On Dropping Sequences for RED

Bruno Gaujal
INRIA, ENS Lyon and Laboratoire ID-IMAG,
51 av. J. Kuntzman,
38330 Montbonnot
Email: Bruno.Gaujal@imag.fr

Arie Hordijk
Mathematical Institute,
Leiden University, The Netherlands.
Email: Arie.Hordijk@math.leidenuniv.nl

Abstract—1 In this note, we show how to compare the dropping sequences in the Random Early Detection Algorithm once all the parameters have been fixed. This is done for a single node, as well as for one TCP connection. The method proposed here uses convexity properties to provide comparing tools for all types of dropping schemes. In particular, we compare Bernoulli with uniform dropping. We also provide the optimal dropping scheme in terms of workload for the single node and end-to-end delay for the connection.

I. INTRODUCTION

Random Early Detection (RED) algorithms have been proposed to reduce bursts of losses that occur when a congestion occurs in a TCP connection in [1], [2]. The main idea of the algorithm is to anticipate the congestion at each node in the connection and to start dropping packets before the congestion actually occurs (for example when the size of the buffer reaches a critical threshold). This smoothes the loss process and may help to increase the overall throughput of the TCP connection.

An extensive literature exists on RED that addresses the issues of parameters choices [3], [4], [5], [6], as this proves to be a difficult task [7]. Many papers are trying to overcome those difficulties by adapting the parameters of RED to the network state [8], [7] while some others even question RED’s efficiency [9].

In this paper we focus on the classical RED proposed in [1] where the dropping policy depends on a single parameter $q$, the dropping probability. Of course $q$ is a function on the moving average queue length $\hat{K}$. Many experimental studies where devoted to the choice of the best function $q(\hat{K})$. Some of these experimental studies [8], [10] suggest that $q$ should not change too fast over time: the better results where obtained when $q$ remains constant for long periods of time.

However in this huge literature, it seems that, one issue has been overlooked.

Once $q$ (and all other parameters of RED) have been chosen, what should the dropping policy be? This paper addresses this issue on several levels.

First, (Section II) we consider a single node and the queue length (or the waiting time) as the cost function to be minimized. Assuming that $q$ remains constant over long period of times, we compare two classical dropping schemes proposed for RED: geometrically distributed and uniform. By using convex stochastic ordering techniques, we show that the uniform dropping yields better results than the geometrical one.

Secondly (Section III) by using multimodularity properties, we show that the best dropping scheme is deterministic, driven by a Sturmian sequence.

The last part considers a complete TCP connection (and not a single node) where RED is only used at one node. Again, we show that the best dropping scheme is Sturmian.

II. DROPPING IN AN ISOLATED NODE

A. The model

Let us first introduce the model used here. In this part we consider a stationary $G/G/1$ queue where RED is used to drop packets.

We assume that the inter-arrival times of the packets form a stationary sequence $\tau_n$ for all $n \in \mathbb{N}$. The service times $\sigma_n$ also form a stationary sequence, independent of the arrival times. We also assume that the queue is initially empty.

In the following, $\tau_1 + \cdots + \tau_n$ is called the $n$-th arrival slot. Note that because of the dropping, this time may not correspond to the actual arrival of a packet in the queue. 

---

1This work has been done while Arie Hordijk was invited professor at ENS Lyon. This work was also supported by the Van Gogh grant on Discrete Event Systems and Optimization.
Let \( D(q) \) (or \( D \) for short when no confusion is possible) denote the infinite packet interval dropping sequence for a given fixed \( q \). This means that the packets arriving at time slots \( \tau_1 + \cdot \cdot \cdot + \tau_{D_1}, \cdot \cdot \cdot, \tau_1 + \cdot \cdot \cdot + \tau_{D_1+\cdot \cdot \cdot + D_n}, \cdot \cdot \cdot \) are dropped.

The main assumption used here is that \( q \) varies slowly in time (as recommended by most practical users of RED as well as by new versions of the algorithm [10]). Therefore, we will assume that \( q \) remains constant for some time (large w.r.t. the dropping sequence). For simplicity, we also assume that the dropping schemes have a negligible influence on \( q \) as long as they all have the same average dropping intensity.

Let \( N(q) = [1/2q] \). When no confusion is possible, we denote it by \( N \).

In the following, we will compare several classical dropping schemes recommended for RED:

1. \( D_k(q) \) is uniform over \( [N - k + 1, N + k] \) for all \( k = 0, 1, \cdot \cdot \cdot, N \). \( (D_U(q) \) is uniform over \( [1, 2N] \)

2. \( D_G(q) \) is geometric with dropping probabilities \( q \).

**B. Convex ordering theorems**

Let us first mention one theorem on the convex ordering and see how they can be applied in our context.

**Theorem 1 ([11]):** \( D_1 \) is smaller or equal in the (increasing) convex ordering than \( D_2 \) (denoted \( D_1 \preceq \) \( D_2 \)) if for any (increasing) convex function \( f \), it holds that

\[
\mathbb{E}f(D_1) \leq \mathbb{E}f(D_2).
\]

Let us state a first result on \( D_k(q) \) and \( D_G(q) \)

**Lemma 2:**

\[
D_k(q) \preceq D_\ell(q) \quad k, \ell, \\
D_U(q) \preceq D_G(q).
\]

**Proof:** First note that \( D_k \) and \( D_G \) have the same mean: \( \mathbb{E}D = N \). We will use the cut criterion of Karlin and Nomikoff (see Def. 1.5.16 and Th. 1.5.17 in [11]) which says that \( D_1 \preceq D_2 \) if for some \( t_0 \), the distribution function of \( D_1 \) (denoted \( F_1 \)) is below that of \( D_2 \) (denoted \( F_2 \)) for all \( t \leq t_0 \) and \( F_1(t) \leq F_2(t) \) for all \( t \geq t_0 \).

It is easy to see that if \( k \leq \ell \), then

\[
F_{D_k(q)}(t) \leq F_{D_\ell(q)}(t) \quad \forall t \leq N,
\]

and

\[
F_{D_k(q)}(t) \geq F_{D_\ell(q)}(t) \quad \forall t \geq N.
\]

As for the second inequality, \( P(D_G \geq k) = (1-q)^k, \quad k = 0, 1, \cdot \cdot \cdot \). If we consider the continuous distribution \( F(t) = 1 - \exp(t \ln(1 - q)) \), it is easily checked that it is concave and that the derivative at \( t = 0 \) is \( -\ln(1 - q) \) with \( q = 1/2N \). This is larger than \( 1/2N \), the slope of \( F_{D_U} \) at \( t = 0 \).

Hence \( F \) and \( F_U \) verifies the cutting criterion. Since \( F \) equals \( F_{D_G} \) on the integer points, \( F_G \) and \( F_U \) also verifies the cutting criterion. This completes the proof of the lemma.

Now let us assume that the parameter \( q \) is random and distributed independently of the \( D \)'s. Then the convex ordering in the above lemma remains true (see [11] th. 1.5.6).

Consider the \( n \) tuple of independent dropping times say

\[
D_1^i, \cdot \cdot \cdot D_n^i
\]

where for \( i = 0, \cdot \cdot \cdot, N \) and \( i = G \), they are independent copies of \( D_i(q) \) and of \( D_G(q) \) respectively.

As an application of Strassen’s theorem and using the above lemma, there exist independent stochastic variables \( D \) such that \( D^i_k \) has the same distribution as \( D_k \) and

\[
\mathbb{E} (\hat{D}^i_k | \hat{D}^j_k) = \hat{D}^i_k \quad \text{if} \quad j \leq i \quad \text{or} \quad i = G.
\]

Hence,

\[
\mathbb{E} (\hat{D}^i_1 \cdot \cdot \cdot \hat{D}^i_n) | (\hat{D}^j_1 \cdot \cdot \cdot \hat{D}^j_n) = (\hat{D}^j_1 \cdot \cdot \cdot \hat{D}^j_n)
\]

This means that

\[
(D^i_1 \cdot \cdot \cdot D^i_n) \preceq \text{ex} (D^j_1 \cdot \cdot \cdot D^j_n).
\]

**C. Multimodularity**

Now, we show that the performance criterion mentioned in the abstract (expected workload) is multimodular.

A function \( f : \mathbb{Z}^n \rightarrow \mathbb{R} \) is multimodular with respect to the base \( b_0 \cdot \cdot \cdot b_n \) of vectors that sum to zero \((b_0 + \cdot \cdot \cdot + b_n = 0)\) if for all \( 0 \leq i \neq j \leq n \) and all \( x \in \mathbb{Z}^n \),

\[
f(x + b_i) + f(x + b_j) \geq f(x) + f(x + b_i + b_j).
\]

One of the most common base is the left shift base:

\[
s_0 = (-1, 0, 0, \cdot \cdot \cdot, 0) \\
s_1 = (1, -1, 0, \cdot \cdot \cdot, 0) \\
\vdots \\
s_n = (0, \cdot \cdot \cdot, 0, 1, -1) \\
s_n = (0, \cdot \cdot \cdot, 0, 1, 0, \cdot \cdot \cdot).
\]

For more on multimodularity and its links with convexity, please refer to [12].
Consider a fixed $n$-tuple of inter-dropping times say $D = D_1 \cdots D_n$ (note that this sequence is finite: only $n$ packets are dropped in total).

Let $W_k(D)$ be the expectation w.r.t. the service times and the inter-arrival times, of the workload at slot $k$, under the dropping sequence $D_1 \cdots D_n$.

**Theorem 3:** For all $k \in \mathbb{N}$, $W_k(D)$ is multimodal in $D$ with respect to the left shift base.

**Proof:** The inter-dropping times $D_i$ determine an infinite binary *admission* sequence $(a_k)_{k \in \mathbb{N}}$ on the slots, saying which packets are accepted in the queue: $a_k = 0$ if $k = D_1 + \cdots + D_m$ for some $m, 1 \leq m \leq n$, and $a_k = 1$ otherwise. Therefore, one can also write $W_k(D) = W_k(a_1 \cdots a_k)$, for all $k \in \mathbb{N}$.

Now, we consider a restriction to the case where all inter-dropping times are bounded by $B$: $D_i \leq B$, $0 \leq i \leq n$. We call $W_k^B(D)$ the expected workload, under that restriction, so that $W_k(D) = W_k^\infty(D)$, and we set $k_B = nB + 1$.

We first check multimodularity of the function $W_{k_B}^B(D)$. For that, we consider the left shifts of size $n$:

\[
\begin{align*}
S_0 &= (-1,0,0,\ldots,0) \\
S_1 &= (1,-1,0,\ldots,0) \\
&\vdots \\
S_{n-1} &= (0,0,\ldots,0,1,-1) \\
S_n &= ((0,0,\ldots,0,1,1),
\end{align*}
\]

and the left shifts of size $k_B$:

\[
\begin{align*}
s_0 &= (-1,0,0,\ldots,0) \\
s_1 &= (1,-1,0,\ldots,0) \\
&\vdots \\
s_{k_B-1} &= (0,\ldots,0,1,-1) \\
s_{k_B} &= ((0,\ldots,0,0,1).
\end{align*}
\]

It should be clear that by definition of the admission sequence, $W_{k_B}^B(D + S_i) = W_{k_B}^B(a + s_{D_1 + \cdots + D_i})$.

The multimodularity of the function $W_{k_B}^\infty(a)$ was shown in [12, Theorem 17]. This implies the multimodularity of the function $W_{k_B}^B(a)$, which is the restriction of $W_{k_B}^\infty(a)$ to the sub-mesh where the inter-dropping times are all bounded by $B$ (by [12, Lemma 69]). In turn, this means that for all $i,j \leq k_B$, $\tilde{W}_{k_B}^B(a+s_i)+\tilde{W}_{k_B}^B(a+s_j) \geq \tilde{W}_{k_B}^B(a) + \tilde{W}_{k_B}^B(a+s_i+s_j)$. In particular (by choosing the good shifts in the second base), this implies

\[
\begin{align*}
W_{k_B}^B(D + S_i) + W_{k_B}^B(D + S_m) \\
\geq W_{k_B}^B(D) + W_{k_B}^B(D + S_{\ell} + S_m).
\end{align*}
\]

By restricting to the sub-mesh of the $k$ first coordinates ($k \leq k_B$), one gets by [12, Lemma 69], the multimodularity of $W_k^B(D)$.

To finish the proof, it is enough to take $B_D \geq \max(D_1,\ldots,D_n)$, to get the multimodularity of $W_k^B(D) = W_k^B(D)$.

As a corollary, the number of packets in the queue at the $k$th slot is convex in $D_1 \cdots D_n$.

**D. Applications**

By a direct application of Lemma 62 of [12], where the $D_i$ tuple is replacing the $A_i$ tuple, we get the following result for the expected workloads at slots $1,\ldots,n$ (denoted $W_1^i,\ldots,W_n^i$) under the respective dropping schemes $i = 0,\ldots,N$ or $i = G$.

**Theorem 4:** Consider the dropping schemes $i$ and $j$ with $i \leq j$ or $i = G$. Then the expected workloads are convex ordered

\[ (W_1^i,\ldots,W_n^i) \leq_{\text{lex}} (W_1^j,\ldots,W_n^j). \]

Moreover, Corollary 67 in [12] implies a similar result for the stationary workloads, say $W_1^\infty$ and $W_\infty$. We get in particular $W_1^\infty \leq_{\text{lex}} W_\infty^G$, in other words, the stationary workload is in convex ordering smaller under the uniform dropping than under the geometric dropping. Since $W_1^\infty$ and $W_\infty^G$ are non-negative random variables, this means that not only the mean workload is smaller under the uniform dropping scheme but also all higher moments are smaller:

\[ \mathbb{E}(W_1^\infty)^k \leq_{\text{lex}} \mathbb{E}(W_\infty^G)^k. \]

**Remark** In order to have a fair comparison between the uniform and the geometric scheme, we chose the uniform distribution on $[1,\cdots,2N]$, in order to compare schemes with the same mean $N$.

**E. Optimal dropping sequence**

By using [12, Theorem 20] on the minimization of multimodal functions, the best dropping sequence with respect to minimizing the expected workload in a single node is a deterministic dropping scheme with $i = 0$, i.e. dropping only the $N$-th arriving packet. It is easily implemented in the RED algorithm as follows:

\[ \text{drop packet number } k \text{ if } [kq] - [(k-1)q] = 1. \]

**III. Dropping in One Node of a TCP Connection**

The previous section shows how to drop in a single node. This may not work in a complete TCP connection,
since the effect of the dropping scheme on the end to end TCP connection is still to be evaluated.

Here we consider a single saturated TCP connection (there is always one packet waiting to be sent) where RED is used in a single node of the connection. We sketch how this can be modeled by a multimodular function of the dropping sequence.

Here are the main assumptions used to construct the model:

- cross traffic is seen as perturbations of the service time in all queues, independent of the arrival times.

- Losses are also independent of the arrival and service times.

Therefore, services times in all queues form stationary sequences (possibly correlated with each other) but independent of the arrival times.

- The key remark here is that for a fixed sequence \( w \) the \( X \) is a matrix corresponding to the service times, \( S \) is the inter-arrival times after droppings. \( S \) goes similarly to that of Section 3.5 of [12], the following result can be shown.

\[
\text{Theorem 5: Under the foregoing assumption, the expected sojourn time (w.r.t. the service times in all nodes) of packet } n \text{ in a saturated TCP connection with one RED node, } \mathbb{E}S_n(D), \text{ is multimodular in } D \text{ w.r.t. the left shifts.}
\]

Using this result, we can compare two inter-dropping sequences just as we did for the G/G/1 case and have the following lemma with a proof similar to that of Lemma 62 of [12].

\[
\text{Theorem 6: Consider the dropping schemes } i \text{ and } j \text{ with } i \leq j \text{ or } j = G \text{ in one node of a saturated TCP connection. Then the waiting times are convex ordered,}
\]

\[
(\mathbb{E}S^i_1, \ldots, \mathbb{E}S^i_n) \leq_{\text{exc}} (\mathbb{E}S^j_1, \ldots, \mathbb{E}S^j_n).
\]

Furthermore, the optimal dropping sequence minimizing the expected end to end delay is deterministic and drops one every \( N(q) \) packets.

### IV. Conclusion and Perspectives

While we have shown how to make the dropping in RED optimal, we do not address the classical problems of fixing \( q \), nor do we claim that our proposal makes RED worth implementing. Further work is needed to assess the effectiveness of the methods proposed here. In particular, it would be interesting to see (by simulation for example) how the choice of optimal dropping sequences may or may not make RED really efficient.

If RED is used in more than one node of the TCP connection, then the approach used here may still work by conditioning of the dropping sequences of all RED nodes but one, for the behavior of the TCP connection can still be put under (max,plus) form. However, this needs further investigations.

The deterministic nature of the optimal dropping sequence at each node may affect the overall performance. Indeed, it may introduce some bad synchronization mechanisms.

To avoid synchronization problems for the end-to-end delay, the optimal dropping can be randomized by generating a random variable \( \theta \in [0, 1) \) with a uniform distribution any time \( q \) is updated. The new optimal dropping scheme becomes: drop packet number \( k \) if \( \lfloor kq + \theta \rfloor - \lfloor (k-1)q + \theta \rfloor = 1 \). It would be interesting to see how these random initial shifts at each node can cancel these synchronizations.
REFERENCES


