An implementation of a service class providing assured TCP rates within the AQUILA framework *

Christof Brandauer and Peter Dorfinger

Salzburg Research, Jakob-Haringer-Str. 5/III,
A-5020 Salzburg, Austria
christof.brandauer@salzburgresearch.at
peter.dorfinger@salzburgresearch.at

Abstract. This paper investigates an attempt to establish a QoS class that supports long-lived, bulk-data TCP flows that require a minimum rate from the network. The approach is based on a model for TCP flows subject to token bucket marking at the network edge and preferential dropping in the core network. The service class adds admission control functionality and a model for multi-RED queue management to the token bucket marker. The difficulty of parameterizing the mechanisms is discussed and analyzed in an explorative simulation study. A set of configuration parameters that enables a successful operation of the service class is identified and the achievable service provisioning is shown.

1 Introduction

The European IST project AQUILA (Adaptive Resource Control for QoS Using an IP-based Layered Architecture) [1] implements an IP-based quality of service (QoS) architecture on the basis of the Differentiated Services [2, 3] philosophy. In the AQUILA approach, the network operator provides a set of *Network Services* to its costumers. A network service defines i) requirements concerning the admissible traffic and ii) the QoS properties provided for the admitted traffic.

One particular network service studied in AQUILA is called Premium Multimedia (PMM). It is intended for greedy TCP-based applications that require a minimum sending rate. Example applications are premium FTP data transfers or TCP-based streaming media applications. The network operator implements a network service by means of admission control, traffic conditioning, queue management, and scheduling.

In order to be able to perform the testbed / real-user trials we strive for an approach that could be implemented with the router equipment available in the project. In this paper we investigate the feasibility of a PMM implementation on the basis of *token bucket marking* and *preferential dropping*.

* This work was partly funded by the European IST project AQUILA under contract IST-1999-10077

The foundation of the approach is an analytical model [4] for the TCP sending rate when a TCP flow is subject to a token bucket marker and preferential queue management. The model shows that there are conditions where the sending rate of the TCP flow can be regulated by the configuration of the token bucket parameters. The goal of the PMM implementation is to permanently enforce these conditions by means of admission control and queue management.

If an application wants to utilize the PMM service it sends a reservation request to the AC entity. The request contains the requested rate R. The AC entity decides whether or not the request can be accepted. If the request is accepted, the requester's ingress edge device is reconfigured which involves the setup of a classifier and the token bucket marker. In any case, the AC decision is signalled back to the requester using the signaling facilities provided by AQUILA framework.

The paper is structured as follows: after briefly summarizing related work in the next section we review in more detail the results of [4] as relevant for this work in section 3. Subsequently, appropriate admission control (section 4) and queue management (section 5) entities are derived. The difficulty of finding appropriate parameter sets is discussed in section 6. The feasibility of the approach is investigated in a large simulation study reported in sections 7 and 8.

2 Related Work

There exist several works [5–8] that enforce TCP rate control by relying on an "invasive" mechanism where TCP header fields are modified. Packeteer [9] seems to have originally come up with this concept. Their TCP rate controller [10] modifies the receiver window and acknowledgement number and additionally modulates the rate of acknowledgements.

Several "non-invasive" mechanism based on packet marking algorithms have been investigated in the context of Differentiated Services. Many of these algorithms use a static marking profile [11–17]. Adaptive marking algorithms are proposed in [18–21].

Unlike these works the AQUILA architecture explicitly acts upon the assumption of an admission control entity limiting access to the offered service classes. The TCP rate controller (TRC) for long-lived TCP flows proposed in [22] essentially requires an admission control entity. It operates as a traffic conditioner at the edge of a domain. Given a requested rate and an estimation of the domain's delay it computes a target packet drop probability and a target delay on the the basis of a TCP model [23]. By enforcing the drop rate and introducing artificial delays the TCP flows are trimmed to the requested rate. The TRC is shown to be unbiased to the requested rate and RTT.

3 Token bucket marker

The authors of [4] model the impact of token bucket marking on greedy TCP flows. On the basis of a model for TCP sending behavior [24] they develop an

analytical model for determining the sending rate of a TCP flow when edge-routers use token bucket marking (with statically configured token bucket parameters) and core routers employ active queue management with preferential packet dropping. Using this model (we denote it as *TBM model* in the following) it is shown that there exist conditions where it is not possible to influence the service achieved by a TCP flow through a marking profile. For a different set of conditions it is, however, feasible to achieve a requested sending rate. In that case the sending rate A of a greedy TCP flow is given by eq. 1.

$$A = \begin{cases} R - \frac{3}{2Rp_2T^2} & R \leq \frac{3W}{2T} \\ \frac{4}{3}(R - \frac{3}{2T\sqrt{2}}\sqrt{Z + \frac{1}{p_2}} & R > \frac{3W}{2T} \end{cases} , where\ W = \sqrt{2(Z + 1/p_2)} + 2\sqrt{2Z} \tag{1}$$

Table 1 describes the parameters involved. The necessary condition is a so-called *under-subscribed* scenario which is defined as $p_1 = 0$ and $p_2 > 0$.

Parameter	Meaning	Unit
A	token bucket rate	packet/s
Z	token bucket size	packet
R	requested rate	packet/s
p_1	packet drop probability for in-profile packets	-
p_2	packet drop probability for out-profile packets	-
T	round-trip-time (RTT)	s

Table 1. Token bucket parameters

The results of [4] make the TBM model a promising candidate for a PMM implementation. The model, which is analytically derived from an accurate TCP model, provides closed-loop formulae for the computation of the appropriate marking profile required to achieve a target sending rate. In a simulation study [4] the TBM model is shown to be very accurate over a wide range of values for p_2, T, and R. Given the accuracy of the model and the possibility to practically implement the token bucket marking approach using today's routers we construct the PMM class on the basis of the TBM model.

The goal is to operate the service class in the region where the achieved TCP rate can be regulated through the configuration of the token bucket parameters. We seek to establish the required under-subscribed scenario by combining the token bucket marking with adequate admission control and queue management.

4 Admission control

The goal of any admission control (AC) functionality is to limit the amount of traffic admitted to a particular service class such that the QoS objectives are

reached for all admitted flows. At the same time, the service class utilization should be maximized.

For PMM a declaration based approach is employed: the requested rate R is part of the reservation request sent to the AC entity. Using the TBM model, the AC entity computes the token bucket rate A according to eq. 1. Now, if $A > R$, the flow requires at least an available bandwidth of A in order to obtain R. If $A \leq R$, an amount of R resources must be available.

On this basis a simple AC rule can be formulated. The resources required by a single flow are expressed by the greater value of the token rate A and the requested rate R. The inequality in eq. 2 ensures that the bandwidth required by the aggregate traffic submitted to the PMM class is smaller than ρ times the reserved capacity C, where ρ is a (tunable) over-provisioning factor; it denotes the fraction of reserved capacity that is at most allocated to resource requests.

$$\sum_{i=1}^{N} \max(A_i, R_i) \leq \rho C \tag{2}$$

The number of flows in the PMM class, including the new one if being admitted, is denoted by N. Note that eq. 2 implicitly assumes that the aggregate PMM traffic is able to fully utilize the reserved capacity C. It is discussed in section 5 why this is a valid assumption. In the AQUILA framework the bandwidth is allocated to the different network services by means of WFQ-based scheduler at each router output port.

It follows from the TBM model that in an under-subscribed scenario each TCP flow achieves a minimum sending rate R_0 when the token bucket marks all packets as out-profile ($A = Z = 0$).

$$R_0 = \frac{1}{T}\sqrt{\frac{3}{2p_2}} \tag{3}$$

A reservation request for a rate $R < R_0$ can be either principally denied (because a rate as small as R cannot be achieved) or it has to be handled in the following way:

- the token bucket parameters are set to: $A = Z = 0$.
- for that particular request, R is replaced by R_0 in the computation of the sum in eq. 2 to take into account that the flow will in fact consume R_0 bandwidth.

5 Queue management

In order to be combineable with the TBM model, the queue management mechanism must be able to enforce an under-subscribed scenario, i.e. $p_1 = 0$ and $p_2 > 0$. First of all, this requires the ability to distinguish between in-profile and out-profile packets, respectively. We employ a two-color extension [25] of RED [26] queue management. There is one parameter set for in-profile packets

$\{minth_i, maxth_i, maxp_i\}$ and one set for out-profile packets $\{minth_o, maxth_o, maxp_o\}$. A single average queue size is calculated over all arriving packets and depending on the color of the packet the corresponding set of parameters for that color is used. This approach is generally referred to as Weighted RED (WRED).

In order to optimally support the PMM class, the following queue size behavior should be enforced:

- the average queue size converges within the control range for out-profile packets, i.e., between $minth_o$ and $maxth_o$.
- the amplitude of oscillation of the average queue size is bounded and significantly smaller than the difference between $minth_o$ and $maxth_o$.
- the instantaneous queue size is (mostly) greater than zero and smaller than the buffer size.

Such a behavior is clearly beneficial for the PMM class: besides establishing the required under-subscribed scenario, the available link capacity can be fully utilized. First, this provides for optimal resource usage. Second, the predictability of bandwidth utilization is an important input for the AC algorithm. In fact, the AC rule has to take into account the amount of bandwidth the aggregate traffic stream is able to consume – not the capacity reserved for that traffic class.

Moreover, due to the enforced queue size behavior the controlled TCP flows experience a rather constant drop probability for out-profile packets and should be able to handle these drops without resorting to timeouts. Consequently, the sending behavior of the flows is rather smooth.

It must be noted that a careful selection of WRED parameters is required to achieve the above described performance. If the queue management parameters are chosen rather incidentally the average queue size will generally not exhibit such a behavior. See [27] and [28] for a discussion of these effects.

5.1 Implementation

We develop a quantitative model (subsequently called *WRED model*) for setting the parameters of WRED queue management. The WRED model is an extension of the quantitative RED model [28] which can be accessed via the Web under [29]. This RED model calculates the RED parameters $minth$, $maxth$, max_p, w_q, and the buffer size as a function of the scenario parameters bottleneck bandwidth, RTT, and number of TCP flows. The RED model has been developed by assembling an accurate analytical model of TCP sending behavior [24], an analytical model for setting w_q [30] and an empirical model providing the required difference between $minth$ and $maxth$. It is shown in [28] that under the load of long-lived TCP flows, RED's average queue size converges between $minth$ and $maxth$ and the amplitude of average queue size oscillation is about one third of the difference between $minth$ and $maxth$.

The idea of the WRED model is to achieve the same convergence behavior in a two-color environment but without having to drop in-profile packets. This would establish the under-subscribed condition as required by the TBM model.

With (W)RED queue management the long term average queue size is mostly dependent on the maximum drop probability max_p. This parameter determines the aggressiveness of dropping packets when incipient congestion is detected.

If in-profile packets are excluded from the dropping process the WRED parameter $maxp_o$ must be higher than the RED parameter max_p in order to achieve the same overall drop probability. Therefore, to adapt max_p correctly, knowledge of the expected ratio of out-profile packets is required. We define the number of out-profile packets divided by the total number of packets that arrive at the queue as the out-share. The out-share is estimated by a parameter called U. It is influenced by several factors as discussed in section 6.

Due to the existence of an AC framework, convergence of the average queue size within the control range for out-profile packets is feasible. This eliminates the need to drop in-profile packets for the sake of congestion avoidance / control. Thus, in order to exclude in-profile packets from the dropping process, we recommend to set the WRED parameters $minth_i$ and $maxth_i$ to the total buffer size and $maxp_i$ to 1. Note that with this approach there is practically no difference between WRED and RIO [31].

Due to space limitations we must omit here the derivation of the equations for the WRED model. Please refer to [32] for the details. A Web interface to the model can be accessed under [33].

6 Parameterization

The WRED model requires an estimate of the out-share as an input parameter. As discussed below, the out-share is influenced by many factors and may oscillate substantially. For the WRED model to be practically applicable it is thus crucial to achieve the desired convergence behavior even if the real out-share differs significantly from the estimated value U. We investigate in [32] the model's sensitivity on a correct estimate of the out-share. In this study, U is set to 0.5 in order to minimize the deviation from the real out-share. In the simulations where the out-share is significantly smaller than estimated (10% instead of 50%) the average queue size converges to a value higher than $(minth_o + maxth_o)/2$ but remains within the control range for out-profile packets. If the real out-share is higher (90% instead of 50%) the dropping is too aggressive and the average converges to a value lower than $(minth_o+maxth_o)/2$, but again remains between $minth_o$ and $maxth_o$. In any case, the convergence behavior is as desired and the performance of the WRED model is shown to not critically depend on U. The details can be found in [32].

Another critical input parameter needed for the WRED model is an estimate N of the number of flows. This dependency cannot be avoided as the RED behavior is intrinsically influenced by the number of flows. In order to compute a WRED parameter set it is therefore required to estimate the expected number of flows. As it will change over time there is no correct value for N. Clearly, there is a lower bound (zero or more flows) as well as an upper bound for the number of flows that can be active at any time in the system. The upper bound is

determined by the finite reserved capacity and the smallest acceptable requested rate.

Additionally, the TBM model requires the expected drop probability for out-profile packets p_2 as in input parameter. The value of p_2 must thus be estimated a priori through a constant value. Clearly, p_2 can fluctuate heavily as it depends on the level of congestion (number of flows) and the out-share. The out-share itself is influenced by:

- the size of the requested rates:
  it is a general property that TCP flows with smaller window sizes exhibit more "aggressiveness" than flows with larger windows. As a consequence, flows with lower bandwidth requests produce more out-profile packets than flows with higher bandwidth requests. This general effect is reinforced if the token bucket rate A is computed according to the TBM model (eq. 1), as in the TBM model $(A - R)$ is strictly monotonic increasing with increasing R.
- the portion of reserved capacity allocated to accepted requests:
  The PMM class is utilized by greedy TCP flows which always fully utilize the available capacity - independently of the requested rate. Clearly, the token bucket parameters A and Z limit the amount of packets that are marked as in-profile but the amount of out-profile packets is only limited by the portion of unallocated capacity.

Unfortunately it is not possible to make a worst-case estimation of p_2. In the one case, if p_2 is estimated too low, the sending rates of the TCP flows are over-estimated and the resulting token rates are too small. For those requests where $A > R$, the $\max(A, R)$ is smaller than the amount of bandwidth that would in fact be needed by that flow. Thus, in general, too many flows would be admitted.

In the other case, if p_2 is estimated too high, the resulting sending rates of the TCP flows are under-estimated and the computed token rates are too high. This again leads in general to a situation where too many flows are admitted because the real TCP rates are higher than the ones used for the admission control algorithm.

The TBM model requires an estimate T of the RTT. It could be estimated as the propagation plus transmission delays plus the average queueing delay at the WRED bottleneck. It is however difficult to know in advance the number of bottleneck routers in a flow' s path. Additionally, different flows generally have different destinations with different RTTs.

In lack of an analytical model that captures the behavior of the PMM implementation we try to discover parameter configuration dependencies by executing a large amount of packet level simulations. The goal is to discriminate between configurations which enable a successful PMM operation (if possible at all) from those configurations where the probability for a sending rate smaller than R is much larger than zero. Moreover, we investigate the influence of deviations in the parameter estimation on the usability of the various traffic control components.

7 Simulation study

We use the well-known packet-level network simulator `ns-2` from [34] and extend it by admission control functionality. The AC entity decides on the basis of eq. 1, 2, 3 whether the request can be accepted.

The simulated topology is shown in figure 1. Applications are distributed over hosts S1–Sn and send to hosts R1–Rm, where n and m are chosen such that drops occur only at the output interface of B1. The AC entity controls access to the bottleneck link between routers B1 and B2. The bottleneck link has a capacity of 10 Mbps and a propagation delay of 20 ms. We simulate only PMM traffic and thus the full 10 Mbps are reserved for PMM. All other links have a capacity of 100 Mbps. The propagation delay of the links between B2 and hosts R1–Rm is either set as 70 ms for all links in the RTT_{equal} scenarios or set as 40 ms (B1–R1), 70 ms (B2–R2), and 100 ms (B2–R3) in the RTT_{var} scenarios. This $\{40, 70, 100\}$ cycle is repeated from R4–Rm. The RTT_{var} setup enables the study of competing flows under different RTTs.

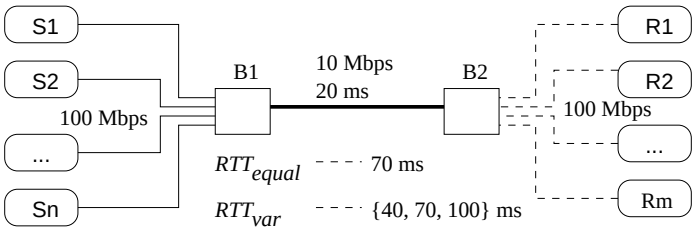


Fig. 1. Simulation topology

We use the `ns-2` built-in FTP application to simulate greedy, TCP based bulk-data transfers. The application lifetime is uniformly distributed between 50 and 150 seconds. While this is arguably not a choice matching real-world observations, we are initially not interested in very long transfers. The goal of the study is to first get an understanding what parameter configuration is useless / useful for the operation of TCL 3. Once this large parameter space can be reduced, more detailed studies can be performed and we argue that a configuration that optimizes transfers of FTP flows lasting for 150 seconds will also be beneficial for longer flow durations.

We have seen in initial simulations that the higher the ratio of the highest and smallest requested rate, the more difficult it is to provide the requested sending rates. This is a consequence of the weak estimation of certain parameters like p_2 or T. In fact, flows with a smaller requested rate tend to "steal" bandwidth from flows with a higher rate. To study this effect, we define a requested rate factor rF as $rF := \frac{R_{max}}{R_{min}}$ and include rF into the list of parameters that are varied between simulation runs.

We put an additional restriction on the rates that can be requested by allowing only a finite set of rates within the range $[R_{min} \dots R_{max}]$. The difference between R_{i+1} and R_i must be equal to the requestable rate distance rD. Like rF, rD is also varied over the simulation scenarios. In total, the requestable rates are the ones shown in table 2.

rF = 3	
$rD = 100$:	$R \in \{200, 300, 400, 500, 600\}$
$rD = 200$:	$R \in \{200, 400, 600\}$
$rD = 300$:	$R \in \{200, 500\}$
rF = 5	
$rD = 100$:	$R \in \{200, 300, 400, 500, 600, 700, 800, 900, 1000\}$
$rD = 200$:	$R \in \{200, 400, 600, 800, 1000\}$
$rD = 300$:	$R \in \{200, 500, 800, 1100\}$
rF = 8	
$rD = 200$:	$R \in \{200, 400, 600, 800, 1000, 1200, 1400, 1600\}$
$rD = 300$:	$R \in \{200, 500, 800, 1100, 1400\}$
$rD = 400$:	$R \in \{200, 600, 1000, 1400\}$

Table 2. Requestable rates

Resource reservation requests are generated according to an exponentially distributed inter-arrival time with a mean on 2 seconds. This results in a high-load scenario where at almost any time all resources are allocated to flows and the probability that a new request has to be denied is high. Although such a request blocking probability may be unrealistically high in an operative network, it is a worst case scenario where flows cannot consume unallocated bandwidth and thus more easily reach the required sending rate.

We introduce the term *traffic template* which represents exactly one possible combination of requested rate R and RTT T. When a new rresourcerequest is generated, one such traffic template is randomly (uniform) chosen and a request for the template's requested rate R is sent to the AC entity.

Besides choosing T according to the RTT_{equal} and RTT_{var} scenarios, respectively, we also vary the error in RTT estimation, called T_{dev}, as {0%, 25%, 50%, 75%, 100%}. This enables us to study the influence of a wrong RTT estimation. A T_{dev} of 0% means that T is set as a value that closely matches the RTT of the simulation scenario (propagation delays plus transmission delays plus average WRED queueing delay). If T_{dev} is larger than zero, the real RTT is around $T * (1 + T_{dev})$, i.e. T underestimates the real RTT. Underestimation is the more difficult case as this overestimates the TCP sending rate.

Concerning the WRED model, we investigate two approaches for setting the number of flows N. In one case, called N_{high}, we set $N = \frac{\rho C}{R_{min}}$. This estimation assumes that all requests are for the minimum rate R_{min} and thus N_{high} is generally an overestimation. In the other case, called N_{low}, we set $N = \frac{\rho C}{R_{avg}}$, where $R_{avg} = \frac{R_{min}+R_{max}}{2}$. Due to an unfairness on the request level – smaller

requests have a higher probability of being accepted – N_{low} is thus generally an underestimation.

Finally, we use values of $\{0.7, 0.8, 0.9\}$ for the ρ parameter of the AC formula (eq. 2).

Some parameters have been fixed for all simulations. The packet size is set to 1500 bytes. The WRED model is configured with $U = 0.5$. For the TBM model, the p_2 value of 0.1 is configured. This choice is the result of a prior study not shown here.

ρ of admission control	$\in \{0.7, 0.8, 0.9\}$
number of flows for the WRED model	$\in \{N_{low}, N_{high}\}$
link delays in topology	$\in \{RTT_{equal}, RTT_{var}\}$
error in RTT estimation T_{dev}	$\in \{0\%, 25\%, 50\%, 75\%, 100\%\}$
requested rate factor rF	$\in \{3, 5, 8\}$
requested rate distance rD	$\in \{100, 200, 300[, 400]\}$

Table 3. Summary: variation of input parameters

The input parameter space is summarized in table 3. In order to exhaustively explore this input space we simulate all 540 possible combinations of input parameters. Each simulation is run for 50000 simulated seconds. As the request blocking probability increases with larger requested rates, this long simulation time is needed to get enough results for traffic templates which request a rate of R_{max}.

8 Simulation results

The primary QoS goal of the PMM class is to achieve a sending rate that is at least as high as the requested rate. We therefore define a *success* value s_t for each traffic template t in the following way:

$$s_t := \frac{card\{flow \in t : rate \geq R\}}{card\{flow \in t\}} \tag{4}$$

A whole simulation run is characterized by one *success* parameter S, where $S = \min(s_t)$. Interestingly, the resulting success values vary between 0% (i.e. there is no traffic template where all flows reach at least R) and 100% (i.e. all flows in all templates reach at least R).

To evaluate the optimality of the different parameter configurations we represent each simulation with a 7-dimensional vector: one dimension for the success S of the simulation and 6 dimensions due to the input parameters (see table 3).

In order to gain insight on how to ideally configure the PMM service we select those simulation vectors with a success $S \geq 0.99$. Such high success values can only be obtained if the input parameters are chosen among the ones shown in table 4

ρ of admission control	$\in \{0.7\}$
number of flows for the WRED model	$\in \{N_{high}\}$
link delays in topology	$\in \{RTT_{equal}, RTT_{var}\}$
error in RTT estimation T_{dev}	$\in \{0\%, 25\%, 50\%, 75\%, 100\%\}$
requested rate factor rF	$\in \{3\}$
requested rate distance rD	$\in \{100, 200, 300\}$

Table 4. Input parameters resulting in high success S

We subsequently discuss the results and thereby look in detail at the influence of each parameter involved. The following paragraphs provide guidelines on how to optimally configure the PMM service class.

The choice of $\rho = 0.7$ provides enough safety margin (unallocated capacity) to compensate for the imperfections of the PMM traffic control combination. Among these imperfections are the impossibility to perfectly estimate changing parameters by a single static value and deviations between analytical models and the traffic under control.

Concerning the WRED model, better results can be achieved if the number of flows is set as N_{high}. The convergence behavior of the average queue size is indeed according to the objective of the WRED model. This justifies the correctness and usability of the WRED model even in an environment where the input parameters are not exactly known.

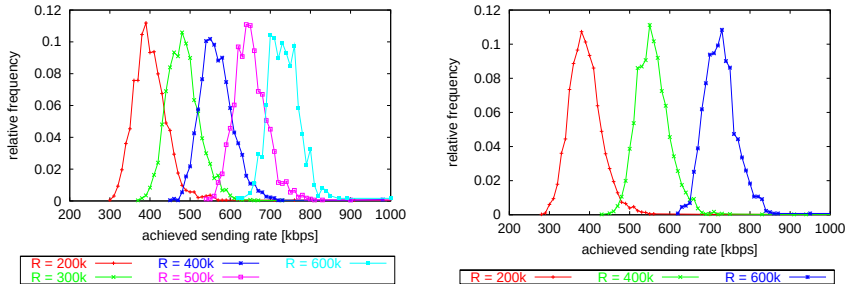
Another positive outcome of the simulation study is the result that under the restricted choice of ρ, rF, and rD as shown in table 4, differing RTTs (RTT_{var}) as well as wrongly estimated RTTs ($T_{dev} > 0$) do not present a major problem. This is a convenient property as finding a good estimate for the RTT is a difficult task.

Providing requestable rates over a broad range seems an impossible objective with the design as investigated in this study. In fact, good results can only be achieved if rF is not larger than 3. The distance rD between requestable rates has no impact on the success S.

For the further analysis the focus is on simulation scenarios where success $S \geq 0.99$. We fix the input parameters to $\rho = 0.7$, N_{high}, $rF = 3$, $T_{dev} = 50\%$ and take a detailed look at the distribution of sending rates for the RTT_{equal} / RTT_{var} scenarios and different values of rD. The sending rates are classified into bins of 10 kbps. Figure 2 shows the relative frequency of sending rates for each requestable rate in the RTT_{equal}, $rD = 100$ scenario. The requestable rates are listed in table 2.

As can be seen in figure 2, each flow achieves at least the requested sending rate, i.e. the success S of both simulations is 100%.

As far as the service differentiation within the PMM class is concerned, the scenario with $rD = 100$ (subfigure 2(a)) shows a suboptimal behavior. The user is offered a high number of requestable rates. However, the resulting service curves are significantly overlapping and the service offerings are thus not clearly distinguishable.



(a) $rD = 100$ (b) $rD = 200$

Fig. 2. Relative frequency of achieved sending rates in the RTT_{equal} scenario

In the $rD = 200$ case (subfigure 2(b)) the service curves are hardly overlapping. This provides for a clear distinction of the service delivered for different requests. The service distinction is even more pronounced in the $rD = 300$ scenario where only two rates (200 kbps, 500 kbps) are requestable.

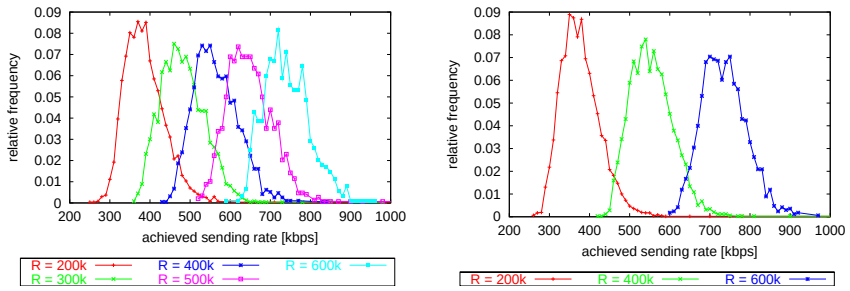
Looking at the shape of the service curves it makes sense to offer only a discrete set of requestable rates instead of a continuous range between $[R_{min} \ldots R_{max}]$. By offering a finite set of rates the operator can tune the service differentiation within the PMM class. This approves the usefulness of the discrete rates approach. A choice of $rD = 200$ results in a reasonable trade-off between the number of requestable rates and a clear service distinction.

In the RTT_{var} scenario, there are flows with different RTTs within each requested rate. Compared to the RTT_{equal} scenario, the resulting sending rates fluctuate more and the curves in figure 3 are thus slightly broader and lower. For $rD = 200$ the service differentiation within the PMM class is still well pronounced.

9 Conclusion

In this paper we report on an attempt to establish a QoS class for long-lived, bulk-data TCP flows that require a minimum rate from the network. The approach is based on a model for TCP flows subject to token bucket marking at the network edge-device and preferential dropping in the core network. This model is combined with an admission control functionality and a model for the parameterization of multi-RED queue management. The goal of the additional components is to enforce conditions under which the sending rate of the flows can be regulated through the token bucket marking profile.

The difficulty of finding a proper parameter set for the various input parameters of the service class is discussed. In a large simulation study a broad



(a) $rD = 100$ (b) $rD = 200$

Fig. 3. Relative frequency of achieved sending rates in the RTT_{var} scenario

spectrum of the input parameter space is explored in order to identify inter-parameter dependencies and discriminate between useless / useful service class configurations.

From these results guidelines on how to configure the PMM class are derived. While the models derived for admission control and queue management can be generally applied in the context of long-lived TCP flows and token bucket marking, the guidelines only apply to the PMM implementation studied in this paper.

Although ideal traffic handling is not feasible with a *static* marking profile the simulation results encourage the practicability of a real-world implementation. Despite simulations were run at a very high service class utilization (and thus an unrealistically high service request blocking probability) a set of configuration parameters that enables a successful operation could be identified. The QoS objectives can be more easily reached under a lower service utilization where more unallocated resources are available.

Initial testbed measurements have been performed but they were heavily influenced by the maximum queue size that could be configured in the router when the full AQUILA scheduling approach is employed (a combination of Priority Queueing and WFQ). This restriction did not allow the use of the WRED model as discussed in section 5 and consequently led to throughput degradations. We plan to repeat the measurements with the PMM class only as this allows the use of a FIFO scheduler where our equipment does not have the mentioned buffer size restrictions.

References

1. Adaptive Resource Control for QoS Using an IP-based Layered Architecture (AQUILA), IST-1999-10077, `http://www.ist-aquila.org/`.
2. Nichols, K., Blake, S., Baker, F., Black, D.: RFC 2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers (1998)

3. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: RFC 2475: An architecture for differentiated services (1998)
4. Sahu, S., Nain, P., Towsley, D., Diot, C., Firoiu, V.: On Achievable Service Differentiation with Tocken Bucket Marking for TCP. In: Proc. of the ACM SIGMETRICS'2000 Int. Conf. on Measurement Modeling of Computer Systems, Santa Clara, CA, USA. (2000)
5. Narvaez, P., Siu, K.Y.: An Acknowledgment Bucket Scheme for Regulating TCP Flow over ATM. In: Proceedings of IEEE Globecom. (1997)
6. Koike, A.: TCP flow control with ACR information (1997) ATM Forum/97-09989.
7. Kalampoukas, L., Varma, A., Ramakrishnan, K.K.: Explicit Window Adaptation: A Method to Enhance TCP Performance. In: Proceedings of INFOCOM '98. (1998)
8. Satyavolu, R., Duvedi, K., Kalyanaraman, S.: Explicit rate control of TCP applications (1998) ATM Forum Doc. Number 98-0152R1.
9. (Inc., P.) `http://www.packeteer.com/` (Feb. 2003).
10. Karandikar, S., Kalyanaraman, S., Bagal, P., Packer, B.: TCP rate control. ACM Computer Communications Review (2000)
11. Elizondo-Armengol, A.: TCP-Friendly Policy Functions: Capped Leaky Buckets. In: Seventeenth International Teletraffic Congress (ITC17). (2001)
12. Heinanen, J., Guerin, R.: RFC 2698: A Two Rate Three Color Marker (1999)
13. Fang, W., Seddigh, N., Nandy, B.: RFC 2859: A Time Sliding Window Three Colour Marker (TSWTCM) (2000)
14. Lin, W., Zheng, R., Hou, J.C.: How to make assured service more assured. In: ICNP. (1999) 182+
15. Seddigh, N., Nandy, B., Pieda, P.: Bandwidth assurance issues for tcp flows in a differentiated services network (1999)
16. Makkar, R., Lambadaris, I., Salim, J., Seddigh, N., Nandy, B.: Empirical Study of Buffer Management Scheme for Diffserv Assured Forwarding PHB. In: Proceedings Ninth International Conferance on Computer Communications and Networks, Las Vegas, Nevada. (2000)
17. Azeem, F., Rao, A., Kalyanaraman, S.: A tcp-friendly traffic marker for ip differentiated services (2000)
18. Feng, W., Kandlur, D., Saha, D., Shin, K.: Adaptive Packet Marking for Maintaining End-to-End Throughput in a Differentiated Services Internet. In: IEEE/ACM Transactions on Networking. Volume 7. (1999) 685–697
19. Nandy, B., Seddigh, N., Pieda, P., Ethridge, J.: Intelligent traffic conditioners for assured forwarding based differentiated services networks. In: IFIP High Performance Networking (HPN 2000). (2000)
20. El-Gendy, M.A., Shin, K.G.: Equation-based packet marking for assured forwarding services. Infocom (2002)
21. Chait, Y., Hollot, C., Misra, V., Towsley, D., Zhang, H.: Providing throughput differentiation for tcp flows using adaptive two color marking and multi-level aqm. In: Proceedings of Infocom 2002. (2002)
22. Dorfinger, P., Brandauer, C., Hofmann, U.: A rate controller for long-lived TCP flows. In: Joint International Workshop on Interactive Distributed Multimedia Systems and Protocols for Multimedia Systems (IDMS/PROMS). (2002) 154–165
23. Padhye, J., Firoiu, V., Towsley, D., Kurose, J.: Modeling TCP Throughput: A Simple Model and its Empirical Validation. In: ACM SIGCOMM'98. (1998)
24. Padhye, J., Firoiu, V., Towsley, D., Krusoe, J.: Modeling TCP throughput: A simple model and its empirical validation. Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication (1998) 303–314

25. Seddigh, N., Nandy, B., Pieda, P., Salim, J.H., Chapman, A.: An Experimental Study of Assured Services in a Diffserv IP QoS Network (1998)
26. Floyd, S., Jacobson, V.: Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking **1** (1993) 397–413
27. Floyd, S.: The RED Web Page (1997) `http://www.aciri.org/floyd/red.html`.
28. Ziegler, T., Brandauer, C., Fdida, S.: A quantitive Model for Parameter Setting of RED with TCP traffic. In: Proceedings of the Ninth International Workshop on Quality of Service (IWQoS), 2001, Karlsruhe, Germany. (2001)
29. Brandauer, C.: Web interface to the RED model developed in [28] (2000) `http://www.salzburgresearch.at/~cbrand/REDmodel`.
30. Firoiu, V., Borden, M.: A study of active queue management for congestion control. In: Proceedings of IEEE Infocom, Tel Aviv 2000 IEEE Computer and Communications Societies Conference on Computer Communications (INFOCOM-00). (2000) 1435–1444
31. Clark, D., Fang, W.: Explicit allocation of best effort packet delivery service. IEEE/ACM Transactions on Networking **6** (1998) 362–373
32. et al., S.S.: Traffic handling studies (2003)
33. Brandauer, C.: Web interface to the WRED model (2001) `http://www.salzburgresearch.at/~cbrand/WREDmodel`.
34. Network Simulator ns-2, see `http://www.isi.edu/nsnam/ns/`.