Value-Aware Resource Allocation for Service Guarantees in Networks

Parimal Parag, *Student Member, IEEE*, Sankalp Sah,
Srinivas Shakkottai, *Member, IEEE*, Jean-Francois Chamberland, *Senior Member, IEEE*

***Abstract*—The traditional formulation of the total value of information transfer is a multi-commodity flow problem. Each data source is seen as generating a commodity along a fixed route, and the objective is to maximize the total system throughput under some concept of fairness, subject to capacity constraints of the links used. This problem is well studied under the framework of network utility maximization and has led to several different distributed congestion control schemes. However, this view of value does not capture the fact that flows may associate value, not just with throughput, but with link-quality metrics such as packet delay and jitter. In this work, the congestion control problem is redefined to include individual source preferences. It is assumed that degradation in link quality seen by a flow adds up on the links it traverses, and the total utility is maximized in such a way that the end-to-end quality degradation seen by each source is bounded by a value that it declares. Decoupling source-dissatisfaction and link-degradation through an effective capacity variable, a distributed and provably optimal resource allocation algorithm is designed to maximize system utility subject to these quality constraints. The applicability of the controller in different situations is supported by numerical simulations, and a protocol developed using the controller is simulated on ns-2 to illustrate its performance.**

***Index Terms*—Utility maximization, quality of service (QoS), distributed algorithm, resource allocation, congestion control.**

I. Introduction

RECENT years have seen an enormous growth in demand for Internet access, with applications ranging from personal use to commercial and military operations. Several of these applications are sensitive to a quality of packet delivery. For instance, archiving data transfers can tolerate long delays, while voice over Internet protocol (VoIP) is very sensitive to latency. Between these extreme examples lies a spectrum of applications with varying service requirements, e.g. video conferencing, electronic commerce and online gaming. All these applications require the allocation of enough network resources for satisfactory performance.

The design of efficient network control systems demands that end-user value be taken into consideration when allocating resources. The Internet architecture is built around the concept of a *flow*, which is a transfer of data between a fixed source-destination pair. How do we quantify the value of such a flow? The classical formulation of the total value of information transfer is a multi-commodity flow problem, in which each data source is seen as generating a commodity along a fixed route; the objective is to maximize the sum throughput under some concept of fairness, subject to capacity constraints on the links used [2]–[5]. If the flow from source r has a rate $x_r \geq 0$ and the system utility associated with such a flow is represented by a concave, increasing function $U_r(x_r)$, the objective can be stated as

$$\max \sum_{r \in \mathcal{S}} U_r(x_r) \quad \text{subject to} \quad y_l \leq c_l,\ \forall\ l \in \mathcal{L} \qquad (1)$$

where $\mathcal{S}$ is the set of sources, $\mathcal{L}$ denotes the set of links, and c_l is the capacity of link $l \in \mathcal{L}$. Also, the load on link l is equal to $y_l = \sum_{r \in \mathcal{S}} R_{lr} x_r$ where R denotes the routing matrix of the network, with $R_{lr} = 1$ if the flow associated with source r is routed through link l. Note that we refer to flows and sources interchangeably; if there are multiple flows between a source and a destination, we simply give them different names. This is a convex optimization problem that is well studied under the framework of network utility maximization [2]–[5].

This approach to network resource allocation can often be used to decompose the problem into several subproblems, each of which is amenable to a distributed solution. This so-called *optimization decomposition framework* has yielded a rich set of control schemes and protocols whose architectural implications are discussed in [6]. For example, there is a strong connection between the *primal solution* to the utility maximization problem and TCP-Reno [7], [8] characterized in [2], [3]. Similarly, one can identify connections between TCP-Vegas and the *dual solution* of the problem [9]. The same approach has been taken in the design of several new protocols such as Scalable TCP [10], [11] (which allows scaling of rate increases/decreases based on network characteristics), FAST-TCP [12] (meant for high bandwidth environments), TCP-Illinois [13] (which uses loss and delay signals to attain high throughput), and TRUMP [14] (a multipath protocol with fast convergence properties). A good tutorial on network utility maximization algorithms is [15].

Still, there is a growing realization that throughput cannot be considered as the sole value metric. As mentioned above, in applications such as voice calls, data is rendered useless after a certain delay threshold. Thus, simply ensuring that link capacities are not exceeded is not sufficient to capture value in this scenario. How do we ensure that the user is not dissatisfied with the quality of service? In many cases the quality of data transfer over a link decreases with load. For example, metrics

Manuscript received 1 July 2010; revised 23 December 2010. Research supported in part by NSF grants CNS-0904520, CCF-0747363, DTRA grant HDTRA1-09-1-0051, and Qatar Telecom, Doha, Qatar. Initial results were presented in [1]. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the authors and do not necessarily reflect NSF's, DTRA's or Qtel's views.

P. Parag, S. Shakkottai and J.-F. Chamberland are with the Dept. of Electrical and Computer Engineering at Texas A&M University (email: {parimal, sshakkot, chmbrlnd}@tamu.edu).

S. Sah is with Ericsson Inc., San Jose, CA (email: sankalp.sah@ericsson.com).

Digital Object Identifier 10.1109/JSAC.2011.110506.

such as the delay and the jitter experienced by packets as they pass through a router depend on the total load on the corresponding links. Such quality degradation may also add up over multiple hops. Indeed, the delay experienced by packets in a flow is the sum of the delays over each hop taken.

Once we have a clear conception of quality degradation as a function of link load, a pertinent question becomes: *Can we design a simple distributed algorithm for fair resource allocation under which each users' quality is no worse than a prescribed value?* To address this question, we need to redefine the traditional congestion control problem to explicitly account for the tradeoff between throughput and quality degradation. We denote the degradation in the quality of link l with load y_l by a convex increasing function $V_l(y_l)$, and assume that degradation in link quality seen by a flow is additive over the links it traverses. Furthermore, we assume that the quality degradation is inherent to a link, and is identical for all flows sharing the link. Thus, there are no priorities assigned to particular flows. We maximize utility in such a way that the total degradation seen by source r is no greater than a pre-specified positive value σ_r. The modified objective becomes

$$\max \sum_{r\in\mathcal{S}} U_r(x_r) \quad \text{subject to} \quad \sum_{l\in\mathcal{L}} R_{lr}V_l\left(y_l\right) \le \sigma_r, \qquad (2)$$

where, again, $x_r \ge 0$, $y = Rx$ and under the assumption that $\lim_{y\to c_l} V_l(y) = \infty$. We emphasize that this convex optimization problem requires the quality degradation on each route to remain bounded. In this article, our objective is to design a distributed control scheme that can approach the optimal operating point which trades off throughput and quality, without maintaining per-flow information or prioritizing certain packets at intermediate hops. We overview our main contributions below, with details contained in subsequent sections.

Classical optimization-decomposition techniques typically yield a "source-rate responds to link-price" type of controller [2]–[6], wherein each link's price increases with the link-load in order to prevent the link-capacity from being exceeded. As the link-price increases, sources cut down their transmission rates, where the aggressiveness of the source controller is determined by its utility function. However, the solution to our delay-aware problem has remained elusive due to the strong coupling between the quality seen at source, which requires a hard guarantee, and *link quality degradation*, which depends on the link-loads along its route.

We present illustrative examples of what quality degradation functions may look like in Section II, and discuss an example that we use later in the paper. We then proceed to provide a centralized solution to our problem of interest in Section III. We develop two algorithms to this end. A primal algorithm is proposed in Section IV. Our main contribution, a dual algorithm, is presented in Section V, and stems from the realization that it is possible to decouple the QoS guarantees at sources and link quality degradation using an *effective capacity* that is based on the link-price and user-dissatisfaction. Once we set the effective capacity for a link, the quality degradation depends solely on this decision and not on the actual link-load.

Each source declares its *dissatisfaction* to the links it uses based on the difference between the quality it sees and what it requires. The links select a *price* based on the difference between load and *effective capacity*, which in turn depends on link-load and aggregate dissatisfaction of users sharing the link. This decoupling of link-load and effective capacity appears to have the correct properties to allow a distributed solution. Finally, sources employ route-prices (the sum of all link-prices on a route) to determine their source-rates.

We prove that the algorithm is indeed capable of solving our resource allocation problem using Lyapunov techniques [16]. We report numerical results about the simulated operation of the controller in Section VI to illustrate its performance. A new contribution over our earlier version [1] is the development of a realistic protocol based on the proposed controller. The protocol is presented in Section VII, where we report simulation experiments on ns-2 to show that it performs as desired. We conclude with pointers to future work in Section VIII.

II. Examples of Quality Degradation Functions

We begin this section by discussing candidate measures of link quality degradation with load. We make several assumptions on the properties of link quality degradation functions. They can be expressed in the following manner; if the total sum-rate on link l is $y_l = \sum_{r\in\mathcal{S}} R_{lr}x_r$ then

- the quality degradation function $V_l(y_l)$ is non-negative, convex increasing in link-load y_l,
- the total quality degradation seen by flow r is $\beta_r(x) = \sum_{l\in\mathcal{L}} R_{lr}V_l(y_l)$ (i.e., quality degradation adds up over multiple hops), and
- the service process at one link does not impact the arrival process at the succeeding link.

The above assumptions ensure the analytical tractability of our optimization problem. We also believe that they provide acceptable models of quality degradations in communication systems with queues. Below, we support these assumptions with common examples of quality degradation functions.

For an M/M/1 queue with arrival rate y and service rate c, the expected waiting time in the queue is $y/(c\,(c-y))$ for a stable queue, that is when $y < c$. In this case, one can select the quality degradation function to be the expected waiting time for any packet in the queue,

$$V(y) = y/(c\,(c-y)).$$

We note that the quality degradation function is non-negative, convex, and increases from 0 to ∞ when x ranges in $[0, c)$.

As a second example, consider a single server fluid queue with constant-rate arrival y and a two-state on-off service process where on and off times are exponentially distributed with rates μ and λ, respectively. When the service is on and the buffer is non-empty, it is serviced at a constant rate r such that $y < r\lambda/(\lambda+\mu)$. It can be shown [17] that the probability of buffer exceeding a threshold z is exponentially decreasing as z increases. A possible quality degradation function in this case is the inverse of this decay-rate. One can write the decay-rate explicitly in terms of the above parameters as

$$1/V(y) = -\lim_{z\to\infty} z^{-1}\log \Pr\left(L > z\right) = \lambda/y - \mu/(r-y).$$

If we denote $r\lambda/(\lambda+\mu)$ by c then, one can write

$$V(y) = y(c-y)^{-1}\left(c\lambda^{-1} - y(\mu+\lambda)^{-1}\right).$$

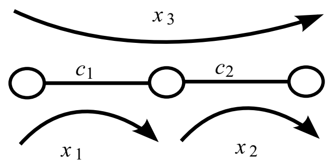


Fig. 1. Three flows sharing a two-link network.

Again, note that the quality degradation function is non-negative, convex, and increases from 0 to ∞ when x ranges in $[0, c)$. Recent results [18] suggest that, under appropriate conditions, the delay seen in a queue is independent of other queues even though packets traverse the network along connected paths.

III. Centralized Resource Allocation

We begin by developing ideas on how to solve the resource allocation problem of (2) in a centralized fashion, and we create model networks that we will use as examples to illustrate the performance of various control loops throughout. Consider the scenario where the utility functions assume unbounded negative values when $x_r = 0$ and the quality degradation functions grow unbounded when sum-rates y_l approach c_l. In this case, an optimal solution is characterized by $x_r > 0$ and $y_l < c_l$. Let $x^* = \{x_r^* : r \in \mathcal{S}\}$ be a feasible point such that $y^* = Rx^*$, and suppose there exist constants $w_r \geq 0$ such that

$$\begin{aligned} U_r'(x_r^*) - \sum_{s \in \mathcal{S}} w_s \sum_{l \in \mathcal{L}} R_{ls} R_{lr} V_l'\left(y_l^*\right) = 0, \\ w_r\left(\beta_r(x^*) - \sigma_r\right) = 0, \end{aligned} \tag{3}$$

for all $r \in \mathcal{S}$, then x^* is a global maximum. Moreover, if $U_r(\cdot)$ is strictly concave, then x^* is the unique global maximum.

We illustrate by the following example how our model takes into consideration all of the desired properties of the quality degradation function and how they impact resource allocation with service guarantees.

Example [Three-Flows Two-Hop Network] Consider a network composed of three sources transmitting over two links, as shown on Fig. 1. Let link i have capacity c_i and suppose that we are using logarithmic utility and quality degradation functions. Then, the resource allocation problem becomes

$$\begin{aligned} &\max \ \sum_{i=1}^{n} a_i \log x_i \ \text{ subject to} \\ &-\log\left(1 - (x_i + x_3)/c_i\right) \leq \sigma_i, \quad i = 1, 2 \\ &-\sum_{i=1}^{2} \log\left(1 - \frac{x_i + x_3}{c_i}\right) \leq \sigma_3. \end{aligned} \tag{4}$$

Let w_i be the Lagrange multipliers corresponding to the quality degradation constraint of flow i. The *Lagrangian* is

$$\begin{aligned} L(x, w) = &\sum_{i=1}^{3} a_i \log x_i + \sum_{i=1}^{2} w_i \left(\log\left(1 - \frac{x_i + x_3}{c_i}\right) + \sigma_i\right) \\ &+ w_3 \left(\sum_{i=1}^{2} \log\left(1 - \frac{x_i + x_3}{c_i}\right) + \sigma_3\right). \end{aligned}$$

Therefore, we can derive the KKT conditions for this system,

$$\begin{aligned} a_i/x_i^* - (w_i + w_3)/(c_i - x_i^* - x_3^*) = 0, \quad i = 1, 2 \\ a_3/x_3^* - \sum_{i=1}^{2} (w_i + w_3)/(c_i - x_i^* - x_3^*) = 0, \\ w_i\left(\log\left(1 - (x_i + x_3)/c_i\right) + \sigma_i\right) = 0, \quad i = 1, 2 \\ w_3 \left(\sum_{i=1}^{2} \log\left(1 - (x_i + x_3)/c_i\right) + \sigma_3\right) = 0. \end{aligned} \tag{5}$$

Let us consider the situation where $\sigma_3 < \min\{\sigma_1, \sigma_2\}$. In this case, $w_1 = w_2 = 0$ and

$$\sum_{i=1}^{2} \log\left(1 - \frac{x_i + x_3}{c_i}\right) + \sigma_3 = 0.$$

For the simple case where $a_i = 1$ and $c_i = c$ for $i = 1, 2, 3$, we get optimal rates

$$x_1^* = x_2^* = 2x_3^* = \frac{2c}{3}\left(1 - e^{-\sigma_3/2}\right). \tag{6}$$

This illustrative example provides supporting evidence that our modeling intuition is accurate for resource allocation with service guarantees. We list pertinent observations derived from this model:

- for any finite service requirement, the sum-rate is always less than the capacity of each link;
- throughputs decrease with the number of hops due to service requirements;
- when quality degradation is inherent to a link, the flow with the most stringent service requirements limits the throughput of every neighboring flow.

IV. Primal Algorithm

In this section, we develop an algorithm that can be employed to obtain an approximate solution to our optimization problem. The approach that we adopt is called the *Primal* method, as it follows from the primal formulation of the problem. The main idea is to relax the constraints by incorporating them as a cost into the objective. Essentially, the idea is that there is a price for violating the quality constraints and, as such, we can maximize the difference between utility and cost. We consider the objective

$$J(x) = \sum_{r \in \mathcal{S}} \left(U_r(x_r) - B_r\left(\beta_r(x)\right)\right), \tag{7}$$

where $B_r(\cdot)$ is a convex barrier function that increases from zero to infinity as its argument ranges from zero to σ_r. To maximize this function, we can use a gradient descent approach,

$$\begin{aligned} \dot{x}_r &= k_r(x_r)\left(U_r'(x_r) - q_r\right), \\ q_r &= \sum_{s \in \mathcal{S}} B_s'\left(\beta_s(x)\right) \sum_{l \in \mathcal{L}} R_{ls} R_{lr} V_l'\left(y_l\right). \end{aligned} \tag{8}$$

Since the problem is convex, it is straightforward to show using Lyapunov techniques [2], [3], [16] that this algorithm converges and leads to a maximizer of (7). To this end, note that $J(x)$, as defined in (7), is a strictly concave function. We denote its unique maximizer by $\hat{x}$. Then, $J(\hat{x}) - J(x)$ is non-negative and equals zero only at $x = \hat{x}$. This makes

$W(x) \triangleq J(\hat{x}) - J(x)$ a natural candidate Lyapunov function; we use it in the following proposition, which has a similar proof to that found in [3].

Proposition 1. *Consider a network in which all sources follow the primal control algorithm* (8). *Suppose $J(x)$ is as defined in* (7) *and let functions $U_r(\cdot)$, $k_r(\cdot)$, $V_l(\cdot)$ and $B_s(\cdot)$ be such that $W(x)$ grows unbounded as $\|x\| \to \infty$, and $\hat{x}_i > 0$ for all i. Then, the controller in* (8) *is globally asymptotically stable and the equilibrium value maximizes* (7).

Proof: Differentiating $W(x)$ with time t, we get

$$\dot{W} = -\sum_{r \in \mathcal{S}} \frac{\partial J}{\partial x_r} \dot{x}_r = -\sum_{r \in \mathcal{S}} k_r(x_r) \left(U_r'(x_r) - q_r\right)^2 < 0,$$

for all $x \neq \hat{x}$, and $\dot{W} = 0$ for $x = \hat{x}$. Note that the second equality follows from (7) and (8). Thus, all the conditions of the Lyapunov theorem are satisfied, which ensures that the system state necessarily converges to $\hat{x}$. ■

Convergence is a highly desirable attribute. Yet, the primal controller suffers from the following drawbacks. The approach is not optimal; the relaxation will produce an acceptable solution only if the barrier values at the optimal solution of our original objective (2) are small. Further, the above formulation does not allow for optimal points on the boundary of the constraint set. To overcome these limitations, we can approach the problem from a dual perspective and hope for better performance.

V. Dual Algorithm

We start by writing a *dual* formulation for the resource allocation problem defined in (2). This provides new insight on how to obtain a distributed means of achieving optimal resource allocation. The dual formulation corresponding to our optimization problem is given by

$$D(w) = \max_{x_s \geq 0} \sum_{s \in \mathcal{S}} U_s(x_s) - w_s \left(\beta_s(x) - \sigma_s\right).$$

Let x^* be the optimal maximizer, then for all $r \in \mathcal{S}$

$$U_r'(x_r^*) = \sum_{s \in \mathcal{S}} w_s \sum_{l \in \mathcal{L}} R_{ls} R_{lr} V_l'\left(y_l^*\right).$$

This gives us a system of equations that can be solved to find the optimal x^* for any vector w. However, it requires knowledge of the load on every link a flow traverses. Therefore, this approach is not completely distributed.

Nevertheless, this formulation gives us the hint that instead of link load and link-degradation being dependent on each other directly with load $y = Rx$ and degradation $V(y)$, we can break up their coupling. We do this by introducing a new variable $\tilde{y}_l$, we refer to as *effective capacity* of link l. This variable upper bounds the total link load y_l, and $V_l(\tilde{y}_l)$ upper bounds the link degradation. We then define "effective quality degradation" $\tilde{\sigma}_r \triangleq \sum_{l \in \mathcal{L}} R_{lr} V_l(\tilde{y}_l)$ seen by flow r. Then, the relaxed version of the resource allocation problem becomes

$$\begin{aligned} &\max \sum_{r \in \mathcal{S}} U_r(x_r) \\ &\text{subject to } y_l \leq \tilde{y}_l, \ \forall\, l \in \mathcal{L} \\ &\qquad \tilde{\sigma}_r \leq \sigma_r, \ \text{and } x_r \geq 0 \ \forall\, r \in \mathcal{S}. \end{aligned} \tag{9}$$

Assuming that our concave utility and convex quality degradation functions ensure that values of x_r and $(c_l - \tilde{y}_l)$ are always positive, we can express the dual problem in terms of positive Lagrange multipliers p_l and w_r,

$$\min_{p, w \geq 0} D(p, w). \tag{10}$$

Here, $D(p, w)$ is the maximum of the *Lagrangian* function $L(x, \tilde{y}, p, w)$ with respect to $x, \tilde{y}$, where

$$L = \sum_{s \in \mathcal{S}} U_s(x_s) - \sum_{l \in \mathcal{L}} p_l \left(y_l - \tilde{y}_l\right) - \sum_{s \in \mathcal{S}} w_s \left(\tilde{\sigma}_s - \sigma_s\right).$$

Let $x^*, \tilde{y}^*$ be the maximizers for L for any p, w, then

$$U_r'(x_r^*) = \sum_{l \in \mathcal{L}} R_{lr} p_l, \qquad p_l = V_l'(\tilde{y}_l^*) \sum_{r \in \mathcal{S}} R_{lr} w_r.$$

We find the partial derivatives of $D(p, w)$ with respect to variables p and w,

$$\frac{\partial D}{\partial p_l} = \tilde{y}_l^* - y_l^*, \qquad \frac{\partial D}{\partial w_r} = \sigma_r - \tilde{\sigma}_r^*, \tag{11}$$

where y_l^* and $\tilde{\sigma}_r^*$ are link-load and effective degradation for maximizing rate x^* and effective-capacity $\tilde{y}^*$. We employ gradient descent method for minimizing the dual of the relaxed problem. Therefore, the update equations for Lagrange multipliers p, w can be written as

$$\dot{p}_l = h_l(p_l) \left(-\frac{\partial D}{\partial p_l}\right)^+_{p_l}, \quad \dot{w}_r = k_r(w_r) \left(-\frac{\partial D}{\partial p_l}\right)^+_{w_r}, \tag{12}$$

where $h_l(\cdot), k_r(\cdot)$ are positive functions and the notation $(z)_\rho^+$ is used to represent the function

$$(z)_\rho^+ = \begin{cases} z & \rho > 0 \\ \max\{z, 0\} & \rho = 0. \end{cases}$$

Function $(z)_\rho^+$ can be thought of as net input-rate into a fluid queue ρ. Clearly, when queue is non-empty fluid can enter at rate z, or leave at rate $-z$. However, when queue is empty fluid can only enter, but not leave.

We can now easily see that the above algorithm is distributed in nature. At any time during the evolution of our algorithm, we can treat Lagrange multipliers p_l and w_r as link-price and route-dissatisfaction, respectively. A flow r needs to "pay" link-price p_l for congesting link l if it uses the link (with the route-price being the sum of all such p_l), and w_r is its end-to-end dissatisfaction under the current system state. The effective capacity of link l is $\tilde{y}_l$ and is decoupled from the actual load on this link, $y_l = \sum_{r \in \mathcal{S}} R_{lr} x_r$. We denote the sum of link-prices by $q_r = \sum_{l \in \mathcal{L}} R_{lr} p_l$ for any flow r, and the sum of route-dissatisfaction on a link l by $\nu_l = \sum_{r \in \mathcal{S}} R_{lr} w_r$ to yield the total dissatisfaction on that link. Note that such a total implies that there is no need to maintain per-flow information at the link. Notice again that due to decoupling through $\tilde{y}$, the perceived quality degradation is a function of the effective capacity, and not the actual link-load.

The algorithm is illustrated in Fig. 2. Although the diagram is reminiscent of traditional "source-rate responds to link-price" [2]–[5] corresponding to the congestion control problem defined in (1), the system is actually very different. The system may be described as follows:

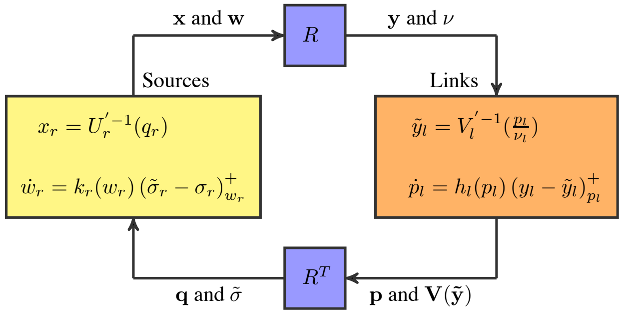


Fig. 2. Block diagram of value-aware resource allocation with decoupling of user-dissatisfaction on the source side, and quality on the link side.

- Each flow r, as it traverses its route, accrues the price q_r that it needs to "pay" for using each of the links l. Using this route-price, each source computes a feasible rate

$$x_r = U_r^{'-1}(q_r).$$

  Furthermore, each source declares its *dissatisfaction* w_r to the links it uses based on the difference between the quality degradation $\tilde{\sigma}_r$ that it sees and the degradation σ_r that it is willing to tolerate. The dissatisfaction is updated using

$$\dot{w}_r = k_r(w_r)\,(\tilde{\sigma}_r - \sigma_r)^+_{w_r}.$$

- Each link detects the total dissatisfaction ν_l of flows it accommodates, and it computes effective capacity

$$\tilde{y}_l = V_l^{'-1}(p_l/\nu_l)$$

  and updates the link price by

$$\dot{p}_l = h_l(p_l)\,(y_l - \tilde{y}_l)^+_{p_l}.$$

  The link ensures that the quality degradation inflicted on its users is $V_l(\tilde{y}_l)$ by increasing or reducing its aggregate flow as needed.

In summary, along with the two traditional elements of source-rate x_r and link-price p_l, we have two additional control variables: source-dissatisfaction w_r and effective capacity $\tilde{y}_l$ (with link-degradation $V_l(\tilde{y}_l)$) that provide two further dimensions of control that are required for a distributed solution. We next show that for admissible functions $V_l(\cdot)$, the *effective capacity* is equal to the sum rate for all links for which the capacity constraint is binding.

Proposition 2. *Assume that $V_l(\cdot)$ is strictly convex and increasing, and σ_r is finite for all $r \in \mathcal{S}$. Then, at equilibrium, for all $l \in \mathcal{L}_1 \triangleq \{l \in \mathcal{L} : \hat{y}_l < \hat{\tilde{y}}_l\}$, we have $\hat{p}_l = \hat{\nu}_l = 0$.*

Proof: Our proof is by contradiction. Let us assume that at the equilibrium, there is a $l \in \mathcal{L}$ such that $\hat{y}_l < \hat{\tilde{y}}_l$ and $\hat{\nu}_l \neq 0$. For this l, we have $\hat{p}_l = 0$ by KKT conditions. Note that $V_l'(\cdot)$ is a non-negative increasing function. That is, either $V_l'(0) = 0$ or $V_l'(z) > 0$ for all $z \in [0, c_l]$. For the former case, $0 \leq \hat{y}_l \leq \hat{\tilde{y}}_l = 0 = V_l^{'-1}(0)$, i.e., $\hat{y}_l = \hat{\tilde{y}}_l$. For the latter case, $\hat{p}_l$ cannot be zero since $V_l^{'-1}(0)$ is not in the feasible range of $\hat{y}_l$ and hence this will force $\hat{y}_l = \hat{\tilde{y}}_l$ at the equilibrium. Therefore, the result holds. ∎

We have in effect, shown that the equilibrium conditions of our control loop satisfy the KKT conditions of the original optimization problem defined in (2). The conditions are easy to verify, and we may state this result as a corollary to Proposition 2.

Corollary 3. *The stationary point of* (12) *is a maximizer of the convex optimization problem described in* (2).

It is quite easy to show that the above algorithm is globally asymptotically stable. To show this, we choose our Lyapunov function to be

$$Q(p, w) = D(p, w) - D(\hat{p}, \hat{w}), \tag{13}$$

where $\hat{p}, \hat{w}$ are the unique minimizers of $D(p, w)$. It is clear that $Q(p, w) \geq 0$ for all values of p, w. Also, it is easily seen that $D(p, w)$ grows radially unbounded in p, w for our choice of $V_l(\cdot)$. Therefore, to prove that the above algorithm is stable it suffices to show $\dot{D}(p, w) \leq 0$, with equality if and only if $p = \hat{p}$ and $w = \hat{w}$. Note that at $\hat{p}, \hat{w}$, one would have $\dot{p}_l = \dot{w}_r = 0$.

Proposition 4. *Let $Q(p, w)$ be as defined in* (13) *and functions $U_r(\cdot), V_l(\cdot), k_r(\cdot)$ and $h_l(\cdot)$ be such that $Q(p, w)$ grows unbounded with $\|p\|$ and $\|w\|$. The controller in* (12) *is globally asymptotically stable and the equilibrium value maximizes* (2).

Proof: Differentiating D with respect to time, we get

$$\begin{aligned}
\dot{D}(p, w) &= \sum_{l \in \mathcal{L}} \frac{\partial D}{\partial p_l} \dot{p}_l + \sum_{r \in \mathcal{S}} \frac{\partial D}{\partial w_r} \dot{w}_r \\
&= -\sum_{l \in \mathcal{L}} h_l(p_l)\,(y_l - \tilde{y}_l)\,(y_l - \tilde{y}_l)^+_{p_l} \\
&\quad - \sum_{r \in \mathcal{S}} k_r(w_r)\,(\tilde{\sigma}_r - \sigma_r)\,(\tilde{\sigma}_r - \sigma_r)^+_{w_r} < 0,
\end{aligned}$$

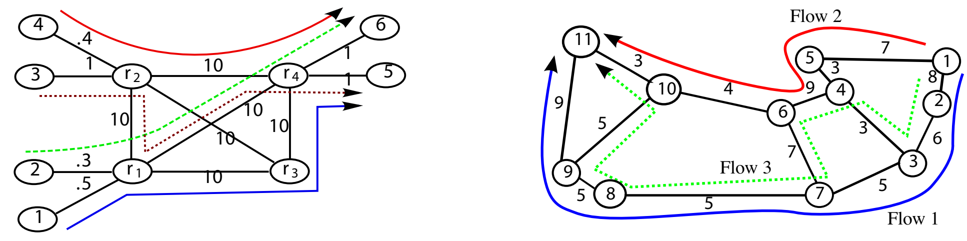
for all $(p, w) \neq (\hat{p}, \hat{w})$ and $\dot{D}(\hat{p}, \hat{w}) = 0$. The second equality follows from equations (11) and (12). Thus, all the conditions of the Lyapunov theorem [16] are satisfied and we have proved that the Lagrange multipliers converge to $\hat{p}, \hat{w}$. Hence, the system converges to the minimizer of (10). From the convexity of our original problem (2) and Corollary 3, it follows that the stable point is the maximizer of (2). ∎

VI. Numerical Studies

We utilize two realistic topologies presented in [14], illustrated in Fig. 3, to conduct numerical experiments. Our objective is to study the performance of our value-aware controller in different networking scenarios. We simulated our distributed resource allocation algorithm in Matlab using discrete-time evolution of *link-prices* and *end-to-end dissatisfaction*. Sources send packets at the rate generated by the controller, and links average this rate out using an exponential averaging factor α. Links base their decision on this average rate.

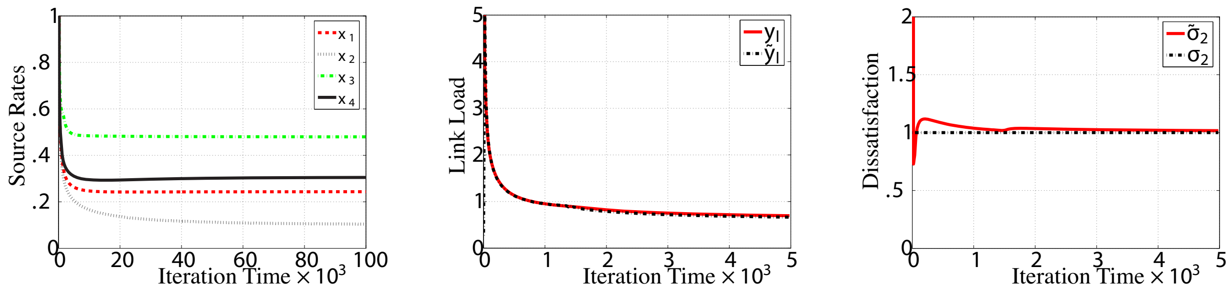
A. Access-Core Topology

Our first network is an *access-core* topology shown in Fig. 3(a). It represents a paradigm similar to commercially available Internet access, wherein users have a relatively small access bandwidth (from homes and businesses), connected together by a resource rich core-network. User bandwidth is



(a) Access Core Topology (b) Abilene Topology

Fig. 3. Two realistic topologies for numerical experimentation.



(a) Source-Rate (b) Load & effective capacity (c) QoS constraint for flow 2

Fig. 4. Convergence performance on Access-core topology.

constrained, either directly at the final hop into the home, or at a neighborhood head-end. Applications such as P2P file transfers (low quality constraint), as well as voice and video calls (higher quality constraints) result in end-to-end traffic on such a topology.

We consider the situation when nodes 1 and 3 wish to communicate to node 5; and similarly nodes 2 and 4, to node 6 over an access-core network as shown in Fig. 3(a). The labels on the links denote their respective capacity. We refer to a flow by its origin node. The QoS constraints on quality of degradation for flows 1-4 are $2, 1, 3, 2$, respectively. We plot the convergence of the source rates in Fig. 4(a). We also plot load y_l and *effective capacity* $\tilde{y}_l$ for the diagonal core link used by flows 2 & 3 in Fig. 4(b). We also plotted the quality seen by the flow 2 and its constraint $\sigma_2 = 1$ in Fig. 4(c). Note, we have different rates of convergence of different parameters.

We assume that core links have a capacity of much higher order than that of access links. Thereby, every link on the core is taken to be of capacity 10, whereas access node 1 connects to the core with capacity 0.5. Similarly, capacities for nodes 2-4 are $0.3, 1, 0.4$ respectively. We chose nodes $5, 6$ to have identical access link capacities of 1.

B. Abilene Topology

Our second network represents the major nodes of the Abilene network topology [19], shown in Fig. 3(b). The network consists of high bandwidth links, and connects several universities and research labs. Traffic consists of large scale data transfers (low quality constraints) and distributed computation (where flows have strict delay constraints).

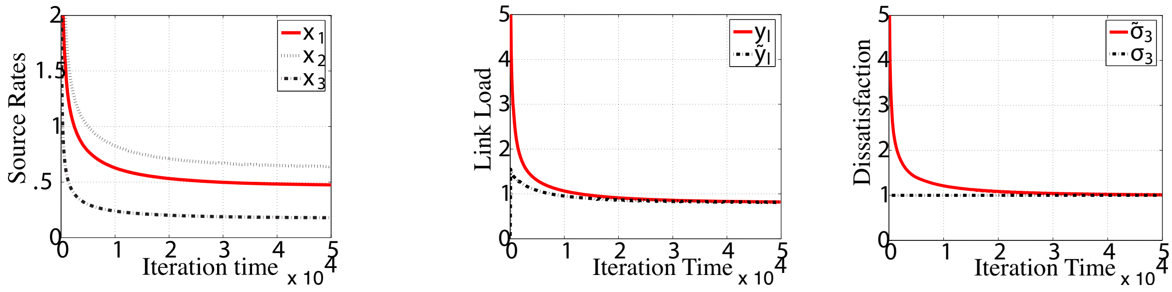
We consider 3 flows over the Abilene network as shown in Fig. 3(b) with labels denoting the capacity of the corresponding link. Note, they are of same order. We call the flow on bottom to be flow 1 and one on the top, flow 2. These two flows have QoS constraint on dissatisfaction 8 and 15 respectively. Flow 3 has the zigzag path and has the most stringent QoS constraint of 1. We plot the convergence of flow rates in Fig. 5(a). We also plot load y_l and *effective capacity* $\tilde{y}_l$ for the link of capacity 3 shared by flows 2 and 3, in Fig. 5(b). We have also plotted the quality seen by the zigzag flow and it's acceptable constraint $\sigma_3 = 1$ in Fig. 5(c).

The conclusions that we draw from our simulations are (i) our value-aware resource allocation algorithm converges to a stable solution, (ii) user quality constraints are satisfied at equilibrium, i.e., the algorithm performs as designed, and (iii) the effective capacity is identical to the actual link load at equilibrium showing that our relaxation produces a tight solution.

VII. NS-2 EXPERIMENTS

We adopt utility functions of the form $U_r(x_r) = a_r \log(x_r)$. We take $a_r = 100$ and normalize source-rates x_r to Megabits per second. The header of each packet transmitted by a source contains two additional fields, (i) source dissatisfaction w_r and (ii) route price q_r. Dissatisfaction is updated for each packet at the destination node as indicated by our dual algorithm. We selected the scaling function $k_r(w_r)$ to be a constant 10^3. Route price q_r is initialized to zero and is updated by the links. Depending on the current value, sources update their current rate as indicated by the algorithm.

As indicated by our algorithm, the price-update can be implemented by a virtual queue being served at rate equal to the *effective capacity* with an arrival rate of y_l. This virtual queue is implemented by traffic shaping (TS) queue, described below. We also need to compute the quality degradation at each link $V_l(\tilde{y}_l)$. We choose quality degradation per link to be the delay experienced by packets arriving in a queue with a rate $\tilde{y}_l$ and being served at link-capacity c_l. This queue is implemented by the router queue that buffers packets pending transmission over the link. The queues are shown in Fig. 6.



(a) Source-Rate (b) Load & effective capacity (c) QoS constraint for flow 3

Fig. 5. Convergence performance on Abilene topology.

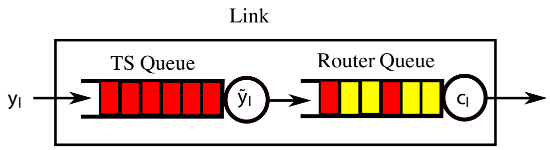


Fig. 6. Each node in our system contains a TS-queue and a router queue for each out-going link.

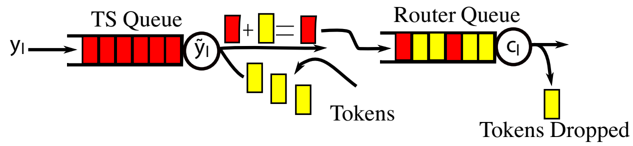


Fig. 7. The dynamics of the TS-queue and router queue in our system. Tokens are used in order to modulate the arrival rate into the router queue. Tokens are dropped when they reach the head of the router queue.

The Traffic Shaping Queue: The purpose of this queue is to shape traffic entering the router queue, and is illustrated by the queue on the left in Fig. 7. Since our system requires decoupling of the real load y_l on link l from the effective capacity $\tilde{y}_l$, we need to either add or subtract packets arriving at the link. For example, if two sources (S_1, S_2) are using link l, then a packet arrival from S_1 or S_2 is enqueued into the TS queue. The TS queue is drained at a rate $\tilde{y}_l$. The queue dynamics are implemented using a token generator at each link. Tokens are created at rate $\tilde{y}_l$. Each token observes the TS queue and if it is non-empty, it places the packet at the head of the TS queue into the router queue. Otherwise, the token itself is placed in the router queue. Delays are created due to the time that packets spend in the TS queue, but we will see that these delays are small. We choose a large buffer size ($10,000$ packets) to ensure that very few packets are lost.

The Router Queue: We approximate the delay in router queue with a decreasing function of difference in link-capacity c_l and the effective capacity $\tilde{y}_l$. In particular, we take $V_l'(\tilde{y}_l) \approx K/(c_l - \tilde{y}_l)^2$. Therefore, we can update effective capacity $\tilde{y}_l$ at periodic time intervals according to the following equation

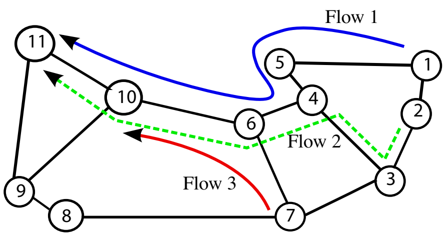
$$\tilde{y}_l = c_l - \sqrt{K\nu_l/p_l}. \tag{14}$$

We choose $K = 10^9$ for fast convergence. Unlike the analytical result presented earlier, we note that in reality it takes a while for the impact of changing effective capacity $\tilde{y}_l$ to be felt on the end-to-end delay. Therefore, we do not change $\tilde{y}_l$ at the same time scale as source rates. Instead, we do so periodically at each link. The times at which each link changes its value of $\tilde{y}_l$ are not synchronized. The objective is to get the source rates to converge to effective capacity $\tilde{y}_l$. The value of $\tilde{y}_l$ is again changed after a time interval, assuming that the sum of source rates converged to that $\tilde{y}_l$, and hence the observed delay is $V_l(\tilde{y}_l)$.

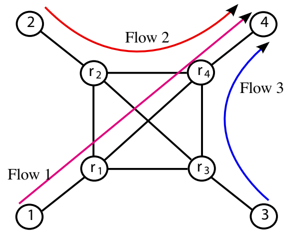
We now study the performance of our protocol using a network simulator (ns-2) on the two realistic topologies presented that we saw earlier in Section VI. Let T_i denote the propagation delay for flow i with no queueing delays. Our objective is to study the performance of our value-aware controller in different networking scenarios, and our example situations are shown in Fig. 8.

Abilene Topology: As shown in Fig. 8(a), the rates of the three flows are x_1, x_2 and x_3. In the system considered, $T_1 = 50$ ms, $T_2 = 45$ ms and $T_3 = 25$ms. We consider two cases. In the first case, flow 1 has stringent quality degradation constraint. In particular, respective degradation tolerances in terms of delays are $\sigma_1 = 55$ ms and $\sigma_2 = \sigma_3 = 1000$ s. Thus, we have set a very high delay tolerance for flows 2 and 3, whereas for flow 1 this tolerance is very low and nearly equals the propagation delay of 50 ms, and hence allows a queuing delay of only 5 ms for the five intermediate queues at routers 1, 5, 4, 6 and 10. The tight delay constraint on flow 1 has an effect on the other two flows which share the link between routers 6 and 10 with it. In Fig. 9(a) we plot the rates associated with individual flows. Figure 9(b) shows the acceptable delay for packets for flow 1, the delay through the router queues (the control delay), and the actual total delay (TS queue plus router queue) for packets to reach node 11 from node 1. Figure 9(c) plots the effective capacity $\tilde{y}$ for the link between nodes 6 and 10 (shared by all the three flows), which is less than the total link capacity of 25 Mbps. It is clear that the protocol is successful in ensuring a good delay performance for flow 1, at the expense of overall system throughput.

Access-Core Topology: Our other experiment involves the Access-Core topology, with flows of interest shown in Fig. 8(b). In the system considered $T_1 = T_3 = 20$ ms and $T_2 = 25$ ms. Flow 1 has very stringent delay constraint. In particular, delay tolerances are $\sigma_1 = 25$ ms, $\sigma_2 = \sigma_3 = 1000$ s. In this case, flows 2 and 3 are highly delay tolerant, while flow 1 has low delay tolerance and it nearly equals the propagation delay of 20ms, and allows a queuing delay of 5ms for the three intermediate queues at routers 1, r_1 and r_4. The tight delay constraint on flow 1 has an effect on the other two flows which share the link between r_4 and 4 with

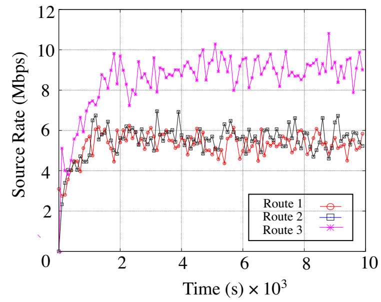


(a) Abilene Topology

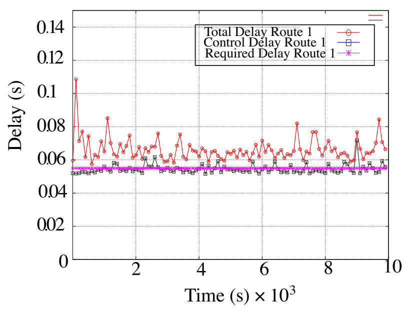


(b) Access-Core Topology

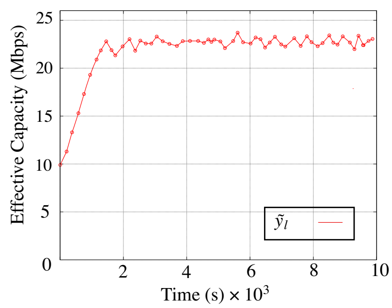
Fig. 8. Flows considered for ns-2 experimentation.



(a) Source-Rate

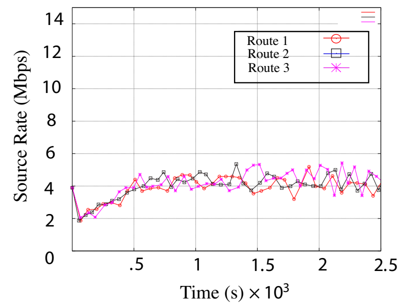


(b) Delay experienced by flow 1

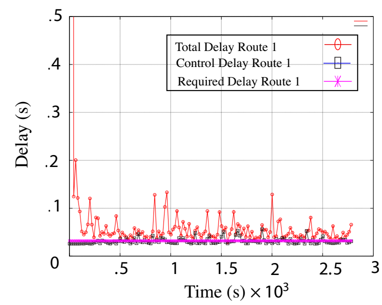


(c) Effective capacity

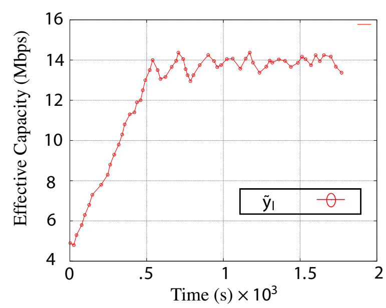
Fig. 9. Convergence performance on Abilene network topology.



(a) Source-Rate



(b) Delay experienced by flow 1



(c) Effective capacity

Fig. 10. Convergence performance on Access-Core network topology.

it. In Fig. 10(a) we plot the three rates for flows 1, 2 and 3. Fig. 10(b) shows the acceptable delay for packets for flow 1, the delay through the router queue (control delay) and the actual total delay (TS queue plus router queue) for packets to reach node 4 from node 1. The total delay is close to the target. Fig. 10(c) plots the $\tilde{y}$ at router r_4 which is shared by all the three flows. Again, the throughput is less than the link capacity of 15 Mbps in the interest of reducing delay.

VIII. Conclusions

In this paper we considered the design of a distributed resource allocation algorithm that would allow each individual flow to specify its measure of value. We assumed that every flow passing through a link suffers a certain quality-degradation due to the load on the link, and that such degradation adds up over the multiple links that the flow traverses. The objective is to ensure that the system throughput is maximized in a fair manner, subject to each flow's quality of service satisfying a hard constraint. Our aim was to ensure that the algorithm should be simple, use local information, and the relays need not maintain per-flow information.

We first showed that attempting to solve this problem by the usual optimization decomposition techniques using the primal formulation yields an approximate solution, and using the dual formulation yields a centralized solution. However, the observation that decoupling the link-load from the quality degradation using a secondary variable that we call *effective capacity*, allows us to design such a controller. Under our scheme, the source chooses its rate based on a *route price*, and it declares a *dissatisfaction* based on the quality of service that it sees. Links choose an effective capacity based on dissatisfaction and *link-price*, and modify the price as if the effective capacity were the actual capacity of the link. The control scheme only requires that links be aware of *aggregate* quantities of the flows using them, and the sources perform computations solely based on the parameters obtained from the links they traverse, hence satisfying our requirements.

We studied illustrative examples of quality-degradation functions that helped us gain insight into the working of the system. We performed simulations on realistic topologies to illustrate the performance of our algorithm, and used it as a basis for developing a delay-sensitive protocol. We showed that the protocol does indeed trade delay and throughput, so as to maximize the total utility of the system. In the future we would like to test out our ideas on a real network.

References

[1] P. Parag, S. Shakkottai, and J.-F. Chamberland, "Value-aware resource allocation for service guarantees in networks," in *Proc. INFOCOM*, March 2010, pp. 2730–2738.

[2] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, January-February 1997.

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

[4] R. Srikant, *The Mathematics of Internet Congestion Control*, ser. Systems & Control: Foundations & Applications. Birkhäuser, 2004.

[5] S. Shakkottai and R. Srikant, *Network Optimization and Control*, ser. Foundations and Trends in Networking. Now Publishers, 2008, vol. 2, no. 3.

[6] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proc. IEEE*, vol. 95, no. 1, pp. 255–312, January 2007.

[7] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, 1st ed. Addison-Wesley Professional, 1994.

[8] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*, 2nd ed. Morgan-Kaufman, 2000.

[9] S. H. Low and D. E. Lapsley, "Optimization flow control–I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, pp. 861–874, December 1999.

[10] G. Vinnicombe, "On the stability of networks operating TCP-like congestion control," in *Proc. IFAC World Congress*, Barcelona, Spain, 2002.

[11] T. Kelly, "Scalable TCP: improving performance in highspeed wide area networks," *ACM SIGCOMM Computer Communication Review*, vol. 33, pp. 83–91, April 2003.

[12] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "Fast tcp: Motivation, architecture, algorithms, performance," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1246–1259, December 2006.

[13] S. Liu, T. Başar, and R. Srikant, "TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks," *Performance Evaluation*, vol. 65, pp. 417–440, June 2008.

[14] J. He, M. Suchara, J. Rexford, and M. Chiang, "Rethinking internet traffic management: From multiple decompositions to a practical protocol," in *Proc. CoNEXT*, New York, December 2007.

[15] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1439–1451, August 2006.

[16] H. K. Khalil, *Nonlinear Systems*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1996.

[17] L. Liu, P. Parag, J. Tang, W.-Y. Chen, and J.-F. Chamberland, "Resource allocation and quality of service evaluation for wireless communication systems using fluid models," *IEEE Trans. Inf. Theory*, vol. 53, no. 5, pp. 1767–1777, May 2007.

[18] W. N. Kang, F. P. Kelly, N. H. Lee, and R. J. Williams, "State space collapse and diffusion approximation for a network operating under a fair bandwidth sharing policy," *The Annals of Applied Probability*, vol. 19, no. 5, pp. 1719–1780, May 2009.

[19] "Internet2," 2009. [Online]. Available: http://www.internet2.edu/network/

Parimal Parag (S '05) received his dual degree (B. Tech. and M. Tech.) in electrical engineering, with specialization in communication systems, from Indian Institute of Technology, Madras, India, in 2004. He is currently working towards the Ph.D. degree at Texas A&M University, College Station. He visited Los Alamos National Laboratory in summer 2007, and Stanford University in autumn 2010.

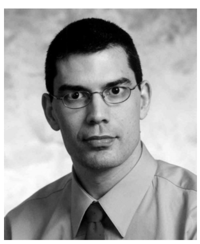
His research interests include applied probability, estimation & detection theory, optimization methods, and their application to communication networks, .

Sankalp Sah received his Bachelor of Technology degree in ECE from the Indian Institute of Technology, Guwahati, India, in 2005 and his M.S. degree in Computer Engineering at Texas A&M University in 2010. He is currently working at Ericsson in the Packet Forwarding group for next generation wireless systems like LTE.

His interests in communication networks lie specifically at the router level handling/processing of packets and in forwarding algorithms.

Srinivas Shakkottai (S '00-M '08) received his Bachelor of Engineering degree in electronics and communication engineering from the Bangalore University, India, in 2001 and his M.S. and Ph.D degrees from the University of Illinois at Urbana-Champaign in 2003 and 2007, respectively, both in electrical engineering. He was Postdoctoral Scholar at Stanford University until December 2007, and is currently an Assistant Professor at the Dept. of ECE, Texas A&M University. He has received the Defense Threat reduction Agency Young Investigator Award (2009) as well as research awards from Cisco (2008) and Google (2010).

His research interests include peer-to-peer systems, pricing approaches to network resource allocation, game theory, congestion control, and the measurement and analysis of Internet data.



Jean-Francois Chamberland (S '98 -M '04-SM '09) received the Ph.D. degree, in 2004, from the University of Illinois at Urbana-Champaign; the M.S. degree, in 2000, from Cornell University; and the B.S. degree, in 1998, from McGill University, Canada. He joined Texas A&M University, in 2004, where he is currently an Associate Professor in the Department of Electrical and Computer Engineering. Among the awards he has received for research and teaching are a Young Author Best Paper Award from the IEEE Signal Processing Society, in 2006; and an Early Career Development (CAREER) Award from the National Science Foundation, in 2008.

He is currently conducting research in statistical signal processing, applied probability, optimization methods, and in their applications to information systems and networks.