# Network Aware Applications: A Background Transfer Service

Peter Key, Laurent Massoulié
Microsoft Research
Roger Needham Building
7 J J Thomson Avenue
Cambridge, CB3 0FB, U.K.
{peterkey, lmassoul} @microsoft.com,

Bing Wang
Computer Science Department
Computer Science Building
University of Massachusetts
Amherst, MA 01003-4610
bing@cs.umass.edu

**Abstract**

Network aware applications react to changing network conditions, offering potential quality of service differentiation without network support. We describe an application level approach to designing a low priority service — one that is 'lower than best-effort' in the context of the Internet. Such applications are appropriate for background file transfers, such as OS updates. We use a receive window control to limit the transfer rate of the application, and the optimal rate is determined by detecting a change-point. We motivate this joint control-estimation problem by considering a fluid-based optimisation framework, and describe practical solutions, based on stochastic approximation and binary search techniques. Simulation results demonstrate the effectiveness of the approach.

## 1 Introduction

Network aware applications offer a way of introducing service differentiation into the current internet without requiring changes to the current infrastructure. By reacting to changes in underlying network conditions, applications can seek to assure their own desired quality of service. The least contentious type of service is one that is 'lower than best effort', i.e. lower than the current base-line quality offered. In this paper we focus on a background transfer service, where the objective is to transfer the data as quickly as possible, but without affecting other 'foreground' applications. Thus we aim to fully utilise the available network resources (those 'spare') whilst ensuring that background traffic is low priority compared to the foreground. Examples of background transfer include large file backup, transferring updates to the current operating system (e.g., Microsoft's Background Intelligent Transfer Service, (BITS)), contents prefetching [1] and distribution, storage management and caching in peer-to-peer systems [2].

One way to a achieve low-priority background transfer is to use priority queues at routers. However, this is not currently practical due to lack of both consensus and router support. Two alternative approaches, TCP Nice [3] and TCP-LP [4] require no support from the routers but instead modify the congestion detection and avoidance in TCP to achieve a low-priority transfer. They react both earlier and more aggressively than standard loss-based TCP (Reno). End-to-end delays are used to trigger a congestion signal before loss occurs, while more aggressive back-off mechanisms than those of standard TCP are used to give a lower share of network resources. Although only sender-side modification is required, this modification of TCP stack may still be difficult to deploy widely.

In this paper, we design an application-level approach for background transfers that uses TCP as the underlying transport protocol. Two components are needed: to infer the available capacity and to adjust the sending rate of the background transfer accordingly. There is a large literature on available capacity inference, however most current work relies on probing the end-to-end path, which requires coordination between sender and receiver. Furthermore, UDP probes can be blocked by firewalls and also impose extra load on the network. Using TCP directly to infer the available capacity, on the other hand, tends to cause overestimation [5].

In our application-level approach, we tightly couple the available capacity inference and rate adjustment: the rate of the background transfer is controlled by adjusting the receiver-advertised window size, which enforces a limitation on the rate used by the application. In turn, the rate obtained for a given receiver window is used to infer whether that rate is above or below the available capacity, which in turn triggers an adaptation of the receiver window.

The outline of this paper is as follows. We identify the relationship between receiver window and achieved rate in Section 2. In Section 3, we propose two application-level approaches for background transfer: one is based on binary search and the other is based on stochastic approximation. We evaluate the performance of the two approaches using *ns* simulation in Section 4.

## 2 Relationship between receiver window and achieved rate

In this section, we investigate the relationship between the achieved rate and the receiver window advertised by the receiver. The findings of this section underpin the method to be described in the next section. We investigate this relationship under two different assumptions: first, in Section 2.1, we assume buffer space at all links is large enough to prevent any packet losses. Secondly, in Section 2.2, we then consider the case where buffering is minimal and flows may experience loss.

We consider a general network comprising a collection $\mathcal{L}$ of links, each link $\ell$ having an associated nominal capacity $C_\ell$. There is a set $\mathcal{R}$ of foreground flows, each flow $r \in \mathcal{R}$ identifies a subset of $\mathcal{L}$, the links it uses, and each foreground flows implements TCP's congestion control. Finally, the background flow of interest is denoted by $b$, also identified with a subset of $\mathcal{L}$.

### 2.1 The large buffers case

Under the assumption of sufficiently large buffers, all flows are controlled by a fixed window, namely their receiver window. Denote by $w_r$ the receiver window of foreground flow $r$, and by $w_b$ that of the background flow $b$, and let $\tau_r$, $\tau_b$ denote the respective round-trip propagation delays. Then under FIFO scheduling, Massoulié and Roberts [6] showed that achieved rates $x = (x_r)$ and $x_b$ solve the optimisation problem

$$\begin{aligned} &\text{Maximise} && \sum_{r\in\mathcal{R}} w_r \log x_r - x_r\tau_r + w_b \log x_b - x_b\tau_b \\ &\text{subject to} && \sum_{r:\ell\in r} x_r + x_b \mathbf{1}_{\ell\in b} \le C_\ell, \quad \ell \in \mathcal{L} \quad \text{over} \quad (x_r) \in \mathbb{R}_+^{\mathcal{R}}, x_b \ge 0. \end{aligned} \tag{1}$$

Let $x_r(w_b)$ denote the corresponding unique optimal solution, where we have emphasised its dependence on the parameter of interest, $w_b$. The rates $x_r(0)$ correspond to the allocations achieved when the background flow is absent. Hence the spare capacity available to $b$ can be written

$$x_b^* = \min_{\ell \in b} \left( C_\ell - \sum_{r:\ell \in r} x_r(0) \right).$$

We assume that $x_b^* < \min_{\ell \in b}(C_\ell)$, so that foreground and background flows could interact. We then have the following result:

**Theorem 1.** *Let $w_b^* := x_b^* \tau_b$. In the present setting, foreground flows are affected by the background flow if and only if $w_b > w_b^*$. In addition, the function $x_b(w_b)$ is linear on $[0, w_b^*]$ with slope $1/\tau_b$, such that $x_b(w_b) < w_b/\tau_b$ on $(w_b^*, +\infty)$ and the function $x_b(w_b)/w_b$ is non-increasing on $\mathbb{R}_+$.*

*Proof.* By considering the Lagrangean for the optimisation (1), it is easily seen that for $w_b \in [0, w_b^*]$, the problem is solved by taking $x_r(w_b) = x_r(0)$, and $x_b(w_b) = w_b/\tau_b$. Thus when $w_b$ is in the range $[0, w_b^*]$, the background flow does not interfere with any other flows. When $w_b > w_b^*$, necessarily the Lagrange multiplier $\mu_\ell$ associated with the capacity constraint $C_\ell$ for some $\ell \in b$ needs to be strictly positive, hence $x_b(w_b)$, which equals $w_b/(\tau_b + \sum_{\ell \in b} \mu_b)$, is indeed strictly less than $w_b/\tau_b$. The last statement is established by noticing that, if we perform the optimisation in (1) first on the $x_r$, and then on $x_b$, the solution $x_b(w_b)$ is identified with the solution of the maximisation problem:

$$\text{Maximise} \quad w_b \log x_b - \tau_b x_b + \psi(x_b) \quad \text{over} \quad x_b \geq 0,$$

where the function $\psi$ is defined as

$$\psi(x_b) = \max_{(x_r) \in \mathcal{S}(x_b)} \left\{ \sum_{r \in \mathcal{R}} w_r \log x_r - x_r \tau_r \right\}$$

where $\mathcal{S}(x_b) = \left\{ (x_r) \in \mathbb{R}_+^{\mathcal{R}} : \sum_{r:\ell \in r} x_r \leq C_\ell - x_b \mathbf{1}_{\ell \in b} \right\}$ is the set of achievable foreground flows given the background flow rate $x_b$. It is easy to see that $\psi$ is non-increasing and concave. Assuming for simplicity that $\psi$ is differentiable[1], the function $x_b(w_b)$ also reads $f^{-1}(w_b)$, where $f(x) = x(\tau_b - \psi'(x))$. The function $f^{-1}(w_b)/w_b$ is non-increasing if and only if the function $f(x)/x$ is non-decreasing, that is if and only if $-\psi'(x)$ is non-decreasing. The latter property is implied by concavity of $\psi$. Finally, observing that $f$ and hence $f^{-1}$ are non-decreasing, it can be seen that necessarily, for any $w_b > w_b^*$, one has $x_b(w_b) > x_b^*$. It then follows that for $w_b > w_b^*$, there must exist some foreground flow $r$ such that $x_r(w_b) < x_r(0)$. □

## 2.2 The small buffers case

We now assume that buffers are small, and hence congestion control relies on losses rather than on buffering. Many studies have shown that the rates achieved by TCP flows in such a regime may be interpreted as solving an implicit optimisation problem; see e.g. [8, 9, 10]. We follow this approach, and assume that the rates $(x_r)$, $x_b$ are the solution to the problem:

$$\begin{aligned} \text{Maximise} \quad & \sum_{r \in \mathcal{R}} U_r(x_r) + U_b(x_b) - \sum_{\ell} \Gamma_\ell \left( \sum_{r:\ell \in r} x_r + x_b \mathbf{1}_{\ell \in b} \right) \\ \text{subject to} \quad & 0 \leq x_b \leq \frac{w_b}{\tau_b} \quad \text{over } (x_r) \in \mathbb{R}_+^{\mathcal{R}},\ x_b \geq 0. \end{aligned} \tag{2}$$

[1]This is actually not the case; however a rigorous argument can be constructed along the same lines, based on sub-differentials of convex functions (see Rockafellar [7], Chapter 23) rather than ordinary differentials.

The constraint captures the impact of the receiver window size $w_b$. In the above, the so-called utility functions $U_r$ are assumed to be strictly concave and increasing. It has been suggested (see [11], [8]) that adequate choices for modelling TCP capacity sharing could be $U_r(x) = -1/(\tau_r^2 x)$, or $U_r(x) = \tau_r^{-1} \arctan(\tau_r x)$. The other key component in (2) is the penalty function $\Gamma_\ell$. It is related to the loss probability $p_\ell(x)$ at link $\ell$ when it carries data at rate $x$ via the equation

$$\Gamma_\ell(x) = \int_0^x p_\ell(y)dy.$$

A special choice advocated in [8] consists in setting $p_\ell(y) = (y - C_\ell)^+/y$. The results below rely solely on the assumption that the utility functions $U_r$, $U_b$ are strictly convex increasing, and that the loss rate functions $p_\ell$ are non-decreasing and continuous. As before, let $(x_r(w_b))$ and $x_b(w_b)$ denote the rates achieving the maximum in the optimisation problem (2). We also make use of the notation $\pi_\ell(w_b) := p_\ell\left(\sum_{r:\ell\in r} x_r(w_b) + x_b(w_b)\mathbf{1}_{\ell\in b}\right)$. The function $\pi_\ell$ is naturally interpreted as the loss probability along link $\ell$. We shall also denote the goodput obtained by the background flow as

$$y_b(w_b) := x_b(w_b)\left(1 - \sum_{\ell\in b}\pi_\ell(w_b)\right).$$

Again, $x_r(0)$ is the rate obtained by foreground flow $r$ in the absence of background flows. We define the spare capacity available to flow $b$ as

$$x_b^* := \min_{\ell\in b}\inf\left\{x > 0 : p_\ell(x + \sum_{r:\ell\in r} x_r(0)) = 0\right\}.$$

We assume that there exists a link $\ell \in b$ and a flow $r'$, $\ell \in r'$ such that $x_{r'}(0) > 0$ and $p_\ell(x_b^* + \sum_{r:\ell\in r} x_r(0) + z) > 0$ for all $z > 0$, so that background and foreground flows could interact. In this context, similarly to Theorem 1, we have the following.

**Theorem 2.** *Let $w_b^* := x_b^*\tau_b$. In the present setting, foreground flows are affected by the background flow if and only if $w_b > w_b^*$. Moreover, the goodput $y_b(w_b)$ received by the background flow is linear in $w_b$ on the interval $[0, w_b]$, with slope $1/\tau_b$, is such that $y_b(w_b)/w_b < \tau_b^{-1}$ when $w_b > w_b^*$, and the function $y_b(w_b)/w_b$ is non-increasing in $w_b$.*

*Proof.* (sketched). We only provide those steps that differ significantly from those in the proof of Theorem 1. It is readily seen that, for $w_b \in [0, w_b^*]$, the optimisation problem (2) is solved by setting $x_r(w_b) = x_r(0)$ and $x_b(w_b) = w_b/\tau_b$. The case where $w_b > w_b^*$ can be further divided into two sub-cases, according to whether the constraint imposed by $w_b$ is binding or not. Denote by $\hat{w}_b$ the critical value for $w_b$ where the receiver window constraint ceases to be binding. In the interval $(w_b^*, \hat{w}_b)$, one has $x_b(w_b) \equiv w_b/\tau_b$. Similarly to the proof of Theorem 1, define

$$\psi(x_b) := \max_{(x_r)\in\mathbb{R}_+^{\mathcal{R}}}\left\{\sum_{r\in\mathcal{R}} U_r(x_r) - \sum_\ell \Gamma_\ell\left(\sum_{r:\ell\in r} x_r + x_b\mathbf{1}_{\ell\in b}\right)\right\}.$$

This function is non-increasing and concave. The interval $(w_b^*, \hat{w}_b)$ may then be alternatively characterised as the one for which $U_b'(w_b/\tau_b) + \psi'(w_b/\tau_b) > 0$. In addition, it can be shown that the derivative $-\psi'(w_b/\tau_b)$ coincides with the loss rate $\sum_{\ell\in b}\pi_\ell(w_b)$ over the range $(w_b^*, \hat{w}_b)$. By concavity of $\psi$, $-\psi'$ is non-decreasing. As a result, the normalised goodput function $y_b(w_b)/w_b$, which reads $\tau_b^{-1}(1 - \sum_{\ell\in b}\pi_\ell(w_b))$, is indeed non-increasing. Finally, in the range $w_b > \hat{w}_b$, one has $y_b(w_b) \equiv y_b(\hat{w}_b)$, and hence the normalised goodput is decreasing there as well. $\square$

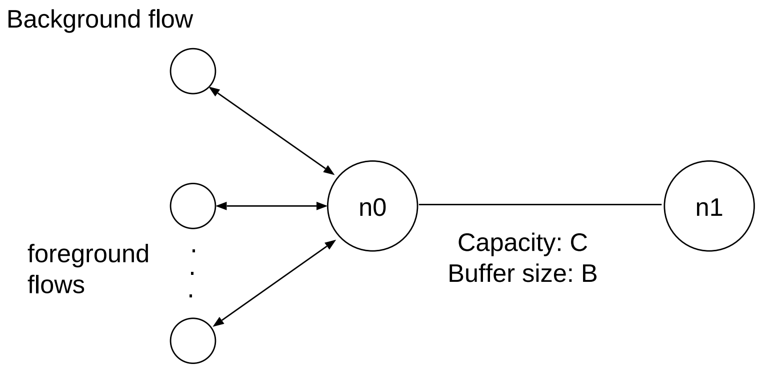


Figure 1: The topology in *ns*: The link between node node $n0$ and $n1$ forms a bottleneck link.

In both contexts, we observe that goodput normalised by receiver window is constant over a range $[0, w_b^*]$, and decreasing over $(w_b^*, +\infty)$, where the critical window $w_b^*$ is precisely the one we should use to maximise background goodput while not interfering with foreground flows.

## 2.3 Experimental validation

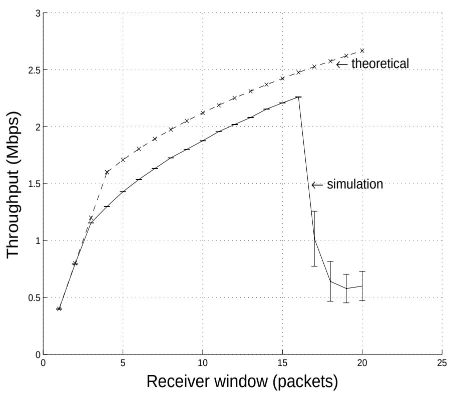
Although we have analysed only two extreme cases, namely large buffers (no loss), and no buffering but losses, we expect the same qualitative behaviour to hold in mixed situations where there is both significant buffering and loss. We now confirm this expectation using *n*s simulations. The topology used in *ns* is shown in Figure 1. The link between nodes $n0$ and $n1$ forms a bottleneck link with capacity $C$. The buffer size of node $n0$ is $B$. Here we have a number $N$ of foreground flows, each a long-lived TCP connection, with a round trip propagation delay $\tau_f$ and a maximum window size of $w_f$. The generic network optimisation problem (1), specialised to the present situation, reads

$$\text{Maximise} \quad N(w_f \log(x_f) - \tau_f x_f) + w_b \log x_b - \tau_b x_b \tag{3}$$

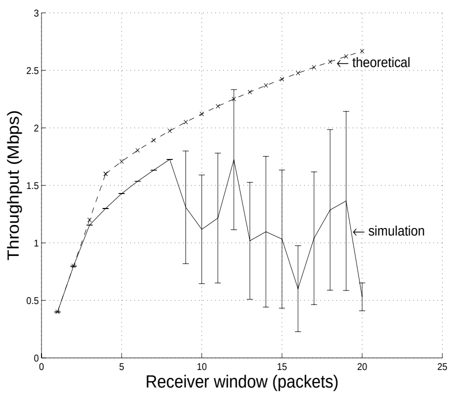
subject to $Nx_f + x_b \leq C$. We would expect its solution to predict accurately the actual achieved rates in the absence of losses, that is when the buffer size $B$ is large.

We now report experiments for $N = 8$ foreground flows, each with characteristics $w_f = 20$ packets and $\tau_f = 100$ ms, a capacity $C$ of 2000 packets per second. $\tau_b$ is set to $10$ ms. The packet size is fixed to 500 bytes. The maximum aggregate bandwidth usage of the foreground TCP flows is then $1600$ packets per second. The foreground TCP flows are started at time $0$ and the background flow is started at the $10$th second. The window size of the background flow is initially $1$ packet, and increases by 1 packet every 40 seconds. For each background flow window size, we obtain the average background flow throughput every 1 second with a total of $40$ samples. We then obtain the mean and the confidence interval from the $40$ samples.

The solid line in Figure 2(a) shows the background flow throughput as a function of the window size, with 95% confidence intervals, when $B = 40$ packets. The result from solving the simple optimization problem (3) is also shown in the figure, which indicates that the optimal window size for the background flow is $4$ packets. From the simulation, the optimal value is lower (3 packets). Packet losses start to occur when $w_b = 16$ packets. Fig. 2 (b) shows the background flow throughput versus the window size for a smaller buffer size at node $n0$ ($B = 20$ packets). Packet losses start to occur when $w_b = 8$ packets. Note that, when loss occurs, the variance in the background flow throughput increases dramatically; furthermore, the throughput is not necessarily an increasing function of the window size.



(a) $B = 40$ packets.



(b) $B = 20$ packets.

Figure 2: The throughput of the background flow versus the window size with $8$ foreground flows, when $C = 2000$ packets per second.

## 3 The joint control and estimation problem

The previous results suggest that one could adapt the receiver window $w_b$ of the background flow, based on the observed achieved rates, so as to tune it to the critical value $w_b^*$ at which interference with foreground flows would start occurring. In what follows, we assume that time is divided in control intervals, each of length $T$. In the $n$th interval, a receiver window of size $w_n$ is applied[2] and an amount $R_n$ of data is received, both $w_n$ and $R_n$ being expressed in the same unit, which is a fixed number of bytes $u$.

Let $\rho_n := R_n/w_n$. Based on the results of the previous section, we postulate the following model for the observed quantities $\rho_n$:

$$\rho_n = \phi(W_n) + Z_n, \tag{4}$$

where the $Z_n$ represent observation noise, assumed to be centred, and the function $\phi$ is the normalised goodput function, which, in view of Theorems 1 and 2, is assumed to be non-increasing, and such that:

$$\phi(w) \begin{cases} \equiv \rho := T/\tau_b, & \text{if } w \in [0, w_b^*], \\ < \rho & \text{if } w > w_b^*. \end{cases}$$

In the special case where the parameter $\rho$ is known, and there is no measurement noise, at the end of the $n$th control interval, by comparing $\rho_n$ to $\rho$ one can directly infer whether $w_n \leq w_b^*$ or not. Based on such binary feedback, one can reduce the search interval in which the unknown critical window lies. In this situation, binary search constitutes a very natural candidate strategy. In fact, Karp et al. [12] address such a problem. They show (among other things) that, if cost at each control interval is measured by the absolute difference between $w_b^*$ and $w_n$, and the unknown critical window is uniformly distributed among the integers in $[1, A]$, then binary search is asymptotically optimal for both the worst and average cumulative cost in the limit $A \to \infty$.

Our problem differs from that tackled in [12] mainly because the parameter $\rho$ is unknown, and the observations are noisy. Rather than trying to identify the ideal window $w_b^*$, we try instead to find the window size $w^*$ solving

$$\phi(w^*) = \rho - \epsilon, \tag{5}$$

[2]Subscripts now denote control intervals, and not flows as in the previous section.

where $\epsilon$ is some positive parameter. This makes the search problem easier, and alleviates the impact of observation noise, although now $w^* > w_b^*$, and hence overestimates the target window size. We discuss the impact of using $w^*$ on foreground flows in Section 3.3.

In the two control methods we propose next, we maintain an Exponentially Weighted Moving Averge (EWMA) estimate $\hat{\rho}_n$ of $\rho$,

$$\hat{\rho}_n = \begin{cases} (1-\delta)\hat{\rho}_{n-1} + \delta\rho_n & \text{if } \rho_n \geq \hat{\rho}_{n-1} - \epsilon', \\ \hat{\rho}_{n-1} & \text{otherwise.} \end{cases} \tag{6}$$

This features two positive parameters $\delta \in (0,1)$ and $\epsilon'$, and can be initialised by taking $\hat{\rho}_1 = \rho_1$. We now describe our two candidate control methods.

## 3.1 The Binary Search Method

In this approach we maintain a search range $[w_{min}, w_{max}]$ for the desired window $w^*$. The lower limit $w_{min}$ is intially set to 1, and the upper limit $w_{max}$ set to a system dependent limit[3]. At the beginning of a control interval, we set $w_n = \lfloor (w_{min} + w_{max})/2 \rfloor$, where $\lfloor \cdot \rfloor$ denotes integer part. At the end of the control interval, if $\rho_n > \hat{\rho}_{n-1} - \epsilon$, we let $w_{min} = w_n$; otherwise we let $w_{max} = w_n$. Such standard binary search proceeds until the difference $w_{max} - w_{min}$ is 0 or 1. In the absence of noise, and in a static environment we would then be guaranteed to have reached the optimal window $w^*$, and the search could be stopped there.

These assumptions are not verified in practice, and we describe how we proceed when the search interval has length at most 1. If $\rho_n > \hat{\rho}_{n-1} - \epsilon$, we make the change $w_{max} = w_{max} + 2$; otherwise, we make the changes $w_{max} = w_{max} - 1$ and $w_{min} = 1$.

In view of the optimality properties of binary search derived in [12], we expect the above approach to perform well. An interesting issue concerns how the presence of the measurement noise terms $Z_n$ in (4) affect such optimality properties.

## 3.2 The Stochastic Approximation Method

Stochastic approximation is a general technique for finding solutions to an equation $f(x) = 0$, given noisy measurements $f(x) + Z$, using

$$X_n = X_{n-1} - \gamma_n \theta_n,$$

where $\theta_n = f(X_{n-1}) + Z_n$ is the noisy observation of $f(X_{n-1})$. Such algorithms are also known as Robbins-Monro algorithms [13]. Conditions on the gain sequence $\gamma_n$, and on the noise variables $Z_n$, under which $X_n$ converges almost surely to a desired solution are known[4].

The control problem at hand can be cast in this framework. Indeed, the equation (5) we are trying to solve is precisely of the form $f(w) = 0$, with $f(w) = \phi(w) - \rho + \epsilon$, and, ignoring for now the fact that $\rho$ is unknown, we do have access to noisy estimates of $f(w_n)$, namely $\rho_n - \rho - \epsilon$. This thus suggests to use updates of the form:

$$w_n = w_{n-1} + \gamma_n(\epsilon + \rho_{n-1} - \hat{\rho}_{n-2}). \tag{7}$$

[3]Alternatively, a preliminary search for suitable value of $w_{max}$ can be made, starting from $w_1 = 1$, and doubling $w_n$ at in each subsequent interval, until the condition $\rho_n > \hat{\rho}_{n-1} - \epsilon$ fails, at which point we set $w_{max} = w_n$.

[4]Essentially requiring $\gamma_n$ to converge slowly, e.g. $\sum_{n>0} \gamma_n$ unbounded but $\sum_{n>0} \gamma_n^2$ finite, with $Z_n$ having zero mean and finite variance.

Provided the estimates $\hat{\rho}_n$ converge to $\rho$, we would expect the corresponding sequence $w_n$ to converge to $w^*$ under mild assumptions on the observation noises $Z_n$. However we have not pursued this yet, and have instead experimented with the fixed gain version of the update rule (7), where the $\gamma_n$ are held constant to a common value $\gamma$. This is more appropriate in a dynamic environment where the target $w^*$ itself might evolve over time.

### 3.2.1 Tradeoffs between the two approaches

In binary search, when the search interval is of length at most 1 and the receiver window size is too large, it is decreased by half. This is in the same spirit as the change in congestion window size in TCP, and ensures that binary search adapts quickly to the dynamic available capacity. However, in a stable environment, the receiver window size may still fluctuate, which leads to less efficient capacity utilization. The stochastic approximation algorithm can be tuned so that the receiver window size converges in a stable environment. However, since the increment and decrement in the receiver window size is linear (see (7)), it reacts more slowly to changes in the available capacity.

## 3.3 Tuning $\epsilon$: The interference impact on foreground flows

We next explore the interference of a background flow on TCP flows, caused by using the window $w^*$ solving (5) rather than $w_b^*$. Here we consider only the same simple situation as Section 2.3, with one background flow and $N$ homogeneous TCP flows sharing a single link of capacity of the $C$, with enough buffering to avoid any data loss. Suppose the window size of the TCP flow is fixed and the background flow adapts its window as in the previous sections to track $\rho - \epsilon$, where $\rho = T/\tau_b$. In equilibrium, denote the window size of the background flow by $w_b^\epsilon$. Then the equilibrium rates $x_f^\epsilon$ and $x_b^\epsilon$ are again characterised as the unique solution to the optimisation problem (3). It is then straightforward to show

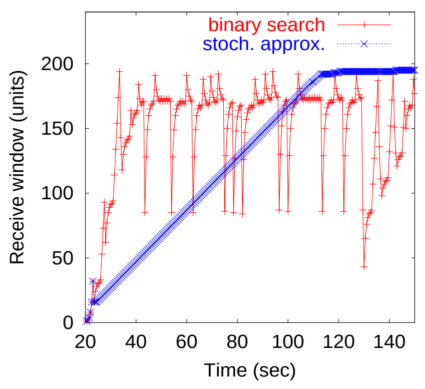
**Theorem 3.** *In the case that the round-trip times are the same ($\tau_f = \tau_b = \tau$), the relative reduction in the TCP rate in the presence of the background flow caused by using a positive threshold $\epsilon$ is bounded above by*

$$D = \frac{\tau\epsilon}{T}.$$

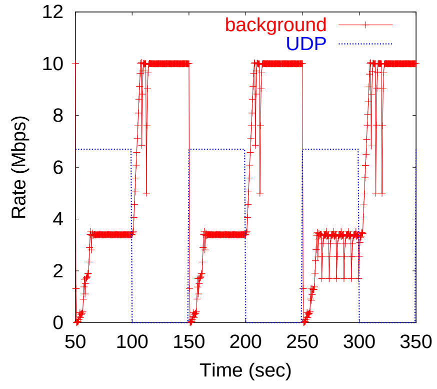
*Proof.* Outline: In the absence of the background flow, $x_f = \min(w_f/\tau, C/N)$. Solving the optimisation problem and using the fact that $x_b^\epsilon T = (\rho - \epsilon)w_b^\epsilon$ enables one to solve explicitly for $w_b^\epsilon$ and $x_f^\epsilon$, and show that the relative reduction in the TCP rate caused by the background flow is indeed bounded above by $D$. □

# 4 Results

We now give some illustrative simulation results of our algorithm. The simulation set-up is as in Figure 1, with the capacity of C=10Mb/s (with the background flow having a 100Mb/s link to the bottleneck). The round-trip propagation for the background flow is 44ms. The data unit is $u = 100$ bytes, and we take $\epsilon = 1$ and $\epsilon' = 1$. The EWMA parameter $\delta$ for $\rho_n$ was set to $\delta = 0.1$, and the control time $T$ varied. Figure 3 shows the reaction time to two different types of background traffic. Figure 3(a) compares binary search and stochastic approximation (with gain parameter $\gamma = 1$) for FTP background traffic, with a control interval of $T = 0.5$ seconds. The FTP foreground traffic uses 58% of the bottleneck (i.e. is window constrained),



(a) FTP foreground traffic



(b) UPD on-off foreground

Figure 3: Reactivity of algorithms to FTP foreground traffic and UDP on-off foreground.

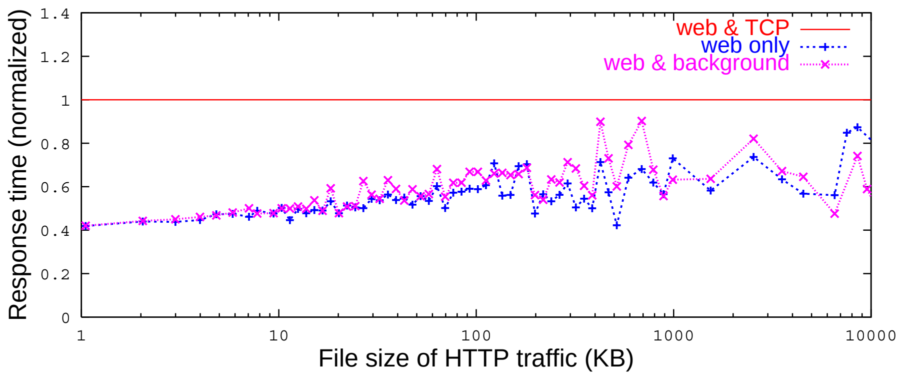


Figure 4: Average response time for file dowloads with and without ftp and background traffic

and the utilisation raised to 95% (stochastic approximation) or 98%(binary search) when the background flow is added, with a corresponding reduction of foreground throughput of 2% or 6% respectively. Note that both approaches do achieve the design goals, and the simulations reflect the trade-off between the speed of reaction and variability discussed in the previous section. Figure 3(b) shows the reactivity of the binary search method when the foreground traffic is on-off UDP, with a peak of 6Mb/s and equal on/off periods of 50 seconds. The background flow reacts well to changes in available capacity, and is indeed able to use around 90% of what is available.

We now examine the behaviour of the binary search algorithm in a more dynamic environment when there is Web traffic. Each web session contains several web pages, and each web pages contains $10$ objects. The inter-page and inter-object time distributions are exponential with means of $1$ sec and $1$ msec respectively. The object size is distributed according to a Pareto distribution with shape parameter of $1.2$. Figure 4 plots the average response time of the web pages versus the size of the web page when there is no background FTP flow, when the FTP uses our algorithm (with $T = 0.5$ sec) and when the FTP uses TCP Reno. The response times in the first two situations are similar and lower than those when the FTP uses TCP Reno. When $T = 0.5$ sec, the average bandwidth usage of the background flow and the Reno TCP flow is $0.32$ Mbps and $0.69$ Mbps respectively. In other words TCP Reno attempts to get a fair share, and interferes with Web traffic by grabbing bandwidth, whilst our algorithm is able to adapt dynamically and has little effect on the foreground Web traffic.

## 5 Concluding remarks

We have looked at the problem of creating a background transfer service using an application layer reaction, adapting a receiver window to create a low-priority service. By phrasing the idealised problem as an optimisation, we were able to characterise the solution as dual control problem, where the aim is to control the window to a critical value which represents a change point. We have given two ways of attacking this control problem, using binary search or stochastic optimisation techniques. Simulations have given credence to our framework, and initial results on the algorithms are encouraging.

## References

[1] A. Venkataramani, P. Yalagandula, R. Kokku, S. Sharif, and M. Dahlin. The potential costs and benefits of long term prefetching for content distribution. *Computer Communication Journal*, 25(4):367–375, 2002.

[2] Antony I. T. Rowstron and Peter Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *SOSP*, pages 188–201, 2001.

[3] A. Kuzmanovic and E. Knightly. TCP-LP: A distributed algorithm for low priority data transfer. In *Proc. IEEE INFOCOM*, 2003.

[4] A. Venkataramani, R. Kokku, and M. Dahlin. TCP Nice: A mechanism for background transfers. In *Proc. Operating Systems Design and Implementation*, December 2002.

[5] M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. In *Proc. ACM SIGCOMM*, 2002.

[6] Laurent Massoulié and James Roberts. Bandwidth sharing: Objectives and algorithms. In *Proc. IEEE INFOCOM*, volume 3, pages 1395–1403, 1999.

[7] T.R. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[8] S. Kunniyur and R. Srikant. End-to-end congestion control schemes: Utility functions, random losses and ECN marks. In *INFOCOM*, 2000.

[9] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35:1969–1985, 1999.

[10] F. P. Kelly, A. K. Maulloo, and D. K. H Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *J. Op. Res. Soc.*, 49:237–252, 1998.

[11] F. P. Kelly. Mathematical modelling of the Iinternet. In *Proceedings of the Fourth International Congress on Industrial and Applied Mathematics*, 2000.

[12] Richard M. Karp, Elias Koutsoupias, Christos H. Papadimitriou, and Scott Shenker. Optimization problems in congestion control. In *IEEE Symposium on Foundations of Computer Science*, pages 66–74, 2000.

[13] H. Robbins and S Munro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.