Minimally Intrusive Server Policies for Background Data Transfers

Costas Courcoubetis, Antonis Dimakis, and Michalis Kanakakis

Abstract We consider the problem of designing access control protocols for servers distributing background data, such as software and database updates, so that they cause the least possible disruption to flows carrying delay-sensitive data. Using a Markov decision process formulation we obtain the optimal policy analytically, which is not easy to implement in practice. A mean-field argument is employed to show that another policy, which is easier to implement and is based on water-filling, converges to the optimal as the number of bottleneck links increases. Using simulations we compare the performance of this policy with the standard case where no control is exercised by the server.

1 Introduction

In this paper we look at the problem of designing data transmission policies at the server side for data which are not delay-sensitive, such as software and database updates, backup and content replication, content cache prefetches, all commonly referred to as *background*. These policies are important as they usually involve the transfer of big volumes of data which can potentially cause increased download delays to delay-sensitive data such as web traffic, video and audio streaming over TCP when they share the same network resources. This problem has been recognized in Deb et al (2005); Key et al (2005); Shai et al (2007); Courcoubetis and Dimakis

C. Courcoubetis
Department of Engineering Systems and Design, Singapore University of Technology and Design, Singapore, e-mail: costas@sutd.edu.sg

A. Dimakis, M. Kanakakis
Deparment of Informatics, Athens University of Economics and Business, Athens, Greece, e-mail: dimakis@aueb.gr, e-mail: kanakakis@aueb.gr. A. Dimakis was supported by a grant funded and administered by the Research Centre of Athens University of Economics and Business (RC/AUEB).

(2012), and end-to-end congestion control mechanisms were proposed which offer explicit or implicit throughput guarantees to background data, while at the same time attempt to have a minimal impact on the download delay on delay-sensitive data. All this cited work critically depends on the assumption of a single bottleneck link. As servers are associated with multiple clients and the server-side access network is rarely the bottleneck, such mechanisms are not useful in this context because multiple bottlenecks may exist.

In this paper we start from the Markov decision problem for a single link in Courcoubetis and Dimakis (2012) but we narrow the set of policies by imposing additional constraints on action space which exclude policies which are not practical to implement on the basis of link congestion information alone. In Theorem 1 we analytically derive the optimal policy which still rests on a single bottleneck link assumption. However the structure of this optimal policy can be approximated by a suboptimal policy –the Water-Filling Algorithm (WFA) introduced in Sect. 3– which is easy to implement when multiple bottlenecks exist. In Theorem 2, using a mean-field argument, we show that the approximation becomes tight as the number of bottlenecks increase to infinity. In fact when the link and traffic characteristics are homogeneous, Theorem 2 implies that WFA is optimal in the limit (Corollary 1). WFA essentially limits the maximum number of data transfers and prioritizes faster transmissions. Interestingly, this is just what a BitTorrent seeder does (e.g., see Cohen (2003)), and so our analysis reveals the conditions under which it behaves optimally and in what sense. In Sect. 4 we develop a prototype implementation of WFA in the ns2 simulator, and compare its performance to the standard case where no control is exercised by the server. Our findings indicate that WFA can result to significant (20% to 30%) reductions in the average download delay of delay-sensitive flows.

2 Optimal Policy for a Single Bottleneck

Consider a server which wants to send background data with long-term average rate b. The server uses a link with capacity C shared also by a constant number k of persistent TCP flows, and a dynamically arriving stream of delay-sensitive TCP flows. The latter concern transfers of files with independent and exponentially distributed file sizes, of mean μ^{-1}, and arrive at the link according to a Poisson process with rate λ arrivals per unit time. Define the offered load of delay-sensitive flows as $\rho = \lambda/(\mu C)$. All TCP flows (whether persistent or not) are assumed to receive an equal bandwidth share, x_n, when the number of delay-sensitive flows in the system is n. The server influences x_n by picking the number a_n of TCP flows to use at state n, through the formula $x_n = C/(n+k+a_n)$. Since any value of x_n can be achieved for some choice of a_n when the latter is permitted to be nonintegral, we consider $(x_n, n \geq 0)$ to be the decision variables. The number n of short flows evolves according to a Markov chain with state space $\{0,1,2,\dots\}$ and transition rates:

$$n \to \begin{cases} n+1\,, & \text{with rate } \lambda\,, n \geq 0\,, \\ n-1\,, & \text{with rate } \mu n x_n\,, n \geq 1\,. \end{cases} \tag{1}$$

Notice that n is not known directly by the server but may be inferred indirectly through congestion indicators, such as packet losses or packet delay. This is easy to do if these indicators are monotonic functions of n, but it becomes highly nontrivial if they are not. For example, whenever $x_n < x_{n+1}, x_{n+2} = x_n$, one cannot differentiate between states $n, n+2$ on the basis of congestion indicators alone unless past information is kept. We would like to avoid such complex policies and for this reason we require that x_n is decreasing in n.

Now let $(\pi_n, n = 0, 1, \ldots)$ be the stationary distribution of the Markov chain when it does exist. The *average download delay* of the delay-sensitive flows is $\sum_{n=0}^{\infty} n\pi_n/\lambda$, by Little's law, and the problem we solve is the following:

$$\min \sum_{n=0}^{\infty} n\pi_n \tag{2}$$

$$\text{such that: } (\pi_n, n \geq 0) \text{ is the stationary distribution of (1)} \tag{3}$$

$$x_n \leq \frac{C}{k+n}\,, n = 0, 1, \ldots \tag{4}$$

$$b + \sum_{n=0}^{\infty} k x_n \pi_n = C(1-\rho) \tag{5}$$

$$x_n \geq x_{n+1}\,, n = 0, 1, \ldots \tag{6}$$

$$\text{over } x_n, \pi_n \geq 0,\ n = 0, 1, \ldots \tag{7}$$

Equation (4) is due to the link capacity constraint and the fact that all TCP flows get equal bandwidth shares, and (5) requires the sum of the rate offered by background data and the throughput of persistent TCP flows to equal the long-term capacity not used by delay-sensitive flows, i.e., the throughput obtained by background data matches the offered load.

The first main result concerns the structure of the optimal policy:

Theorem 1 (Structure of the optimal policy). *The optimal policy $(x_n, n \geq 0)$ satisfies*

$$x_n = \begin{cases} x_{n-1}\,, & \text{if } n \leq n_*\,, \\ \frac{C}{k+n} & \text{if } n > n_* \end{cases}\,, n = 1, 2, \ldots \tag{8}$$

for some finite nonnegative integer n_*.

Proof. A straightforward modification to Lemma 2 in Courcoubetis and Dimakis (2012) implies that under the identification

$$\bar{\pi}_n = x_n \pi_n \frac{k}{C(1-\rho)-b}\,, y_{n+1} = \frac{\bar{\pi}_n}{\bar{\pi}_{n+1}}\,, n = 0, 1, \ldots\,, \tag{9}$$

the problem (2)-(7) is equivalent to the problem (21) in Courcoubetis and Dimakis (2012) with the additional constraint $y_{n+1}/(n+1) \leq y_n/n$ for all $n = 1,2,\ldots$, over the variables $y_{n+1}, \bar{\pi}_n, n \geq 0$. We will show that the optimal solution satisfies

$$y_n = \begin{cases} \frac{n}{\rho(k+n-1)} & n \geq m \\ \frac{n y_{n-1}}{n-1} & 1 \leq n < m \end{cases}, \tag{10}$$

for some $m \geq 1$. Assume for the moment that this is true. Then (10) is equivalent to (8) by the identification (9) and $n_* = m-1$. Thus it suffices to show (10) to conclude the proof. Observe that the second eq. in (9) and the fact that $(\bar{\pi}_n)$ is a probability distribution imply that $(\bar{\pi}_n, n \geq 0)$ can be interpreted as the stationary distribution of a birth-death chain with unit birth rate and death rate y_n in state $n \geq 1$. Now suppose $y_{m+1} < (m+1)/(\rho(k+m))$ and $y_m > m y_{m-1}/(m-1)$ for some $m \geq 1$. Then there exist some other set of death rates $(y'_n, n \geq 1)$ with $y_{m+1} < y'_{m+1} \leq (m+1)/(\rho(k+m))$, $y_m > y'_m \geq m y_{m-1}/(m-1)$, $y'_n = y_n$ for all $n \notin \{m, m+1\}$, for which the corresponding stationary distribution $(\bar{\pi}'_n, n \geq 0)$ of the birth-death chain continues to satisfy the target throughput constraint (19) in Courcoubetis and Dimakis (2012).

Lemma 1. $(\bar{\pi}_n, n \geq 0)$ *dominates* $(\bar{\pi}'_n, n \geq 0)$ *in the convex stochastic order.*

Proof. First note that $\mathrm{sgn}(\bar{\pi}_n - \bar{\pi}'_n) = \mathrm{sgn}(\bar{\pi}_0 - \bar{\pi}'_0)$[1] for all $n < m$ since $y_n = y'_n$ in that range. Similarly $\mathrm{sgn}(\bar{\pi}_n - \bar{\pi}'_n) = \mathrm{sgn}(\bar{\pi}_{m+1} - \bar{\pi}'_{m+1})$ for all $n > m$ is true. Now $\mathrm{sgn}(\bar{\pi}_m - \bar{\pi}'_m) = \mathrm{sgn}(\bar{\pi}_{m-1} y_m^{-1} - \bar{\pi}'_{m-1} {y'_m}^{-1}) \leq \mathrm{sgn}(\bar{\pi}_{m-1} - \bar{\pi}'_{m-1})$, since $y'_m \leq y_m$ which follows from the fact that $(\bar{\pi}_n)$ and $(\bar{\pi}'_n)$ have the same mean (cf. Eq. (19) in Courcoubetis and Dimakis (2012)) and $y'_{m+1} \geq y_{m+1}$. In turn this implies $\mathrm{sgn}(\bar{\pi}_m - \bar{\pi}'_m) \leq \mathrm{sgn}(\bar{\pi}_{m+1} - \bar{\pi}'_{m+1})$. Thus, $\bar{\pi}_n - \bar{\pi}'_n$ can change sign at most twice as n goes from 0 to ∞. It is easy to see that the distributions $(\bar{\pi}_n)$ and $(\bar{\pi}'_n)$ are not stochastically ordered so Theorem 1.A.12 in Shaked and Shanthikumar (2007) implies that $\bar{\pi}_n - \bar{\pi}'_n$ cannot change sign only once. Thus there are exactly two sign changes and so by Theorem 3.A.57, $(\bar{\pi}_n)$ dominates $(\bar{\pi}'_n)$ in the convex stochastic order. □

The Lemma implies $\sum_n n^2 \bar{\pi}'_n < \sum_n n^2 \bar{\pi}_n$ and so $(y_n, n \geq 1)$ is not optimal which contradicts our assumption. Therefore it must be that no such rates $(y'_n, n \geq 1)$ as above exist, i.e., the optimal policy satisfies $y_{n+1} = (n+1)/(\rho(k+n))$ or $y_n/n = y_{n+1}/(n+1)$ for all $n \geq 1$. Notice that if $y_n = n/(\rho(k+n-1))$ then $(n+1)y_n/n = (n+1)/(\rho(k+n-1)) > (n+1)/(\rho(k+n)) \geq y_{n+1}$, and so $y_{n+1} = (n+1)/(\rho(k+n))$ must be true. This proves (10). □

Still, this algorithm is difficult to implement in the case where multiple bottlenecks between the servers and its clients exist. A straightforward implementation, where an independent version of the optimal policy (8) runs on each bottleneck link, requires the clients to be classified according to the bottleneck links they share.

[1] $\mathrm{sgn}(x) = 1$ is the sign function: it takes the values -1,0,1 if $x < 0$, $x = 0$, or $x > 0$ respectively.

Since the identification of bottlenecks is nontrivial and approximate, we do not pursue this approach in favour of a simpler one. In the next section we consider an algorithm that does not need to know about bottleneck links. It is not optimal, but its performance approximates the optimal as the number of clients and their bottleneck links increase.

3 A Water-filling Algorithm

In this section we consider a model of a system with multiple parallel bottleneck links and present a "Water-Filling Algorithm" (WFA) which does not explicitly keep track of links.

3.1 Multiple Bottleneck Model

Consider L parallel links indexed by $l = 1, \ldots, L$, where link l has capacity C_l, it is used by k_l persistent (non-background) TCP flows and an arriving stream of delay-sensitive flows of a finite size under the same distributional assumptions used in Sect. 2. The arrival rate, average size and load is $\lambda_l, 1/\mu_l$ and ρ_l respectively. The amount of background traffic on this link is b_l which we do not assume to be known to the server. Let n_l be the number of delay-sensitive flows on link l, and $\mathbf{n} = (n_l, l = 1, \ldots, L)$. The bandwidth share of each TCP flow on link l is

$$x_{\mathbf{n}}^{l} = \frac{C_l}{n_l + k_l + a_{\mathbf{n}}^{l}} \, , \tag{11}$$

where $a_{\mathbf{n}}^{l}$ is the number of background TCP connections the controller allows (on link l) when in state $\mathbf{n}$.

Obviously, the problem of minimizing the average download delay is decomposable into a set of L independent minimization problems, one for each link. Thus the optimal $x_{\mathbf{n}}^{l}$ is given by (8) using the parameters of the l-th link. Instead, we are going to consider the following policy: for any positive $a > 0$, the $x_{\mathbf{n}}^{l}$ are chosen according to the optimum solution to the problem $\max \sum_l C_l \log x_{\mathbf{n}}^{l}$ such that (11) and $\sum_l a_{\mathbf{n}}^{l} \geq a$ over $a_{\mathbf{n}}^{l} \geq 0, l = 1, \ldots, L$. Since at the optimum $\sum_l a_{\mathbf{n}}^{l} = a$ holds, the algorithm operates such that in each state a total number a of background connections is assigned to the links such that $\sum_l C_l \log x_{\mathbf{n}}^{l}$ is maximized. As time passes and the state evolves, the a connections are continuously reassigned on different links. By considering the KKT conditions it is readily shown that the optimum solution is characterized by the property that there exists a value of throughput $x(a)$ for which $\sum_l a_{\mathbf{n}}^{l} = a$ and the following holds:

$$\text{If } a_l > 0 \text{ then } x_{\mathbf{n}}^{l} = x(a)\,;\ \text{if } \frac{C_l}{n^l + k_l} < x(a) \text{ then } a_l = 0\,. \tag{12}$$

This characterization implies that the optimum solution $(a^l_{\mathbf{n}}, l = 1,\dots,L)$ can be found by a so-called "water-filling" algorithm: start pouring a volume a of water in L tanks of infinite height, unit length and width C_l for the l-th tank. Also, let the bottom of the l-th tank be located at a height $(n^l + k^l)/C_l$ above ground. If tank l holds $a^l_{\mathbf{n}}$ volume of water then the water surface is located at height $(n^l + k_l + a^l_{\mathbf{n}})/C_l$ above ground. If we assume that tanks have no walls and water can freely circulate between them, the tanks with smaller heights start to fill first. When the pouring of water stops the "water level" will be located at some height, say $1/x(a)$, above ground for any tank with $a^l_{\mathbf{n}} > 0$. For empty tanks, their depth raises above water level, i.e., $(n^l + k_l)/C_l > 1/x(a)$ if $a^l_{\mathbf{n}} = 0$. Thus the throughput $x(a)$ corresponds to the inverse of the water level. In the context of a data server, the same water-filling effect can be achieved by the following conceptual continuous reshuffling of TCP connections:

1. Let $a^l_{\mathbf{n}}$ be the initial allocation of TCP connections to link l, which may not be optimal.
2. Start shutting down infinitesimally small fractions ε of connections from links of smaller throughput (which corresponds to a higher water level), i.e., decrease $a^l_{\mathbf{n}}$ by ε, and start using them in any link l' with higher throughput $x^{l'}_{\mathbf{n}} > x^l_{\mathbf{n}}$, i.e., by increasing $a^{l'}_{\mathbf{n}}$ by ε.
3. Repeat the previous step.

The only equilibrium of this procedure is characterized by (12), for if there existed links l, l' with $a^l_{\mathbf{n}}, a^{l'}_{\mathbf{n}} > 0$ and $x^l_{\mathbf{n}} \neq x^{l'}_{\mathbf{n}}$ then TCP connections will "move" from the link of the higher throughput to that with the lower one.

3.2 Asymptotic Optimality of the Water-filling Algorithm

We consider a mean-field limit as the number of links increases. Assume that for each $R \geq 1$ we construct a replica of the set of L links defined in Sect. 3 such that there are R links with capacity C_l, accepting independent arrivals of delay-sensitive flows with parameters λ_l, μ_l, ρ_l for each $l = 1,\dots,L$. Thus each link is indexed by l and the replica index $r = 1,\dots,R$. Let $n^{r,R}_l(t)$ be the number of delay-sensitive flows on the link of type l in the r-th replica. Also define $\mathbf{N}^{\mathbf{R}}_{\mathbf{r}}(t) = (n^{r,R}_l(t), l = 1,\dots,L) \in \mathbb{N}^L$, the state of links of the r-th replica, and $\mathbf{N}^{\mathbf{R}}(t) = (\mathbf{N}^{\mathbf{R}}_{\mathbf{r}}(t), r = 1,\dots,R) \in \mathbb{N}^{RL}$ the system state at time t. Under WFA, $\mathbf{N}^{\mathbf{R}}(t)$ is a Markov chain and we will consider the empirical distribution defined by

$$f^R(t)(A) = \frac{1}{R}\sum_{r=1}^{R} \delta_A\left(\mathbf{N}^{\mathbf{R}}_{\mathbf{r}}(t)\right) , \tag{13}$$

for any $A \subset \mathbb{N}^L$, where δ_A is the unit mass at A.

The RL links are coupled through the bandwidth sharing resulting from WFA for a total number Ra of connections, where $a > 0$ is a constant. Thus as the number of replicas increase, the average number of background connections per link is constant. As we are not interested in the mean-field convergence proof itself, we will assume that the processes converge in a certain strong sense and then proceed to show that their limit satisfies (8).

Theorem 2. *If $f^R(t) \xrightarrow{d} f$ a.s. as $t, R \to \infty$, where f is a nonrandom probability measure on $\mathbb{N}^L$ then the proportion of type l links that are in state $\mathbf{n}$ is given by the stationary distribution under the single link optimal policy* (8) *applied to link type l.*

In other words, as $R \to \infty$ WFA operates each link of type l according to the optimal policy corresponding to l for some threshold n_*, which does not need to coincide with the single link optimal.

Proof. The key is that the link states are coupled only through the water level, which in turn depends on $\mathbf{N^R}(t)$ only through the distribution $f^R(t)$. Let $x(Ra, Rf^R(t))$ be the inverse of the water level at time t, i.e., the level of TCP throughput attained by links with nonzero background flows. We write this as a function of Ra, the total number of background TCP connections allocated by WFA, as well as $Rf^R(t)$ which corresponds to the count of replicas in each replica state at time t. First notice that $x(\cdot,\cdot)$ is homogenous, i.e., $x(MRa, MRf^R(t)) = x(Ra, Rf^R(t))$ for any integer $M \geq 1$, so $x(Ra, Rf^R) = x(a, f^R)$ where we have used the obvious extension for fractional M. Now let $f_i^{l,R}(t)$ be the proportion of type l links that are in state i, i.e., $f_i^{l,R}(t) = f^R(t)(A_i^l)$ for $A_i^l = \mathbb{N}^{l-1} \times \{i\} \times \mathbb{N}^{L-l}$. Then,

$$\frac{\mathrm{d}}{\mathrm{d}t} E f_i^{l,R}(t) = \lambda_l E f_{i-1}^{l,R}(t) + \mu_l (i+1) E \left[f_{i+1}^{l,R}(t) \min \left(\frac{C_l}{i+1+k_l}, x(Ra, Rf^R(t)) \right) \right]$$
$$- E \left\{ f_i^{l,R}(t) \left[\lambda_l + \mu_l i \min \left(\frac{C_l}{i+k_l}, x(Ra, Rf^R(t)) \right) \right] \right\}$$

Taking limits in t, R and using the homogeneity and continuity of $x(\cdot,\cdot)$ yields

$$0 = \lambda_l f_{i-1}^l + \mu_l (i+1) f_{i+1}^l \min \left(\frac{C_l}{i+1+k_l}, x(a,f) \right)$$
$$- f_i^l \left[\lambda_l + \mu_l i \min \left(\frac{C_l}{i+k_l}, x(a,f) \right) \right], \quad (14)$$

for each $i = 0, 1, 2, \ldots$, where $f_i^l = f(A_i^l), f_{-1}^l = 0$.

Notice that this is just the global balance equations characterizing the stationary distribution of the chain (8) for a n_* corresponding to the inverse water level $x(a,f)$, i.e., $x(a,f) = C_l/(n_* + k_l)$. □

In certain cases, WFA not only shares the same structure with the optimal policy but the two coincide.

Corollary 1. *If the links are homogenous, i.e., $L = 1$ then WFA coincides with* (8) *for a such that WFA obtains throughput b_1, i.e., the same throughput as* (8).

Proof. Theorem 2 guarantees that WFA and (8) have the same structure but not necessarily the same threshold. Since they obtain the same throughput, their thresholds must coincide as the throughput in (8) depends monotonically on n^*. □

4 Water-filling Algorithm Implementation

In this section we consider a prototype implementation of WFA in the ns2 simulator and compare its performance with the case where no control is exercised by the server, i.e., every background file transmission request is served by one TCP connection as soon as the request arrives.

As in the ideal WFA, the implementation utilizes at most a number a of TCP connections at any given time. This number is slowly adapted in order to match throughput with offered demand. Although the ideal WFA is able to know exactly the TCP flow throughput over any link l, this is not possible here. Neither the number of links is known. The idea is to use the previous transmissions to estimate the throughput of upcoming transmissions to the same client. Thus each background file is send over multiple transmissions of 1 MB chunks. Each chunk is transmitted over one TCP connection which terminates after chunk's completion. The maximum number of simultaneous TCP connections used for chunk transmissions is controlled by the parameter a. Thus no more than a chunks are transmitted simultaneously and these chunks may belong to different files. For each file we keep track of its pending chunks as well as the throughput obtained by its last transmitted or currently transmitting chunks. Each time a chunk completes, the next transmitted chunk is the one which belongs to the file with the highest throughput. In this way, more TCP connections are allocated on links (actually files) offering higher TCP bandwidth share. Thus a throughput balancing effect across TCP flows on different links arises, similar to ideal WFA.

We next compare the performance of this WFA implementation against the case of a server which does not exercise any control to its flows. In order to separately evaluate the effect of limiting the number of connections to a and the prioritization of links with higher throughput in WFA, we also compare with a First-Come-First-Served (FCFS) policy. This policy limits the number of connections as WFA does, but it does not do any prioritization; all files are served in a FCFS basis. We consider the case of 2, 4, and 8 parallel identical links of capacity $C_l = 10$ Mbps, where each is traversed by a single (nonbackground) persistent TCP flow, i.e., $k_l = 1$ and a delay-sensitive stream of flows with $\lambda_l = 1/4.8$ arrivals/sec and average size $1/\mu_l = 3$ MB. One background file transmission request arrives every 24 sec uniformly distributed across links and according to a Poisson process, where the file size is exponentially distributed with mean 12 MB. In Fig. 1 we depict the average number of delay-sensitive flows in the system versus the average number of background flows. The points in the curves for WFA, FCFS were obtained for different selections of the parameter a. WFA reduces the delay of delay-sensitive flows relative to 'no control' from 21% to 31% depending on the case. Since FCFS lies almost in the middle

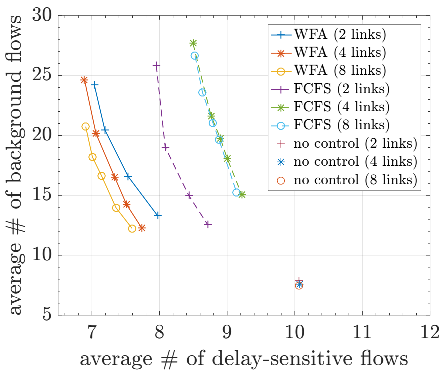

Fig. 1 Average number of delay-sensitive flows versus the average number of background flows in the system. The implementation of WFA brings significant delay reductions compared to 'no control'.

between WFA and 'no control', with respect to delay, we see that half of the delay reductions were due to limiting the number of simultaneous TCP connections (i.e., the *a* parameter) and the rest were due to prioritizing links with higher throughput. The increase of links from 2 to 8 brings a 2-8% delay drop under WFA, depending on the case.

5 Conclusions

In this paper we have shown that a simple background data transfer policy such as limiting the maximum number of active TCP flows transferring data to clients, and prioritizing faster flows yields significant download delay reductions to delay-sensitive TCP flows, compared to exercising no control on client flows. In theory (suggested by Theorem 2) more elaborate data transfer policies in large servers do not bring any significant additional reductions, unless nonstationary policies exploiting past information are used.

References

Cohen B (2003) Incentives build robustness in BitTorrent, available in http://www.bittorrent.org/bittorrentecon.pdf

Courcoubetis C, Dimakis A (2012) Fair background data transfers of minimal delay impact. INFOCOM, 2012 Proceedings IEEE pp 1053–1061

Deb S, Ganesh A, Key P (2005) Resource allocation between persistent and transient flows. IEEE/ACM Trans Netw 13(2):302–315

Key P, Massoulie L, Vojnovic M (2005) Farsighted users harness network time-diversity. In: INFOCOM, 2005 Proceedings IEEE, vol 4, pp 2383–2394

Shai L, Vojnovic M, Gunawardena D (2007) Competitive and considerate congestion control for bulk data. In: 15th IEEE International Workshop on Quality of Service, pp 1–9

Shaked M, Shanthikumar J (2007) Stochastic Orders. Springer-Verlag New York