Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

A Two-Time Scale Design for Detection and Rectification of Uncooperative Network Flows

X. Fan*, K. Chandrayana, M. Arcak, S. Kalyanaraman, J. T. Wen
Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, NY 12180

***Abstract*— Existing Internet protocols rely on cooperative behavior of end users. We present a control-theoretic algorithm to counteract *uncooperative* users which change their congestion control schemes to gain larger bandwidth. This algorithm *rectifies* uncooperative users; that is, forces them to comply with their fair share, by adjusting the prices fed back to them. It is to be implemented at the edge of the network (e.g. by ISPs), and can be used with any congestion notification policy deployed by the network. Our design achieves a separation of time-scales between the network congestion feedback loop and the price-adjustment loop, thus recovering the fair allocation of bandwidth upon a fast transient phase.**

I. INTRODUCTION

In a network which does not differentiate among users, the equilibrium rate for any user is primarily determined by the congestion control being used [1]. With new software advancements, however, "*uncooperative*" users can change their congestion control schemes to gain more than their fair share of bandwidth, at the cost of cooperative users. This uncooperative behavior can lead to TCP unfriendliness, congestion collapse [2], [3] and, to a traffic-based denial-of-service to cooperative users [4], [5]. Detecting uncooperative users, and "*rectifying*" their flow rates to comply with cooperative rates, is thus an important emerging problem in network management.

Among rectification mechanisms proposed in the literature, the majority are "router-based" that is, they modify the router algorithm to detect and limit uncooperative flows, e.g. Active Queue Management (AQM) schemes or scheduling disciplines [2], [3], [6], [7], [8]. More recently, *edge-based* price-adjustment mechanisms have been proposed in [9] and [10], which manage uncooperative flows only at edge routers. A significant advantage of this approach is that it does not require core network upgrades and can be implemented without performing per flow management at routers. By estimating each flow's incoming rate and using it to label flow's packet, the Core-Stateless Fair Queueing (CSFQ) algorithm in [9] computes the forwarding probability from link fair rate estimation. However, this design only applies to network in which all nodes implement Fair Queueing. In [10], the authors manage uncooperative flows by mapping their utility function to a specified target network behavior at the edge. This study, however is restricted to a specific form of TCP.

In this paper, we develop an edge-based price-adjustment algorithm using tools from control theory. Rather than address a specific protocol, we develop our design within the optimization framework of Kelly [1], [11], [12], which is applicable to diverse types of networks, and encompasses numerous protocols such as TCP Reno, TCP Vegas, FAST etc. Our algorithm recovers the cooperative share of bandwidth prescribed in Kelly's framework, with a new feedback loop implemented at the edge router, and, hence, referred to as the "edge supervisor". It detects uncooperative users by comparing their sending rates with "*audit*" rates calculated according to an ideal, cooperative, model, and increases their price feedback. Although in this design edge supervisor does perform per flow management by this price adjustment loop, core routers, which are in general more complex than edge routers, do not perform per flow management, and therefore the implementation complexity is significantly reduced. Our algorithm is independent of congestion notification policy deployed by the network, and thus, can be used with any Active Queue Management scheme.

We design the price adjustment loop to evolve in a faster time-scale than the existing price feedback loop from the links, because, then, uncooperative flows are rectified during a fast transient phase, after which stability and convergence properties of the desired cooperative network model is recovered. Indeed, using singular perturbations tools [13], we prove that the fast and slow feedback loops, when combined, ensure convergence of the sending rates to their cooperative values. The type of convergence established is "semi-global" [13], which means that any desired region of attraction can be achieved by increasing the feedback gain of the price-adjustment loop.

The paper is organized as follows: Section 2 overviews Kelly's primal and dual flow control algorithms. Section 3 studies the primal algorithm and presents our price adjustment design for uncooperative users. Section 4 extends this design to the dual algorithm. In Section 5, we implement our price adjustment algorithms in NS-2 and evaluate their performance for a multi-bottleneck topology. In particular, we show that given a standard network behavior like TCP-Friendliness, our algorithm forces uncooperative users to comply with their fair-share of the bandwidth. Conclusions

*Corresponding author. Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA. Tel.: +1-518-276-8205; fax: +1-518-276-6261.

are given in Section 6. In the paper, some proofs are omitted due to space limitations, please refer to [14] for details.

II. Overview of Kelly's primal and dual flow control algorithms

In Kelly's framework [1], network flows are modeled as the interconnection of users and communication links. Packets from each user (with sending rate x_i) are routed through the links with the aggregate link rate

$$y = R_f x \tag{1}$$

where R_f is the forward routing matrix. Each link j has a fixed capacity c_j, and based on its congestion and queue size, a link price, p_j is computed:

$$p_j = h_j(y_j), \quad j = 1, \cdots, L. \tag{2}$$

The link price information is then sent back to each source with the aggregate source price,

$$q = R_b p. \tag{3}$$

where $R_b = R_f^T$, since the links only feed back price information to the users that utilize them.

Kelly formulated the flow control as the combination of a static optimization and a dynamic stabilization problem. The static optimization problem computes the desired equilibrium by maximizing the sum of the source utility functions $U_i(x_i)$, while complying with capacity constraints in the links:

$$\max_{x \geq 0} \sum_{i=1}^{N} U_i(x_i) \quad \text{subject to } \underbrace{Rx}_{y} \leq c. \tag{4}$$

The dynamic problem is to design the source rate update law based on the aggregate price, and the link price update law based on the aggregate rate, to guarantee stability of the equilibrium. For this problem, Kelly introduced two dynamic algorithms: The *Primal Algorithm* consists of a first order source update law, and a static penalty function for the link to keep the aggregate rate below its capacity:

$$\dot{x}_i = \kappa_i\left(U_i'(x_i) - q_i\right), \quad p_j = h_j(y_j). \tag{5}$$

The penalty functions $h_l(y_l)$ are designed to enforce the link capacity constraints $y_l \leq c_l$, $l = 1, \cdots, L$, i.e., to keep the aggregate rate y_l below its capacity c_l.

The *Dual Algorithm* consists of a static source update and a first order dynamic price update:

$$x_i = U_i'^{-1}(q_i), \dot{p}_j \quad = \gamma_j (y_j - c_j)^+_{p_j}. \tag{6}$$

where the positive projection $(\cdot)^+$ for a general function $f(\cdot)$ is defined as

$$(f(x))^+_x := \begin{cases} f(x) & \text{if } x > 0, \text{ or } x = 0 \text{ and } f(x) \geq 0 \\ 0 & \text{if } x = 0 \text{ and } f(x) < 0. \end{cases}$$

From (6), the unique equilibrium for the dual control law is obtained from the equations

$$q_i^* = U_i'(x_i^*), \quad i = 1, \cdots, N \tag{7}$$

$$p_l^* \begin{cases} = 0 & \text{if } y_l^* \leq c_l \\ \geq 0 & \text{if } y_l^* = c_l \end{cases} \qquad l = 1, \cdots, L, \tag{8}$$

which as shown in [1], correspond to the solution of the optimization problem (4), in which p_l's play the role of Lagrange multipliers for the capacity constraints. For the primal control law (5), the equilibrium obtained from

$$q_i^* = U_i'(x_i^*), \quad i = 1, \cdots, N \tag{9}$$

$$p_l^* = h_l(y_l^*) \quad l = 1, \cdots, L, \tag{10}$$

approximates the optimality condition (7)-(8) with the help of the penalty functions $h_l(y_l)$. The stability of these two algorithms and their extensions has been established in [11], [12], [15], [16].

III. Uncooperative users in Kelly's primal algorithm

We now assume that some users, which we call "uncooperative", use more aggressive utility functions to increase their share of bandwidth; that is, instead of $U_i(x_i)$ in (5), they implement $\tilde{U}_i(x_i)$:

$$\dot{x}_i = \kappa_i\left(\tilde{U}_i'(x_i) - \tilde{q}_i\right). \tag{11}$$

To rectify these uncooperative users, we propose that the supervisor at the edge of the network (e.g., internet service providers) adjust the price feedback from its nominal value q_i to $\tilde{q}_i$. An ideal design of $\tilde{q}_i$ would be

$$\tilde{q}_i = q_i + \tilde{U}_i'(x_i) - U_i'(x_i), \tag{12}$$

which replaces $\tilde{U}_i'(x_i)$ in (11) with the cooperative $U_i'(x_i)$. However, this design is not implementable because $\tilde{U}_i(x_i)$ is not known to the supervisor. Instead, in our design, we obtain an estimate of $\tilde{U}_i(x)$ with the help of the cooperative reference model:

$$\dot{\hat{x}}_i = \kappa_i\left(U_i'(x_i) - q_i\right), \quad \hat{x}_i(0) = x_i(0). \tag{13}$$

The $\hat{x}_i$ thus calculated differs from x_i by $e_i := \hat{x}_i - x_i$, which, from (11)-(13), is governed by

$$\dot{e}_i = \kappa_i\left(\tilde{q}_i - q_i - \tilde{U}_i'(x_i) + U_i'(x_i)\right). \tag{14}$$

This means that, if we design the price adjustment to be

$$\tilde{q}_i = q_i - \rho_i e_i, \tag{15}$$

with a sufficiently high gain $\rho_i > 0$, then the variable e_i evolves in a faster time scale than x_i, and reaches the quasi-steady state $\rho_i e_i \approx -\tilde{U}_i'(x_i) + U_i'(x_i)$. Thus, after a fast transient, our design (13), (15) approximates the non-implementable scheme (12). For cooperative users, where $\tilde{U}_i(x_i) = U_i(x_i)$, (13) and (15) yield $\tilde{q}_i = q_i$, which means that no price adjustment is applied. Note that U_i in (13) is not necessarily the same for each user. This means that the supervisor can intentionally set up different utility functions, and use this flexibility to only adjust high bandwidth flows while leaving low bandwidth flows without rectification.

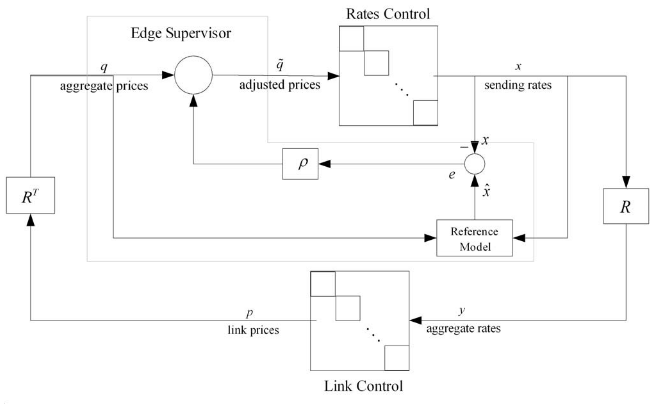


Fig. 1. Price adjustment in Kelly's primal algorithm.

The algorithm (13), (15) is depicted with a block diagram in Figure 2. In Theorem 1 below, we use tools from singular-perturbations theory [13] to prove that (13), (15) achieves asymptotic stability of the cooperative value x^* in (9)-(10):

Theorem 1: Consider the network (1)-(3), where some users implement the uncooperative algorithm (11), rather than (5). Suppose $U_i(x_i) : \mathrm{R}_+ \to \mathrm{R}$ are increasing and sufficiently smooth functions, $U_i''(x_i) < 0 \quad \forall x_i \in \mathrm{R}_+$, and $U_i(x_i) \to -\infty$ and $\tilde{U}_i(x_i) \to -\infty$ as $x_i \to 0$ for $i = 1, \cdots, N$. Then, the price adjustment algorithm (13), (15) ensures that, for any compact set $\Omega \subset \mathrm{R}_+^N$ of initial conditions $x(0)$, there exists $\rho_i^* > 0$ such that, if $\rho_i > \rho_i^*$, then $x(t)$ and $\hat{x}(t)$ remain bounded, and $x(t)$ converges to the cooperative value x^* in (9)-(10).

The assumptions of Theorem 1 on the utility functions $U_i(x_i)$ are standard in the literature [1], [18]. In particular, the assumption $U_i(x_i) \to -\infty$ as $x_i \to 0$ ensures that R_+^N is positively-invariant, i.e., if x is initially in R_+^N, it will remain in R_+^N for all $t \geq 0$. It is satisfied by commonly used utility functions such as $U_i(x_i) = -\frac{a_i}{x_i}$ (variant of TCP Reno) and $U_i(x_i) = a_i \log x_i$ (TCP Vegas). For others, such as $U_i(x_i) = \frac{\sqrt{2}}{\tau_i}\tan^{-1}\left(\frac{\tau_i x_i}{\sqrt{2}}\right)$ (TCP Reno), we can modify Theorem 1 and prove stability by using positive projection functions as in [12]. It is reasonable to make the same assumptions for $\tilde{U}_i'(\cdot)$ as for $U_i'(\cdot)$, because cheating users would typically change the parameters of the nominal utility functions, such as a_i in TCP Vegas above. However, this assumption excludes some traditional unresponsive flows referred to as UDP or CBR, where users send data at a constant rate without acknowledging any feedback. □

Proof: To represent the algorithm (11), (13) and (15) in the standard singularly perturbed form [13], we let

$$\omega_i := \rho_i e_i \tag{16}$$

$$\varepsilon_i = \frac{1}{\rho_i} \tag{17}$$

and obtain:

$$\dot{x}_i = \kappa_i\left(\tilde{U}_i'(x_i) - q_i + \omega_i\right). \tag{18}$$

$$\varepsilon_i\dot{\omega}_i = -\kappa_i\left(\omega_i + \tilde{U}_i'(x_i) - U_i'(x_i)\right). \tag{19}$$

An inspection of (18) and (19) shows that the equilibrium for x_i is same as the cooperative x_i^* in (9)-(10), and the equilibrium for ω_i is

$$\omega_i^* = -\tilde{U}_i'(x_i^*) + U_i'(x_i^*). \tag{20}$$

To shift this equilibrium to 0, we define

$$\varpi_i := \omega_i + \tilde{U}_i'(x_i) - U_i'(x_i) \tag{21}$$

and rewrite (18)- (19) as

$$\begin{aligned}\dot{x} &= K\left(U'(x) - R^T h(Rx) + \varpi\right)\\ \varepsilon\dot{\varpi} &= -K\left(\varpi - \varepsilon\frac{\partial\left(\tilde{U}'(x) - U'(x)\right)}{\partial x}\left(U'(x) - R^T h(Rx) + \varpi\right)\right)\end{aligned} \tag{22}$$

where we use the vector notation $x = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix}^T$, $\varpi = \begin{bmatrix} \varpi_1 & \varpi_2 & \cdots & \varpi_N \end{bmatrix}^T$. $K = \mathrm{diag}\{\kappa_i\}$ and $\varepsilon = \mathrm{diag}\{\varepsilon_i\}$ are diagonal matrixes of the source controller gains $\kappa_i > 0$ and $\varepsilon_i > 0$, $i = 1, \cdots, N$, and $U'(x) \in \mathrm{R}^N$ is a vector whose ith component is the derivative $U_i'(x_i)$ of the utility function $U_i(x_i)$. Likewise, $h(y) \in \mathrm{R}^L$ and $\tilde{U}'(x) \in \mathrm{R}^N$ consist of the penalty functions $h_l(y_l)$ and uncooperative utility functions $\tilde{U}_i'(x_i)$.

To prove asymptotic stability of $(x, \varpi) = (x^*, 0)$ we use the Lyapunov function

$$\begin{aligned}V = &\sum_{i=1}^{N}\left(-\left(U_i(x_i) - U_i(x_i^*)\right) + q_i^*(x_i - x_i^*)\right)\\ &+ \sum_{l=1}^{L}\left(\int_{y_l^*}^{y_l}\left(h_l(\sigma) - h_l(y_l^*)\right)d\sigma\right) + \frac{1}{2}\varpi^T K^{-1}\varpi\end{aligned} \tag{23}$$

in which, the first and the second terms, are identical to the Lyapunov function used in [1], [12] for the proof of the stability of Kelly's Primal algorithm, while the third term is a quadratic Lyapunov function for the dynamics of ϖ subsystem. This Lyapunov function is positive definite and radially unbounded in R_+^N, and yields the derivative

$$\begin{aligned}\dot{V} \leq &- f_1(x)^T K f_1(x) - \varpi^T\varepsilon^{-1}\varpi\\ &+ \varpi^T\frac{\partial\left(\tilde{U}'(x) - U'(x)\right)}{\partial x}\varpi + \varpi^T f_2(x),\end{aligned} \tag{24}$$

where

$$f_1(x) := U'(x) - R^T h(Rx), \tag{25}$$

$$\begin{aligned}f_2(x) := &\frac{\partial\left(\tilde{U}'(x) - U'(x)\right)}{\partial x}\\ &\left(\left(U'(x) - R^T h(Rx)\right) + K\left(-U'(x) + R^T h(Rx)\right)\right).\end{aligned} \tag{26}$$

We show in Lemma 1 below that, on any compact set of (x, ϖ) that includes $(x^*, 0)$, we can choose ε small enough to ensure $\dot{V}$ is negative definite. The conclusion of Theorem 1 follows from this lemma because, from $\hat{x}(0) = x(0)$, we have $\omega(0) = 0$ and, thus $\varpi(0) = -\tilde{U}'(x(0)) + U'(x(0))$, which means that for any set Ω as in the statement of the theorem, we can find a corresponding region of attraction in (x, ϖ) coordinates, which does not depend on ε. Since V is also independent of ε, we can select a level set of V that encompasses this region of attraction, and design ε from Lemma 1 to render $\dot{V}$ negative definite in this level set. □

Lemma 1: Let the assumptions of Theorem 1 hold, and let $f_1(x)$ and $f_2(x)$ be defined as in (25)-(26). Then, for any compact set Λ of (x, ϖ) that includes $(x^*, 0)$, there exists $\varepsilon^* > 0$ such that if $\varepsilon_i \in (0, \varepsilon^*]$ for all $i = 1, \cdots, N$, then $\dot{V}(x)$ given in (24) is negative definite on Λ.

In Theorem 1, we require that the edge supervisor set $\hat{x}(0)$ equal to $x(0)$. However, it is not difficult to show that the proof holds true for small errors between $\hat{x}(0)$ and $x(0)$.

IV. Price adjustment for Kelly's dual algorithm

We next study Kelly's dual algorithm where uncooperative users implement, instead of (6),

$$x_i = \tilde{U}_i'^{-1}(\tilde{q}_i). \tag{27}$$

We assume $\tilde{U}_i'^{-1}(s) \geq U_i'^{-1}(s)$, $\forall s \geq 0$, which means that the uncooperative sending rate is larger than the cooperative rate. To counteract such uncooperative users, the supervisor must replace the nominal price feedback q_i with

$$\tilde{q}_i = \tilde{U}_i' \circ U_i'^{-1}(q_i), \tag{28}$$

which, when substituted in (27), results in the cooperative rate (6). Because a direct solution of (28) would require the knowledge of $\tilde{U}_i'(\cdot)$, which is not available to the supervisor, we propose the dynamic algorithm

$$\tilde{q}_i = q_i + \omega_i, \tag{29}$$

$$\dot{\omega}_i = \rho_i \left(x_i - U_i'^{-1}(q_i)\right), \quad \omega_i(0) = 0, \quad \rho_i > 0, \tag{30}$$

depicted in Figure 3. The equilibrium of (30) is achieved when

$$x_i = U_i'^{-1}(q_i), \tag{31}$$

which indeed coincides with the cooperative rate (6). We achieve asymptotic stability of this equilibrium, again, by designing the adaptation gain ρ_i to be sufficiently high:

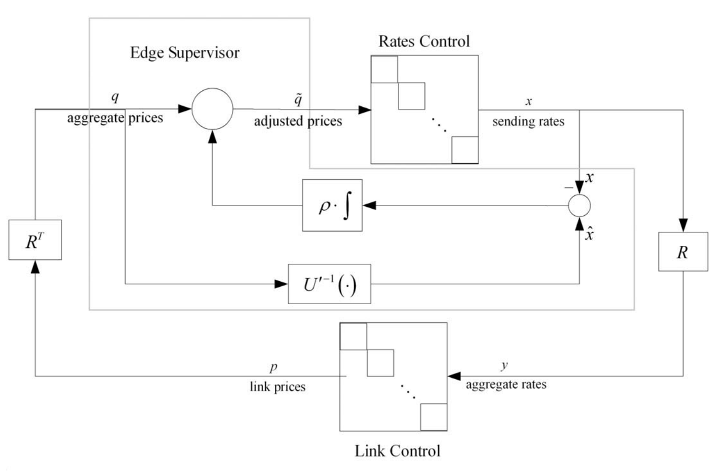


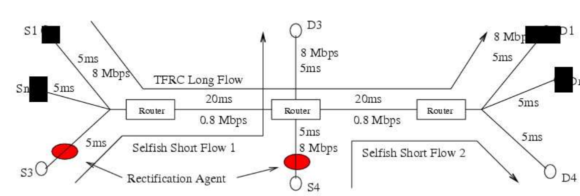
Fig. 2. Price adjustment for uncooperative users in Kelly's dual algorithm.

Theorem 2: Consider the network (1)-(3), (6) and (27), where $U_i(x_i)$ and $\tilde{U}_i(x_i)$ are as in Theorem 1, and $\tilde{U}_i'^{-1}(s) \geq U_i'^{-1}(s)$, $\forall s \in \mathrm{R}_+$. Then, the price adjustment algorithm (27), (30), ensures that, for any compact set $\Omega \subset \mathrm{R}_+^N$ of initial conditions $p(0)$, there exists $\rho_i^* > 0$ such that, if $\rho_i > \rho_i^*$, then $p(t)$, $x(t)$ and $\tilde{q}(t)$ remain bounded, and $x(t)$ and $p(t)$ converge to the cooperative values x^* and p^* in (7)-(8).

V. Implementation and Simulations

We have implemented the uncooperative framework presented in this paper in the Network Simulator (NS-2). While we have studied both dynamic (Section III) and static (Section IV) users, in simulations we implement the method of Section III because of the prevalence of TCP, which is dynamic and can be modeled as in (11) (see [1]). We added an edge-based supervisor, which adjusts the price feedback according to (15). The implementation of this feedback adjustment depends upon the congestion notification policy deployed in the network. We note that, unlike the static link assumption in Section III, AQM and Drop-Tail in simulations make use of queue length and, hence, are dynamic algorithms. An extension of the proof of dynamic-source dynamic-link algorithms would be possible, but lengthy. The stability properties observed in simulations are indeed consistent with those predicted by Theorem 1.

Due to space limitations, we only present part of our simulation results for multi-bottleneck topologies, depicted in Figure 3. All the access links are configured to have a capacity equal to four times that of bottleneck links. The bottleneck links capacity and delay is fixed at 0.8Mbps and 20ms respectively unless specifically stated. For all simulations reported in this paper, rate (or throughput) measurements are taken every 0.5 seconds. Each router has a buffer equal to one bandwidth delay product. In setups where the bottleneck routers have Random Early Drop (RED) buffer management policy deployed, the corresponding maximum and minimum threshold are set at $0.8 \times B$ and $0.3 \times B$ where B is the total buffer length; the queue weight was set to 0.002 and the maximum dropping probability to 0.1. In the topology, the flow between source S1 and destination



(b) Multi-Bottleneck

Fig. 3. Topologies used in simulations.

D1 is referred to as a *long flow*, while the flows [S2-D2] and [S3-D3] are referred to as *short flows*.

Since almost 90% of the traffic carried on the Internet uses TCP, we chose TCP-Friendliness as our definition of conformant flows. For those flows, which under same operating conditions, get more rate than TCP, we refer them as selfish flows. In this paper all transport protocols are rate based. Thus, all TCP-Friendly schemes use equation based rate control scheme (TCP Friendly Rate Control - TFRC) presented in [20] and all selfish schemes are variants of TFRC which have conservative decrease algorithms, i.e. upon congestion they decrease more slowly than TCP.

In the simulation, RED is deployed on the routers, the TCP-Friendly long-flow with $U(x) = -1/x$, competes for bandwidth against the two uncooperative short-flows. Figure 4 a) shows the result where both the long and short flows use TCP-Friendly rate control scheme. Figure 4 b) shows the result for the setup where we replace the TCP-Friendly short flows with uncooperative rate control schemes ($U(x) = -1/\sqrt{x}$). We see that, the uncooperative flows almost force a traffic volume based denial of service attack. When we employ our edge-based supervisor, with $\rho = 2.5 \times 10^{-5}$, we recover the ideal bandwidth sharing of bottleneck links, as shown in Figure 4 a). The value of ρ is comparatively large, because dropping or marking probability is less than 1 and so is the price adjustment $\rho \times e$.

A. Higher Flow Multiplexing with Background Traffic and Reverse Path Congestion

To further present the efficiency and the robustness of our scheme we increase the number of competing flows, and add HTTP sources to the persistent flows and also short TCP-Friendly flows to the reverse paths. The capacity of the bottleneck links is set to 8Mbps and that of access links to 80Mbps. The bottleneck buffer is set to one bandwidth delay product. Figure 5 shows the results for the scenario where 5 TFRC flows compete for bandwidth against selfish flows. On each bottleneck there were 5 selfish short flows. To these persistent flows, we added short web transfers which occupied 10% of the bottleneck bandwidth. On each bottleneck in the reverse path there were 5 flows and thus creating congestion on the reverse path. Figure 5 shows the throughput of one flow from each group: TFRC Long flows, selfish short flows from Group 1 which go over the first bottleneck only; and, finally, the selfish short flows from Group 2 which go over the last bottleneck only.

Figure 5 a) shows the ideal sharing of the bottleneck when the short flows are also TCP-Friendly. Figure 5 b) shows that in the absence of any policing the uncooperative flows get more share of the bandwidth at the expense of TFRC flows. With our rectification algorithm (ρ as 2.5×10^{-5}) the fair share of the TFRC flows is restored; see Figure 5 c).

B. Effect of Gain ρ on Rectification of Selfish Users

The performance of our edge-based rectification algorithm depends on the gain ρ in equation (15). As detailed below, simulation studies indicate that too small or too large values of this ρ may deteriorate the performance. Indeed, Theorem 1 disallows small values of ρ because, otherwise, the desired two-time-scale behavior is not achieved. Although Theorem 1 allows arbitrarily large values for ρ, in practice, such high-gain leads to saturation of dropping or marking schemes, which violate the "small marking probability" approximation in Equation (3). We see in the following simulations that large ρ might result in "*over-penalization*", which means that uncooperative flows receive even less than their fair share.

In Figure 4 we presented simulations with $\rho = 2.5 \times 10^{-5}$. In Figure 6 we compare this result with $\rho = 10^{-5}$ (Figure 6 a)) and with $\rho = 10^{-4}$ (Figure 6 c)). We note that a high value of ρ may result in "*over-penalization*", and uncooperative flows may receive even less than their fair share. Similarly, with a very small value of ρ the selfish users are not sufficiently penalized and they continue to get more share of the bottleneck link(s) at the expense of cooperative users. However, for intermediate values, such as $\rho = 2.5 \times 10^{-5}$ in Figure 6 b), we recover the ideal shares for the uncooperative and the cooperative users.

For all the results reported in this paper we have found that the ideal range of ρ lies between the interval 10^{-4} to 10^{-5}. We also extensively evaluated the edge-based rectification model for different value of selfishness, i.e. users chose different values of $U(x)$, and found observation on ρ consistent with those reported above.

VI. CONCLUSIONS

We have presented a price adjustment algorithm for both Kelly's primal and dual network flow control models, and tested it on the Network Simulator. This algorithm is to be implemented at the edge of the network and, thus, does not require costly hardware upgrades in the entire network. It is independent of congestion notification policy deployed by the network, and thus, can be used with any Active Queue Management scheme, as well as Drop Tail queueing. Although a suitable range for the gain ρ in our algorithm was determined by simulations, a judicious choice of this gain deserves further investigation. An on-line adaptation for ρ may be possible, and is being investigated by the authors.

REFERENCES

[1] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability, *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[2] A. Akella, S. Seshan, R. Karp, S. Shenker and C. Papadimitriou. Selfish Behavior and stability of the Internet: A Game Theoretic Analysis of TCP. *Proceedings of ACM Sigcomm*, Aug 2002.

[3] S. Floyd and K. Fall. Promoting the Use of End-to-end Congestion Control in the Internet. *IEEE/ACM Transactions on Networking*, 7(4):458-472, 1999.

[4] A. Kuzmanovic and E. Knightly. Low-Rate TCP-Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants). *Proceedings of ACM SIGCOMM*, Aug 2003.

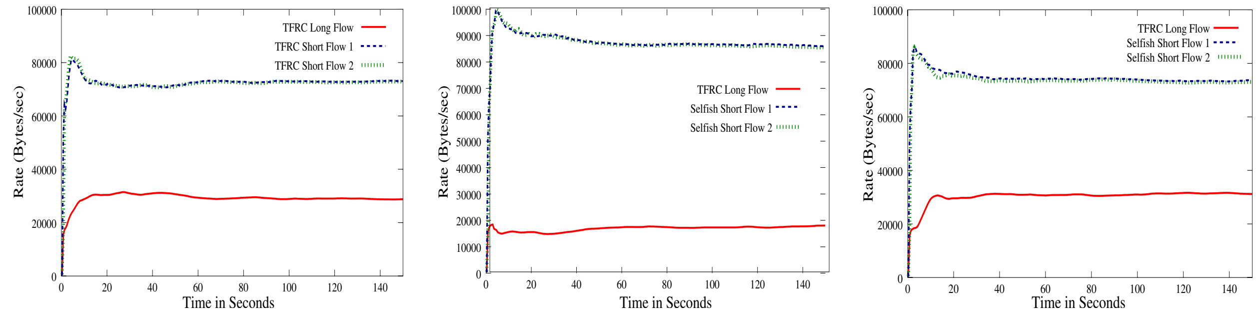
[5] S. Gorinsky, S. Jain, H. Vin and Y. Zhang. Robustness to Inflated Subscription in Multicast Congestion Control. *Proceedings of ACM SIGCOMM*, Aug 2003.

[6] D. Lin and R. Morris. Dynamics of Random Early Detection, *Proceedings of ACM SIGCOMM*, Augutst, 1997.

[7] W. Feng et. al. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness. *Proceedings of INFOCOM*, April 2001.

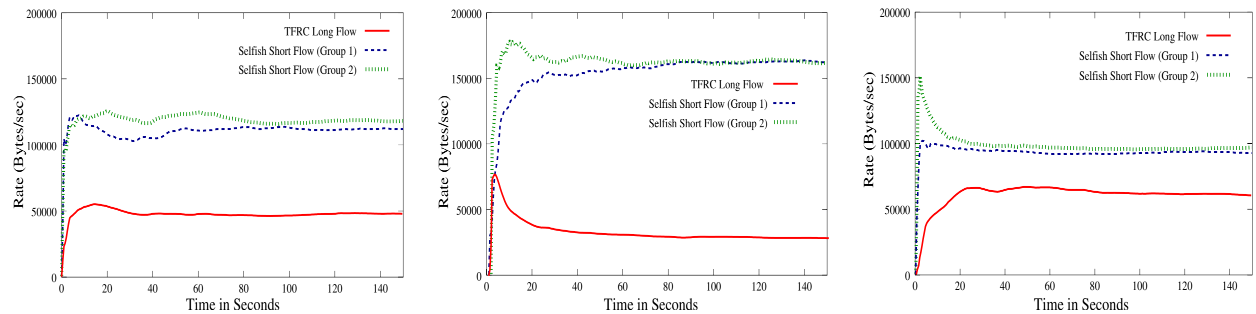
[8] R. Mahajan and S. Floyd. Controlling High-Bandwidth Flows at the Congested Routers. In ICNP 2001.

[9] I. Stoica, S. Shenker and Hui Zhang.Core-Stateless Fair Queueing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks. *SIGCOMM'98*.



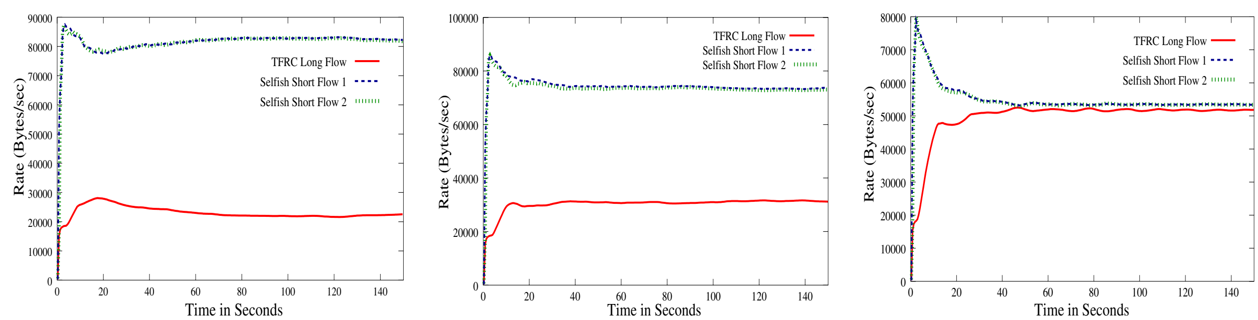
(a) Ideal Sharing (b) Without Rectification (c) With Rectification

Fig. 4. Multi-Bottleneck scenario where (a) shows the ideal bandwidth sharing (b) shows the aggravated unfair sharing in the presence of uncooperative flows and (c) shows the rectification of uncooperative flows with our edge supervisor.



(a) Ideal Sharing (b) Without Rectification (c) With Rectification

Fig. 5. Higher flow multiplexing with background traffic and reverse path congestion in a multi-bottleneck setup.



a) $\rho = 10^{-5}$ b) $\rho = 2.5 \times 10^{-5}$ c) $\rho = 10^{-4}$

Fig. 6. Effect of gain ρ on the steady state rates of uncooperative and TFRC flows with our edge supervisor.

[10] K. Chandrayana and S. Kalyanaraman. Uncooperative Congestion Control. *Proceddings of ACM SIGMETRICS 2004*.

[11] S. Low and D. Lapsley, Optimization flow control - I: basic algorithm and convergence, *IEEE/ACM Transaction on Networking*, vol. 7, no. 6, pp. 861-874, 1999.

[12] J. Wen and M. Arcak, A unifying passivity framework for network flow control, *IEEE Transactions on Automatic Control,* vol. 49, no. 2, pp. 162–174, 2004.

[13] H.K. Khalil. Nonlinear Systems. Prentice Hall, Englewood Cliffs, NJ, third edition, 2002.

[14] X.Fan, K. Chandrayana, M. Arcak, Shiv Kalyanaraman and J. T. Wen, A Two-Time-Scale Design for Edge-Based Detection and Rectification of Uncooperative Flows, Submitted to IEEE/ACM Transactions on Networking, 2005.

[15] F. Paganini, A global stability result in network flow control, *Systems and Control Letters,* vol. 46, pp. 165-172, 2002.

[16] S. Deb and R. Srikant, Global stability of congestion controllers for the Internet, University of Illinois, Urbana, IL, Internal Report, Feb. 2002.

[17] On the Stability of End-to-end Congestion Control for the Internet. Univ. of Cambridge Tech Report CUED/F-INFENG/TR.398, December 2000.

[18] R. Srikant. The Mathematics of Internet Congestion Control. Birkhauser, 2004.

[19] G. H. Hardy, J. E. Littlewood and G. Polya. Inequalities. Cambridge University Press, second edition, 1988.

[20] S. Floyd, M. Handley, J. Padhye and J. Widmer. Equation-Based Congestion Control for Unicast Applications. *Proceedings of ACM SIGCOMM 2000*, Stockholm, Sweden, August 2000.