Analysis of the Early Packet Discard with per-VC Queueing Mechanism

K. Spaey C. Blondia
University of Antwerp
Department of Mathematics and Computer Science
Performance Analysis of Telecommunication Systems Research Group
Universiteitsplein 1
B-2610 Wilrijk - Belgium
{spaey,blondia}@uia.ua.ac.be

Abstract

The Early Packet Discard (EPD) with per-VC Queueing mechanism is a packet discard scheme proposed in the literature to improve EPD performance in terms of fairness. This paper develops an analytical approximation model for evaluating the performance of the mechanism at a single node. As illustrated in the paper by numerical examples, the model allows the study of the evolution of performance measures such as buffer occupation, packet discard ratio and cell loss ratio when parameters of the input traffic or of the discarding mechanism are changed.

1 Introduction

The Unspecified Bit Rate (UBR) service category is intended for data applications that are not sensitive to delay and which generate bursty traffic that is difficult to characterize. Since no QoS commitments are made to connections of this service category, the network delivers a best effort service. It is however clear that when congestion in a network element occurs, each lost cell may belong to a different data packet. This leads to a low packet throughput (also called goodput), since losing a cell of a packet implies that the packet to which the cell belongs cannot be reassembled at the destination. Thus, once a cell of a packet is lost, all transmissions of subsequent cells of the packet are useless cell transmissions. Therefore, intelligent packet discarding mechanisms have been proposed and associated with the UBR service category, although they are not restricted to this service category only [3].

Packet discard mechanisms drop AAL5 packets through the use of the ATM User-to-User (AUU) bit in the Payload Type Identifier (PTI) field in the ATM header. Two well-known examples are Early Packet Discard (EPD) and Partial Packet Discard (PPD) [11]. EPD discards complete packets when a buffer threshold is exceeded, while PPD is triggered by buffer overflow and discards the remaining cells of packets from which a cell is lost.

Although the goodput may be increased considerably by using these packet discard schemes, they can not guarantee fairness [5, 10]. To avoid unfairness between competing flows, also per-flow information has to be incorporated into the mechanisms. Unfairness occurs because connections which lose packets will be forced by the transport protocol (e.g. TCP) to slow down their transmission rate, allowing other connections which did not loose packets to use more bandwidth and buffer space. Simulations show that schemes such as e.g. EPD with per-VC accounting, selective drop with per-VC accounting and EPD with per-VC queueing, alleviate the unfairness problem [5, 10]. The two schemes with per-VC accounting maintain fair buffer allocation at the time of cell discarding, while EPD with per-VC queueing also maintains fairness in the throughput as long as each VC has some cells in its queue.

In this paper we consider an analytical model for EPD with per-VC queueing. In the literature, a lot of exact analytical models are available for the EPD and PPD schemes [7, 9, 8]. A common factor in those papers is that the distinction between the normal mode of the switch in which arriving cells are admitted and the discarding mode in which arriving cells are discarded is made in the source models. Since in EPD with per-VC queueing a round robin scheduler is used, constructing an exact model for this scheme will lead to an extremely large and thus analytically untractable model. Therefore, our analysis will follow an approximate approach, combining the source models that are used in the exact models for EPD and the approximate vacation models that are often used to model cyclic service systems [1, 4].
This paper is organized as follows. In Section 2, we recall the EPD with per-VC queueing discarding scheme in detail. Section 3 introduces the model of the system, and Section 4 discusses the iterative algorithm to solve the system. Section 5 shows how performance measures can be obtained, while Section 6 presents a validation of the model and several numerical results. Section 7 concludes the paper.

2 Early Packet Discard with per-VC Queueing

This section describes the EPD with per-VC queueing mechanism using the pseudocode of [10]. Cells from different VCs heading for the same output interface of an ATM switch will be buffered into different VC queues. In order to maintain fair buffer allocation, a round robin scheduler is employed to serve all VC queues. Denote Q_j as the queue length for VC_j, Q as the total queue size, Th as the EPD threshold, N as the number of active VCs (a VC is said to be active if it has at least one cell in its queue) and $Q_{\max}$ as the maximum available buffer capacity. K is a control parameter (typically, $1 \leq K \leq 2$). In pseudocode, the algorithm then looks as follows:

```
When a cell of VC_j arrives at the buffer:
    if the cell is the first cell of a packet
        calculate \overline{Th} = K*Q / N
        if Q ≥ Th and Q_j ≥ \overline{Th}
            discard the cell
        else
            accept the cell into VC_j's queue
            Q_j = Q_j + 1
    else
        if the first cell of the packet has been discarded
            discard the cell
        else
            if Q = Q_max
                discard the cell
            else
                accept the cell into VC_j's queue
                Q_j = Q_j + 1
```

Remark that whether a packet is discarded or not depends on two thresholds. If the buffer occupancy is below the fixed EPD threshold Th, no packets are discarded. If this threshold is exceeded, packet discarding is possible, depending on the second threshold $\overline{Th}$, which is calculated as $\overline{Th} = \frac{K*Q}{N}$. Since Q and N vary in time, also $\overline{Th}$ varies in time. When $K = 1$, $\overline{Th}$ represents the average buffer occupancy per VC. In the case that cells of a VC from which a new packet arrives already occupy more than a fair amount $\overline{Th}$ of the buffer space, all the cells of the new packet are discarded.

3 Model Description

3.1 The System Model

Consider a discrete-time queueing system in which the time needed to transmit a cell, called a time slot, is chosen as time unit. The system consists of a buffer with maximum capacity $Q_{\max}$ in which the traffic of $R+1$ VCs arrives. One of these VCs is tagged, while the R other VCs are considered as background VCs. Denote $Q_t(n)$ as the queue length of the tagged VC in time slot n ($0 \leq Q_t(n) \leq Q_{\max}$) and $Q(n)$ as the total queue size at time n ($Q_t(n) \leq Q(n) \leq Q_{\max}$).

3.2 The Input Process

Traffic for each of the VCs is generated by an on/off source, which changes its state from active to silent with probability α, and from silent to active with probability β. When the source is active, a cell is generated in a time slot with probability λ_{on}. All cells which arrive during the same active period of the source constitute a packet. The packet length in number of cells is geometrically distributed with mean (see [7])

$$q = \frac{1-(1-\alpha)(1-\lambda_{\text{on}})}{\alpha}. \tag{1}$$

Because of the packet discard mechanism, not all cells generated by the sources will effectively enter the buffer. To model this, we define the states of each source similar as in [7]:

- off: the source does not generate a packet,
- on: the source is transmitting a packet of which the cells will not be discarded by the discarding mechanism (but they may be lost due to buffer overflow),
- on*: the source is transmitting a packet of which the cells will be discarded by the discarding mechanism.

So the source is active while it is in the on or on* state, but only the cells generated during an on period will effectively enter the buffer (except if they are lost due to buffer overflow).
We will suppose that all $R+1$ sources are homogeneous in terms of traffic characteristics, and call the source which corresponds with the tagged VC the tagged source, the other R sources are called background sources.
The state of the input process at the n-th slot can then completely be described by

- $S_{\text{on}}(n)$: the number of background sources in the on state,
- $S_{\text{on}^*}(n)$: the number of background sources in the on* state,
- $S_t(n)$: the state of the tagged source,

with $0 \leq S_{\text{on}}(n) \leq R$, $0 \leq S_{\text{on}^*}(n) \leq R - S_{\text{on}}(n)$, $S_t(n) \in \{\text{on}, \text{on}^*, \text{off}\}$.
The actual cell arrival process at the switch will be different depending on if (i) $Q < Th$, (ii) $Q \geq Th$, $Q_t < \overline{Th}$ and d of the R background queues exceed $\overline{Th}$, or (iii) $Q \geq Th$, $Q_t \geq \overline{Th}$ and d of the R background queues exceed $\overline{Th}$. We refer to [12] for the mathematical description of this arrival process.

3.3 The Service Process

The queues are served according to a non-exhaustive (limited to one) cyclic service strategy, where one cell (if present) is served at each scanning epoch of a queue. The service time equals one time slot. At the end of this service, the server moves to the next queue. The switch-over time is supposed to be zero.

In principle, we would have to derive the joint queue length distribution of this multi-queueing system. But as the state space may become very large, we make an approximate analysis for each queue in the system separately by deriving the joint probability distribution of the length of this queue and the total queue length. We model each queue as a queueing system with repeated server vacations. The time between two consecutive scanning instants of a queue k is called a k-cycle. As before, we tag a queue and refer to the other queues as background queues. If the tagged queue is not empty at a scanning instant, its first cell is served. If after this service not all background queues are empty, the server will be unavailable for the tagged queue during a random time v_2, servicing one cell in each of the other non-empty queues of the system. Let $v_{2,l} = P\{v_2 = l\}$, where $l \in \{1, \dots, R\}$. So the cycle of the tagged queue consists in this case of a service followed by a vacation. If after the service of the tagged queue all background queues are empty, the cycle consists of only one slot, namely that of the service. If the tagged queue is empty at a scanning instant, but the system is not, the server will be unavailable for the tagged queue during a random time v_1, with $v_{1,l} = P\{v_1 = l\}$ ($l \in \{1, \dots, R\}$), to serve one cell in each of the non-empty background queues. In this case, the length of the cycle of the tagged queue equals the length of the vacation. If the system is completely empty at the scanning instant, the cycle of the tagged queue will be exactly one slot, after which the server will check the system again, to see if new cells have arrived meanwhile.
Remark that during a vacation, the system may not enter a state from which a service of the background queues is impossible, i.e. a state in which there are no cells in the background queues. But this implies that at all but the last slot of the vacation, the system may not be in a state where there is only one cell in the background queues and no background source in the on state, since if this would be the case, then at the next slot the system would be in one of the states we excluded before. On its turn, this restriction implies that at all but the last and last but one slot of the vacation, the system may not enter a state where there are two cells in the background queues and all background sources are in state on*, since a source has always to pass from on* via off before it can possibly go to on, and thus the system again would enter at the next slot a state of the excluded type.

3.4 The Queue Length Distribution

We describe the state of our approximate system at time n by the discrete-time Markov chain with states $\big(Q_t(n), Q(n), S_{\text{on}}(n), S_{\text{on}^*}(n), S_t(n)\big)$. Denote the state space of this chain by $\mathcal{A}$. To describe how this process evolves from slot to slot, consider all different types of time slots which exist: a time slot can fall

- during a vacation of the tagged queue,
  - as the last slot of the vacation,
  - as the last but one slot of the vacation,
  - as the third-last slot of the vacation,
  - as a slot which is not of the type mentioned above,
- during a service of the tagged queue,
- during an idle period of the server because the system is empty.

To describe the evolution of the system from slot n to slot $n+1$, we have to be able to calculate $\overline{Th}$ at every slot, and to decide how many of the background queues exceed this $\overline{Th}$. But since the system state only provides the queue length of the tagged queue and of the total queue, we have to approximate the number of queues which are non-empty and the number of background queues which exceed $\overline{Th}$ at time n. Define $p_i(u, v, j, k, l)$ and $q_d(u, v, i, j, k, l)$ as

> $p_i(u, v, j, k, l)$ is the probability that there are i VCs with a non-empty queue when $Q_t = u$, $Q = v$, $S_{\text{on}} = j$, $S_{\text{on}^*} = k$ and $S_t = l$,

and

> $q_d(u,v,i,j,k,l)$ is the probability that d of the background queues exceed $\overline{Th}$ when $Q_t = u$, $Q = v$, $S_{\text{on}} = j$, $S_{\text{on}^*} = k$, $S_t = l$ and there are i VCs with a non-empty queue.

For a discussion of how to determine $p_i(u,v,j,k,l)$ and $q_d(u,v,i,R,0,\text{off})$, and for a mathematical derivation of the slot to slot evolution of the system for the different types of slots mentioned above, we refer to [12].
From the slot to slot evolution of the system, the evolution of the system between two scanning instants can be derived. Suppose that the latter is described by the transition matrix $\boldsymbol{Q}$, with $\mathcal{A} \times \mathcal{A}$ as state space. If we divide $\boldsymbol{Q}$ into blocks $\boldsymbol{Q}_{i,j}$ $(i,j \in \{0,\ldots,Q_{\max}\})$, where i and j correspond with the number of customers in the tagged queue, then the blocks $\boldsymbol{Q}_{i,j}$ with $j < i-1$ are $\boldsymbol{0}$, since at most one cell can leave the tagged queue between two scanning instants. Also the blocks $\boldsymbol{Q}_{0,j}$ with $j > R$ and the blocks $\boldsymbol{Q}_{i,j}$ with $i > 0$ and $j > R+i$ are $\boldsymbol{0}$, since a vacation of the tagged queue lasts maximum R slots, so at most R cells can arrive in the tagged queue when $i = 0$, and at most $R+1$ cells can arrive in the tagged queue when $i > 0$. But in the last case, one cell will leave the system during service of the tagged queue, resulting in a maximum net growth of R cells. Remark that only the diagonal blocks $\boldsymbol{Q}_{i,i}$ of $\boldsymbol{Q}$ are square, and they are becoming smaller for increasing i. Denote the stationary distribution of $\boldsymbol{Q}$ by $\boldsymbol{q}$, i.e.

$$\boldsymbol{q}\boldsymbol{Q} = \boldsymbol{q} \quad \text{and} \quad \boldsymbol{q}\boldsymbol{e} = 1. \tag{2}$$

Because of the upper block-Hessenberg structure of $\boldsymbol{Q}$, we can find $\boldsymbol{q}$ by applying for example the algorithm of Grassmann et al. [6] extended to block partitioned matrices as explained in [2]. This algorithm has no problems dealing with the non-equal sized blocks $\boldsymbol{Q}_{i,j}$. From $\boldsymbol{q}$ and the slot to slot evolution of the system it is possible to derive $\boldsymbol{z}$, the stationary joint distribution of the queue length of the tagged queue and the queue length of the total queue at arbitrary instants. Again, we refer to [12] for the details of this derivation.

4 Calculating the Queue Length Distributions

In Section 3 we have described how to model the multi-queueing system by considering one tagged queue together with the total queue. The following algorithm describes how $\boldsymbol{q}$, resp. $\boldsymbol{z}$, the stationary joint distribution of the queue length of the tagged and the queue length of the total queue can be computed at a scanning instant of the tagged queue, resp. at an arbitrary instant. Define $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ as

> $\boldsymbol{\phi}$, resp. $\boldsymbol{\psi}$, is the stationary joint queue length distribution of the tagged and the total queue at scanning instants of the tagged queue during a cycle of a particular background queue, knowing that this background queue was empty, resp. non-empty at its scanning instant,

and define $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$ as

> $\hat{\boldsymbol{\phi}}$, resp. $\hat{\boldsymbol{\psi}}$, is the stationary joint queue length distribution of the tagged and the total queue at an arbitrary instant, knowing that during a cycle of a particular background queue, this background queue was empty, resp. non-empty at its scanning instant.

We will calculate these stationary distributions in an iterative way: in every iteration step we assume we have these distributions available (the 'old' distributions), we use them to calculate all unknows (vacation distribution, $p_i(u,v,j,k,l)$ and $q_d(u,v,i,j,k,l)$) needed to construct $\boldsymbol{Q}$, and from $\boldsymbol{Q}$ and the vacation distributions we obtain the 'new' stationary distributions. This results in the algorithm below, where the vacation distributions $v_3,\ldots,v_8$ are defined as follows:

v_3, v_4, v_5 and v_6 are the lengths of a vacation during a cycle of the tagged queue, conditioned on the facts that the tagged queue was empty (for v_3 and v_5) / non-empty (for v_4 and v_6) at the start of the cycle and that a particular background queue is empty (for v_3 and v_4) / non-empty (for v_5 and v_6) at its scanning instant during this cycle.

v_7 resp. v_8 is the length of a vacation during a cycle of the tagged queue, conditioned on the fact that the tagged queue was empty, resp. non-empty at the start of the cycle.

Formulas to calculate $v_3 \ldots v_8$ can be found in [12].

Algorithm to calculate the queue length distributions:

1. Assign initial distributions to $\boldsymbol{\phi}$, $\boldsymbol{\psi}$, $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$.
2. Calculate
   (a) the vacation distributions v_3, v_4, v_5 and v_6 using $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$,
   (b) all the values $p_i(u, v, j, k, l)$ and $q_d(u, v, i, j, k, l)$ using $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$.
3. Compose $\boldsymbol{Q}$, and calculate its stationary distribution $\boldsymbol{q}$,
   (a) using $v_1 := v_3$ and $v_2 := v_4$, resulting in $\boldsymbol{\phi} := \boldsymbol{q}$,
   (b) using $v_1 := v_5$ and $v_2 := v_6$, resulting in $\boldsymbol{\psi} := \boldsymbol{q}$.
4. Compute the stationary distribution $\boldsymbol{z}$ at arbitrary instants
   (a) using $v_1 := v_3$ and $v_2 := v_4$ and $\boldsymbol{\phi}$ as the stationary distribution at scanning instants, resulting in $\hat{\boldsymbol{\phi}} := \boldsymbol{z}$,
   (b) using $v_1 := v_5$ and $v_2 := v_6$ and $\boldsymbol{\psi}$ as the stationary distribution at scanning instants, resulting in $\hat{\boldsymbol{\psi}} := \boldsymbol{z}$.
5. Repeat steps 2–4 until in two consecutive steps, the difference between the 'old' and the 'new' distribution $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ are element-wise less than a given ϵ.
6. Calculate
   (a) the vacation distributions v_7 and v_8 using $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$,
   (b) all values $p_i(u, v, j, k, l)$ and $q_d(u, v, i, j, k, l)$ using $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$.
7. Compose $\boldsymbol{Q}$, and calculate its stationary distribution $\boldsymbol{q}$. Compute $\boldsymbol{z}$ from $\boldsymbol{q}$. Use $v_1 := v_7$ and $v_2 := v_8$.

5 Performance Measures

Important performance measures we can calculate from the obtained queue length distributions are the packet discard ratio and the cell loss ratio.
The packet discard ratio is defined as

$$P_{\text{disc}} = \frac{\text{mean number of packets discarded by the discarding mechanism in a time slot}}{\text{mean number of packets generated in a time slot}}, \tag{3}$$

while the cell loss ratio is defined as

$$\text{CLR} = \frac{\text{mean number of cells which are lost due to buffer overflow in a time slot}}{\text{mean number of cells which arrive at the buffer in a time slot}}. \tag{4}$$

Because of the homogeneity of all input sources, we can calculate P_{disc} and CLR also by

$$P_{\text{disc}} = \frac{P\{\text{a packet from the tagged source is discarded in a time slot}\}}{P\{\text{the tagged source generates a packet in a time slot}\}}. \tag{5}$$

and

$$\text{CLR} = \frac{P\{\text{a cell of the tagged source is lost due to buffer overflow in a time slot}\}}{P\{\text{a cell of the tagged source arrives at the buffer in a time slot}\}}. \tag{6}$$

In [12] we derived that P_{disc} equals

$$P_{\text{disc}} = \frac{\alpha \sum_{(u,v,j,k,\text{on}^*)\in\mathcal{A}} z_{(u,v,j,k,\text{on}^*)}}{\beta \sum_{(u,v,j,k,\text{off})\in\mathcal{A}} z_{(u,v,j,k,\text{off})}}, \tag{7}$$

and that CLR equals

$$\text{CLR} = \Bigg[\sum_{v=1}^{Q_{\max}} \sum_{j=1-v+Q_{\max}}^{R} \Bigg(\sum_{(u,v,j,k,\text{on})\in\mathcal{A}} z_{(u,v,j,k,\text{on})}\Bigg)\Bigg(\sum_{w=1}^{j+v-Q_{\max}} \frac{w}{w-v+Q_{\max}+1}$$
$$\binom{j}{w-v+Q_{\max}}(\lambda_{\text{on}})^{w-v+Q_{\max}+1}(1-\lambda_{\text{on}})^{j-w+v-Q_{\max}}\Bigg) + \sum_{j=Q_{\max}}^{R}\Bigg(\sum_{(u,0,j,k,\text{on})\in\mathcal{A}} z_{(u,0,j,k,\text{on})}\Bigg)$$
$$\Bigg(\sum_{w=1}^{j+1-Q_{\max}} \frac{w}{w+Q_{\max}}\binom{j}{w-1+Q_{\max}}(\lambda_{\text{on}})^{w+Q_{\max}}(1-\lambda_{\text{on}})^{j-w+1-Q_{\max}}\Bigg)\Bigg]$$
$$\Bigg(\lambda_{\text{on}} \sum_{(u,v,j,k,\text{on})\in\mathcal{A}} z_{(u,v,j,k,\text{on})}\Bigg)^{-1}. \tag{8}$$

These results will be used in the next section.

6 Model Validation and Numerical Results

To validate our model, results obtained by the analytical model are compared with simulation results. Figure 1 shows the distribution of the length of the total queue and of a VC queue for three examples with parameters as listed in the table below:

	R	λ_{on}	λ_{all}	q	$Q_{\max}$	Th	K
Example A	4	0.8	1.00	4	20	8	1
Example B	3	0.4	0.70	4	20	10	1
Example C	2	0.77777	0.50	8	20	12	1

The parameter λ_{all} in this table is the traffic generated by all $R+1$ sources, all other parameters are as defined before.

$$\lambda_{\text{all}} = (R+1)\frac{\beta}{\alpha+\beta}\lambda_{\text{on}}. \tag{9}$$

To obtain the analytical results, the modified algorithm (see below) was used. All simulation results are obtained by running ten independent runs, in each of which 10,000,000 packets are generated. Confidence intervals are calculated with a confidence coefficient of 0.95. The obtained packet discard and cell loss ratios are shown in the table below. The results obtained by using the analytical model are of the same order as the results obtained by simulation.

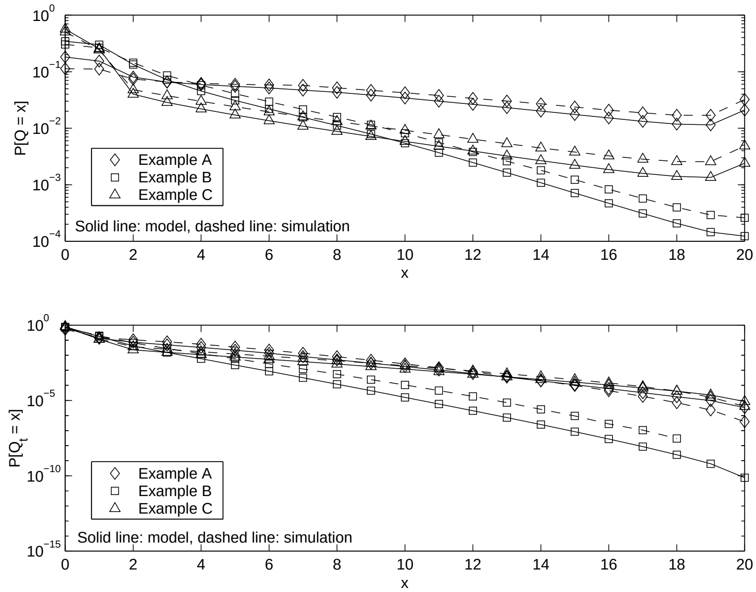

Figure 1: Comparison of the queue length distributions obtained with the analytical model and by simulation.

	Analytical results		Simulation results	
	P_{disc}	CLR	P_{disc}	CLR
Example A	5.0635e-02	1.4439e-02	9.4388e-02 ± 7.4920e-05	2.0298e-02 ± 2.9549e-05
Example B	1.3479e-03	7.1061e-05	4.1358e-03 ± 1.8414e-05	1.3807e-04 ± 1.7511e-06
Example C	3.6210e-03	2.6094e-03	5.7155e-03 ± 8.2142e-06	4.9142e-03 ± 1.9795e-05

In the algorithm as proposed in Section 4 we calculate in every iteration step the stationary distributions $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ (step 3) and $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$ (step 4). Steps 2–4 are repeated several times because of step 5. From those steps, step 4 is clearly the most time consuming one. As we have noticed in several examples, our results will be of the same order if we change step 2(b) in

2(b)': Calculate all the values $p_i(u, v, j, k, l)$ and $q_d(u, v, i, j, k, l)$ using $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$,

skip step 4(b) and only use $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\psi}}$ at the end of the algorithm (step 6). This however makes a significant difference in computation time. To illustrate that the results of the modified algorithm are of the same order as the results of the original algorithm, consider the plots of Figure 2. This figure shows the distribution of the total queue length and of a VC queue length for three examples with the following parameters:

	R	λ_{on}	λ_{all}	q	$Q_{\max}$	Th	K
Example D	2	0.5	0.75	5.5	20	5	1
Example E	2	0.77777	0.50	8.0	20	12	1
Example F	3	1.0	1.00	5.0	20	8	1

The obtained packet discard and cell loss ratios are summarized in the table below:

	Original algorithm		Modified algorithm	
	P_{disc}	CLR	P_{disc}	CLR
Example D	1.9594e-02	1.4075e-04	2.1756e-02	1.3706e-04
Example E	3.5613e-03	2.6224e-03	3.6210e-03	2.6094e-03
Example F	5.8654e-02	2.0118e-02	6.1643e-02	1.9469e-02

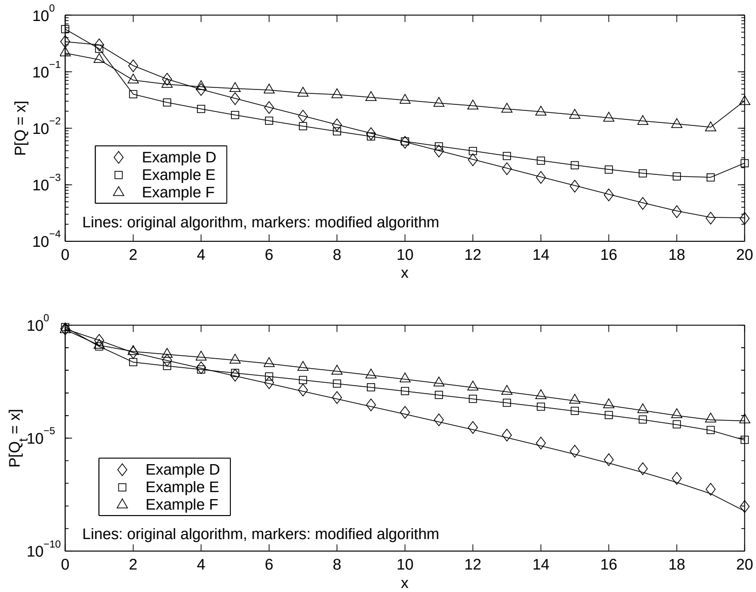

Figure 2: Comparison of the queue length distributions obtained with the original and the modified algorithm.

Since we are more interested in the evolution of the curves if parameters are changed than in the absolute values, all following numerical results presented are obtained with the modified algorithm. Figure 3 shows the impact on the queue length distributions if the same total amount of traffic ($\lambda_{\text{all}} = 0.7$) is generated by a different number of sources $R + 1$. Figure 4 shows the impact on the packet discard ratio and the cell loss ratio. If there are more sources, the burstiness of the generated traffic is larger, which enlarges the probability that Q exceeds the EPD threshold Th (see Figure 3). If there are more sources, the probability that N, the number of active VCs, will be larger increases. But a larger N means also a smaller threshold $\overline{Th}$, so more packets will be discarded if there are more sources. However, the larger N, the smaller the effect on $\overline{Th}$ if N evolves from N to $N + 1$. This explains the smaller increase of P_{disc} for growing N in Figure 4. Although more packets are discarded if the number of sources increases, the cell loss ratio does not decrease, but on the contrary increases. This is because a larger burstiness of the traffic has a larger negative effect on the cell loss than that a small reduction of the amount of traffic that enters the buffer has a positive effect.

Figures 5 and 6 show the impact of the EPD threshold Th on the queue length distributions, the packet discard ratio and the cell loss ratio for a scenario with offered load $\lambda_{\text{all}} = 0.7$. As can be seen from Figure 5, the influence of a change of Th on the queue length distribution of one VC is only a minor one. We noticed from results with a larger offered load λ_{all} that this influence becomes larger if λ_{all} increases. As can be seen in Figure 5, the curves of the total queue length distribution start to diverge downwards from the other curves from the value Th on, since once the total queue is over this threshold, it becomes possible to discard packets, so less of the generated traffic will effectively enter the buffer. As expected, an increase of Th leads to a decrease of P_{disc}, as shown in Figure 6. This is however at the cost of an increasing cell loss ratio, since a larger part of the traffic is allowed to enter the buffer.

7 Conclusions

This paper adresses the EPD with per-VC queueing mechanism. This mechanism was proposed in [10] because the EPD scheme with a single FIFO queue can not guarantee fairness between

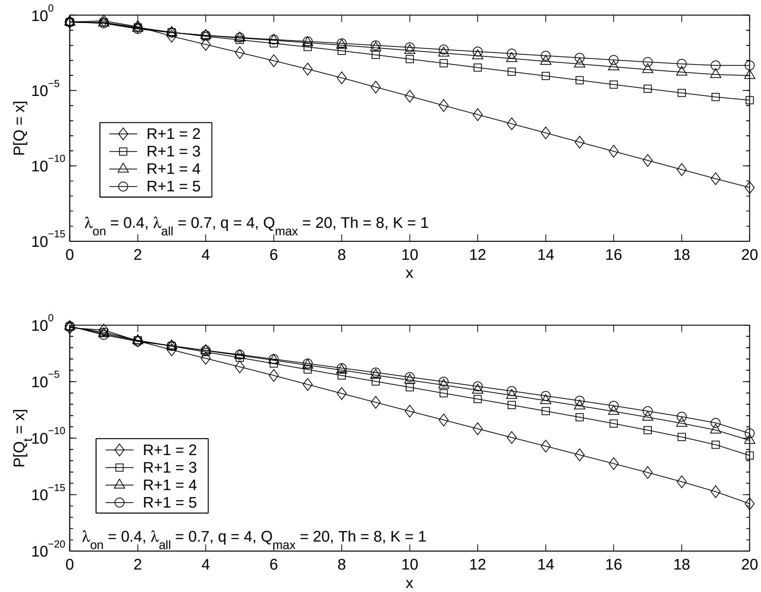

Figure 3: Queue length distributions when the same amount of traffic ($\lambda_{\text{all}} = 0.7$) is generated by a different number of sources $R + 1$.

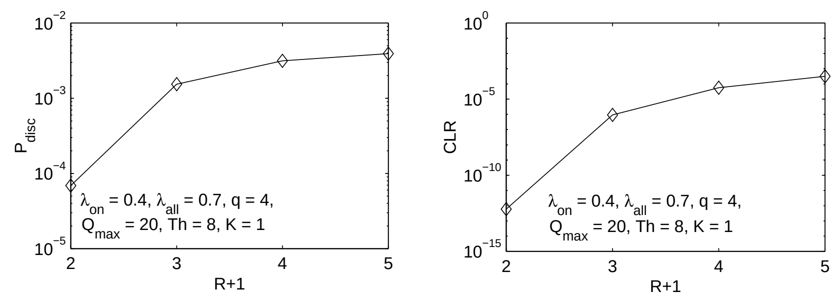

Figure 4: Evolution of P_{disc} and CLR when the same amount of traffic ($\lambda_{\text{all}} = 0.7$) is generated by a different number of sources $R + 1$.

competing flows. In this paper we developed an analytical approximation model for the mechanism. It combines the source models used in the literature for EPD and PPD and the approximate vacation analysis often used for cyclic service systems.

To validate our model, the obtained results were compared with simulation results. Both methods lead to results that are shown to have the same order of magnitude. So they can be used very well to study the evolution of performance measures such as buffer occupation, packet discard ratio and cell loss ratio when parameters of the input traffic or of the discarding mechanism are changed. In this paper we looked at the effect of a change in the number of sources which generate the same amount of traffic and of a change in the setting of the EPD threshold Th. In the first scenario we saw that the important influence of increasing burstiness on the CLR could not be overthrown by the increasing packet discard ratio of the discarding scheme. The second scenario indicated the minor influence of Th on the queue occupation of a VC queue, and the expected influence (decreasing packet discard ratio and increasing cell loss ratio for increasing Th) on packet discard ratio and cell loss ratio.

Future work will include the investigation of the influence of other parameters on the performance,

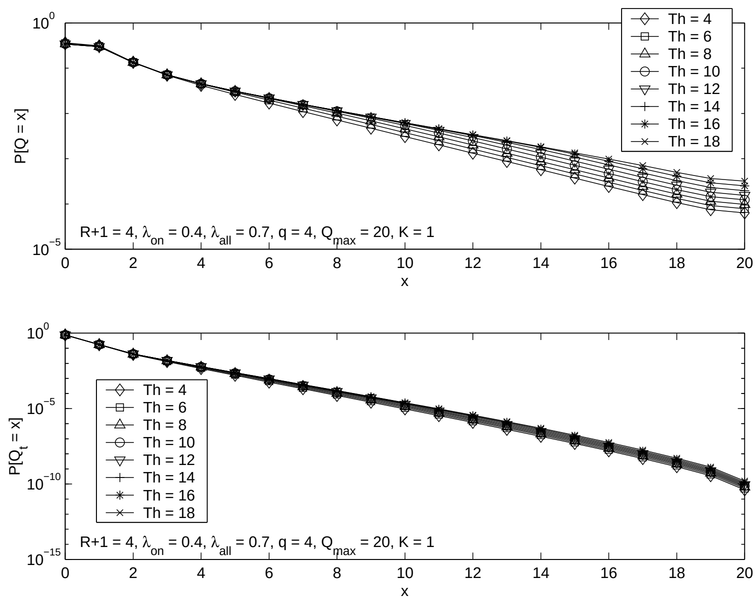

Figure 5: Queue length distributions for different values of the EPD threshold Th.

and a comparison between the results obtained by this packet discard scheme and the EPD scheme with a single FIFO queue used in [7].

References

[1] Blondia, C. (1989). *A discrete-time approximate performance analysis of an ATM switching element with non-renewal input*, Research Report.

[2] Blondia, C. (1993). *A discrete-time batch Markovian arrival process as B-ISDN traffic model*, Belgian Journal op Operations Research, Statistics and Computer Science, Vol. 32, No. (3,4).

[3] Chan, C. T., Y. C, Chen and P. C. Wang (1998). *An efficient traffic control approach for GFR services in IP/ATM internetworks*, Proceedings of IEEE Globecom'98.

[4] Doshi, B. T. (1986). *Queueing systems with vacations - a survey*, Queueing Systems, Vol. 1.

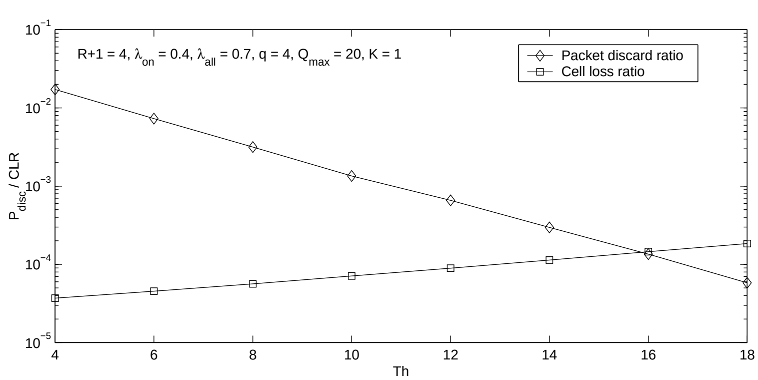

Figure 6: Evolution of P_{disc} and CLR for different values of the EPD threshold Th.

[5] Goyal, R., R. Jain, S. Kalyanaraman, S. Fahmy and S. C. Kim (1997). *UBR+: Improving performance of TCP over ATM-UBR service*, Proceedings of ICC'97.

[6] Grassmann, W., M. Taksar and D. Heyman (1985). *Regenerative analysis and steady state distributions for Markov chains*, Operations Research, Vol. 33, No. 5.

[7] Kawahara, K., K. Kitajima, T. Takine and Y. Oie (1997). *Packet loss performance of selective cell discard schemes in ATM switches*, IEEE Journal on Selected Areas in Communications, Vol. 15, No. 5.

[8] Kim, Y., and S. Q. Li (1997). *Performance analysis of data packet discarding in ATM networks*, Proceedings of ITC 15.

[9] Lapid, Y., R. Rom and M. Sidi (1998). *Analysis of discarding policies in high-speed networks*, IEEE Journal on Selected Areas in Communications, Vol. 16, No. 5.

[10] Li, H., K. Siu, H. Tzeng, C. Ikeda and H. Suzuki (1996). *TCP performance over ABR and UBR services in ATM.*

[11] A. Romanow and S. Floyd (1995). *Dynamics of TCP traffic over ATM networks*, IEEE Journal on Selected Areas in Communications, Vol. 13, No. 4.

[12] K. Spaey (1999). *Analysis of the early packet discard with per-VC queueing mechanism*, Technical Report, available from ftp://wins.uia.ac.be/pub/pats/reports as 99-EPDperVC.ps.gz.