A Queue Management Algorithm for Intra-Flow Service Differentiation in the "Best Effort" Internet

Henning Sanneck and Georg Carle
GMD Fokus
Kaiserin-Augusta-Allee 31
D-10589 Berlin, Germany

Abstract—**While there is a high demand for IP based real-time services, it is still an open issue how to provide these services efficiently and scalable with a Quality-of-Service appropriate for audio and video. Mechanisms of service differentiation between flows ("inter-flow" QoS) have been explored extensively. However, actual deployment has been delayed, mainly due to the complexity of maintaining a complete QoS architecture (including admission/policy control and charging/accounting). End-to-end QoS can however also be improved by exploiting knowledge about the flow structure to influence the flow within the network. This leads to a graceful degradation under congestion ("intra-flow" QoS). Typically this is accomplished by filtering higher-layer information within the network, which is both expensive in terms of resources, as well as undesirable with regard to network security. In this paper, we present a queue management algorithm called DiffRED (Differential Random Early Detection) that allows to enhance intra-flow QoS without higher-layer filtering. The algorithm differentiates between packets marked by the sender as either more or less eligible to be dropped in comparison to unmarked packets. This gives the application some control over the packet loss process and thus enhances the performance of available end-to-end loss recovery mechanisms, end-to-end fairness and finally the perceived quality. The algorithm helps to bridge the huge gap between the current "best effort" Internet and full deployment of inter-flow service differentiation. We introduce simple metrics to describe the loss process of individual flows and present simulation results for a voice service using the proposed scheme. We demonstrate how the algorithm provides a significant intra-flow QoS enhancement and evaluate the impact on conventional traffic.**

I. Motivation: Intra-flow QoS

Recently, we have seen research efforts on how to use information on a flow's structure (e.g. the association of packets to frames in an Application Data Unit ADU) to allow a graceful degradation of the flow when either no mechanisms for differentiation between flows are present ("best effort" Internet) or the flow violates its contracted traffic profile. We describe these mechanisms with the term "*intra-flow*" QoS enhancement (as opposed to "*inter-flow*" QoS where differentiation between flows takes place). For video traffic there have been several proposals (e.g. Frame-Induced Packet Discarding [1], Transcoding, Transform Coefficient Filters), some of which also include an alignment with inter-flow QoS mechanisms [2], [3]. However, these application-level approaches typically suffer from adding significant (application layer) complexity to nodes interior to the network, contradict with network security constraints and are generally very dependent on the supported payload types which are subject to change over time.

Due to the low per-flow bandwidth for real-time voice, most of the approaches mentioned above for video do not apply. A low-bitrate voice stream typically can neither be source-rate-adaptive nor easily be filtered/transcoded further. Thus, in the absence of inter-flow protection, it needs to be augmented with end-to-end loss recovery. Due to real-time constraints, open-loop error control is used (FEC, [4], [5]), in particular by adding an additional lower quality low bitrate source coding [6]. Alternatively or in combination with FEC, we can exploit long-term correlation within the speech signal for concealment of the signal degradation [7], [8].

However, FEC and loss concealment are limited in the number of consecutive packet losses which can be treated. For FEC, the limitation lies in the additional data and delay overhead necessary to detect and recover consecutive losses. For concealment, the limitation is due to the assumption of quasi-stationarity for speech. This is only valid for a time period typically equivalent to one or two packets. Given these constraints, concealment and forward error recovery approaches become much less efficient as the loss burstiness increases, as shown e.g. in [9], [10].

The previous arguments underline the importance of mapping application requirements with regard to their ADU format and end-to-end quality enhancement capabilities to network mechanisms effectively controlling the *distribution* of losses within a flow. Such mechanisms also support certain fairness aspects. Without controlling the loss process bursty background traffic could cause bursty losses (dropouts) for a voice flow without affecting a fair long-term bandwidth share.

In this work we present a simple network mechanism

0-7803-5794-9/99/$10.00 © 1999 IEEE

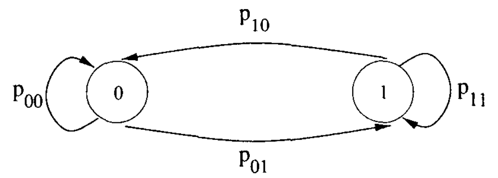

Fig. 1. Gilbert Model

which allows loss control on a per-flow basis which bridges the huge gap between just employing end-to-end loss recovery mechanisms and full deployment of service differentiation/reservation in the network including admission control, negotiation of service level agreements and charging/accounting. We show how applying the algorithm to best-effort voice flows significantly improves application-level QoS without impairing the quality of other flows.

The structure of the paper is as follows: Section II. introduces simple metrics to describe the loss process. In section III. we present our proposed algorithm which is an extension to the Random Early Detection (RED, [11]) concept. Section IV. evaluates a preliminary implementation of the algorithm by simulation. In section V. we summarize our findings and conclude the paper.

II. Simple intra-flow loss metrics

Intra-flow loss metrics[1] introduced up to now (see, e.g., [12], [13]) have been mainly used for admission control, i.e. in the access control path of multiplexers. In contrast, we consider a "best effort" Internet scenario where real-time flows can start and end at any time without explicit setup, i.e. the network has no a-priori knowledge of connections, and thus intra-flow QoS has to be enforced in the data path.

To characterize the behaviour of the network as seen by one flow, we use the well-known Gilbert model (Fig. 1). The system can be completely described by the probability p_{01} for a transition from state 0 (no loss) to state 1 (loss) and the probability p_{11} to remain in state 1. The probability p_{11} represents the *conditional loss probability clp*. The probability of being in state 1, representing the mean loss, is called *unconditional loss probability ulp* and can be computed as follows:

$$ulp = \frac{p_{01}}{1 - p_{11} + p_{01}} \qquad (1)$$

The Gilbert model implies a geometric distribution of the probability for the number of consecutive losses k, $(1 - clp)clp^{k-1}$, which is known to approximate well the head of the loss distribution of actual traces. (The tail of the distribution is typically dominated by few events, caused e.g. by link outages and route flappings, and cannot be captured by a simple model [14].) Fig. 2 shows how the (clp, ulp) space is covered by the Gilbert model using p_{01} as a parameter.

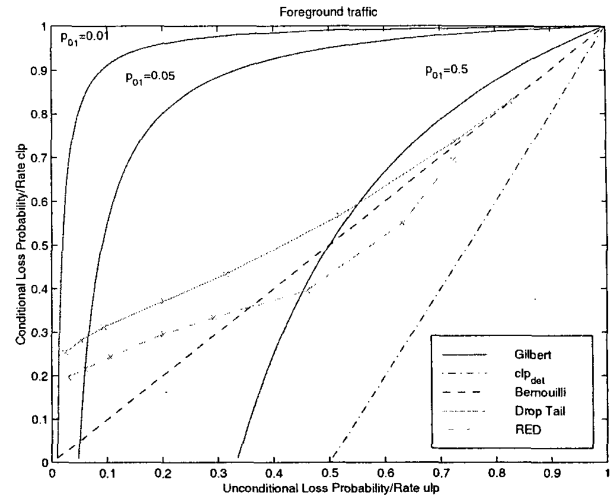

Fig. 2. Conditional Loss Probability vs. Unconditional Loss Probability: Gilbert Model, *clp*-bound clp_{det} and simulations of Drop-Tail and RED algorithms for foreground traffic

If losses of one flow are correlated (i.e. the loss probability of an arriving packet is influenced by the contribution to the state of the queue by a previous packet of the same flow and/or both the previous and the current packet see bursty BT arrivals, [15]) we have $p_{01} \leq clp$ and thus $ulp \leq clp$ (upper half of Fig. 2). For $p_{01} = clp$ the Gilbert model is equivalent to a 1-state (Bernouilli) model with $ulp = clp$ (no loss correlation).

Fig. 2 leads us to the conclusion that simple queue management algorithms can be designed that allow the adjustment of the conditional loss probability for individual flows[2], while keeping the unconditional loss probability within a controlled bound around the value that is determined by the background traffic load, buffer size, and scheduling policy, but not by the queue management algorithm itself. In the following we call flows sharing a queue

[1] "Intra-flow" loss probabilities are also refered to as "short-term" loss probabilities because the are typically using random variables describing "close" loss events in terms of the packet sequence.

[2] Note that by modifying the queue management algorithm, we cannot cannot change the conditional loss probability *clp* below a theoretical limit. This limit can be explained as follows: clearly the *clp* can be zero up to $ulp = 0.5$. Then, a deterministic loss pattern with every other packet lost is reached. When further increasing the loss rate, even when considering a deterministic loss pattern, burst losses cannot be avoided. This lower bound is thus given by the deterministic conditional loss probability clp_{det} :

$$clp_{det} = \begin{cases} 0: 0 \leq ulp < 0.5 \\ 2ulp - 1: 0.5 \leq ulp \leq 1 \end{cases}$$

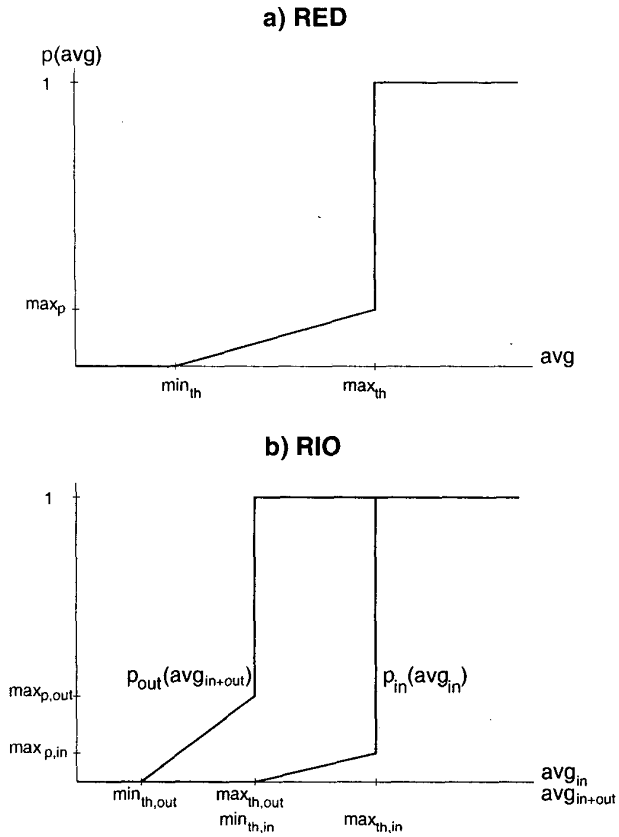

Fig. 3. RED and RIO drop probabilities

under the control of such an algorithm *foreground* traffic (FT) and the remaining flows in that queue *background* traffic (BT).

With the RED [11] algorithm there exists already a queue management algorithm whose modifications to the queue behaviour can be described with the Gilbert model parameters previously introduced. To be able to accomodate bursts in the queue, as well as not to over-react during transient congestion, the instantaneous queue size q is low-pass filtered resulting in an *average queue size* (avg) which is used to compute the drop probability (see Fig. 3 a). By employing RED, the parameter p_{01} of the queue is thus increased by gradually increasing the packet drop probability (according to the measured average queue size) before the queue is completely filled. RED was designed to signal congestion to adaptive flows (TCP) and to reduce the average delay independently of the association of packets to flows. However, being interested in the *clp*, we see from Fig. 2 that for a given *ulp*, increasing p_{01} amounts to a reduction in the *clp*. This effect can be seen in Fig. 2 for simulations we conducted with parameters detailed in the appendix. For all *ulp* values, the conditional loss probability when using RED is below that

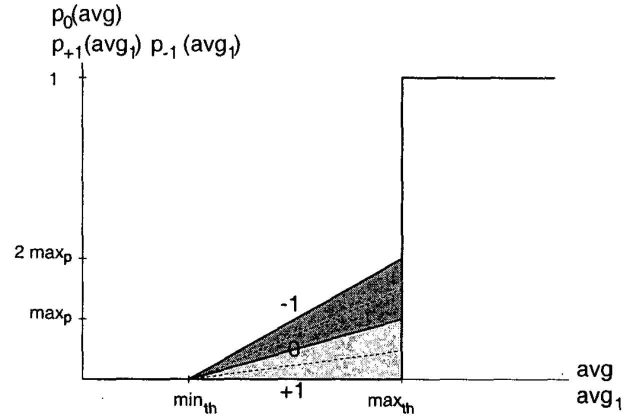

Fig. 4. DiffRED drop probabilities as a function of average queue sizes

for a Drop Tail queue. Only under heavy overload (when the RED algorithm is also just tail dropping most of the time), the RED curve approaches the Drop Tail one.

III. The Differential RED (DiffRED) Algorithm

One approach to realize inter-flow service differentiation using a single queue is RIO ('RED with IN and OUT', [16]). With RIO, two average queue sizes are computed (Fig. 3 b): one just for the IN packets and another for both IN and OUT packets. Packets marked as OUT are dropped earlier (in terms of the average queue size) than IN packets.

RIO has been designed to decrease the *ulp* seen by particular flows at the expense of other flows. In this work however, we want to keep the *ulp* as given by other parameters[3] while modifying the *clp* parameter for the foreground traffic. Fig. 4 shows the conventional RED drop probability curve (p_0 as a function of the average queue size for all arrivals avg), which is applied to all unmarked ("0") traffic (background traffic: BT). Foreground traffic (FT) packets marked as less eligible for a drop ("+1") are dropped with a probability as given by the lower thick line. This lower probability is compensated by the higher drop probability for the foreground traffic packets marked as ("-1"), i.e. packets more eligible for a drop. This implies that the initial ratio of $+1$ to -1 packets of a flow must be 1. Thus, a service differentiation for foreground traffic is possible which does not differ from conventional RED behaviour in the long term average (i.e. in the *ulp*), presumed the shares of $+1$ and -1 at a gateway are equal. Additionally, packets of one flow carrying different markers are not reordered.

[3] Note that this does not preclude a combination with mechanisms enforcing a certain *ulp*, e.g. with a link sharing scheduler.

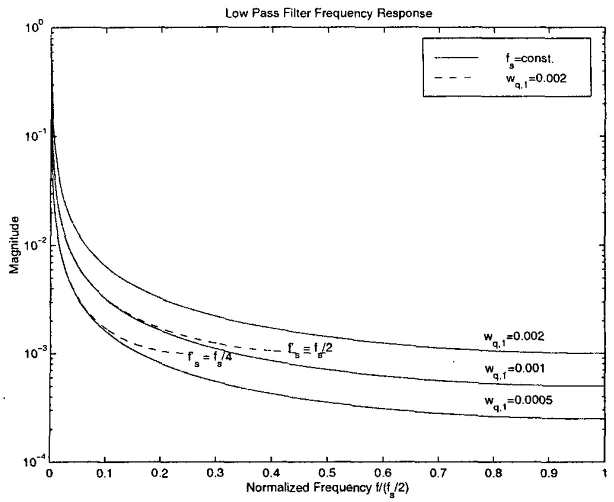

Fig. 5. Low Pass Filter Frequency Response

For the packet marking the IPv4 Type-of-Service (ToS) byte could be used (which is especially suitable in its new DiffServ meaning of the AF (Assured Forwarding, [17]) Per-Hop Behaviour (PHB) which offers three drop precedence levels per class).

A. Queue size sampling

Considering a rather small fraction of FT traffic at the gateway and using the average queue size avg ($avg_1 = avg$, Fig. 4) for the calculation of the $+1,-1$ drop probabilities p_{+1} and p_{-1} we can identify the following problem: the state of the queue and thus the avg value may have changed significantly between consecutive FT arrivals. Thus a value for the drop probability is computed which does not reflect adequately the evolution of the queue state as seen by the FT fraction and its contribution to it. Ideally $\frac{p_0(avg_1(n))+p_0(avg_1(n+1))}{2}$ (where packet n is a $+1$ packet and packet $n+1$ is a -1 packet or vice versa, and $avg_1(n)$ is the value of avg_1 at arrival of packet n) should equal the drop probability computed for the -1 packet (either $p_{-1}(avg_1(n))$ or $p_{-1}(avg_1(n+1))$). If this relation is not approximated by the algorithm, it can lead to an unfair distribution of drops between the FT and the BT fraction.

The described problem can be solved by changing the low pass filter parameter as a function of the ratio of the number of FT arrivals to the overall number of arrivals when sampling the queue size and then computing an additional average queue size for the FT arrivals (avg_1). However, in this case we need to keep additional state about the number of FT arrivals, need to re-calculate the filter parameter and avg_1 at every arrival.

Instead, our approach avoids this complexity by sampling the queue length q only at the FT arrival instants. Now, the avg_1 filter is a sub-sampled version of the avg filter, with a subsampling factor equal to the current ratio of all arrivals to the FT arrivals. Fig. 5 shows the magnitude of the filter frequency response (assuming a time-invariant system) when modifying the filter parameter $w_{q,1}$ (solid lines), as well as when keeping $w_{q,1}$ constant and changing the sampling frequency to f_s' (dashed lines).

Now we can compute the drop probabilities for the different priority packets as follows:

$$p_0(avg) = \begin{cases} 0: & avg < min_{th} \\ max_p \frac{avg - min_{th}}{max_{th} - min_{th}}: & min_{th} \leq avg < max_{th} \\ 1: & avg \geq max_{th} \end{cases} \quad (2)$$

$$p_{-1}(avg_1) = \begin{cases} 0: & avg_1 < min_{th} \\ 2max_p \frac{avg_1 - min_{th}}{max_{th} - min_{th}}: & min_{th} \leq avg_1 < max_{th} \\ 1: & avg_1 \geq max_{th} \end{cases} \quad (3)$$

$$p_{+1}(avg_1) = \begin{cases} 0: & avg_1 < max_{th} \\ 0: & min_{th} \leq avg_1 < max_{th} \\ 1: & avg_1 \geq max_{th} \end{cases} \quad (4)$$

B. Irregular partition of $+1, -1$ arrivals

To discourage abuse by malicious users who could send just $+1$ packets, we compute low-pass filtered values of the arrival function of $+1$ packets (arv_{+1}) and -1 packets (arv_{-1}). The arrival function is defined as follows:

$$a_{x,FT} = \begin{cases} 0: & \text{FT packet type} \neq x \\ 1: & \text{FT packet type} = x \end{cases} \quad (5)$$

Note that the arrival function describes the FT arrival process, and not the sampling of overall arrivals at $+1, -1$ arrival instants. The arrival function for all FT packets $a_{|1|,FT}$ is thus 1 for all samples ($arv_{|1|} \to 1$). The choice of the averaging filter parameter allows to adjust the burst length of $+1, -1$ packets respectively which can be accomodated, while avoiding a persistent mismatch of the partition between $+1$ and -1 packets.

A correction is added to $p_{-1}(avg_1)$ and $p_{+1}(avg_1)$ to decrease the -1 loss probability and to increase the $+1$ probability at the same time thus degrading the service for all users[4]. The correction depends on the mismatch between the $+1$ and -1 arrivals. The shaded areas above and below the $p_0(avg)$ curve (Fig. 4) show the operating area when the correction is added. The corrected values for the $+1, -1$ drop probabilities (Eqs. 3 and 4) for the interval $min_{th} \leq avg_1 < max_{th}$ are:

$$p'_{-1}(avg_1) = p_{-1}(avg_1) - \frac{|arv_{+1} - arv_{-1}|}{arv_{|1|}} p_0(avg_1) \quad (6)$$

$$p'_{+1}(avg_1) = \frac{|arv_{+1} - arv_{-1}|}{arv_{|1|}} p_0(avg_1) \quad (7)$$

[4] Another option, yet with significantly higher overhead, would be to identify and deny access to the misbehaving flows.

Every congested DiffRED hop will increase the mismatch between the number of +1 and −1 packets at the next hop. If this effect becomes significant is a function of the number of congested hops already traversed by the flows present, as well as the congestion situation at a gateway and the relation of the presence of "fresh" flows which enter the network and flows which have already experienced several congested gateways. Note that the higher the individual loss of a flow, the higher is the ratio of +1 to −1 packets of that flow. Thus the flow is protected more at subsequent gateways supporting end-to-end fairness.

C. Packet marking policy

We have a variable marking granularity, i.e. that marking across flows and thus also inter-flow differentiation is possible. This means that a flow sent by a host could receive more +1 marking at the expense of another one sent concurrently, which would mark more packets as −1. However the ratio of packets marked as +1 to the packets marked as −1 must remain 1 over short time intervals (the length of these time intervals depend on the DiffRED gateway filter parameters as described in paragraph B.). Thus, volume-based charging (or ingress monitoring and suppression of mis-behaving flows) is needed, as otherwise users could inject just −1 traffic to completely mark another flow as +1.

IV. Results

We used the same simulation scenario as in section II. with the parameters as given in the appendix. The foreground traffic share of the offered load $\frac{\lambda_{FT}}{\lambda}$ was varied at a fixed traffic intensity level to assess the performance of RED, DiffRED without subsampling ($avg_1 = avg$) and DiffRED with subsampling. The mean of the traffic intensities for the examples is $\bar{\rho} = 0.9521$ with standard deviation of $\sigma_\rho = 0.0013$ (the differences in the traffic intensity levels are due to the changing distribution of flow types and thus traffic patterns). The distributions of flows ranges from 20/7/1 H/D/voice flows at $\frac{\lambda_{FT}}{\lambda} = 0.01$ to 9/3/32 H/D/voice flows at $\frac{\lambda_{FT}}{\lambda} = 0.5$, where "H" and "D" flows constitute the background traffic (BT) fraction and "voice" flows are foreground traffic as described in the appendix. Every voice source marked its packets alternately with +1 and −1.

Fig. 6 shows the average of the mean loss rates of the FT flows $p_{L,FT}$ normalized with the mean loss rate calculated over all traffic p_L. It can be seen that for DiffRED without subsampling, the algorithm drops significantly more packets of the FT flows, due to the missing correlation of the avg and thus p_{+1} and p_{-1} values between consecutive FT arrivals. With subsampling however the FT flows receive a mean loss rate just above p_L except for very low FT shares. For plain RED the figure shows that the algorithm is biased slightly against the non-adaptive bursty H-type

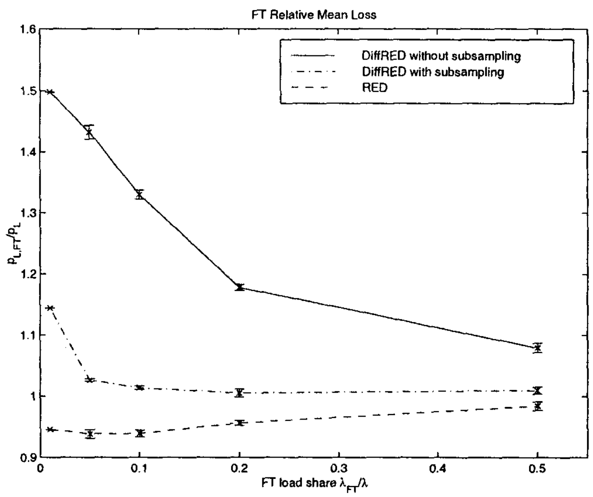

Fig. 6. Foreground traffic relative mean loss

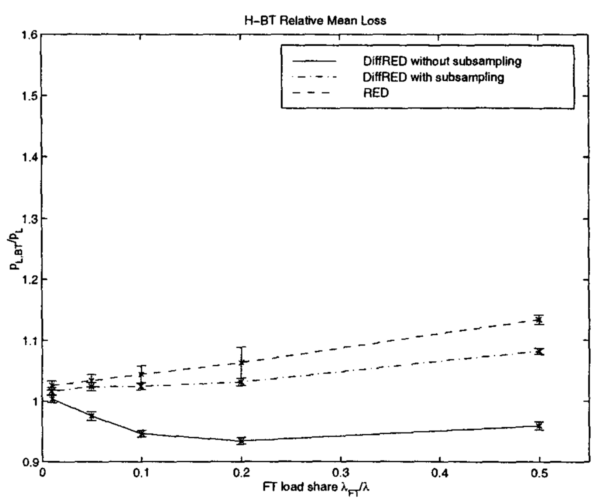

Fig. 7. Background traffic relative mean loss

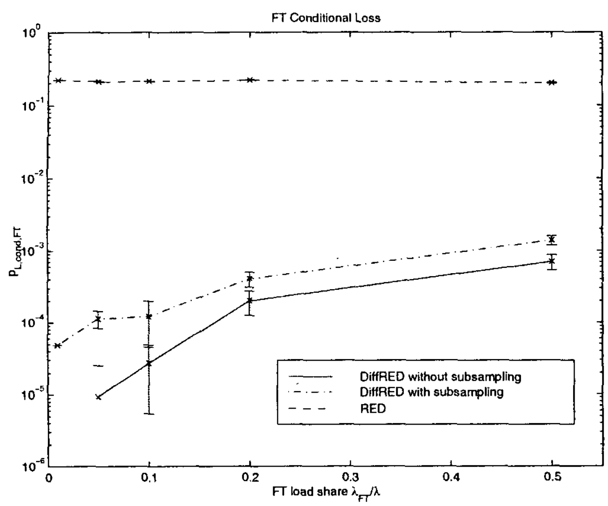

Fig. 8. Foreground traffic conditional loss

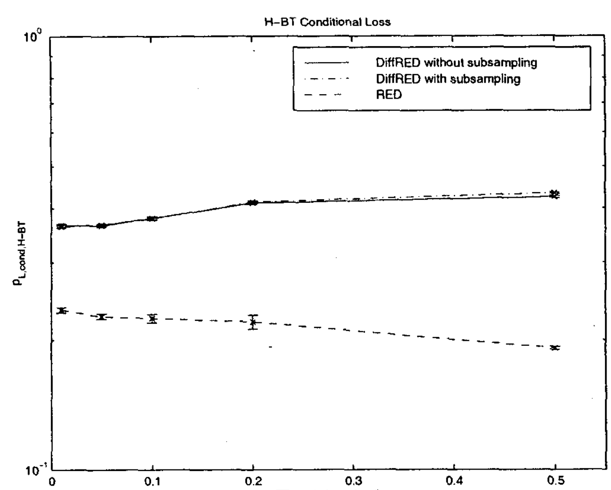

Fig. 9. Background traffic conditional loss

BT traffic and thus is in favor of the non-bursty FT traffic ($\frac{p_{L,FT}}{p_L} < 1$), an effect which decreases with increasing FT share. Fig. 7 shows the described properties in terms of the H-type BT traffic. We obtained the same utilization with any of the three algorithms. This is expected, because all three algorithms use the same minimum and maximum threshold parameters and the behaviour when $min_{th} < avg < max_{th}$ in terms of the *aggregate* traffic seen over time intervals significantly larger than flow burst intervals is identical.

Figs. 8 and 9 show the conditional loss rates $p_{L,cond,FT}$ and $p_{L,cond,H-BT}$ for the foreground and H-type background traffic respectively. Here we give the absolute values as we cannot reasonably define a $p_{L,cond}$ value for the entire system (across different flow types with different traffic envelopes). In the given scenario we can decrease the conditional loss rate for FT traffic by at least two order of magnitude by employing DiffRED instead of RED (Fig. 8). $p_{L,cond,FT}$ is increasing with the flow share as for an increasing number of voice flows we have a higher probability that bursts of +1 packets arrive which might drive the avg just over the max_{th} limit (where p_{+1} jumps from 0 to 1).

Apart from the overhead of keeping an additional average queue size $(avg_1)^5$, we now impose a higher conditional loss rate (Fig. 9) on the (non-adaptive) background traffic. In DiffRED a (burst of) +1 packet(s) has a direct impact on the conditional loss probability of a BT flow. In the approach proposed in [18], we have directly associated $+1, -1$ events, i.e. a +1 packet is only protected if a -1 packet which can be dropped at once instead is already present in the queue. Thus the loss processes of the FT and BT packets are less correlated. The disadvantages are however a potentially larger buffer requirement, the dropping of already queued traffic (including the overhead of searching in the queue) and higher resulting FT conditional loss rate.

V. Conclusions

In this paper we have shown how intra-flow QoS requirements of applications can be mapped to simple differentiated packet marking which is then enforced within the network by a simple queue management mechanism, the DiffRED algorithm. By extending the well-known RED algorithm to comprise two additional drop probability functions per differentiation level, we are able to control the loss characteristics of individual flows while keeping their unconditional loss probability within a controlled bound around the value expected using a conventional RED algorithm. We demonstrated the usefulness of DiffRED for a single level of differentiation, a simple metric (conditional loss probability) and for an application with a simple flow structure (voice).

[5] Plus keeping the low-pass filtered arrival values, if the correction as described in section III.B. is enabled.

The scheme is extensible to several differentiation levels satisfying more complex intra-flow QoS requirements and more complex metrics, based on the specific ADU format (e.g. the frame association of packets).

We have shown how the DiffRED algorithm provides service differentiation *within* a flow. The algorithm improves application-level QoS without impairing the throughput for other flows and without reserving dedicated resources for the considered flows (the flows are still "best effort" flows). Therefore, the deployment of the algorithm does not presume the deployment of major elements of a QoS architecture (admission control, charging/accounting, etc.). Real-time applications can benefit significantly from influencing the loss characteristics. The algorithm can achieve application-level QoS improvements by avoiding audio "dropouts" and "frozen" video due to partial frame loss which might render a frame undecodable. The scheme is highly suitable for QoS enhancement in the migration from a purely best effort Internet to a QoS-enhanced one, as it does not need to be implemented in every router in order to be effective and can finally be extended to provide inter-flow QoS (RIO). DiffRED improves the performance of FEC and error concealment by reducing the probability of consecutive (bursty) packet losses.

The presented simulations used voice flows as a foreground traffic. It was shown how DiffRED can help to assure continous playout and simple loss recovery for such sources. While the simulations demonstrated significant improvements for voice flows, additional studies are required in order to investigate the impact of high bandwidth (video) sources which send bursts of +1 packets followed by bursts of −1 packets.

Appendix

We implemented the DiffRED algorithm into a modified version of the NS-2 network simulator [19], which allows tracing of the occurence o_k of burst losses of length k for individual flows. Thus for a given number of packet arrivals a (experiencing $d = \sum_{k=1}^{\infty} k o_k$ drops) of a flow we have the *mean loss rate* (*ulp* for $a \to \infty$) $p_L = \frac{d}{a}$. With $b = \sum_{k=1}^{\infty}(k-1)o_k$ being the frequency of "two consecutive packets lost", we calculate a *conditional loss rate* as $p_{L,cond} = \frac{b}{d}$ (*clp* for $d \to \infty$).

We use a simple network scenario where several flows experience a single bottleneck link (e.g. a small bandwidth access link connecting a customer LAN to an ISP or a base station connecting mobile hosts to a LAN). In our simulation the bottleneck link has a link-level bandwidth of $\mu = 1920 kBit/s$. Several flows fed to the gateway over $10 Mbit/s$ links are multiplexed to either a DiffRED queue with or without subsampling or a conventional RED[6] output queue.

TABLE I
SOURCE MODEL PARAMETERS

Traffic type	*H-BT*	*D-BT*	*FT (voice)*
flow share (%) of BT	75	25	-
peak bandwidth ($\frac{kBit}{s}$)	256	30...34	83.2
packet size (bytes)	560	128	208
on/off distribution	Pareto	Expo.	Expo.
shape parameter	1.9	–	–
mean burst (packets)	20	4	18
mean ontime (s)	0.35	0.12...0.14	0.36
mean offtime (s)	0.7	0.12...0.14	0.64

We used the same traffic model as in [18], that reflects results from various recent Internet Access-LAN and Internet backbone measurements [20]–[22]: the majority of traffic (in terms of flows and volume) are http transfers ("H-type" background traffic). The rest are mostly short-lived flows dominated by DNS traffic ("D-type" background traffic), which has a relatively large share of the active flows, yet only a small share of the traffic volume[7]. The values we chose for modeling of individual sources are shown in Table I. To model Web traffic we use a Pareto distribution [23] both for the ON and OFF periods of the source. By using a variance-time ($var(X(m)) - m$) plot [24], describing the variance of the process of arrivals X dependent on the scale of averaging m, we determined that the aggregation of the described background traffic sources produces long-range dependent traffic. As the DiffRED algorithm tries to influence the loss burstiness of individual flows, it is crucial to reflect the existing "burstiness on all time scales" of the aggregate arrival process in the model. To model voice sources with silence detection, we employed a model widely used in the literature (see e.g. [13]) where ON (talkspurt) and OFF periods are exponentially distributed with a speaker activity of 36%. Every voice source marks its stream with a $+1, -1, +1, -1, \ldots$ profile. The same profile could be applied meaningfully to video traffic at the *frame* level, for coding schemes where every frame has the same importance (e.g. M-JPEG). Then, however, there is a higher probability (dependent on the degree of multiplexing) that bursts of +1 packets (and bursts of −1 packets) arrive at the gateway (see section III.B.).

Table I also gives "raw" peak bandwidth and packet sizes (i.e. including packet header overhead[8]). The range of $30 \ldots 34 \frac{kBit}{s}$ D-type BT bandwidth and $0.12 \ldots 0.14 s$ for the on-/offtimes is due to the changing number of flows and load for the different experiments. Packet inter-

[6] We used the implementation of the NS-2 distribution.

[7] The small per-flow bandwidth of the D-type BT allows us to set the background traffic load with a relatively fine granularity.

[8] We assume 8 bytes link level overhead and 20, 20, 8, 12 IP-, TCP-, UDP-, RTP-packet overhead respectively.

departure times within a burst are uniformly distributed in the interval $[0.95I, 1.05I]$ (with I being the packet inter-departure time calculated from the values of Table I) to avoid phase effects caused by the exact timing of packet arrivals in the simulator.

We found a simulation time of $5 \cdot 10^4$ seconds (13.9 hrs.[9] with the number of packet arrivals ranging from $16 \cdot 10^6$ to $27 \cdot 10^6$) sufficient for the Pareto sources to "warm up" and thus to guarantee that the traffic shows long-range dependence as well as to result in a statistically relevant number of drop events even for low loss rates as a basis for performance measures ($p_{L,cond}$). We averaged the results for one flow group (H, D, voice). On the figures 6-9 we also plot error bars giving the standard deviation for the averaged values (this is to verify that every flow of a group has identical behaviour seen over the entire simulation time).

The (Diff)RED parameters used for all simulations are as follows: $min_{th} = 5$, $max_{th} = 15$, $max_p = 0.1$, $w_q = w_{q,1} = 0.002$ and maximum queue size is 21 packets.

Acknowledgment

We would like to thank Adam Wolisz, Michael Zander and the anonymous reviewers for valuable comments.

References

[1] S. Ramanathan, P.V. Rangan, and H. Vin, "Frame-induced packet discarding: An efficient strategy for video networking," in *Proceedings of NOSSDAV '93*, 1993, pp. 173–184.

[2] L. Wolf, R. Herrtwich, and L. Delgrossi, "Filtering multimedia data in reservation-based internetworks," Technical Report 43.9608, IBM European Networking Center, August 1994.

[3] R. Wittmann and M. Zitterbart, "AMnet: Active multicasting network," in *Proc. of Intern. Conf. on Communications (ICC'98)*, Atlanta, GA, June 1998.

[4] C. Perkins, O. Hodson, and V. Hardman, "A survey of packet-loss recovery techniques for streaming audio," *IEEE Network Magazine*, Sept./Oct. 1998.

[5] C. Perkins and O. Hodson, "Options for the repair of streaming media," RFC 2354, IETF, June 1998, ftp://ftp.ietf.org/rfc/rfc2354.txt.

[6] C. Perkins, I. Kouvelas, O. Hodson, M. Handley, and J. Bolot, "RTP payload for redundant audio data," RFC 2198, IETF, September 1997, ftp://ftp.ietf.org/rfc/rfc2198.txt.

[7] H. Sanneck, A. Stenger, K. Ben Younes, and B. Girod, "A new technique for audio packet loss concealment," in *Proceedings IEEE Global Internet 1996 (Jon Crowcroft and Henning Schulzrinne, eds.)*, London, England, November 1996, pp. 48–52, ftp://ftp.fokus.gmd.de/pub/glone/papers/Sann9611:New.ps.gz.

[8] H. Sanneck, "Concealment of lost speech packets using adaptive packetization," in *Proceedings IEEE Multimedia Systems*, Austin, TX, June 1998, ftp://ftp.fokus.gmd.de/pub/glone/papers/Sann9806:Adaptive.ps.gz.

[9] N. Shacham and P. McKenney, "Packet recovery in high-speed networks using coding and buffer management," in *Proceedings ACM SIGCOMM '90*, San Francisco, CA, June 1990, pp. 124–131.

[10] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of packet loss processes in high-speed networks," *IEEE Transactions on Information Theory*, vol. 39, pp. 98–108, January 1993.

[11] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.

[12] T. Miyata, H. Fukuda, and S. Ono, "New network QoS measures for FEC-based audio applications on the Internet," in *Proceedings IEEE IPCCC 1998*, Tempe/Phoenix, AZ, USA, February 1998, pp. 355–362.

[13] R. Nagarajan, J. Kurose, and D. Towsley, "Finite-horizon statistical Quality-of-Service measures for high speed networks," *J. High Speed Networks*, December 1994.

[14] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for interactive audio in the Internet," in *Proceedings IEEE Infocom '99*, New York, NY, March 1999.

[15] H. Schulzrinne, J. Kurose, and D. Towsley, "Loss correlation for queues with bursty input streams," in *Proceedings ICC '92*, Chicago, IL, 1992, pp. 219–224.

[16] D. Clark and W. Fang, "Explicit allocation of best effort packet delivery service," Technical Report, MIT LCS, 1997, http://diffserv.lcs.mit.edu/Papers/exp-alloc-ddc-wf.pdf.

[17] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured forwarding PHB group," RFC 2597, IETF Diffserv Working Group, June 1999, ftp://ftp.ietf.org/rfc/rfc2597.txt.

[18] H. Sanneck and G. Carle, "Predictive loss pattern queue management for Internet routers," in *Internet Routing and Quality of Service, S. Civanlar, P. Doolan, J. Luciani, R. Onvural, Editors, Proceedings SPIE Vol.3529A*, Boston, MA, November 1998, ftp://ftp.fokus.gmd.de/pub/glone/papers/Sann9811:Predictive.ps.gz.

[19] UCB/LBNL/VINT, *Network simulator ns-2*, October 1998, http://www-mash.cs.berkeley.edu/ns/ns.html.

[20] K. Claffy, G. Miller, and K. Thompson, "The nature of the beast: Recent traffic measurements from an Internet backbone," in *Proceedings INET '98*, Geneva, Switzerland, July 1998, http://www.isoc.org/inet98/proceedings/6g/6g_3.htm.

[21] P. Newman, T. Lyon, and G. Minshall, "Flow labelled IP: Connectionless ATM under IP," in *Proceedings of Networld + Interop*, Las Vegas, NV, USA, April 1996.

[22] M. Ilvesmäki, K. Kilkki, and M. Luoma, "Packets or ports - the decisions of IP switching," in *Broadband Networking Technologies, Seyhan Civanlar, Indra Widjaja, Editors, Proceedings SPIE Vol.3233*, Dallas, TX, November 1997, pp. 53–64.

[23] M. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM Transactions on Networking*, vol. Vol. 5, no. 6, pp. 835–846, December 1997.

[24] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic," in *Proceedings ACM SIGCOMM*, San Francisco, CA, September 1993.

[9] The initial 10^4 s were discarded from the datasets.