The Alternative Best-Effort Service

Paul Hurley[*]
Jean-Yves Le Boudec
Patrick Thiran
Institute for Computer Communication and Applications (ICA)
Ecole Polytechnique Fédérale de Lausanne (EPFL)
`http://icawww.epfl.ch`

SSC Technical Report SSC/1999/036

September 1999

Abstract

Alternative Best-Effort (ABE) is a novel service for IP networks which offers applications the choice between receiving a lower end-to-end delay and receiving more overall throughput. Every best effort packet is marked as either *green* or *blue*. Green packets receive a low, bounded queueing delay. To ensure blue packets do not suffer as a result, green flows receive less throughput during bouts of congestion.

The unique combination of lower delay with reduced throughput for green makes it different from recent differentiated service proposals such as expedited forwarding [19] and assured forwarding [21]. The incentive to choose one or other is based on the nature of one's traffic and on traffic conditions. Typically, green flows have real-time deadlines (e.g. interactive audio), while blue traffic (e.g. bulk data transfer) seeks to minimise overall transfer time. There is benefit for all traffic in that green traffic achieves a low delay and blue traffic will receive at least as much throughput as it would in a flat best-effort network and usually more. Neither traffic type can be said to be better, thus flat rate pricing may be maintained, and there is no need for reservations or profiles.

We first describe the ABE service. We then describe and simulate a first generation router implementation. It combines packet drop differentiation with differential scheduling for blue and green packets. Green packets have a fixed bounded delay. Differential dropping is done by ensuring blue flows are compensated for the increased delay by higher throughput. Given these constraints, the parameters of the system are regulated to minimise green losses. Simulations show that our implementation is able to implement our definition of the ABE service.

[*]Contact author: paul.hurley@epfl.ch, Ph. +41-21-693-6626, Fax +41-21-693-6610

Keywords Alternative Best-Effort, ABE, Active Queue Management, Traffic Control, Packet Admission Control, real-time traffic, TCP Friendliness.

1 Introduction

We present a new IP service, Alternative Best-Effort (ABE)[1]. The goal of ABE is (1) to provide a low queueing delay service and (2) to operate in best effort mode, requiring no usage control. The first requirement is for applications with stringent real time constraints, such as interactive audio. The second requirement is an attempt to maintain the simplicity of the original Internet. With ABE, it is not required to police how much traffic uses the low delay capability; as explained below, the service is designed to operate equally well in all traffic scenarios.

ABE is designed primarily for rate-adaptive multimedia applications. These are applications that adapt to network state: the rate is reduced when negative feedback is received, and increased with positive feedback. In today's Internet, feedback is based on packet drop. In the future, binary feedback based on Explicit Congestion Notification (ECN) [23] will be used. We assume, as is required in the Internet, that rate adaptation is performed such that the application is *TCP-friendly* [8], namely, it does not receive more throughput than a TCP flow would. It is now established that it is possible to implement adaptive multimedia applications, which perform across a wide range of network conditions [24, 25]. In this context, the key idea of ABE is to provide low-delay at the expense of maybe less throughput. As discussed below, this second point is fundamental in ensuring that ABE requires no usage control.

ABE operates as follows. Best effort IP packets are partitioned into either low delay packets, called green packets, and other best effort packets, called blue packets. The choice of the terms "blue" and "green", two primary colours of equal value, is to indicate that none of the two has priority over the other, while "green", the colour of the traffic light signal for "go", indicates low queueing delay. Green packets are given the guarantee of a low bounded delay in every router. In exchange, these packets receive more losses during bouts of congestion than blue packets. If ECN is used, then green packets are more likely to be marked with the congestion bit than blue packets. For simplicity, in the rest of the document we consider only non ECN-capable systems. Nevertheless, ABE is equally valid, and indeed would work even better with ECN.

A key requirement on the service is that "green does not hurt blue", namely, if some sources send green rather than blue packets, there should be no negative impact on the throughput of those sources which remain blue. In particular, an entirely blue flow receives at least as good average throughput as it would in a flat best-effort network i.e. if all packets were blue. As a result, there is benefit to all: if some application decides to mark some packets green, then it must do so because it values the low delay more than a potential decrease in throughput; otherwise, it would mark the packets blue. In all cases, there is no penalty for other applications, which might choose to mark all their packets blue. Thus, it is not required to police the colour chosen by applications,

[1]For clarity of purpose, the name is changed from the original "Asymmetric" Best-Effort.

provided they are TCP-friendly. The enforcement of friendliness [10, 9] is outside the scope of this paper. Flat rate charging may be applied if the marketing department so decides. We leave for future discussion the issue of how to mark packets (while noting that the differentiated services framework leaves enough room for supporting the marking of ABE packets).

The value of the delay bound offered to the green service depends on how many hops are used by one flow. The exact setting of the green per-hop delay bound is the object of companion work and is outside the scope of this paper. It suffices to mention here that we expect ABE routers to be implemented at network peripherals, where bit rates are of the order of a few Mb/s or less. At very high bit rates, queueing delays are in general expected to be lower and high speed backbones probably will not need any delay differentiation. Thus a multimedia flow probably uses a small number (2 to 6) of low speed hops. An interactive audio application has a delay budget of 100-150 msec, out of which 50 msec may be allocated to network delay. As a result, we expect the green per-hop delay bound to be set to a value of the order of 5 to 10 msec.

In Section 3 we describe the ABE service model and discuss its properties. We also discuss some possible marking strategies for applications. Note that ABE addresses a different market than differentiated or integrated services. Unlike these services, ABE does not offer any guarantee, or even indication of guarantee, on throughput. A highly loaded network offering ABE will give little throughput to all best effort flows, no matter whether green or blue. However, ABE enables a moderately loaded network to offer low delay to some applications (typically, adaptive multimedia applications), as long as such applications are satisfied with the throughput they receive. We expect traditional, byte transfer oriented applications to use only blue packets. More detail on this point is given in Section 3. The design of a multimedia adaptive application that would exploit the new degree of freedom offered by ABE is outside the scope of this document.

As mentioned previously, enforcement of TCP-friendliness is outside this paper's scope. ABE neither makes worse nor improves the enforcement problem. Work is on-going to define a router implementation which combines TCP friendliness enforcement with support for ABE.

In Section 4, we consider the issue of implementing support for ABE in routers. We identify generic router requirements, and then propose one possible implementation. It is based on the combination of a deadline based scheduler and a differential active queue manager called the packet admission control (PAC) module. The PAC module uses Random Early Detection (RED) [14]. It seeks to minimise green losses subject to the constraint that losses must be sufficient to ensure the throughput of blue flows is protected. A unique feature of our implementation is that it incorporates a control loop in order to solve the problem of an optimal tuning of the PAC parameters. Our ABE router implementation is performed in the ns simulation environment, and is publicly available. The mathematical details of our implementation are given in Appendix A.

In Section 5 we provide some simulation results for this implementation. In practice, achieving the strict definition of ABE is hard under all circumstances, and we have yet to prove that our router implementation of ABE definitively satisfies this. However, under all the awkward circumstances we were able to test it under, it provided more throughput to blue than under a flat best-effort network while giving a low bounded delay to green.

2 Related Work

Within the framework of differentiated services, Expedited Forwarding [19, 20] (EF) offers a guaranteed service in the form of a virtual leased line or point-to-point connection to provide extremely low loss and low queueing delay guarantees. In the same framework, Assured Forwarding [21](AF) offers an assurance that IP packets are forwarded with high probability as long as the aggregate traffic entering the network from an edge does not exceed an agreed profile. It divides AF traffic into classes within each there are distinct levels of drop precedence. The authors suggest that an AF PHB (Per Hop Behaviour) group could be used to implement a low delay service where low loss is not an objective, by allocating an AF class with a low maximum buffer space available. SIMA [26] offers applications the choice of a level (0-7) of how "real-time" its traffic is, with each level having relatively lower delay and loss ratio than the previous one.

Crowcroft [2] proposed a low delay service, analysed by May et al [17], where desire for low delay is distinguished by a single bit. Turning on this bit ensures that the packet receives serving priority while constrained to a smaller buffer size. Depending on the input traffic and the buffer sizes of both types of traffic, this may or may not, but typically would, result in the low delay traffic also having more throughput.

Dovrolis et al [22] and Moret and Fdida [18] both describe a system based on a proportional distribution model, where the quality between classes of traffic is proportional and thus can be performed independently of the load within each class. They both propose controlling the relative queueing delays between classes. The former proposes the problem of proportional distribution of coupled delay and loss differentiation as future work. ABE trades-off delay with throughput, and thus, in a non-ECN based context, loss. This is not the same as coupling, which seeks to ameliorate both.

All the services described above are priority in one form or another. If one's traffic is non adaptive such that guaranteed low delay, high throughput and low probability of loss are needed, then priority over and above standard best-effort is the only solution. Priority schemes are complementary to ABE which does not provide any throughput guarantees. In contrast, what ABE adds is a new dimension in the best-effort class to the benefit of all within that class.

3 The ABE Service

3.1 Service Requirements

ABE is an Internet service defined by the following set of requirements.

1. ABE packets are marked either green or blue.

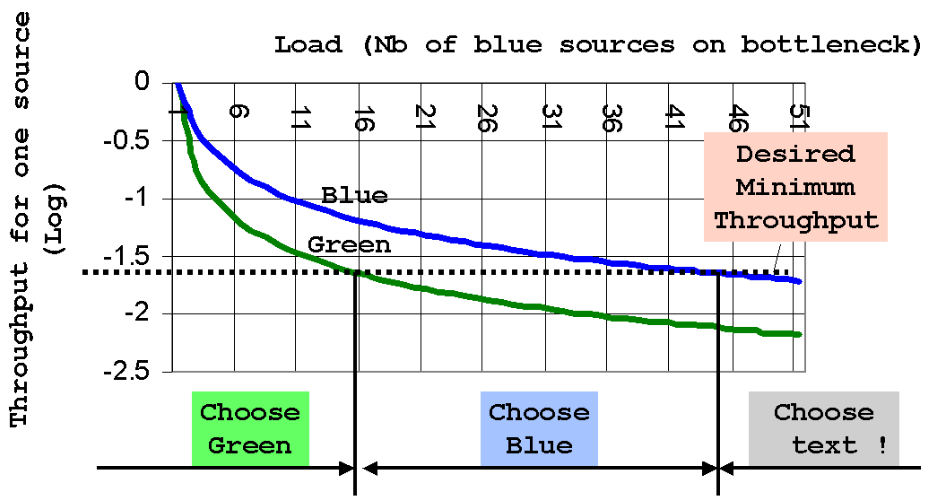


Figure 1: A possible strategy for a multimedia source using the ABE service.

2. Green packets receive a low, bounded delay at every hop. The value of the per-hop delay bound is left to the operator. As discussed in the introduction, the choice of "good" values is for further study.

3. Applications are expected to control their rate in a TCP-friendly manner. Blue and green packets are considered as belonging to the same flow, not two distinct flows.

4. (*"Transparency to Blue"*) If some sources decide to mark some of its packets green rather than blue, then the throughput of sources that mark all their packets blue remains the same or becomes better.

5. All ABE packets belong to one single best effort class. If the total load is high, then every source may receive little throughput. However, entirely green sources may experience less throughput than entirely blue sources sharing the same network resources.

In Section 4 we discuss a router implementation of the service.

3.2 Source Aspects

ABE is intended to provide a low delay service for multimedia, rate-adaptive applications. We discuss now a very simple scenario, in order to illustrate the ABE service.

Figure 1 shows a simple simulation where a multimedia source competes with n background sources for one bottleneck. The other sources are all blue. Assume the source has a required minimum rate R_0 in order to function properly, for a given loss pattern in the network. The rate R_0 is shown by the horizontal dashed line. Also assume that the source is able to forward-correct packet losses, as long as the minimum rate is achieved (see [24] for such an application example; note that this would not be needed if ECN was used). Then, the source is able to mark its packets green in the low load region shown. In this region, the source may also mark its packets blue, in which case it will receive more throughput. The choice between green or blue is left to the application. It depends on its utility function $u(R, D)$, for a given throughput R and end-to-end network delay D. In many situations today throughput (or equivalently packet drop) is the major impediment. However, once a minimum rate R_0 is achieved which provides enough intelligibility, delay becomes the major impediment. Thus, we assume that the utility function for our source satisfies (1) $u(R, D) = 0$ for $R < R_0$ and (2) $u(R, D)$ is a decreasing function of D for $R \geq R_0$. For this source, the optimal strategy is to be green in the low load region, blue in the moderate load region, and to disconnect when the load is too high. This example illustrates that ABE opens up a new region of operation for the best-effort network: in low load scenarios, a source may decide to obtain less throughput at the benefit of low delay. In a flat best effort network, a network without the ABE service, there is no such option. By refraining from sending at a higher rate, there is in general no impact on the queueing delay, because of external sources.

Note that this example is oversimplified. In general we expect more complex utility functions to be used. Note also that the detection of which region the source is currently operating in has to be made automatically by the source itself.

Unlike the multimedia source above, a source using TCP is probably more interested in its throughput and should thus mark all its packets blue. This follows immediately from the definition of the service. Intuitively, it is because ARQ protocols such as TCP are more sensitive to packet loss than to queueing delay, though queueing delay does have an impact. We will see in Figure 10 on page 28 and Figure 12 on page 28 that a blue source has a higher throughput than a green one, It will also be shown that the throughput given to the blue source is higher in the ABE case. The blue source is a TCP Reno source and the green source uses a transport protocol designed to represent TCP friendliness which is described in Appendix B.

More realistic sources would probably use a colour mixing strategy, where they would send some green packets and some blue. This is perfectly permissible and considered normal practice; in fact, apart from possibly policing TCP-friendliness, the network supporting ABE does not need to analyse individual flows. Source strategies would typically be performed at the application level as expected by Application Layer Framing (ALF).

3.3 Global Aspects

The service definition implies that introducing ABE has benefit for all. As illustrated in the previous section, if a source marks some packets green, then it must do so because it sees value in low delay. Otherwise, the source should mark the packet as blue, since a blue packet gets at least the same service as it would in a flat network.

The "single class" requirement (number 5 on the list of service requirements) makes sure that no policing of colour marking is required. We see in Figure 1, that in a network with a large number of blue sources, a green source receives little throughput. Conversely, in a network with many green sources, a blue source also receives little throughput. However, this is not because the green sources are green, but because there are many of them. If they were blue, the blue source would probably get less throughput. However, it does get more than if it were green. Lastly, a network where all sources are blue would probably behave the same as a flat best effort network (this is at least true with our implementation). Conversely, a network with only green sources behaves like a flat best effort network with smaller buffers.

3.4 Can we replace ABE with destination drop ?

Lastly, we conclude this section by discussing what might appear as an alternative to ABE. This alternative would consist in having the destination drop all packets that arrive too late, say after a transit deadline D_{obj}. Intuitively, this should waste network resources, since packets are dropped after being carried by the network. We show, at least on some simulation examples, and for our implementation, that, as long as D_{obj} is smaller than the tail of delays obtained for blue packets, it is better to send green packets. In other words, the throughput reduction due to being green is less than the throughput reduction due to dropping late packets. Of course, if the delay objective D_{obj} is large, then no packet is dropped at the destination, and it is better to mark the packets as blue.

Figure 2 shows the simulation results. The network consists of flows with long and short round-trip times, which each compete on the same best-effort network. The ABE implementation used is as described in Section 4.2. The curves show the number of useful packets received by the application as a function of the tolerated deadline D_{obj}. If D_{obj} is less than the end-to-end delay when there is no queueing no useful packets can be received by the application. After this point the amount of useful packets received by the application is larger while green until the delay tolerance is sufficiently high.

4 Router Support

In this section we give one possible implementation in routers of the ABE service. Since there may be many different implementations, we first discuss generic router requirements. We consider only a non-ECN router implementation in this document (extension to ECN is straightforward).

4.1 General Router Requirements

Requirements on a router implementing ABE are as follows:

1. Provide low, bounded delay to green packets;

2. Provide transparency to blue packets;

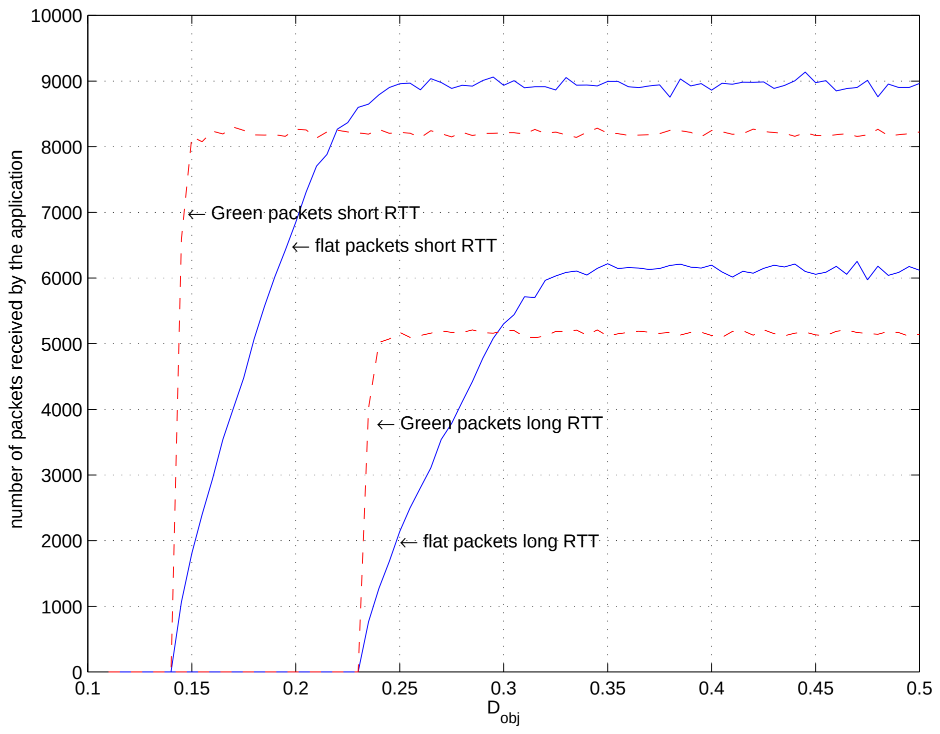


Figure 2: The amount of useful packets received by an application as a function of the end-to-end delay tolerance D_{obj}.

3. Drop green packets with higher probability than blue, such that an entirely green flow gets a lesser or equal throughput than if it were blue;

4. Minimise green packet dropping, subject to the above requirements.

The first three requirements directly map from service requirements 2, 4 and 5 in Section 3. The last requirement is because an implementation should try to minimise green packet loss, in order to make the service attractive.

As mentioned earlier, we do not consider in this document the task of enforcing TCP-friendliness. From service requirement 5 in Section 3, the relation between packet drop ratio and source rate should be enforced independently of packet marking.

4.2 Outline of Our Implementation

We have designed a router implementation of ABE, then implemented and simulated in ns [16]. Our design is outlined in Figure 3. We use two main modules: a Packet Admission Control (PAC) module and a scheduler. The PAC module manages the queue by dropping packets whenever necessary or appropriate, acceptance being biased in favour of blue packets. It must ensure that sufficient green packets are dropped in order to prevent blue flows from suffering. The PAC module must not accept green

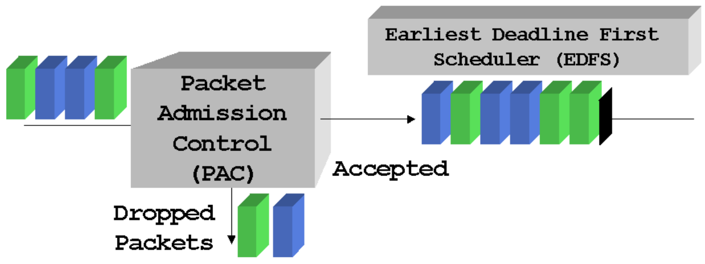

Earliest Deadline First Scheduler (EDFS)
Packet Admission Control (PAC)
Accepted
Dropped Packets

Figure 3: Overview of Implementation's Router Support: α and D

packets if they would experience delay greater than a specified bound, which is the delay guarantee given to green packets by the ABE router.

The PAC controls green packet dropping such that blue traffic receives as much throughput as if the network was flat best-effort, while trying to keep these green losses to a minimum. Let q_b and q_g be the drop ratio of blue and green traffic respectively. The PAC attempts to have the following relation hold,

$$q_g = \alpha q_b$$

where α is a controlled parameter, called the *drop bias*. The drop bias must first provide a drop disadvantage to green packets (a second purpose appears later in this section).

For simplicity we describe the implementation as though packets were of constant size rather than considering variable packet sizes. A byte-oriented description will appear at a later date.

The PAC uses Random Early Detection (RED) [14] to obtain an initial probability p of dropping a packet. RED is a congestion avoidance mechanism, such that when the average queue size exceeds a pre-set threshold, the router drops each arriving packet with a certain probability which is a function of the average queue size. The modification is as shown in Figure 4. The RED dropping probability p is calculated as before. If the packet is green, the probability of dropping p is multiplied by α.

RED actually calculates the drop probability in two parts (the second calculated from an input of the first probability and the amount of packets since the last drop) in order to achieve a more uniform distribution of packet losses. The increase in probability of loss for green packets is calculated after the second part.

The scheduling is Earliest Deadline First [15]. Each packet is assigned a finishing service time deadline, a tag, and the packet currently having the lowest value is served first (i.e. earliest deadline). Each green packet arriving is assigned a finishing service time deadline equal to its arrival time now. A blue packet is assigned a time equal to its arrival time plus the value of the *offset bound* D, namely now + D. The goal of the offset bound is to limit the delay penalty imposed on blue by our implementation. This scheduling is more advantageous than an absolute priority scheme which serves blue packets only if there are no green ones awaiting service. This can prevent service starvation for blue traffic. Also, even when service starvation does not occur with absolute priority, one would be forced to drop a large number of green packets in order

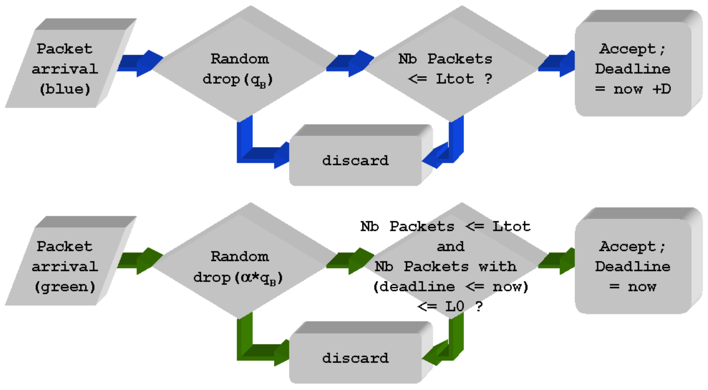


Figure 4: Summary of PAC Algorithm

to preserve transparency to blue.

The scheduling does impose a delay penalty on blue packets, which reduces throughput, and contradicts requirement 2. We could have designed an implementation which simply drops green packets when they cannot be served within the per-hop bound for green packets. However, the throughput for greens of such an implementation would be very little, thus contradicting requirement 4 (This is equivalent to our implementation if D was always 0). We compensate for the delay penalty imposed on blues by artificially reducing their drop rate, done by adjusting the value of the drop bias α. The determination and behaviour of the minimum α required to provide transparency for blue in a given network is analysed in Section 4.4. A sufficient α in the general case is described in Section 4.5.

A green packet is only provisionally accepted at this point. It then undergoes a second stage dropping acceptance decision. This is to ensure the low delay guarantee to green is provided. The packet is accepted only if the number of packets in the queue with a deadline less than the current time is less than some defined value L_0. This bounds the queueing delay for green packets to L_0/c where c is the capacity of the output link in packets per second. In this way, if there are only green packets arriving, the system acts like a flat best-effort network with smaller buffers. The queueing delay for blue packets is at most $L_{\text{tot}}/c + D$ where L_{tot} is the buffer size.

The PAC algorithm can now be summarised as follows:

```
For each packet arrival to output port:
  if buffer full
    drop packet
  else if Blue
    drop if queue size > $L_{\text{tot}}$
    if not dropped
      drop with random probability p
    if still not dropped
      deadline = now + D
      accept packet
```

```
else Green
  drop with random probability $p\alpha$
  if not dropped
    if number of packets with a deadline less than now > $L_0$
      drop packet
    else
      deadline = now
      accept packet
```

We now examine how our implementations satisfies the router requirements.

1. Low delay for green is enforced by the PAC; the per-hop delay bound for green is L_0/c where c is the capacity of the output link in packets per second and L_0 is the queue threshold.
2. Transparency to blue packets is obtained by the PAC ensuring that the delay penalty incurred by blue is compensated by a lower drop ratio. This is implemented by adjusting the offset bound D and the drop bias α.
3. Higher drop probability for greens is implemented with the PAC, using drop bias α.
4. Minimising the green drop ratio is performed by the control loop which adjusts the offset bound D and the drop bias α.

The queue size L_{tot} and threshold L_0 are fixed, whereas the offset bound D and the drop bias α are adjusted automatically, on a slow time scale, by means of the control loop described in Section 4.3. The question of what drop bias α would be sufficient to protect blue flows is addressed in Sections 4.4 and 4.5.

4.3 Control Loop Description

Let d_b and d_g denote the average queueing delay for blue and green packets respectively. Let τ_e be the lowest possible round-trip time of a source through the router, excluding queueing delays in the router. Hence $\tau_g = \tau_e + d_g$ and $\tau_b = \tau_e + d_b$. The drop bias α required to provide transparency to blue is described in Section 4.5, and is given by Equation (5), which is,

$$\alpha = \max\left(1, \left(\frac{\tau_b^3}{\tau_g(2\tau_g^2 - \tau_b^2)}\right)^2\right) \tag{1}$$

Thus, α is not a fixed value, since it is a function of the queueing delays d_b and d_g, which are variable. By responding to measured queueing delay we adapt to the variations in green and blue traffic load.

For a given α, the system will not drop exactly α times more green than blue packets due to the intrinsic randomness in the dropping mechanism, the approximate modelling used in Equation (5), variations in the input process and green drops which occur in the second stage. The actual ratio of green to blue drop ratios may be quite different from the α computed in Equation (1). This calls for a control loop.

We also control the value of the D, the delay priority given to green flows. The reasons we do so are two-fold. We would like to ensure there are few drops due to second stage drops (which is caused if D is too low), thus increasing the number of drops caused by the differential loss. Also, we seek to minimise green first stage losses (4th router requirement), and having too high a D results in too many green losses to protect blue.

Indeed, if D is low, then the delay boost b is low, which in turn makes the required α low since α is an increasing function of b. This reduces the number of first stage green drops, but can increase the number of second stage drops. A lot of second stage drops is a symptom of too low a D.

Conversely, if D is high, then b is high, and thus α is high. This may result in an unnecessarily high number of green losses. The higher the α the lower the average number of green packets in the queue. To minimise green losses, the number of green packets in the queue should be maximised as much as possible while avoiding second stage drops. We control D in such a way as to drive it into this operating region.

The actions required can be be thought of as follows. α should be increased if the delay boost increases or too many blue packets were dropped in the past. Conversely, α should be decreased if the delay boost decreases or too many green packets were dropped in the past. D should be decreased if the number of green packets in the queue has been small, and increased if there were many second stage drops.

These observations lead us to the following control laws for the parameters. Let q_g' denote the ratio of the number of first stage green packet losses to the total number of green arrivals. Let q_g'' denote the ratio of the number of second stage green packet losses to the total number of green arrivals. In this way, $q_g = q_g' + q_g''$.

To protect blue flows we must ensure that Equation (1) holds. This means that for a measured blue loss ratio q_b and delay boost b, the ideal ratio of green first stage losses to green arrivals $q_g^{'*}$ is such that we maintain the relationship,

$$\frac{q_g^{'*}}{q_b} \approx \max(1, \left(\frac{\tau_b^3}{\tau_g(2\tau_g^2 - \tau_b^2)}\right)^2).$$

So we would like to drive the actual value q_g' to be equal to $q_g^{'*}$. This motivates the following update law for α,

$$\log \alpha \leftarrow \log \alpha + K_1(\log q_g^{'*} - \log q_g')$$

for gain parameter $K_1 > 0$ where

$$q_g^{'*} = q_b \max\left(1, \left(\frac{\tau_b^3}{\tau_g(2\tau_g^2 - \tau_b^2)}\right)^2\right). \tag{2}$$

We now discuss the controlling mechanism for the offset bound D. Let w_l be the measured average number of packets with a deadline less than *now* over the time interval T_1. In the control law, a safety margin λ is used to avoid waiting for second stage losses before increasing D, λ being a measure of the tolerance of how close we allow the green buffer to being full. Thus we consider $\lambda L_0 - w_l$ a measure as how "far" we are from a full queue, or, if negative, how much we have exceeded it by. In controlling D we would like to ensure $q_g'' \to 0$ and $w_l \to \lambda L_0$. We arrive at the control law,

$$D \leftarrow D + K_2(\frac{w_l}{\lambda L_0} - 1) + K_3 q_g''$$

for gain parameters $K_2, K_3 > 0$ and $\lambda \in (0, 1]$. D is restricted to lie in the range $[0, D_{\max}]$, where $D_{\max}$ is the highest tolerable offset bound, given by the value beyond which the system behaves effectively as it gave absolute priority to greens. From our experiments, it was found that λ should lie between about 0.4 and 0.6.

We can now describe how this is actualised in the implementation. α and D are updated after each time interval T_1, for some T_1. q_b^m, $q_g^{'m}$, $q_g^{''m}$, d_g^m and d_b^m are the measured values, in this time period T_1, for ratio of blue losses to number of green arrivals, the ratio of first stage green losses to number of green arrivals, the ratio of second stage green losses to number of green arrivals, the average green queueing delay, and the average blue queueing delay respectively.

In order not to be too reactive to the instantaneous values of q_b^m and $q_g^{'m}$, we smooth using an exponentially weighted moving average (EWMA) filter with parameter $\gamma_1 \in [0, 1]$. d_b^m and d_g^m are also smoothed by an EMWA filter with parameter $\gamma_2 \in [0, 1]$, and $q_g^{''m}$ is smoothed by an EWMA filter with parameter $\gamma_3 \in [0, 1]$. b^m and $q_g^{''m}$ are filtered differently to the loss ratios in order to react to higher frequency oscillations in these signals. We use the measured value of w_l without any filtering.

We now summarise the control action for the nth iteration (i.e. at time nT_1):

$$\begin{aligned}
q_b(n) &= \gamma_1 q_b(n-1) + (1-\gamma_1) q_b^m(n) \\
q_g'(n) &= \gamma_1 q_g'(n-1) + (1-\gamma_1) q_g^{'m}(n) \\
q_g''(n) &= \gamma_3 q_g''(n-1) + (1-\gamma_3) q_g^{''m}(n) \\
d_b(n) &= \gamma_2 d_b(n-1) + (1-\gamma_2) d_b^m(n) \\
d_g(n) &= \gamma_2 d_g(n-1) + (1-\gamma_2) d_g^m(n) \\
\tau_b(n) &= \tau_e + d_b(n) \\
\tau_g(n) &= \tau_e + d_g(n) \\
q_g^{'*}(n) &= q_b(n) \max\left(1, \left(\frac{\tau_b(n)^3}{\tau_g(n)(2\tau_g(n)^2 - \tau_b(n)^2)}\right)^2\right) \\
\alpha(n) &= \alpha(n-1)\left(\frac{q_g^{'*}(n)}{q_g'(n)}\right)^{K_1} \\
D(n) &= D(n-1) + K_2(\frac{w_l(n)}{\lambda L_0} - 1) + K_3 q_g''(n)
\end{aligned}$$

To start the system, some initial values are required. The choice of these values is not vital as the system will settle to operating values. $q_b(0)$ is chosen to be an initial blue loss ratio based on observation e.g. $q_b(0) = 0.01$. $q'_g(0)$ is calculated from the initial q_b according to Formula (2) and $q''_g(0) = 0$. $D(0)$ can be chosen to be something reasonable e.g. $D(0) = 0.1$. The queueing delay measurements can be chosen to be $d_g(0) = 0$ and $d_b(0) = D(0)/2$.

We are currently investigating methods for the determination of appropriate parameter settings for the control for a variety of conditions. Optimal settings are not necessary for the service to work, as it is robust enough. Examples of settings used are shown in the simulations section, Section 5.2.

4.4 Analysis of Minimum Drop Bias α

We call b ("delay boost"), for an output queue to a link, the difference between average queueing delay for blue packets and for green packets, namely $d_b - d_g$, for average blue queueing delay d_b, and average green queueing delay where d_g.

In this section we analyse the drop bias α requirements for a given network for a given delay boost b.

We first describe a method to determine α for a given arbitrary network composed of a deterministic number of blue and green sources and links. We then show, under an example but reasonably generic topology, that (i) the minimum α required is dependent on the relative number of green and blue sources, (ii) the determination of the minimum α for all flows is equivalent to choosing the highest α needed for sources with just one bottleneck and (iii) the required α increases approximately exponentially as the boost b increases linearly.

The establishment of (ii) was shown on the example and not in general. This result also held under simulation tests, but we have yet to prove it definitively. Result (iii) has significance in the engineering of a router's delay advantage to green.

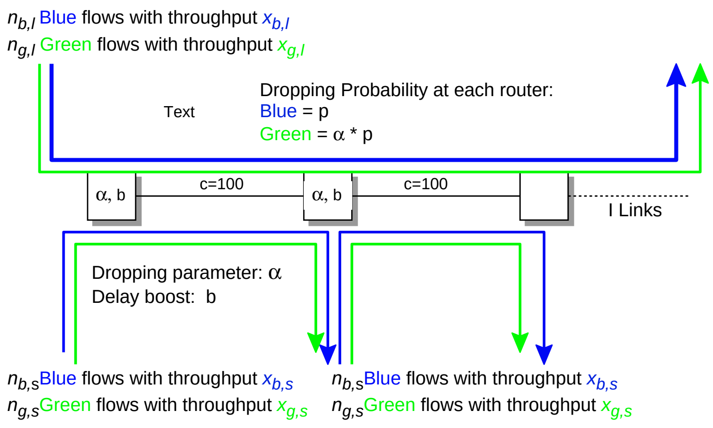
Consider the following abstraction of a general network. It contains L links, a set of blue sources B and a set of green sources G. L_l is the set of all sources who use link l. A source s is on link l if $s \in L_l$. Assume each source s has a long term rate of x_s and a fixed round-trip time τ_s. The capacity of each link l is c_l. Whenever the probability of not accepting a blue packet on a link l is p, the probability of rejecting a green packet is $\alpha_l p$.

Assume a source s responds in an additive increase, multiplicative decrease way i.e. in the event of no loss in a RTT τ_s it increases its rate by r_s, and responds to loss detection by decreasing its rate to be $\eta \in (0,1)$ times the previous rate. This behaviour is a simplification of the congestion response of a TCP flow.

We need to use this method of global modelling, rather than using known TCP equations [6, 7, 3, 4, 1]. Those relate loss rate to throughput for an indiviual source, with no concept of the number of bottlenecks a flow experiences.

Suppose $\alpha = \alpha_l$ for all $l = 1 \ldots L$, namely the drop bias is the same on every link. Then it can be shown that the distribution of long term rates x_s can be determined by the maximisation of the utility function,

$$\sum_{s \in B} \frac{1}{\tau_s} \log \frac{x_s}{r_s + \eta x_s} + \frac{1}{\alpha} \sum_{s \in G} \frac{1}{\tau_s} \log \frac{x_s}{r_s + \eta x_s} \tag{3}$$



$n_{b,l}$ Blue flows with throughput $x_{b,l}$
$n_{g,l}$ Green flows with throughput $x_{g,l}$
Text
Dropping Probability at each router:
Blue = p
Green = α * p
α, b
c=100
α, b
c=100
l Links
Dropping parameter: α
Delay boost: b
$n_{b,s}$Blue flows with throughput $x_{b,s}$
$n_{g,s}$Green flows with throughput $x_{g,s}$
$n_{b,s}$Blue flows with throughput $x_{b,s}$
$n_{g,s}$Green flows with throughput $x_{g,s}$

Figure 5: Parking Lot Scenario used in analysis of α.

subject to the link constraints,

$$\sum_{s \in L_l} x_s \leq c_l \quad l = 1 \ldots L. \tag{4}$$

The derivation, which relies on losses being rare, is an easy consequence of the results in [13] and [5], and is shown in Appendix C. The restriction that $\alpha = \alpha_l$ for all $l = 1 \ldots L$ is of course not true in general. Determining the distribution in this case is not as simple, and not necessary for the network which we analyse here. When $\alpha = 1$, the maximisation results in the flat best-effort distribution of rates.

For a blue source s and a delay boost b, the minimum drop bias α_s should be the smallest value such that:

1. Its throughput with ABE is no less than if the network were flat best-effort. This ensures router requirement 2 (transparency to blue) is satisfied. The minimum in this case, α'_s, is given when both throughputs are the same i.e. it is the solution to $x_s(b, \alpha'_s) = x^f_s$ where $x_s(\alpha'_s)$ is the throughput from an ABE network with drop bias α_s and x^f_s is the throughput the flow receives in a flat best-effort network.

2. If the blue source were to become green it would not receive more throughput (router requirement 3). The minimum in this case, α''_s, is given by the solution to $x_s(\alpha''_s) = x'_s(\alpha''_s)$ where $x'_s(\alpha''_s)$ is the throughput from the ABE network if the source was green rather than blue.

The minimum drop bias for a source s is thus given by $\alpha_s = \max(\alpha'_s, \alpha''_s)$. Note that in general $\alpha'_s \geq \alpha''_s$ may not hold, a point illustrated on the parking lot network.

For a given network, the α that protects all blue flows at minimum cost to green flows is given by $\alpha = \max_{s \in B} \alpha_s$. Dropping with this drop bias α ensures that all blue flows receive at least as much as they would if it were a flat best-effort network, while turning green cannot result in a throughput advantage.

We now look at the parking lot scenario as depicted in Figure 5. There are I links each of capacity c. There are $n_{b,l}$ blue flows with throughput $x_{b,l}$ and $n_{g,l}$ green flows with throughput $x_{g,l}$ which traverse all links. These are the long flows. On each link there are $n_{b,s}$ blue flows with throughput $x_{b,s}$ and $n_{g,s}$ green flows with throughput $x_{g,s}$ which traverse just this link. These are the short flows.

When this is an ABE network, the long and short green flows have a round-trip time (RTT) of $\tau_{g,l}$ and $\tau_{g,s}$ respectively. The long and short blue flows have a round-trip time of $\tau_{b,l}$ and $\tau_{b,s}$ respectively. Let the average extra queueing delay at each link for a blue flow be the delay boost b. Thus, $\tau_{b,s} = \tau_{g,s} + b$ and $\tau_{b,l} = \tau_{g,l} + Ib$. The assumption that the drop bias α is the same on each link is valid given that the load on each link is the same. For the same reason, the delay boost b can also be considered to be the same on each link. The α sufficient to ensure blues do not suffer is given by $\max(\alpha_l, \alpha_s)$, where α_l and α_s are the minimum drop biases for long and short flows respectively.

When this network is flat best-effort, $\alpha = 1$. The round-trip times of the flows are unknown, unless queueing analysis is performed. To bypass this difficulty we consider all possible values to determine a worse-case α.

Let $\eta = 0.5$. The additive increase parameters for a long and short blue flow are given by $r_{b,l}$ and $r_{b,s}$ respectively, and $r_{g,s}$ and $r_{g,s}$ for a long and short green flow. Let $r_{b,l} = 1/\tau_{b,l}$, $r_{b,s} = 1/\tau_{b,s}$, $r_{g,l} = 1/\tau_{g,l}$ and $r_{g,s} = 1/\tau_{g,s}$. This means the sources react like a TCP source by decreasing the rate by two in the event of a lost packet, and increasing their rate accordingly in the event of no loss.

The throughput for the long blue and green flows is given by $x_{b,l}$ and $x_{g,l}$, and by $x_{b,s}$ and $x_{g,s}$ for the short blue and green flows. By maximisation procedures, we can determine from Equations (3) and (4) that the distribution is given by the solution to the following equations:

$$\begin{aligned}
\tau_{b,l}x_{b,l}(2+\tau_{b,l}x_{b,l})I &= \alpha x_{g,s}\tau_{g,s}(2+x_{g,s}\tau_{g,s}) \\
x_{g,l} &= \frac{1}{\alpha\tau_{g,l}}\left(-\alpha+\sqrt{\alpha(\alpha+x_{b,l}\tau_{b,l}(2+x_{b,l}\tau_{b,l}))}\right) \\
x_{b,s} &= \frac{1}{\tau_{b,s}}\left(-1+\sqrt{1+Ix_{b,l}\tau_{b,l}(2+x_{b,l}\tau_{b,l})}\right) \\
x_{g,s} &= \frac{1}{n_{g,s}}\left(c-n_{b,l}x_{b,l}-n_{g,l}x_{g,l}-n_{b,s}x_{b,s}\right).
\end{aligned}$$

The results that follow are determined by numerical solutions to these. As an example, consider that $I = 2$, the capacity of each link $c = 100$, $\tau_{g,l} = 0.2$ and $\tau_{g,s} = 0.1$. We examine first the case of an equal number of green and blue flows, $n_{b,l} = n_{g,l} = 5$, $n_{b,s} = n_{g,s} = 3$. First let the boost be fixed, $b = 0.03$. Figure 6 shows the throughput of each source. The short flows get more throughput than the long ones due to their shorter round-trip times and less number of bottlenecks. In the long flow case, blue flows receive more than green flows when $\alpha > \alpha_l'' \approx 1.40$ but do not receive at least as much they would if it were a flat best-effort network until $\alpha \geq \alpha_l' \approx 1.44$. In the short flow case, blue flows receive as much as they would if it were a flat best-effort network when $\alpha \geq \alpha_s' \approx 1.42$, but do not receive more than green

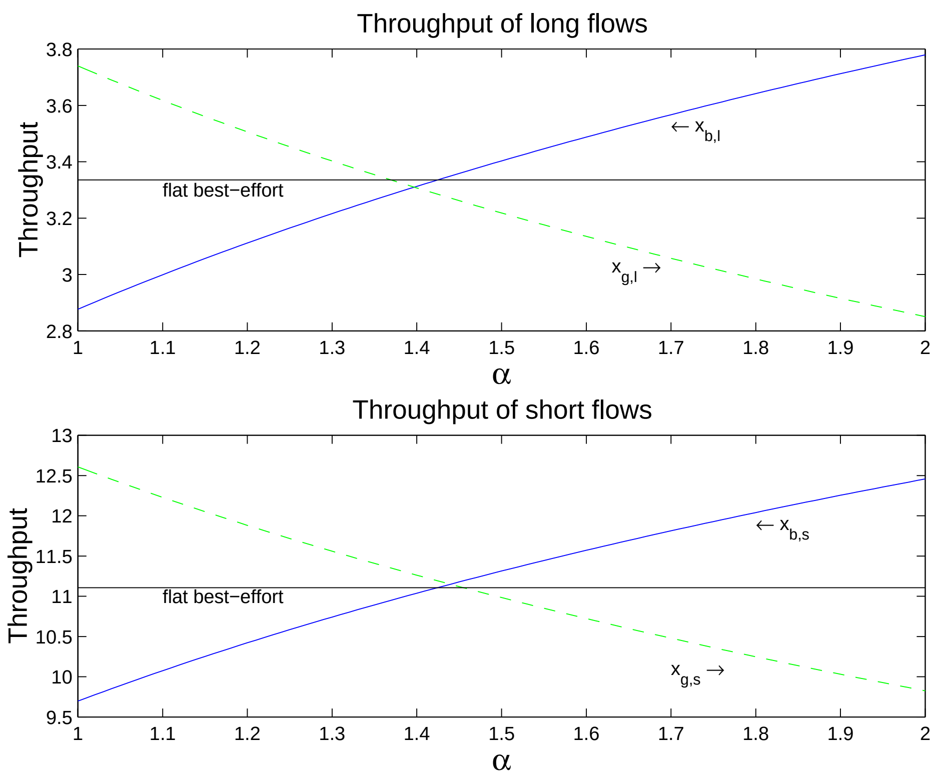


Figure 6: The throughput of long and short flows as a function of the drop bias α. Here, $n_{b,l} = n_{g,l} = 5$, $n_{b,s} = n_{g,s} = 3$, and $b = 0.03$.

flows until $\alpha > \alpha''_s \approx 1.46$. When the short blue flow receives as much as in the flat case, the greens still get more because the long green flows lose sufficient throughput to the benefit of the short green flows. Overall, $\alpha = 1.46$ is sufficient to ensure both blue flow types receive enough throughput.

In Figure 7 we let the delay boost b vary, and measure the required α_s and α_l. We notice two things. Firstly, satisfying the short blue flows requires a higher α than the long blue flows. Secondly, as b increases, we must increase α exponentially to compensate.

Suppose we increase the number of short and long blue and green flows alternatively while keeping the others fixed as in Figure 8. Note that the delay boost is fixed at $b = 0.03$ here. In reality, when we change the number of flows, we also change b. To obtain the relationship between b and the number of flows, a queueing analysis would be needed, which we leave for future work. The number of other flows directly influences the minimum required drop bias. In particular, in this case we see that increasing the number of short blue flows increases the size of α needed while in all other cases it reduces the α required.

The results in this section apply to long lived flows. However, it is known that shorter flows will suffer more from loss than long term ones. Therefore the worst case

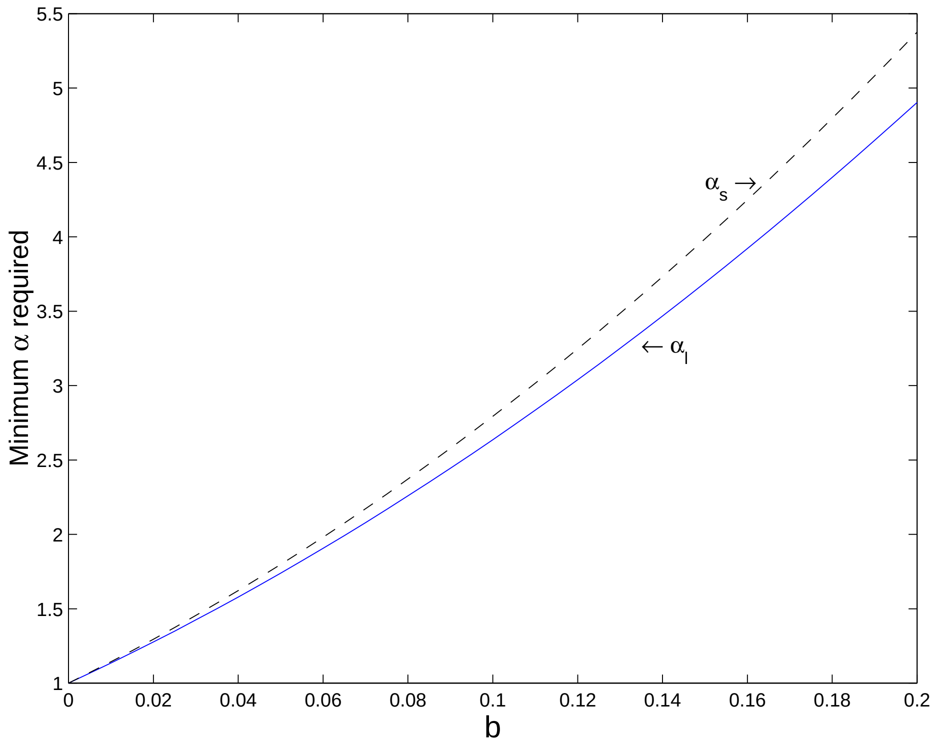


Figure 7: The minimum α for the long and short flows as a function of the delay boost b. $n_{b,l} = n_{g,l} = 5$, $n_{b,s} = 3$, $n_{g,s} = 1$, and $b = 0.03$.

drop bias α needed is obtained by considering all flows as being long lived.

Also, we used TCP-Reno for our definition of TCP-friendliness. It is known that TCP-Reno has a strong negative bias against flows with long round trip times, due to the fact that the window is increased with a period roughly equal to the round trip time. This appears in the modelling above by the fact that, for any given source s, the increase parameter r_s is proportional to $\frac{1}{\tau_s}$. Variants of TCP-Reno have been proposed which correct this bias to give the same value of r_s to all sources. If such a variant were to become the standard for TCP-friendliness, then we would need to correct our analysis accordingly. Note that even with this correction, sources which use many hops usually still get less throughput, because maximising a utility function leads to penalising those sources which use more resources.

4.5 Transparency for blue

In the previous section, we knew the network conditions in order to determine the lowest possible drop bias α. In practice we need to choose a safe α at a router which works under general conditions with an unknown number of blue and green sources, unknown round-trip times, and unknown conditions these flows experience elsewhere. In addition, the previous modelling relied on losses being rare, which may not be the

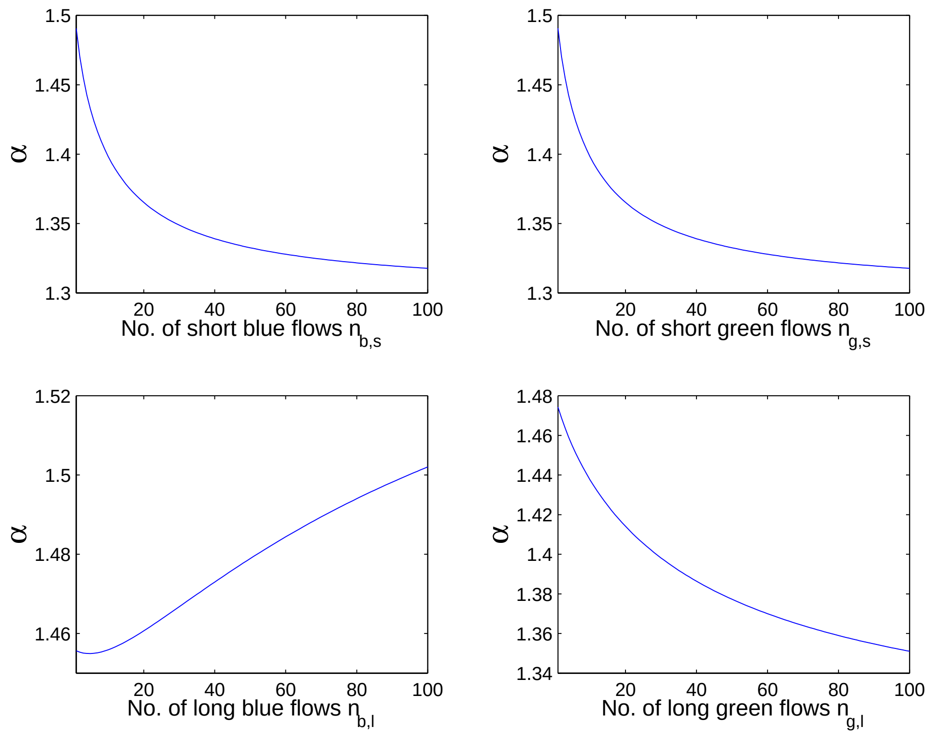


Figure 8: α required when we keep other amount of flows fixed but vary one amount of flows. For fixed $b = 0.03$.

case for green packets.

We derived a sufficient drop bias α, which works under general conditions. As in Section 4.3, τ_e is the reasonable possible round-trip time of a source through the router, excluding queueing delays. As before, $\tau_g = \tau_e + d_g$ and $\tau_b = \tau_e + d_b$, where d_b and d_g are the average queueing delays for blue and green packets respectively. If the router has only flows with one bottleneck then choosing

$$\alpha = \max\left(1, \left(\frac{\tau_b^3}{\tau_g(2\tau_g^2 - \tau_b^2)}\right)^2\right) \tag{5}$$

is sufficient, assuming the modelling holds, to ensure blue flows would not receive a better throughput in a flat best-effort network than with ABE. The derivation is in Appendix A. This is an extremely safe choice of α designed to work in the absence of knowledge of the number of green and blue flows. The safety margin is bigger than the simplified modelling used in deriving this result.

Optimisation of this α is not required for ABE to work. However, lower values would reduce the green loss requirements. This would be possible if we had (approximate) knowledge of the ratio of the number of blue to green flows or queueing analysis to determine the RTT in the hypothetical flat best-effort network, as opposed to using the worst case as we do now. This is the subject of ongoing work.

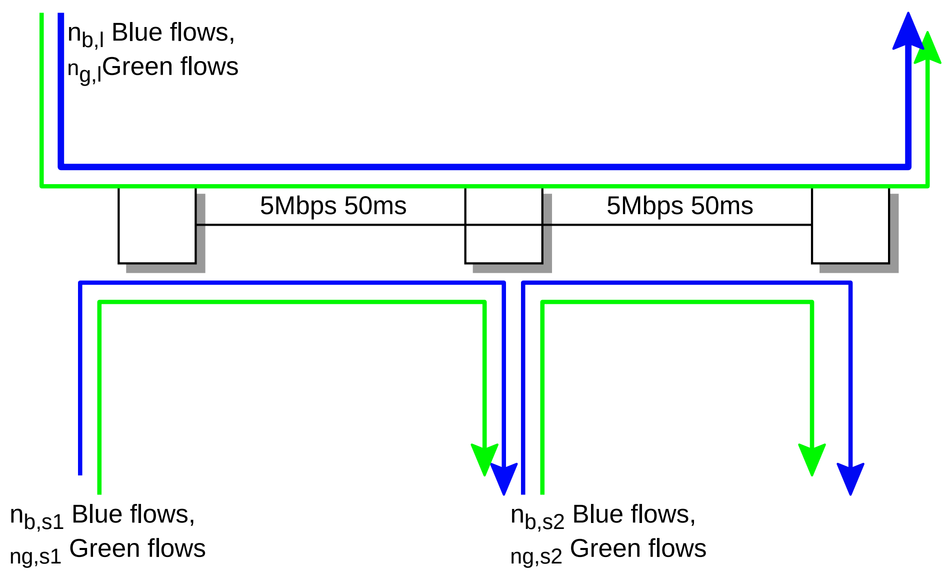


Figure 9: Simulation topology.

The router considers, for the purpose of determining α that all flows experience just it as a bottleneck. We show through simulation in Section 5 and through analysis in Section 4.4 that provides a higher and thus a sufficient α than if α was (able to be) chosen in conjunction with more global state. However, we have yet to prove this completely.

τ_e is a parameter specified in the router. The higher one can choose τ_e, the less green losses there will be. If it is chosen too high, there may be circumstances in which blue flows may not receive as much as they would in a flat best-effort network.

5 Simulations of the Implementation

In this section, we show that the implementation satisfies the router requirements. From the description in the previous section, requirement 3 is automatically satisfied. We now check that green traffic is provided with a low bounded delay (requirement 2) while still receiving acceptable throughput (requirement 4) and blue traffic receives at least as much throughput as it would from a flat best-effort network (requirement 4). These results are in Section 5.2.

5.1 Simulations Description

The general test topology, shown in Figure 9, consists of: $n_{b,l}$ blue and $n_{g,l}$ green sources which traverse both links, the long flows; n_{b,s_1} blue and n_{g,s_1} green sources which traverse the first link, the type 1 short flows; and n_{b,s_2} blue and n_{g,s_2} green sources which traverse the second link, the type 2 short flows. The links from the sources and sinks to the bottleneck links were 10Mbps and had propagation delays of 20ms.

One bit in the header of a packet specifies whether the packet is green or blue. The blue source is a TCP Reno source which has always a packet to send. The green source uses the TCP friendly type algorithm described in Appendix B. It is a simple one for simulation purposes, used to represent the fact that green traffic will most likely be real-time in nature. The TCP friendly protocol is basic, and its success in providing said property is approximate but sufficient for our purposes. More sophisticated rate-based TCP friendly unicast protocols which explore their algorithms ability to provide TCP friendliness effectively are described and evaluated in [12] and [11].

Each router buffer size was 60 packets (i.e. $L_{tot} = 60$) and the apparent buffer size to green was 10 packets (i.e. $L_0 = 10$). The set of RED parameters used, were as follows: The initial threshold $min_{th} = 10$, maximum threshold $max_{th} = 30$, the upper bound on the (blue) marking probability $max_p = 0.15$, and average queue weight $w_q = 0.002$. Throughout, $\gamma_1 = 0.8$, $\gamma_2 = 0.4$, and $\gamma_3 = 0.4$, and the control loop updates α and D every $T_1 = 0.5s$. The gain parameters were $K_1 = 1.1$, $K_2 = 0.02$, $K_3 = 2.0$ and $\lambda = 0.4$. The initial values were $q_b(0) = 0.01$, $w_l(0) = 4$, and $D(0) = 0.1$, $d_b = D(0)/2$. The round-trip time τ_e, used in determining α, was 0.18, which is based on the propagation delays for the short flows.

Given the randomness in dropping, average values were obtained after four simulation runs and confidence interval results obtained. Throughput and delay measurements for the first 30s of simulation time are not measured as we allow the control mechanism to warm up to reflect more realistic operation.

5.2 Service Simulations Results

5.2.1 Equal numbers of blue and green flows

We first look at the case when there are an equal number of blue and green flows for each flow type, explicitly $n_{b,l} = n_{b,s_1} = n_{b,s_2} = 5$ and $n_{g,l} = n_{g,s_1} = n_{g,s_2} = 5$. Figure 10 shows the average number of packets received by each blue and green connection at each time t. The average shown at each time t is the average obtained over 4 simulation results for that time t. For clarity the confidence intervals are not shown. The worst-case interval for 95% confidence seen was small, 80 packets. Figure 11 shows the end-to-end delay distributions received for green packets of each type under ABE and flat best-effort. We show only the type 1 short flows delay distribution, since the type 2 short flow distribution is practically identical.

All ABE router requirements were satisfied. Green packet delay is small and bounded (requirement 1), receiving the benefit of low bounded delay. The blue flows receive at least as much as they would as with flat best-effort (requirement 2). They actually receive more, thus receiving benefit from the use of ABE. They also receive more throughput than if they became green (requirement 3). The green sources receive reasonable throughput (thus providing requirement 4).

Figure 10 provides some additional observations. The long blue flows receive less throughput than the short blue flows due to their longer round-trip time and multiple bottlenecks. A similar observation hold for the green flows. The long blue flows receive proportionally more throughput benefit from ABE than the short blue flows do, which is expected. The minimum round trip-time value used in the calculation of the drop bias α was the short flows' propagation delay. Thus, the long flows benefitted from being considered, for the purposes of protecting it by dropping green packets, as having a lower round-trip time. Also, because of the multiple bottlenecks, the long green flows' losses receive a proportionally lower share.

The short green flows actually receive higher throughput in the ABE case than in the flat best-effort case. This does not violate the requirements of ABE; in fact, they would receive more if blue. The higher throughput with ABE results from them benefiting from the reduced long green throughput, a point also seen in the analysis in Section 4.4.

5.2.2 Unequal numbers of blue and green flows

We now look at the case where there is the same number of long blue and green flows, but there are more green than blue flows of type 1, and more blue than green flows of type 2 - The explicit values are $n_{b,l} = n_{g,l} = 5$, $n_{b,s_1} = 1$, $n_{g,s_1} = 4$, $n_{b,s_2} = 4$, and $n_{g,s_2} = 1$. Figure 12 shows how the average throughput received by each type. For brevity, we omit the end-to-end delay distribution which is similar to the previous case. Again all ABE router requirements are satisfied.

We observe that, when using ABE the type 2 short blue flow get a significantly higher increase in throughput than the type 1 short blue sources. This is expected behaviour. Both compete with the same number of flows on its bottleneck link. However, the type 2 short blue flow has more since it competes with less blue flows and more green flows. If these greens were to become blue then the type 2 short blue flows throughput would reduce.

6 Conclusions

We have described ABE, a new service which enables best-effort traffic to experience a low delay, at the expense of possibly more throughput. ABE is targeted at supporting rate-adaptive multimedia applications, with no concept of reservation or signalling and while retaining the spirit of a flat rate network. The service choice of green or blue is self-policing since the user/application will be coaxed into choosing one or the other or indeed a mixture of both, based on its traffic profile objectives. ABE allows a collection of rate-adaptive multimedia applications to drive the network into a region of moderately high load and low delay. It also allows such an application to trade reduced throughput for low delay, thus in some cases increasing its utility.

It should be stressed that ABE is a new service in its own right and not a substitute for reservation or priority services. In contrast, with ABE, both delay sensitive (green) and throughput sensitive (blue) traffic share the same resources, and high load in any of the two pools affects the other.

We have defined the ABE service and presented a router implementation. Initial simulations tend to indicate that it provides the ABE service. Work is ongoing to refine and validate the implementation.

We have presented ABE with the assumption that negative feedback is based on packet drop. It would be relatively straightforward to map our implementation to a network supporting explicit congestion notification.

The implementation of a multimedia adaptive application that would exploit the new degree of freedom offered by ABE is ongoing work, outside the scope of this document. Ongoing work is also addressing: the optimal value for the maximum green delay in a router (or equivalently, the L_0 parameter of our router implementation); a router implementation that would combine policing of TCP-friendliness with support for ABE (based on per-flow queueing).

Acknowledgement

Thanks to Felix Farkas for implementing the simulations in Section 3.4.

Appendix A: Derivation of a sufficient drop bias

Let n blue flows and m green flows share a link of capacity c. They have all just this link as a bottleneck i.e. they experience all congestion losses at this link.

Let all green flows have a round-trip time τ_g, and blue flows have a round-trip time $\tau_b = \tau_g + b$, where b is the delay boost provided by the preferential scheduling of green packets. In the flat best-effort network all flows have a round-trip time τ_f, where $\tau_f \in [\tau_b, \tau_g]$.

Let x_b and x_g be the throughput received by a blue and green flow respectively in the ABE environment. In the flat best-effort scenario they would have throughput x_f.

Assuming the established loss-throughput formula [6, 7, 3, 4, 1] holds,

$$x_b = \frac{C}{\tau_b\sqrt{q_b}}, \;\; x_g = \frac{C}{\tau_g\sqrt{q_g}} \;\text{ and }\; x_b = \frac{C}{\tau_f\sqrt{q_f}}$$

where q_b, q_g, q_g are the blue, green, and flat loss ratios respectively, and C is a constant.

Assume we can characterise the dropping at the router by a nondecreasing convex function $\phi(x)$ where $\phi(0) = 0$ and $\phi(x) = 1$ for $x \geq c$, where c is the output link capacity. The ABE PAC drops blue and green packets with probability $\phi(x)$ and $\alpha\phi(x)$ respectively, for given load x. The flat best-effort router drops all packets with probability $\phi(x)$.

Then $q_b = \phi(nx_b + mx_g)$, $q_g = \alpha q_b$, and $q_f = \phi((n+m)x_f)$. For simplicity of notation consider that we would like to determine a sufficient β where $\alpha = \beta^2$. We can then deduce that,

$$x_b = \frac{C}{\tau_b\sqrt{\phi((n+\gamma m)x_b)}} \;\text{ and }\; x_f = \frac{C}{\tau_f\sqrt{\phi((n+m)x_f)}}$$

where $\gamma = \frac{\tau_b}{\tau_g \beta}$. This yields that,

$$x_b^2 \tau_b^2 \phi((n + \gamma m)x_b) = x_f^2 \tau_f^2 \phi((n + m)x_f). \tag{6}$$

Now we propose that a sufficient γ for the blue flows to get at least as much throughput as in the flat best-effort case i.e. $x_b \geq x_f$, is

$$\gamma = \frac{1}{m} \left(\frac{\tau_f^2}{\tau_b^2}(n + m) - n \right). \tag{7}$$

The proof is as follows. From Equation (7),

$$\phi((n + \gamma m)x_b) = \phi(\frac{\tau_f^2}{\tau_b^2}(n + m)x_b). \tag{8}$$

Since $\tau_f \leq \tau_b$, $\phi(0) = 0$ and ϕ is convex,

$$\phi(\frac{\tau_f^2}{\tau_b^2}(n + m)x_b) \leq \frac{\tau_f^2}{\tau_b^2}\phi((n + m)x_b) \tag{9}$$

Equations (6), (8) and (9) then yield that,

$$x_f^2 \phi((n + m)x_f) \leq x_b^2 \phi((n + m)x_b)$$

while implies $x_f \leq x_b$ since ϕ is nondecreasing.

Equation (7) translates as a sufficient β being,

$$\beta(a) = \frac{\tau_b^3}{\tau_g(\tau_f^2(1 + a) - a\tau_b^2)} \tag{10}$$

where $a = \frac{m}{n}$. This depends on the RTT in a flat best-effort network $\tau_f \in [\tau_g, \tau_b]$.

In the absence of knowledge of this, all we can say is that for large values of a, $\tau_f \approx \tau_g$ and Equation (10) becomes,

$$\beta = \frac{\tau_b^3}{\tau_g(2\tau_g^2 - \tau_b^2)}. \tag{11}$$

Conversely, for small values of a, $\tau_f \approx \tau_b$ and Equation (10) becomes,

$$\frac{\tau_b}{\tau_g}.$$

In practice, we found that Equation (11) provides a value that works well for large range of a.

Appendix B: A TCP-Friendly protocol

Green traffic would typically be rate rather than window based, and not necessarily concerned with loss recovery given its real-time nature. As such, a protocol was designed which approximates TCP friendliness in a rate based unicast context. The goal was not to define a sophisticated rate based scheme which is proven to be TCP-friendly. Rather it was to have a simple one for simulation purposes. More sophisticated rate-based TCP friendly unicast protocols which explore their algorithms ability to provide TCP friendliness effectively are described and evaluated in [12] and [11].

Each packet contains a sequence number. The receiver acknowledges every packet it receives. A source starts by sending one packet. It is then subjected to additive increase, multiplicative decrease as follows. In the event of a loss, the rate, measured in packets per second, is reduced by half to $r/2$. No re-transmission is attempted. If in a given round-trip time, τ, no loss is detected, the source increases its rate by $1/\tau$.

The round trip time is estimated by the same algorithm as TCP. Losses are detected by two means. Each packet sent is assigned a timeout and a loss is deduced if one of these timeouts expire before receipt of an acknowledgement for that packet.

In the spirit of the Fast Retransmit algorithm in TCP, losses are also deduced when we receive two acknowledgements and a gap remains in the sequence number acknowledgement space. A loss (or losses) causes a restart of all timers for packets up to the most recently acknowledged, and monitoring is restricted to packets after this value.

Appendix C: ABE Fairness Distribution

Let $\hat{\alpha}_s = \alpha$ if $s \in G$ (source is green) and $\hat{\alpha}_s = 1$ if $s \in B$ (source is blue). Equation (22) on page 4 of [5] changes to

$$\dot{x}_s = \frac{r_s}{t_s} - x_s(r_s + \eta x_s)\hat{\alpha}_s \sum_{l \in L, s \in L_l} g_l(f_l)$$

since the dropping probability function for a green source is $\alpha g_l(f_l)$. From this, it follows that

$$J_A^h(\vec{x}) = \sum_{s \in B} \frac{1}{\tau_s} \log \frac{x_s}{r_s + \eta x_s} + \frac{1}{\alpha} \sum_{s \in G} \frac{1}{\tau_s} \log \frac{x_s}{r_s + \eta x_s} - G(\vec{x}).$$

The rest of the derivation is the same.

References

[1] N. Cardwell, S. Savage, T. Anderson. Modeling the Performance of Short TCP Connections. University of Washington. Oct, 1998.

[2] J. Crowcroft. All you need is just 1 bit. Keynote Presentation. *IFIP Conf. on Protocols for High Speed Networks*, Oct. 1996.

[3] S. Floyd. Connections with multiple congested gateways in packet switched networks, part 1: one way traffic. *ACM Computer Communication Review*, 22(5):30–47, October 1991.

[4] J. Padhye, V. Firoiu, D. Towsley and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. *Proceedings of SIGCOMM'98.*

[5] M. Vojnovic, J. Y. Le Boudec, C. Boutremans. Global Fairness of additive-increase and multiplicative-decrease with heterogeneous round-trip times. EPFL/DSC Technical report No. SSC/1999/024, July 1999.

[6] T. V. Lakshman and U. Madhow. The performance of TCP for networks with high bandwidth delay products and random loss. *IEEE/ACM Trans on Networking*, 5(3):336–350, June 1997.

[7] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behaviour of the TCP congestion avoidance algorithm. *Computer Communication Review*, 3, July 1997.

[8] TCP friendly web site. http://www.psc.edu/networking/tcp_friendly.html

[9] Floyd, S., and Fall, K. Promoting the Use of End-to-End Congestion Control in the Internet.

[10] B. Suter, T.V. Lakshman, D. Stiliadis, A. Choudhury. Design Considerations for Supporting TCP with Per-flow Queueing. *Proceedings of IEEE INFOCOM 98.*

[11] Reza Rejaie, Mark Handley, and Deborah Estrin. RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet. *Proceedings of IEEE INFOCOMM 99.*

[12] Jitendra Padhye, Jim Kurose, Don Towsley and Rajeev Koodli. A TCP-Friendly Rate Adjustment Protocol for Continuous Media Flows over Best Effort Networks. *UMass-CMPSCI Technical Report TR 98-04*, October 1998.

[13] Paul Hurley, Jean-Yves Le Boudec, Patrick Thiran. A Note On the Fairness of Additive Increase and Multiplicative Decrease. *International Teletraffic Congress (ITC-16)*, June 1999.

[14] Floyd, S., and Jacobson, V. Random Early Detection gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, V.1 N.4, August 1993, p.397-413.

[15] R. Guerin and V. Peris. Quality-of-Service in Packet Networks - Basic Mechanisms and Directions. *Computer Networks and ISDN Systems. Special issue on multimedia communications over packet based networks*, 1998.

[16] ns v2 simulator. See http://www-mash.cs.berkeley.edu/ns/

[17] M. May, J. Bolot, C. Diot, A. Jean-Marie. 1-bit Schemes for Service Discrimination in the Internet: Analysis and Evaluation. *Technical Report no 3238*, Inria.

[18] Y. Moret, S. Fdida. A Proportional Queue Control Mechanism to Provide Differentiated Services. *International Symposium on Computer Systems*, Belek, Turkey, October 1998.

[19] V. Jacobson, K. Nichols and K. Poduri. An Expedited Forwarding PHB. RFC2598.

[20] K. Nichols, V. Jacobson, L. Zhang. A Two-bit Differentiated Services Architecture for the Internet. RFC2638.

[21] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski. Assured Forwarding PHB Group. RFC2597.

[22] C. Dovrolis, D. Stiliadia, P. Ramanathan Proportional Differentiated Services: Delay Differentiation and Packet Scheduling Proceedings of ACM SIGCOMM'99.

[23] Floyd, S. TCP and Explicit Congestion Notification. *ACM Computer Communication Review*. V. 24 N. 5, October 1994, p. 10-23.

[24] J. Bolot, S. Fosse-Parisis, D. Towsley. Adaptive FEC-Based Error Control for Interactive Audio in the Internet.
it Proceedings of IEEE Infocom 99.

[25] C. Diot, C. Huitema, T. Turletti. Multimedia Application should be Adaptive. *HPCS*, Aug. 1995.

[26] K. Kilkki, Jussi Ruutu. Simple Integrated Media Access - an Internet Service Based on Priorities 6th International Conference on Telecommunication Systems, 1998.

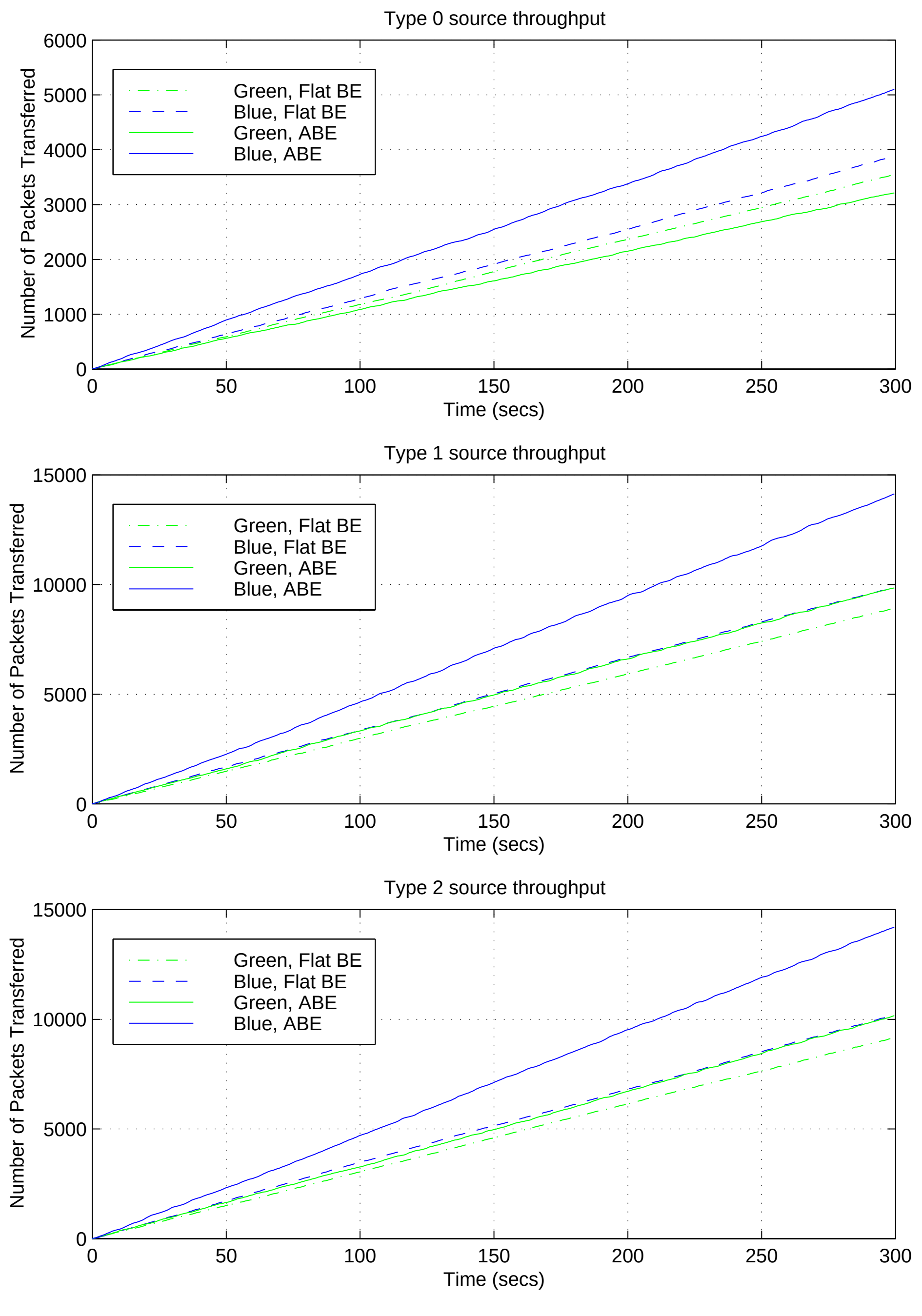
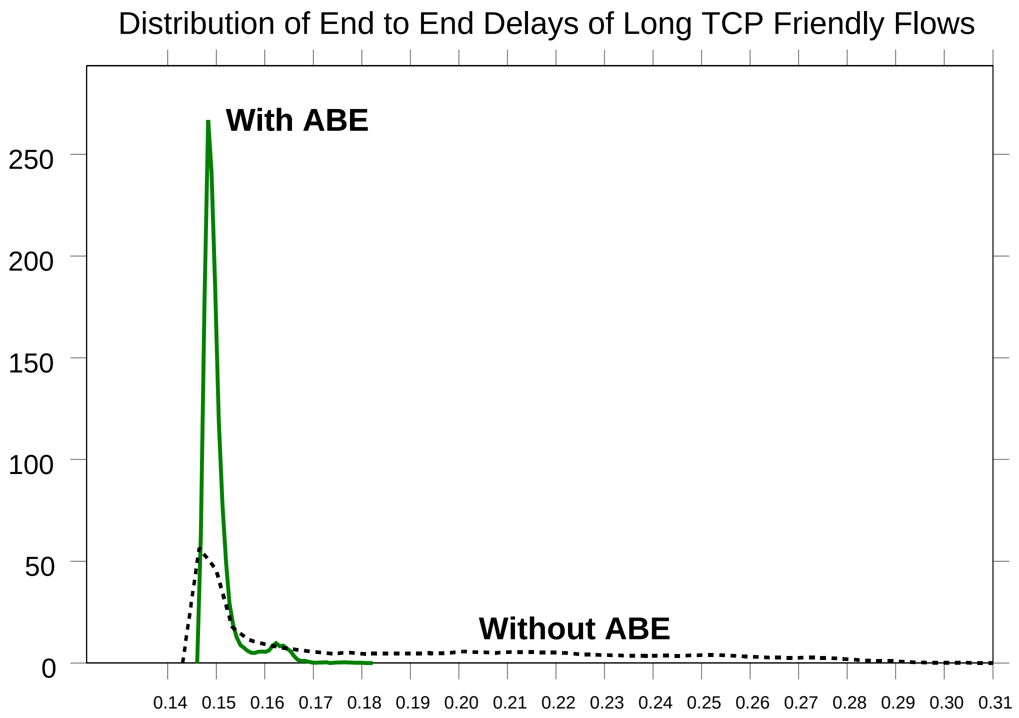


Figure 10: Average number of packets transferred per green and blue connection, for each traffic type, at each time t. When routers implement ABE and not (flat best-effort). There are 5 blue and green flows of each flow type.



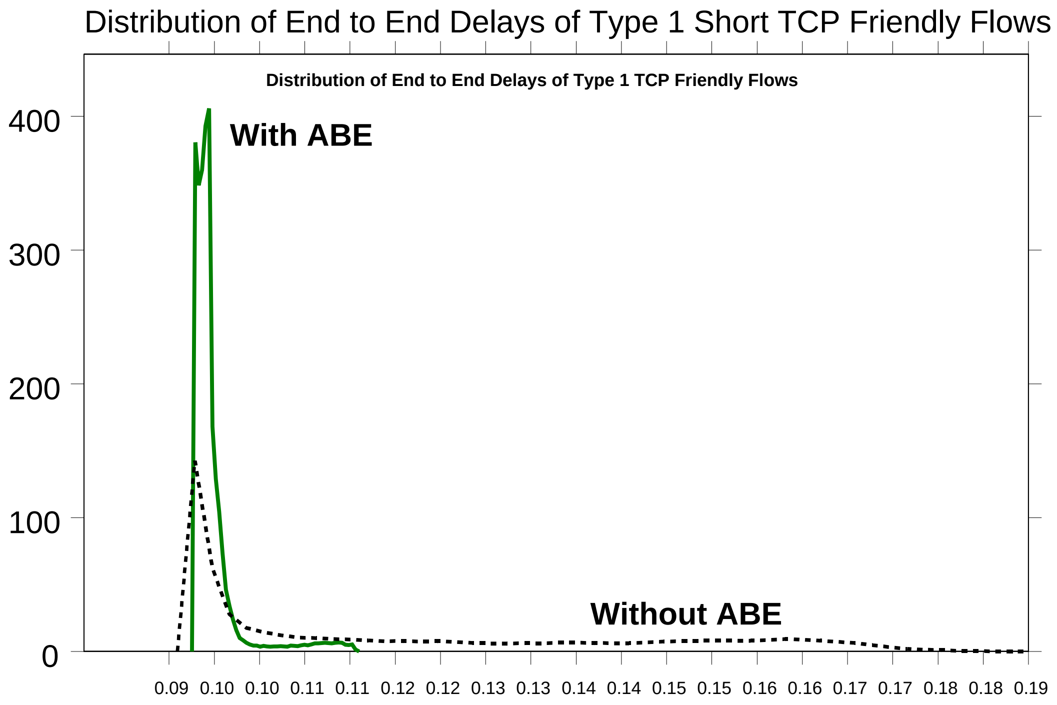


Figure 11: End-to-End Delay distributions received for green packets of each type under ABE and flat best-effort. 5 blue and green flows of each flow type.

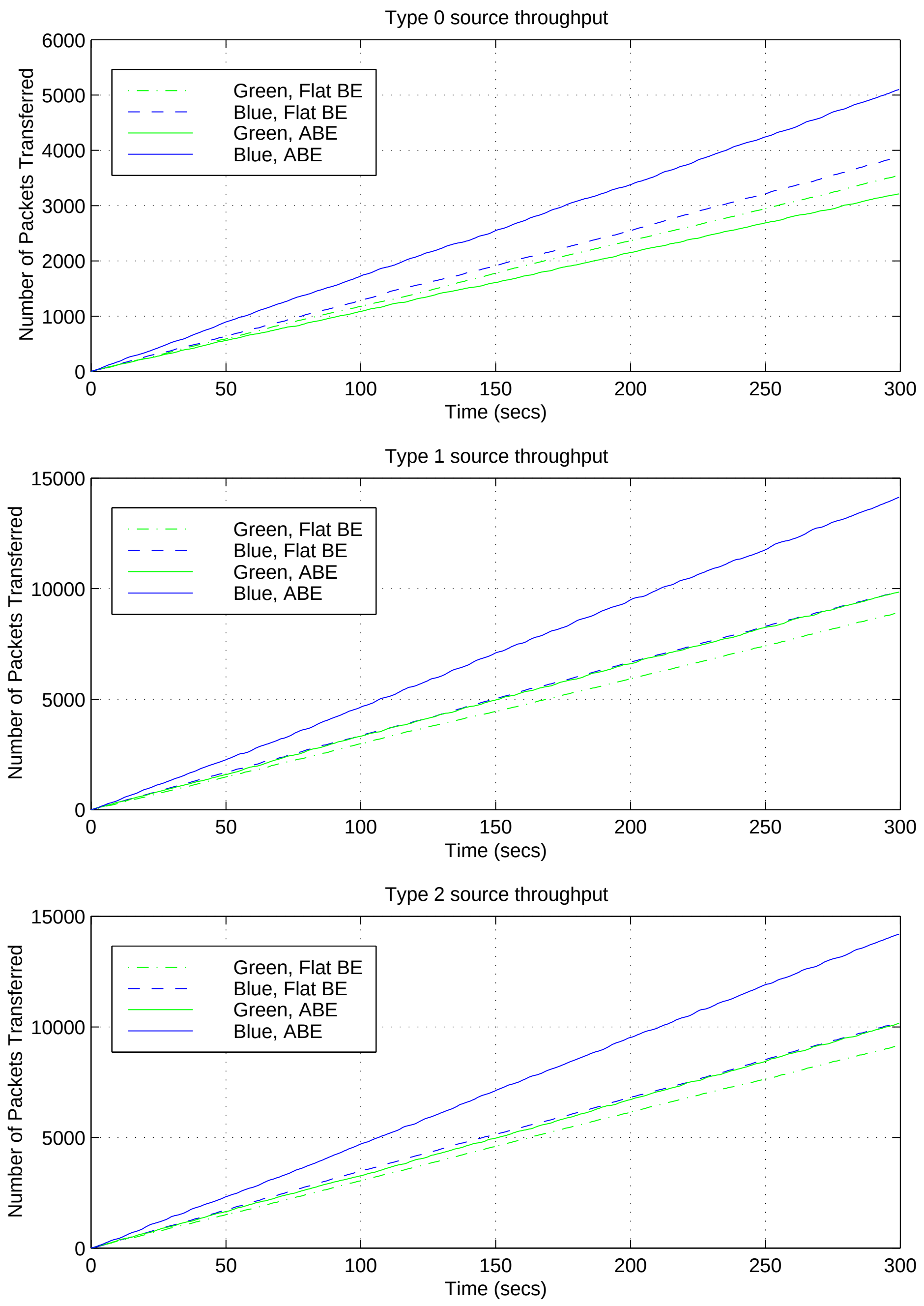


Figure 12: Average number of packets received per connection for each time t for ABE and flat best-effort when $n_{b,l} = n_{g,l} = 5$, $n_{b,s_1} = 1$, $n_{g,s_1} = 4$, $n_{b,s_2} = 4$, $n_{g,s_2} = 1$.