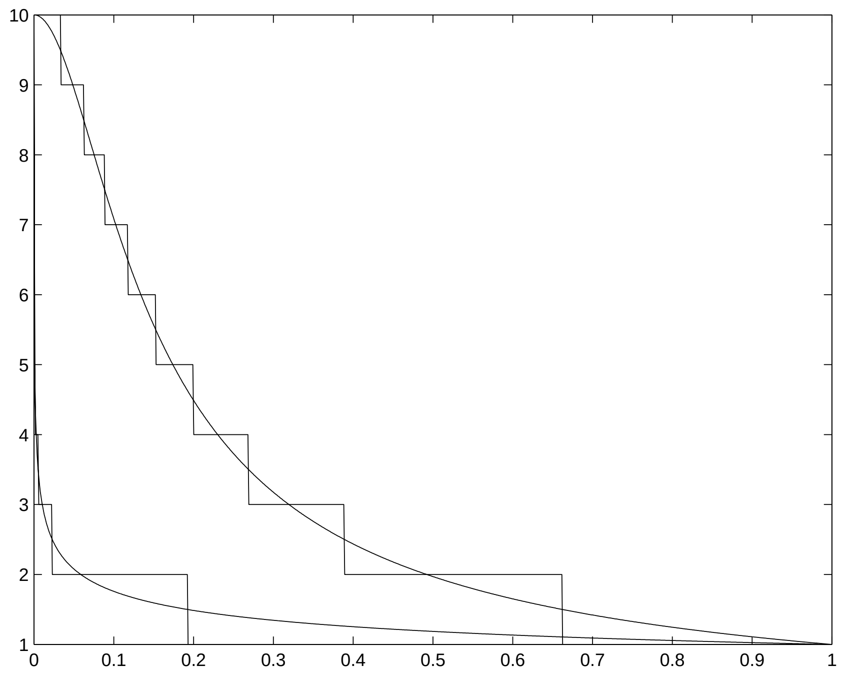

Figure 2: The function (3) with $b = 10$, for $V = 2$ (upper graphs) and for $V = 0.5$ (lower).

always a rapid initial fall in $f(p)$. In fact if p is not 0 then it must be at least $1/b$ (in the case where b packets were submitted and only 1 lost) and then f falls from b to

$$f(1/b) \approx \begin{cases} b/\sqrt{2} & \text{if } V = 2 \\ \sqrt{b} & \text{if } V = 1 \\ b^{1/4} & \text{if } V = 0.5. \end{cases} \tag{4}$$

If $p = 1$, so all packets of a burst are lost, then source X submits its next burst (which will be a single packet) after a certain time-out interval T_0. This $T_0 = 3\overline{RTT}$ is three times the average round trip time (RTT) of previous transmissions, typically calculated by $\overline{RTT}_{\text{new}} = 0.9 * \overline{RTT}_{\text{old}} + 0.1 * RTT$. It should be noted that the transmission times dominate the time spent in the process, so that we shall always consider the time spent loading packets into the buffer, computing the burst size, *etc.*, to be negligible.

1.2 Packet discard policies

The total buffer space available will be denoted by B, and in practice may be of order megabytes but will depend on the application. We are concerned with what happens when this buffer fills up, and there are two packet discard policies that might be in operation:

(1) Drop-tail: this simply means that if a packet arrives when the buffer is full then it, and any subsequent packets of the same burst, are dropped.

(2) Random Early Discard (RED): If the buffer occupancy exceeds some threshold $O_{\min}$, ($O_{\min} < B$), then arriving packets are dropped with a probability varying linearly between 0 at $O_{\min}$ and 1 at B, different packets being independent.

The graphs of packet drop probability against buffer occupancy in these two cases are illustrated in Figure 3.

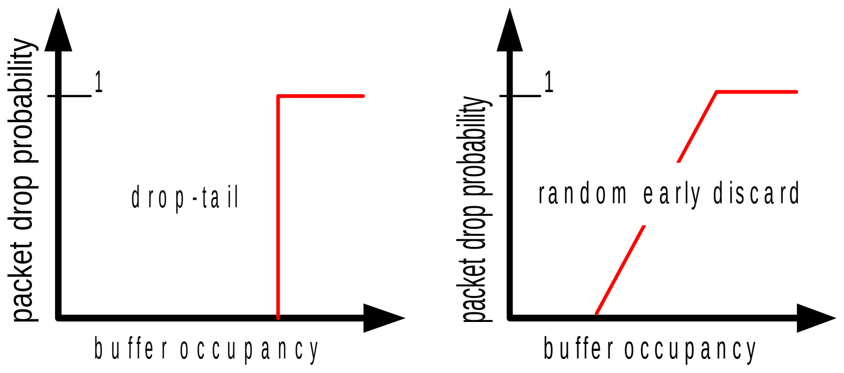

Figure 3: Packet loss probability for the two packet discard policies.

1.3 Service disciplines

There are three queuing disciplines of interest:

(1) First-in-first-out (FIFO). This is the simplest case, in which all packets form a single queue, occupying the whole buffer of length B packets, and are dealt with in order of arrival.

(2) Weighted Fair Queuing (WFQ). Here, the buffer space is divided up into a number of smaller buffers in which separate queues operate allocated to specific services or sources, and there are various ways in which this can be done.

In one variant, which we call WFQ1 (referred to as Quality of Service) there are 4 queues, and each source is allocated to a particular queue, with the FIFO system applying to each queue individually. Each queue is assigned one of four weights w_1, w_2, w_3, w_4. Packets are serviced one from each queue in turn, in a fixed cyclic order (round robin fashion) skipping any queues that are empty, and with an allocation of serving time proportional to the weights. (There are various ways in which this can be done. For instance, suppose the system is deciding which queue to take the m-th packet from, and the previous $m-1$ packets have dealt with m_i from the i-th queue. Then choosing i to maximize $w_i m - m_i$ and taking the m-th packet from that queue will achieve the result that as $m \to \infty$, $m_i/m \to w_i$. There are complications if some of the queues are empty for parts of the time, but we shall not pursue this point.)

In another variant, which we call WFQ2 (referred to as Best Effort), the buffer is divided into M buffers, each of size $B' = B/M$, but each of these buffers just serves for a single source. If $N > M$ then while the M buffers are holding packets for M different sources, the remaining $N - M$ sources will experience 100% packet loss, reducing their bursts to just one packet. If $N \leq M$ then all sources get buffer space. A source will overflow its own buffer when the burst size becomes larger than B'. The queues are serviced in a round robin fashion.

In a third variant, WFQ3, the M buffers of WFQ2 are dealt with according to some weights w_i rather than equally.

(3) Earliest Service Time (EST). This aims to give priority to flows with the least service time: the scheduler knows the average service time for packets from each source, and gives some weighted priority to those packets that can be transmitted fastest. (But while a packet is being served, no preemption occurs if a new incoming packet has shorter service time.)

The queueing patterns in the first two cases are illustrated in Figure 4.

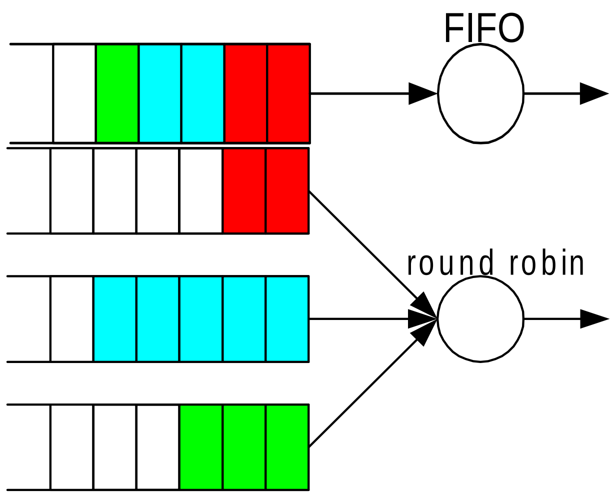

Figure 4: FIFO and round robin queueing systems.

1.4 The channel

The channel is of limited capacity, and is time-variable, making the service time for a packet variable, in a way that can be treated as random. The way that its capacity is used by the different sources is governed by the service discipline in use. Possible assumptions about the packet service time could be

(1) exponential service time;

(2) general service time (as in a M/G/1 queuing system).

1.5 Problems for the Study Group

The combinations of cases, in order of most interest, are

(1) Drop-tail and FIFO

(2) Drop-tail and Weighted Fair Queuing

(3) Random Early Discard and FIFO

(4) Random Early Discard and Weighted Fair Queuing.

2 FIFO queue with drop-tail discard policy

We begin the study by considering what happens in the case of a single queue, operating a FIFO queuing policy, and with a drop-tail discard policy. We initially consider the case in which there are N sources, all ON, and suppose that each source has some packets in the queue. For ease of illustration we shall consider $N = 3$, and label the sources as X, Y and Z, and suppose that at some stage the queue contains numbers of packets Z_j, Y_j, X_j and the X_j packets are about to be dealt with. After they have been processed, source X will receive the signal and will submit its next burst of X^s_{j+1} packets. If they can all be placed in the remaining space in the buffer, none are lost, but if not then the buffer will just be filled up to its capacity B and the remaining packets lost. Thus the number of X-packets that enter the queue is

$$X_{j+1} = \min(X^s_{j+1}, B - Y_j - Z_j). \tag{5}$$

The queue then contains X_{j+1}, Z_j, Y_j and the Y_j packets are dealt with. On completion of them, source Y submits Y^s_{j+1} packets, and

$$Y_{j+1} = \min(Y^s_{j+1}, B - Z_j - X_{j+1}) \tag{6}$$

of them enter the buffer, which then contains Y_{j+1}, X_{j+1}, Z_j. After transmission of the Z_j packets, source Z submits Z^s_{j+1} packets,

$$Z_{j+1} = \min(Z^s_{j+1}, B - X_{j+1} - Y_{j+1}) \tag{7}$$

of them enter the buffer, and we are back to the initial point with j replaced by $j + 1$.

This has the consequence that the sequence of burst sizes is deterministic, governed by (5)–(7) and (2) for each source. The mapping of this deterministic sequence into physical time will be random of course, depending on the service times of the packets, but the sources will continue to transmit in the same cyclic order X, Y, Z, X *etc*, with submitted and actual burst sizes governed by these recurrence relations. No source can get pushed out by this process, since if $X_j \geq 1$ then $X^s_{j+1} \geq 1$ and $B - Y_j - Z_j \geq X_j \geq 1$, so (5) shows $X_{j+1} \geq 1$. Furthermore, if at any point in the cycle a source submits a maximum burst, of b packets, and they are all transmitted successfully, *i.e.* $X_j = X^s_j = b$, then at the next stage it will also submit $X^s_{j+1} = b$ according to (2) and since $B - Y_j - Z_j \geq X_j = b$, equation (5) ensures that $X_{j+1} = b$ also, so that source continues to send maximum bursts successfully.

This discrete system can be simulated easily of course, and is certain to settle into a cycle. Simulations with $N = 3$ sources, buffer size $B = 20$, and maximum burst size $b = 10$ show that

(1) If each $V_i = 2$ then a cycle is reached in which the first source has maximum burst size, and the other 2 sources share the remaining resource;

(2) If two of the V_i are 2 and the other is 0.5, then a cycle is reached in which the first source with $V = 2$ has maximum burst size;

(3) If just one of the V_i is 2 and the others are 0.5 then the cycle gives that (more aggressive) source the maximum burst size;

(4) If each source has $V = 0.5$, then a cycle is reached in which *no* source has maximum burst size, and the buffer is, in effect, underused.

2.1 Sources entering and leaving

When a source leaves a cycle of the kind described above, the immediate benefit is obtained by the source that follows it in order, and the system will settle into a new cycle in which that source obtains a greater share of the service.

More complicated is what happens when a new source, say W, switches ON and tries to join the system. It will initially submit a 1-packet burst, and if that arrives when the buffer is full, it will be discarded and W will wait for the time-out interval T_0 before trying again. Since that time-out interval is not synchronized with the packet transmission process, W will eventually (assuming $N \leq B$) send its packet at a time when there is space in the buffer (because a burst of 2 or more packets is part way through transmission) and so W will enter the system at that point and the system will then settle to a deterministic cycle of 4 sources.

If the rates at which sources switch between ON and OFF are low compared with the rate at which the deterministic system reaches its steady cycle, then we can think of the system as simply moving between these cycles in response to the changing number and character of the sources. However, if the ON/OFF switching rates are comparable with the response rate of the deterministic system, then to handle it properly we would need to have a more complicated Markov chain with quite a large state space. For instance, suppose we take only $N = 2$ sources, a buffer size $B = 3$, and maximum burst size $b = 2$. Then the items that may be in the queue from X are: a 2-packet burst, a 1-packet burst that is going to be followed by a 2-packet burst, and a 1-packet burst that is going to be followed by a 1-packet burst. Denote these by X_2, X_1, X_1', and similarly for Y. Then the states of the queue at the point where a burst has just finished transmission can be: empty, X_2, X_1, X_1', X_2Y_1, X_2Y_1', X_1Y_2, X_1Y_1, X_1Y_1', $X_1'Y_2$, or any of these last 9 states with X and Y reversed, making a total of 19 states of the queue. Each of these has to be combined with the states of the two sources X and Y being either ON or OFF, making a total of 76 states for the system. We did not attempt to analyse this kind of situation further.

3 Weighted Fair Queuing (WFQ) with drop-tail

There are two versions of WFQ that we may consider. In the first version (WFQ1) there are 4 queues, each for a different traffic class. Each source sends its bursts to a particular queue, and each queue is a FIFO buffer of size B/4, served at a rate w_j times the total rate. In this case, each of the buffers individually functions in the manner described earlier, and we shall not discuss this case further.

In the second version (WFQ2), there is a single queue for each source, M queues available, and the number of sources is $N > M$. Each of the $N - M$ waiting sources will send 1-packet bursts until it gets a queue. This will happen only if it sends its packet while one of the existing queues is empty because it is just transmitting the *last* packet of the previous burst: we refer to this as stealing the queue. Once a source steals a queue in this way, it will next send a 2-packet burst, then 4, *etc*, until either it reaches its maximum burst size b (when it will continue to send bursts of that size), or its queue is stolen in turn by one of the waiting sources.

This could be set up as a continuous-time Markov chain, but will be very complicated. Instead we shall set up a simpler approximate model, thinking of N and M as large. Essentially the approximations consist of

(1) treating the retries from the waiting sources as a Poisson process of rate $\lambda = 1/T_0$;

(2) treating the packet service times as exponential with parameter μ;

(3) replacing the Round Robin by a server that picks one of the M queues at random to serve.

These simplifications will enable the model to capture some aspects of the behaviour, but there will be others that are missed. For instance, in practice it is found that the timings are such that if consecutive queues contain the last packets of two sources X and Y, and if X's queue is stolen by a waiting source W, then there is a strong tendency for X to steal Y's queue when Y's last packet is served. A further aspect is that an aggressive source (high V) will be sending larger bursts, and therefore tend to have higher RTT, and therefore a longer time-out interval T_0 than a more moderate source. This means that when it is waiting it is not retrying as often and so is proportionately less likely to be able to steal a queue.

To model the simplified version of the system, suppose that in a steady state it has m_1 queues serving 1-packet bursts, m_2 queues serving 2-packet bursts, *etc.* Then consider how one particular source moves around in this system, and in particular view it in discrete time at the end of each "event": these events are either packet-services, or retries from the waiting sources. The total rate of events we shall denote by $\Lambda = (N-M)\lambda + \mu$. We describe the source's state as 0 (waiting), or 1, 2, 4, 8, *etc*, according to whether its current burst is of 1, 2, 4, 8, *etc* packets. Then the transition probability p_{01} from waiting to having a 1-packet burst in the system is

$$p_{01} = \frac{\lambda}{\Lambda}\frac{m_1 + m_2/2 + m_4/4 + m_8/8 + \ldots}{M} = \frac{\lambda}{\Lambda}\frac{\left(\sum m_r/r\right)}{M}, \tag{8}$$

since the probability that the next event is a retry by source X is λ/Λ, and when this retry happens the server is dealing with one of the M queues. If it is dealing with a 1-packet burst (probability m_1/M) then source X gets in; if it is dealing with a 2-packet burst (probability m_2/M) then X gets in only if it is handling the *second* packet of that burst (probability 1/2), *etc*, giving the second factor in (8). From state 1, the rate of transition back to 0 is

$$p_{10} = \frac{(N-M)\lambda}{\Lambda}.\frac{1}{M}, \tag{9}$$

since the probability that the next event is a retry by *any* of the waiting sources is $(N-M)\lambda/\Lambda$, and the probability that this retry takes place when X's queue is being served is $1/M$. More hopefully (from X's point of view) the rate of transition from 1 to 2 is

$$p_{12} = \frac{\mu}{\Lambda}.\frac{1}{M}, \tag{10}$$

since the first factor is the probability that the next event is a packet service, and the second factor is the probability that it is X's queue that is served. From state 2, the transition rates are

$$p_{20} = \frac{(N-M)\lambda}{\Lambda}.\frac{1}{M}.\frac{1}{2}, \qquad p_{24} = \frac{\mu}{\Lambda}.\frac{1}{M}.\frac{1}{2}, \tag{11}$$

where in each case we have an additional factor of 1/2 to account for the probability that the event occurs during service of the *second* packet of the burst. The transition rates from 4, 8, *etc* will similarly have factors 1/4, 1/8, *etc.* (For each state i there is of course a probability of remaining in that state $p_{ii} = 1 - \sum_{j\neq i} p_{ij}$.) The steady state distribution with m_i buffers allocated to burst size i is then determined by solving $m_i \sum_{j\neq i} p_{ij} = \sum_{j\neq i} m_j p_{ji}$, representing that fact that the rate of flow of probability *out of* state i must be equal to the rate of flow *into* it. If say $b = 8$ then these equations are

$$m_1(p_{10} + p_{12}) = m_0 p_{01}, \tag{12}$$

$$m_2(p_{20} + p_{24}) = m_1 p_{12}, \tag{13}$$

$$m_4(p_{40} + p_{48}) = m_2 p_{24}, \tag{14}$$

$$m_8 p_{80} = m_4 p_{48}. \tag{15}$$

(The corresponding equation for state 0 could be written down but is just equivalent to the sum of these equations.) Together these imply that

$$\frac{m_8}{m_4} = \frac{p_{48}}{p_{80}} = \frac{2\mu}{(N-M)\lambda}, \qquad \frac{m_4}{m_2} = \frac{p_{24}}{p_{40}+p_{48}} = \frac{2\mu}{\Lambda},$$

$$\frac{m_2}{m_1} = \frac{2\mu}{\Lambda}, \qquad \frac{m_1}{m_0} = \frac{\lambda}{\Lambda}\left(\sum m_r/r\right). \tag{16}$$

Hence we find

$$M = m_1 + m_2 + m_4 + m_8 = m_1 \left(1 + \left(\frac{2\mu}{\Lambda}\right) + \left(\frac{2\mu}{\Lambda}\right)^2 + \left(\frac{2\mu}{\Lambda}\right)^2 \frac{2\mu}{(N-M)\lambda}\right), \quad (17)$$

from which all the required quantities can be found, and the final equation of (16) gives $m_0 = N - M$ as it should.

4 Random Early Discard (RED)

In the case of RED, packets arriving at the buffer are placed in the buffer until the occupancy j reaches $O_{\min}$. When $j \geq O_{\min}$, an arriving packet is dropped with probability

$$q_j = \left(\frac{j - O_{\min}}{B - O_{\min}}\right) = (j - O_{\min})\Delta, \quad (O_{\min} \leq j \leq B), \quad (18)$$

where we shall define

$$\Delta = \frac{1}{B - O_{\min}}. \quad (19)$$

The packet is dropped with this probability *independently* of what happened to previous packets in this or previous bursts.

To form a Markov chain of this situation we would like to know the transition probabilities from a state where the occupancy is j and a k-packet burst arrives to the state where the burst has been processed by RED and the occupancy is j' ($j \leq j' \leq \min(j+k, B)$), so $j + k - j'$ packets have been discarded. The complication in dealing with this is that the probability of acceptance of a particular packet depends on the number of acceptances of previous packets. We shall deal with this problem later, but first we address the simpler question of the *expected value* of j'.

4.1 Expected number of packets accepted

We consider the situation where the occupancy is $j \geq O_{\min}$ and a burst of k packets arrives. The first packet is accepted with probability $p_j = 1 - q_j$, varying with j as illustrated in Figure 4.1. If the first packet is accepted then the second packet is accepted with the smaller probability $p_{j+1} = p_j - \Delta$; but if the first packet was discarded then the second packet is accepted with the same probability p_j that applied to the first packet. Counting the different combinations we find that

$$\begin{aligned} a_1 = \mathbb{P}\text{r}(\text{1st packet accepted}) &= p_j, & (20)\\ a_2 = \mathbb{P}\text{r}(\text{2nd packet accepted}) &= p_j p_{j+1} + (1 - p_j) p_j, & (21)\\ a_3 = \mathbb{P}\text{r}(\text{3rd packet accepted}) &= p_j p_{j+1} p_{j+2} + p_j(1 - p_{j+1}) p_{j+1} + & (22)\\ &\qquad (1 - p_j) p_j p_{j+1} + (1 - p_j)^2 p_j, \end{aligned}$$

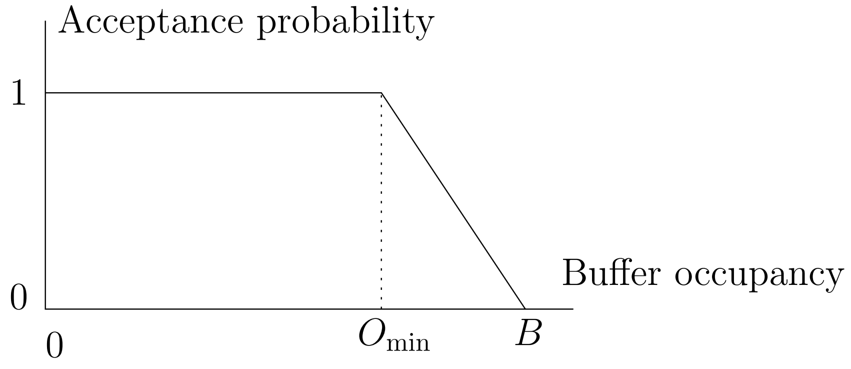

Figure 5: packet acceptance probability as a function of occupancy in the Random Early Discard (RED) case.

and so on. In general the probability a_k that the k-th packet is accepted will be a sum of 2^{k-1} terms, and the expected number of packets accepted out of the k-packet burst will be $a_1 + a_2 + \ldots + a_k$.

We initially left these unpromising-looking expressions and instead considered what would happen if the burst size is large and we regard the packets as a continuum, so that a differential equation approach can be used[2]. The expected number of packets accepted, X, out of a number submitted X^s then obeys

$$\frac{dX}{dX^s} = p = \begin{cases} 1 & \text{if } j + X \leq O_{\min} \\ \dfrac{B - (j + X)}{B - O_{\min}} & \text{if } j + X \geq O_{\min}. \end{cases} \tag{23}$$

With $X = 0$ at $X^s = 0$ this can easily be solved and we find the value of X at $X^s = k$, and then the final occupancy of the buffer is

$$\overline{j'} = \begin{cases} j + k & \text{if } j + k \leq O_{\min} \\ O_{\min} + (B - O_{\min})\,(1 - \exp\left(-(j + k - O_{\min})\Delta\right)) & \text{if } j \leq O_{\min} \leq j + k \\ j + (B - j)\,(1 - \exp(-k\Delta)) & \text{if } j \geq O_{\min}. \end{cases} \tag{24}$$

This represents the way the expected number of packets accepted approaches B exponentially as the number of packets submitted increases without limit.

Having handled this continuum limit successfully[3] we then found we could also deal with the exact expressions (20)–(22) for the discrete case. We let $f_k(x_1, x_2, \ldots, x_k)$ be

[2]following Malwina's remark early in the week: "If you can't do a differential equation, you can't do anything else either."

[3]thereby overcoming the hurdle imposed by Malwina's remark

the probability that the k-th packet is accepted when x_j is the probability of acceptance of a packet after $j-1$ previous acceptances. So for instance

$$\begin{aligned} f_1(x_1) &= x_1, & (25) \\ f_2(x_1, x_2) &= x_1 x_2 + (1 - x_1) x_1, & (26) \end{aligned}$$

and the the quantities we introduced earlier are $a_k = f_k(p_j, p_{j+1}, \ldots, p_{j+k-1})$. Then we can obtain a recurrence relation for the f_k by considering what happens to the first packet: either it is accepted (with probability x_1) in which case the remaining $k-1$ packets face the acceptance probability sequence $x_2, x_3, \ldots, x_k$; or the first packet is discarded (with probability $1 - x_1$) in which case the remaining $k-1$ packets face the acceptance probability sequence $x_1, x_2, \ldots, x_{k-1}$. So

$$f_k(x_1, x_2, \ldots, x_k) = x_1 f_{k-1}(x_2, x_3, \ldots, x_k) + (1 - x_1) f_{k-1}(x_1, x_2, \ldots, x_{k-1}). \qquad (27)$$

Although this does not simplify in general, we are always interested in the case where the x_j decrease linearly with j so $x_j = A - j\Delta$ for some constant A. In this case we shall show that

$$f_k(x_1, x_2, \ldots, x_k) = x_1 (1 - \Delta)^{k-1}. \qquad (28)$$

For when $k = 1$ this is just (25) and for $k > 1$ we proceed by induction, so that under the inductive hypothesis the right-hand-side of (27) is

$$x_1 x_2 (1-\Delta)^{k-2} + (1-x_1) x_1 (1-\Delta)^{k-2} = x_1 (1 - x_1 + x_2)(1-\Delta)^{k-2} = x_1 (1-\Delta)^{k-1}, \quad (29)$$

as required.

An alternative derivation of this result is to use indicator variables

$$Y_j = \begin{cases} 1 & \text{if the } j\text{-th packet is accepted} \\ 0 & \text{if it is discarded.} \end{cases} \qquad (30)$$

Then the dependence of Y_j on the previous acceptances $Y_1, Y_2, \ldots, Y_{j-1}$ is

$$Y_j = \begin{cases} 1 & \text{with probability } p_{Y_1+Y_2+\ldots+Y_{j-1}+1} \\ 0 & \text{otherwise,} \end{cases} \qquad (31)$$

i.e. $Y_j = 1$ with probability $A - (Y_1 + Y_2 + \ldots + Y_{j-1} + 1)\Delta$. Thus the conditional expectation of Y_j is

$$\mathbb{E}\mathrm{x}(Y_j | Y_1, Y_2, \ldots, Y_{j-1}) = A - (Y_1 + Y_2 + \ldots + Y_{j-1} + 1)\Delta. \qquad (32)$$

So a_j, which is the unconditional probability of acceptance of the j-th packet, *i.e.* the unconditional expectation of Y_j, is

$$\begin{aligned} a_j = \mathbb{E}\mathrm{x}(Y_j) &= A - (\mathbb{E}\mathrm{x}(Y_1) + \mathbb{E}\mathrm{x}(Y_2) + \ldots + \mathbb{E}\mathrm{x}(Y_{j-1}) + 1)\Delta \\ &= A - (a_1 + a_2 + \ldots + a_{j-1} + 1)\Delta. & (33) \end{aligned}$$

Applying this recurrence for a_j with j replaced by $j-1$ we see we can write it as $a_j = a_{j-1} - a_{j-1}\Delta$, so $a_j = a_1(1-\Delta)^{j-1}$ and we obtain the same result as previously.

We notice that it depends crucially on the right-hand-side of (32) being *linear* in the $Y_{j'}$ ($j' < j$) so that we can take the expectation regardless of the correlations between the $Y_{j'}$. This linearity in turn arises from the linear dependence of the discard probability on buffer occupancy.

Combining the results we have

$$\mathbb{E}\mathrm{x}(j'|j,k) = \begin{cases} j+k & \text{if } j+k \le O_{\min} \\ O_{\min} + (B - O_{\min})\big(1-(1-\Delta)^{k+j-O_{\min}}\big) & \text{if } j \le O_{\min} \le j+k \\ j + (B-j)\big(1-(1-\Delta)^k\big) & \text{if } j \ge O_{\min}. \end{cases} \tag{34}$$

Furthermore, when we modify these results to represent the continuum limit, by replacing $(1-\Delta)^k$ by $\exp(-\Delta k)$, they agree exactly with those of the differential equation approach. We have given this variety of approaches to illustrate different methods, some of which may generalise more easily than others to future variants of the problem.

4.2 Transition probabilities in detail

We now return to the question that we posed before considering the expected number of packets accepted, namely what are the transition probabilities, and in particular if the occupancy is $j \ge O_{\min}$ and k packets arrive, what is the transition probability to occupancy j' ($j \le j' \le \min(j+k, B)$). We wish to know the full distribution, of which (34) is the mean.

We shall show that this is

$$p(j+l|j,k) = p_j p_{j+1} \cdots p_{j+l-1} \Delta^{k-l} M(k, J+l, l)/l!, \tag{35}$$

where $J = j - O_{\min}$ and $M(k,r,l)$ is the number of mappings of k points into r points that cover a specified l-element subset. Explicitly, using the principle of inclusion and exclusion, we can write

$$\begin{aligned} M(k,r,l) &= \sum_{s=0}^{l} (-1)^s \binom{l}{s} (r-s)^k & (36) \\ &= r^k - \binom{l}{1}(r-1)^k + \binom{l}{2}(r-2)^k - \ldots + (-1)^l (r-l)^k. & (37) \end{aligned}$$

Since $M(k,r,0) = r^k$ and $M(k,r,k) = k!$, (35) certainly gives the correct results $p(j|j,k) = (J\Delta)^k = (1-p_j)^k$ for no acceptances and $p(j+k|j,k) = p_j p_{j+1} \cdots p_{j+l-1}$ for k acceptances. To show the result in full, consider the probability that l packets are accepted out of k submitted: either $l-1$ are accepted out of the first $k-1$ and then the final packet is accepted, or l are accepted out of the first $k-1$ and the final packet is rejected. So

$$p(j+l|j,k) = p(j+l-1|j,k-1)p_{j+l-1} + p(j+l|j,k-1)(1-p_{j+l}). \tag{38}$$

Here the first term is only present if $k \geq 1$ and the second only if $l \leq k-1$. Substituting (35) into this recurrence relation we find we need to check that

$$M(k,r,l) = lM(k-1,r-1,l-1) + rM(k-1,r,l). \tag{39}$$

However, this certainly holds since maps counted on the left can be divided up into those where the first $k-1$ points cover only some $l-1$ of the specified points and those where they already cover all l of the specified points. In the first case there are l choices of which of the l points is *not* covered, and then $M(k-1,r-1,l-1)$ ways in which the $k-1$ points can cover those $l-1$ specified points but not the omitted one. In the second case there are $M(k-1,r,l)$ ways in which the first $k-1$ points can already cover the l specified points, and then the k-th point can map onto any of the r points in the target set.