End-to-End Optimal Algorithms for Integrated QoS, Traffic Engineering, and Failure Recovery

Bernardo A. Movsichoff, *Member, IEEE*, Constantino M. Lagoa, *Member, IEEE*, and Hao Che

Abstract

This paper addresses the problem of optimal Quality of Service (QoS), Traffic Engineering (TE) and Failure Recovery (FR) in Computer Networks by introducing novel algorithms that only use source inferrable information. More precisely, optimal data rate adaptation and load balancing laws are provided which are applicable to networks where multiple paths are available and multiple Classes of Service (CoS) are to be provided. Different types of multiple paths are supported, including point-to-point multiple paths, point-to-multipoint multiple paths, and multicasting. In particular, it is shown that the algorithms presented only need a minimal amount of information to achieve an optimal operating point. More precisely, they only require knowledge of whether a path is congested or not. Hence, the control laws provided in this paper allow source inferred congestion detection without the need for explicit congestion feedback from the network. The proposed approach is applicable to utility functions of a very general form and endows the network with the important property of robustness with respect to node/link failures; i.e., upon the occurrence of such a failure, the presented control laws reroute traffic away from the inoperative node/link and converge to the optimal allocation for the "reduced" network. The proposed control laws set the foundation for the development of feature-rich traffic control protocols at the IP, transport, or higher layers with provable global stability and convergence properties. Highly scalable QoS, TE, and FR features can be implemented based on these control laws, without the involvement of the routers in the network core.

Index Terms

Distributed Traffic control, QoS, Failure Recovery, Sliding Mode Control, Optimization

Bernardo A. Movsichoff and Constantino M. Lagoa are with the Department of Electrical Engineering, The Pennsylvania State University. Email: bernardo@gandalf.ee.psu.edu; lagoa@engr.psu.edu

Hao Che is with the Department of Computer Science and Engineering, University of Texas at Arlington. Email: hche@cse.uta.edu

I. Introduction

The Transport Control Protocol (TCP) window flow control algorithms use minimum information from the network as input to allow fully distributed traffic control. In other words, the only needed feedback information for the TCP window flow control is whether the forwarding path is congested or not. This allows the TCP source node to *infer* path congestion by counting the number of repetitive acknowledgments of the same packet or measuring end-to-end round-trip delay, making TCP a truly end-to-end protocol without the assistance of the underlying internetworking layer infrastructure. This has made the proliferation of the Internet applications at the global scale possible. An excellent example is the fast, ubiquitous adoption of World Wide Web due to its use of TCP as its underlying transport.

However, as the Internet has evolved into a global commercial infrastructure, there has been a great demand for new applications of global reach, for which today's Internet protocols cannot adequately support. For example, realtime applications, such as Voice over IP (VoIP) and video phone, have stringent delay and delay jitter requirements, which cannot be adequately supported by today's Internet protocols. As a result, in recent years, a large number of new Internet protocols were developed in an attempt to meet this demand. For example, Multiprotocol Label Switching (MPLS) has been envisioned as an ideal platform upon which guaranteed services could be developed. Service guarantee is achieved by setting up and managing a set of primary and backup Class-of-Service (CoS) aware label switched paths across an IP domain. In addition to MPLS, this approach requires a suite of protocols be implemented, e.g., DiffServ for Quality of Service (QoS), path protection/fast rerouting for link failure recovery (FR), and constraint-based routing for traffic engineering (TE). This, however, means that, to adequately support realtime applications, a whole suite of protocols with significant involvement of the IP core nodes need to be developed. This raises serious concerns about the scalability and complexity of using these protocols to support realtime applications at a global scale.

Hence a key question to be answered is whether it is possible to enable the above service quality features, including QoS, TE, and FR, with the involvement of communication end points only. In this paper, we put forward a much needed mathematical framework to make this possible. We show that a large family of

Distributed traffic Control Laws (DCLs) exists, which allows optimal, multiple CoSes, multipath[1] based rate adaptation and load balancing. The DCLs drive the network to an operation point where a user defined global utility function is maximized. The DCLs control the traffic independently at different traffic source nodes, e.g., edge nodes or end-hosts. A salient feature of this family of DCLs is that the needed information feedback from the network is minimum, i.e., whether a forwarding path is congested or not, which can be inferred at the source node itself, the same way as TCP congestion notification. This makes it possible to allow this family of DCLs to be operated end-to-end. A core node may be CoS and multipath agnostic and may employ any queue management/scheduling algorithms, e.g., simple FIFO queues, at its output ports. This family of DCLs allows fast timescale TE through multipath load balancing which is robust in the presence of link/node failures. In other words, the DCLs can automatically repartition the traffic in an optimal way among the rest of the multipath in react to any path failures. Hence, this family of DCLs by design has the capability to enable optimal, scalable QoS, TE, and FR, simultaneously. Moreover, since the mathematical formulation allows both point-to-point multipath and point-to-multipoint multipath, the family of DCLs can be applied to a connectionless IP network to enable sophisticated service quality features, solely based on a set of shortest paths from any given ingress node to a set of egress nodes.

The remainder of the paper is organized as follows: Section II reviews the related work. Section III presents notation and assumptions used throughout this paper, Section IV provides a precise statement of the problem to be solved, while Section V introduces the proposed optimal solution. Section VII on the other hand, discuss some implementation issues while Section VIII provides some simulation results. Finally, Section IX provides some conclusions and the Appendix presents the proof of the results in this paper.

II. Related Work

There is extensive literature on distributed traffic control. In particular, algorithms with a focus on TCP types of traffic were developed, including both empirical algorithms (e.g., see [7], [8]) and algorithms based

[1]Here a multipath is defined as a set of paths originated from a given source node to one (i.e.,point-to-point) or a set of (i.e., point-to-multipoint) sink nodes.

on control theory (e.g., see [2], [4]). However, these algorithms assume a single path and the approaches taken are not optimization based.

Since flows with different ingress-egress node pairs share the same network resources, the key challenge in the design of DCLs is the fact that there is a high degree of interaction between different flows due to the resource constraints. One approach to get around this is to incorporate a link congestion cost into the overall utility function, which replaces the link resource constraints. Then, the problem is solved using a gradient type algorithm, resulting in families of DCLs that support point-to-point multi-path load balancing for rate adaptive traffic, e.g., Golestani, et al. [10], Elwalid, et al. [5], and Guven, et al. [11].

Recently, significant research effort has been made in the design of DCLs with link capacity constraints explicitly taken into account. At the core of this endeavor is the development of DCLs which converge to an operation point where a given global utility function is maximized. This line of research has been proven to be fruitful. Large families of DCLs of this kind are obtained based on the nonlinear programming techniques, e.g., the work by Kelly, et al. [14], Low and Lapsley [18] [21], La and Anantharam [16], and Kar, et al. [13]. These families of DCLs generally require that a sum of link “prices" for all the links in the forwarding path to be periodically calculated and fed back to the source. Under the condition that the network is lightly loaded, the DCLs developed in [16] and [13] allow local control without feedback from the network. In particular, in [21], a family of rate adaptive control laws is design that requires only single bit binary feedback indicating whether the path is congested or not. Kelly, et. al. [14] found a TCP-like DCL that allows point-to-point multipath, under the condition that there is no feedback delay. Recently, Han, et al. [12] successfully extended the results in [14] to allow feedback delay. The results were applied to an overlay network of BGP peers with dedicated resources to allow point-to-point multi-path load balancing.

However, all the above results can only be applied to rate adaptive traffic. Recently, the authors of this paper developed a family of DCLs [17] based on nonlinear control theory [15]. This family of DCLs can be applied not only to usual rate adaptive traffic with point-to-point multipath, but also to rate adaptive traffic with minimum service requirements and/or maximum allowed sending rate and to services with targeted rate guarantee, all allowing for point-to-point multipath. The only needed feedback from the network is the number

of congested links along the forwarding paths. Moreover, the technique applies to any utility function that can be expressed as a sum of concave terms.

Nevertheless, due to the needed use of the number of congested links in a forwarding path as the input to a DCL, the family of DCLs proposed in [17] requires explicit congestion feedback from the network. Hence, this family of DCLs can only be applied to a connection-oriented network, such as an MPLS enabled IP network. In this paper, a new family of DCLs is design, free of limitations suffered by the family of DCLs proposed in [17], while retaining all the nice features enjoyed by that family of DCLs. Moreover, the new family of DCLs allows both point-to-point multipath and point-to-multipoint multipath, making it applicable to a connectionless IP network using multiple source rooted shortest paths found by an underlying intradomain routing protocol.

Finally, note that in a related work in [20], the authors of this paper designed a family of DCLs which allows hop-by-hop rate adaptation and load balancing with minimum information exchange between neighboring nodes. It is particularly powerful to provide sophisticated service quality features at the internetworking layer in a connectionless IP network, while the family of DCLs developed in this paper is particularly useful to allow sophisticated service quality features to be developed at the transport or higher layers end-to-end.

III. Preliminaries

Throughout this paper, it is assumed that traffic flows can be described by a fluid flow model, where the only resource taken into account is link bandwidth. For simplicity, we first restrict ourselves to the point-to-point multipath only and address the point-to-multipoint and multicast cases later.

Consider a computer network where calls of different *types* are present. In this paper, *types* denote aggregate of calls with the same ingress and egress node, as well as service requirements; i.e., calls that share a given set of paths connecting the same ingress/egress node pair and whose service requirements are to be satisfied by the aggregate, not by individual calls. Note that when the edge nodes coincide with the end-hosts, the control laws developed in this paper become end-to-end control laws working at the transport layer servicing individual application flows.

More precisely, consider a computer network whose set of links is denoted by $\mathscr{L}$ and let c_l be the capacity of link $l \in \mathscr{L}$. Let n be the number of types of calls, n_i be the number of paths available for calls of type i

and $\mathscr{L}_{i,j}$ be the set of links used by calls of type i taking path j; i.e., if $B_{i,j} = \text{card}(\mathscr{L}_{i,j})$, the cardinality of the set $\mathscr{L}_{i,j}$, then $B_{i,j}$ is the number of links in this path. Given calls of type i, let $x_{i,j}$ be the total data rate of calls of type i using path j. Also, let $\mathbf{x}_i \doteq [x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}] \in \mathbf{R}^{n_i}$ denote the vector containing the data rates allocated to the different paths taken by calls of type i, and $\mathbf{x} \doteq \left[\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_n^T\right]^T \in \mathbf{R}^N$ the vector containing all the data rates allocated to different call types and respective paths, where $N = n_1 + n_2 + \cdots + n_n$.

Now, a link $l \in \mathscr{L}$ is said to be congested if the aggregated data rate of the calls using the link reaches its capacity c_l. The congestion information $cg_{i,j}$ for calls $x_{i,j}$; i.e., calls of type i taking path j, is defined as $cg_{i,j} \doteq 1$ if any link $l \in \mathscr{L}_{i,j}$ is congested, and 0 otherwise. Moreover, $\overline{cg_{i,j}}$ denotes the logical not operation on $cg_{i,j}$; i.e., $\overline{cg_{i,j}} = 1 - cg_{i,j}$.

IV. Problem Statement

In this paper, we aim at solving the same problem as in [17]; i.e., developing data rate adaptation laws that maximize a given utility function subject to CoS requirements. Although addressing the same problem, the solution to the problem presented in this paper is not an incremental improvement on the solution provided in [17]. It sets the foundation for the development of a wide variety of traffic control protocols to enable QoS, TE, and FR features simultaneously, which only use source inferrable congestion information. We now define precisely the problem to be solved.

The results in this paper aim at maximizing utility functions of the form

$$U(\mathbf{x}) \doteq \alpha \sum_{i=1}^{n} U_i(\mathbf{x}_i) \doteq \alpha \sum_{i=1}^{n} U_i\big(x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}\big),$$

subject to network constraints and CoS requirements, where $U_i(\cdot)$, $i = 1, 2, \ldots, n$, are differentiable concave functions, strictly increasing in each of their arguments, and α is a positive scaling constant. Given this, the problem of optimal resource allocation can be formulated (see [17]) as the following optimization problem:

$$\max_{\mathbf{x}} U(\mathbf{x})$$

subject to the network capacity constraints

$$\sum_{i,j:\ l \in \mathscr{L}_{i,j}} x_{i,j} - c_l \leq 0; \qquad l \in \mathscr{L},$$

the CoS requirements: the Assured Forwarding (AF) requirements

$$\sum_{j=1}^{n_i} x_{i,j} = \Lambda_i; \qquad i = 1,2,\ldots,s_1,$$

the Minimum Rate Guaranteed Service (MRGS) requirements

$$\sum_{j=1}^{n_i} x_{i,j} \geq \theta_i; \qquad i = s_1+1, s_1+2,\ldots,s_2,$$

the Upper Bounded Rate Service (UBRS) requirements

$$\sum_{j=1}^{n_i} x_{i,j} \leq \Theta_i; \qquad i = s_2+1, s_2+2,\ldots,s_3,$$

the Minimum Service Guarantee and an Upper Bounded Rate (MRGUBS) requirements

$$\theta_i \leq \sum_{j=1}^{n_i} x_{i,j} \leq \Theta_i; \qquad i = s_3+1, s_3+2,\ldots,s_4$$

and all data rates are nonnegative

$$x_{i,j} \geq 0; \qquad i = 1,2,\ldots,n;\ j = 1,2,\ldots,n_i.$$

Obviously, the optimization problem above is a convex problem; i.e., maximizing a concave function over a convex set. If global information is available then algorithms like gradient descent could be used to solve it. However, generally, global information is not available. The objective of this paper is to provide decentralized adaptation laws that converge to the solution of the problem stated above[2].

A. Point-to-Multipoint Service

The problem formulation so far has only considered point-to-point multipath; i.e., multiple paths from an ingress node to a given egress node. This formulation, however, is too restrictive. It does not account for the possible need for point-to-multipoint multipath forwarding; i.e., forwarding from an ingress node to multiple egress nodes. This feature is particularly useful when traffic is to be balanced among multiple shortest paths

[2]The provisioning of the aggregated resource for AS, MRGS, and MRGUBS traffic running between any pair of nodes does need to ensure that at least one feasible distribution exists, which is beyond the scope of this paper. Some optimization algorithms with global information such as the one proposed by Mitra [19] can be employed to serve this purpose during the network resource planning phase.

to the destination network reachable via multiple egress nodes, as pointed out in [9]. Now, we show that point-to-multipoint multipath can be easily recast into the problem formulation provided above.

Assume that calls of type i use point-to-multipoint multipath and that it has M egress nodes. As before, we assume that there are several paths connecting the ingress node to each of the egress nodes and denote the data rate used by calls to receiver m that use path j by x_{i,j_m}. Moreover, let $\mathscr{L}_{i,j_m}$ be the set of links used by calls to receiver m taking path j_m.

In this case, one defines congestion of a path in the usual way; i.e., path j_m to receiver m is congested if at least one of the links in $\mathscr{L}_{i,j_m}$ is congested. Hence, as far as link constraints are concerned, no modification in the formulation is needed. The main difference between point-to-multipoint multipath and the point-to-point multipath discussed earlier, is the fact that CoS constraints are to be enforced on the total data rate; i.e., CoS constraints are defined in terms of

$$\sum_{m=1}^{M} \sum_{j=1}^{n_{i,m}} x_{i,j_m}$$

where $n_{i,m}$ is the number of paths available to calls whose receiver is m. Hence, in the case of point-to-multipoint multipath, the control laws will "look" at the overall aggregate data rate to all receivers. Apart from that small difference, the constraints involved are of the same form as the ones used for point-to-point multipath and, therefore, the data rate control laws are similar. Hence, to simplify the exposition, from this point on only point-to-point multipath is considered.

In summary, the problem formulation in this paper addresses a very general multipath forwarding problem including point-to-point, point-to-multipoint, and (using a similar formulation to the one described above) multicast multipaths. Fig. 1 gives an example to show different kinds of multipaths that may co-exist in the network. From ingress node 1 to egress node 3, there is a point-to-point multipath with two paths in it. This multipath, together with the path from ingress node 1 to egress node 5, can form a point-to-multipoint multipath with three paths in it. Also in Fig. 1, there is a multicast multipath from ingress node 2 to egress nodes 3, 4, and 5.

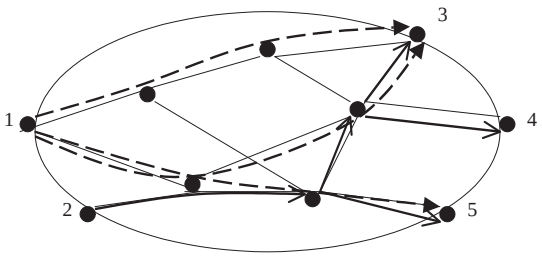

Fig. 1. Examples of point-to-point multipath, point-to-multipoint multipath and a multicast path.

V. A Novel Family of Distributed Rate Adaptation Control Laws

Before presenting the main results in this paper, this section introduces the proposed solution to the optimization problem above, a family of control laws that achieve optimal rate allocation.

Let $f_{i,j}$ be defined as

$$f_{i,j}(\mathbf{x}) \doteq \left(1 - e^{-\partial U/\partial x_{i,j}}\right),$$

and let

$$(y)^{+}_{x=0} = \begin{cases} \max\{y,0\} \text{ if } x = 0; \\ y \text{ if } x \neq 0. \end{cases}$$

Also, let $z_{i,j}(t,\mathbf{x})$ be positive scalar functions for all i and all j. Now, define the following family of control laws: For $i = 1,2,\ldots,s_1$; i.e., AF calls, let

$$\dot{x}_{i,j} = \left(z_{i,j}(t,\mathbf{x})\left[f_{i,j}(\mathbf{x}) - \left(1 - \overline{cg_{i,j}} r_i\right)\right]\right)^{+}_{x_{i,j}=0}, \qquad \text{where} \qquad r_i(\mathbf{x}_i) = \begin{cases} r_{min} < 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \Lambda_i \\ r_{max} > 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \Lambda_i, \end{cases}.$$

For $i = s_1+1, s_1+2, \ldots, s_2$; i.e., MRGS calls, let

$$\dot{x}_{i,j} = \left(z_{i,j}(t,\mathbf{x})\left[f_{i,j}(\mathbf{x}) - \left(1 - \overline{cg_{i,j}} r^m_i\right)\right]\right)^{+}_{x_{i,j}=0}, \qquad \text{where} \qquad r^m_i(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \theta_i \\ r^m_{max} > 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \theta_i, \end{cases}$$

For $i = s_2+1, s_2+2, \ldots, s_3$; i.e., UBRS calls, let

$$\dot{x}_{i,j} = \Big(z_{i,j}(t,\mathbf{x}) \Big[f_{i,j}(\mathbf{x}) - \big(1 - \overline{cg_{i,j}} r_i^M\big) \Big] \Big)^+_{x_{i,j}=0}, \qquad \text{where} \qquad r_i^M(\mathbf{x}_i) = \begin{cases} r_{min}^M < 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \Theta_i \\ 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \Theta_i, \end{cases}.$$

For $i = s_3+1, s_3+2, \ldots, s_4$; i.e., MRGUBS calls, let

$$\dot{x}_{i,j} = \Big(z_{i,j}(t,\mathbf{x}) \Big[f_{i,j}(\mathbf{x}) - \big(1 - \overline{cg_{i,j}} r_i^m r_i^M\big) \Big] \Big)^+_{x_{i,j}=0},$$

where

$$r_i^m(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \theta_i \\ r_{max}^m > 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \theta_i, \end{cases} \qquad r_i^M(\mathbf{x}_i) = \begin{cases} r_{min}^M < 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \Theta_i \\ 1 & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \Theta_i, \end{cases}.$$

The quantities r_{min}, r_{max}, r_{max}^m, and r_{min}^M are predetermined positive constants chosen to satisfy convergence of the algorithm, as shown in Theorem 1.

Finally, for $i = s_4+1, s_4+2, \ldots, n$; i.e., BE calls, let

$$\dot{x}_{i,j} = \Big(z_{i,j}(t,\mathbf{x}) \Big[f_{i,j}(\mathbf{x}) - \big(1 - \overline{cg_{i,j}}\big) \Big] \Big)^+_{x_{i,j}=0}.$$

A. Main Result

The main result of this paper establishes that the control laws presented above, converge to the solution of the optimization problem posed. This is formally stated in the following theorem.

Theorem 1: Assume that all data rates are bounded; i.e., there exists $\rho \in \mathbf{R}$ such that the data rate vector $\mathbf{x}$ always belongs to the set

$$\mathscr{X} \doteq \{\mathbf{x} \in \mathbf{R}^{n_1+n_2+\cdots+n_n} : \; x_{i,j} \le \rho, \; l \in \mathscr{L}_{i,j}, \; j = 1,2,\ldots,n_i, \; i = 1,2,\ldots,n\}.$$

Also, assume that at the optimal traffic allocation, each congested link has at least one BE call with non-zero data rate and that the elements of the gradient of the utility function are bounded in $\mathscr{X}$. Let $\zeta > 0$ be a given (arbitrarily small) constant and let $z_{i,j}(t,\mathbf{x})$ be scalar continuous functions satisfying $z_{i,j}(t,\mathbf{x}) > \zeta$, for all $t > 0$ and all $\mathbf{x} \in \mathscr{X}$. Furthermore, let

$$0 < r_{min}, r_{min}^M < r_{lower} < 1 < r_{upper} < r_{max}, r_{max}^m,$$

where $r_{lower} = e^{-v_{k,\max}}$, $r_{upper} = e^{v_{k,\max}}$, and

$$v_{k,\max} = \max_{i,j} B_{i,j} \max_{i,j,\mathbf{x}\in\mathscr{X}} \frac{\partial U}{\partial x_{i,j}}.$$

The quantity $B_{i,j}$, as defined in Section III, is the number of links in path j taken by calls of type i. Then, the control laws presented above converge to a traffic allocation that maximizes the utility function $U(\mathbf{x})$ subject to the network's capacity constraints, CoS requirements and non-negativity of all the data rates.

VI. A TCP-like Control Law For Multipath BE Traffic

It turns out that the linear increase/exponential decrease behavior of the TCP algorithm in its congestion avoidance phase is a particular case of the control laws provided in the previous section. Moreover, these control laws indicate how one can generalize the TCP algorithm to the multipath case. To see this, consider calls of type i belonging to the BE CoS and assume that the aggregate data rate is bounded away from zero. Moreover, assume that the aggregate rate is "large." Now, assume that the associated factor in the utility function is

$$U_i\left(x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}\right) = \log\left(\sum_{j=1}^{n_i} x_{i,n_i}\right).$$

Moreover, take

$$z_{i,j}(t,\mathbf{x}) = \frac{\zeta}{1 - e^{-\alpha \partial U_i/\partial x_{i,j}}}, \quad \text{for some } \zeta > 0.$$

It turns out that, with these parameters and if $\sum_{j=1}^{n_i} x_{i,j}$ is large, the control laws exhibit a TCP-like behavior; i.e., if there is no congestion, the data rate increases linearly. If congestion is detected, the data rates decrease exponentially. More precisely, if no congestion is detected, one has $\dot{x}_{i,j} = \zeta$. If congestion is detected, since it is assumed that the data rate is large

$$e^{\alpha \partial U_i/\partial x_{i,j}} \approx 1 + \frac{\alpha}{\sum_{j=1}^{n_i} x_{i,j}}$$

and, hence,

$$\dot{x}_{i,j} \approx -\frac{\zeta}{\alpha} \sum_{j=1}^{n_i} x_{i,j}.$$

In other words, in multipath case, a TCP-like congestion control law should decrease the sending window by an amount proportional to the aggregate data rate. Obviously, this reduces to the usual TCP algorithm if one just has one path.

VII. Implementation Issues

It is important to note that the new family of DCLs provides the much needed mathematical foundation which allows the use of source inferred congestion detection and notification to maintain layer abstraction. Also important is to realize that the new family of DCLs allows the rate control to be decoupled from the congestion detection mechanisms in use. This means that any queue management algorithm and queue scheduling discipline used in the core nodes, can coexist with the family of DCLs running at the edge nodes or end-hosts. In other words, the implementation of any DCL in this family only needs to consider the two end nodes, provided that a source inferred congestion detection and notification is available. However, having said that, one must realize that different queue management algorithms and queue scheduling disciplines do have an impact on the overall performance for any end-to-end traffic control mechanism (see [3]).

As a result, there are two key components in the implementation of the family of DCLs; i.e., the implementation of the DCL in the edge nodes or end-hosts and the design of source inferred congestion detection and notification mechanisms. The implementation of the DCL control plane and data plane functions in the edge nodes or end hosts are similar to the one described in [20]. In this paper, we focus on the issues related to the design of source inferred congestion detection and notification mechanisms.

Note that due to the wide applicability of the new family of DCLs with respect to rate adaptation, multi-path load balancing, and multiple CoSs, for both connectionless and connection-oriented networks, it is difficult to address detailed implementation issues, unless the network architecture to which the DCL applies is defined. In what follows, we only discuss the general aspects of the implementation issues.

A. Discretization, Delays and Quantization

When implementing the control laws developed in this paper, one is faced with several issues: First, one has to implement a discrete time version of the control algorithms. Second, usually one uses finite word length which leads to a quantization of the possible data rate values. Finally, there is delay in the propagation of the congestion information. All of these lead to a well known phenomenon: Oscillation. Even in this case, the discretization of the control laws presented in this paper is approximately optimal. We now state the precise result.

Proposition 1: Let $\mathbf{x}(t)$ be the trajectory obtained using the control laws in Section V and let $\mathbf{x}^r(t)$ be the corresponding discrete time trajectory obtained using the discretization algorithm above and in the presence of delays in the propagation of the congestion information. Let t_r be an upper bound on the largest delay and t_d be the discretization period. Again, define $\mathscr{X}$ as in Theorem 1.

Given any time interval $[t_0, t_1]$ and constant $\varepsilon > 0$, there exists a $\delta > 0$ such that if

$$\max\{t_d, t_r\} z_{i,j}(t, \mathbf{x}) < \delta$$

for all $t > 0$ and $\mathbf{x} \in \mathscr{X}$, then

$$\|\mathbf{x}(t) - \mathbf{x}^r(t)\| < \varepsilon$$

for all $t \in [t_0, t_1]$.

Proof: Direct application of result 2, page 95 of [6]. ■

Remark: One can sharpen the result above. More precisely, one can prove that the control laws proposed in this paper are asymptotically stable in the presence of both delays and discretization if the gains $z_{i,j}$ converge "slowly" to zero as $t \to \infty$.[3] However, in this case, the network would react very slowly to changes in operating conditions (such as change in traffic demand and/or link/node failure). Hence, this case is not studied further in this paper.

B. Congestion Detection and Notification

To mantain the transport or higher layers abstraction, a source inferred congestion detection and notification mechanism is desirable for the implementation of this family of DCLs in a connectionless IP network. However, unless the transport or higher layer protocol that implements this family of DCLs is defined, the exact source inferred congestion detection and notification mechanism cannot be decided. For example, if a DCL in this family is used in association with a TCP-like reliable transport protocol, a source inferred congestion detection and notification mechanism based on, for example, ACK counts can then be adopted. On the other hand, if the DCL is used in association with an UDP-like unreliable transport protocol, the forwarding path congestion

[3]Stability of networks under delays has been addressed by several authors; e.g., see [1], [21], [24]. However, these results, as opposed to the ones presented in this paper, require a "tight cooperation" between the sending nodes and the network routers.

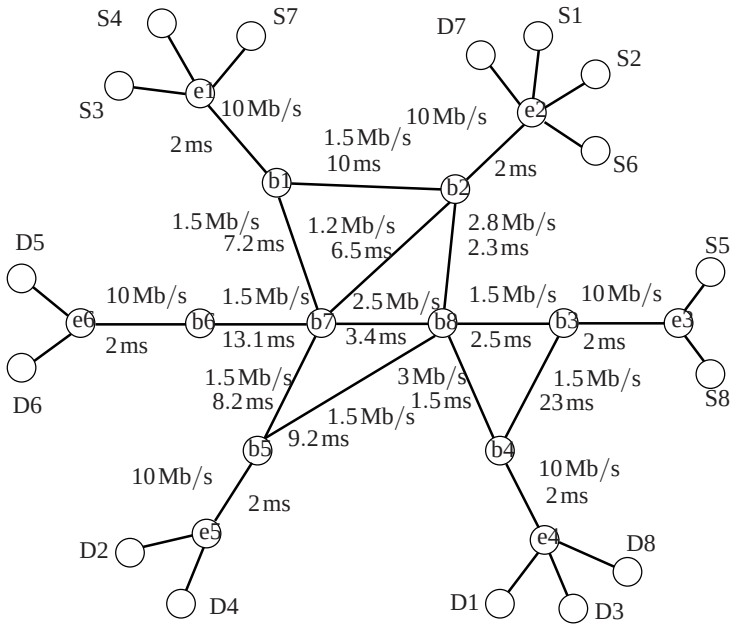

Fig. 2. Topology of the network

may be detected and notified by periodically sending an echo packet to the destination node and measuring the round-trip time of the echoed packet.

The above source inferred congestion detection and notification approaches can also be used in the context of a connection-oriented network, such as an MPLS one. In addition, other mechanisms can be employed; e.g., mechanisms using a signaling protocol for congestion detection and notification or the one described in [17].

C. Failure Detection and Notification

The node/link failure detection and notification may or may not be integrated with the congestion detection and notification mechanism. Again, they are dependent on the actual protocol that implements an DCL in this familiy. For example, a source inferred congestion detection and notification using echo packets to infer path congestion may also be used to infer possible node/link failures. On the other hand, in an MPLS network, the path protection mechanism under development [22] can be leveraged to allow failure detection and notification, separate from the congestion detection and notification mechanisms.

VIII. Simulation Examples

In this section simulation examples are presented, that help in the understanding of the behavior of the proposed control laws. In particular, it is shown that the control laws converge to the optimal traffic allocation while satisfying service requirements and that they provide an optimal way of reacting to link failures. These examples use a discrete-time version of the control laws and a flow approximation for the calls. Furthermore, for simplicity, only calls of AF and BE CoS categories are taken into account. Given the structure of the algorithm, the behavior with other CoSs will be similar.

A. Simulation Setup

The model of the network used for these examples is the same as in [17] which was originally used by La, et al. [16]. The topology is shown in Fig. 2 along with all link capacities and delays. There are overall $n = 8$ types of calls corresponding to the source/destination pairs indicated in the figure. The paths available for each one of these calls are indicated in Table I, where n_i is the number of paths available for calls of type i.

Utilization will be measured by the function

$$U(\mathbf{x}) = \sum_{i=1}^{8} 0.1 \log\left(0.5 + \sum_{j=1}^{n_i} x_{i,j}\right);$$

i.e., $\alpha = 0.1$, where n_i is again indicated in the table. The term 0.5 is included to avoid an infinite derivative at 0 data rate. As for the AF service requirements, calls of types $i = 3$ and $i = 5$ are assumed to have target rates $\Lambda_3 = \Lambda_5 = 1\,\mathrm{Mb/s}$.

Given this, the control laws presented in Section V are of the following form: For $i = 1,3$ and $j = 1,2$; i.e., AF calls

$$\dot{x}_{i,j} = z_{i,j}(t,\mathbf{x})\left[\left(1 - e^{-0.1\left(\sum_{j=1}^{n_i} x_{i,j} + 0.5\right)^{-1}}\right) - \left(1 - \overline{cg_{i,j}} r_i\right)\right]$$

and for $i = 1,2,4,6,7,8$ and $j = 1,\ldots,n_i$; i.e., BE calls

$$\dot{x}_{i,j} = z_{i,j}(t,\mathbf{x})\left[\left(1 - e^{-0.1\left(\sum_{j=1}^{n_i} x_{i,j} + 0.5\right)^{-1}}\right) - \left(1 - \overline{cg_{i,j}}\right)\right],$$

where r_i was chosen with a margin of ± 0.001 with respect to the bounds set forth in Theorem 1. The same oscillation reduction scheme as in [17] was used with $z_{i,j}$ taken as $z_{i,j}(t) = \omega(t - t_0)$, where $\omega(t) = 1.8(0.25 +$

TABLE I

PATHS AVAILABLE FOR EACH TYPE OF CALLS

type 1 - $n_1 = 4$	type 2 - $n_2 = 3$	type 3 - $n_3 = 2$	type 4 - $n_4 = 4$
$x_{1,1} : e_2b_2b_8b_4e_4$	$x_{2,1} : e_2b_2b_8b_5e_5$	$x_{3,1} : e_1b_1b_7b_8b_4e_4$	$x_{4,1} : e_1b_1b_7b_5e_5$
$x_{1,2} : e_2b_2b_8b_3b_4e_4$	$x_{2,2} : e_2b_2b_7b_5e_5$	$x_{3,2} : e_1b_1b_2b_8b_4e_4$	$x_{4,2} : e_1b_1b_7b_8b_5e_5$
$x_{1,3} : e_2b_2b_7b_8b_3b_4e_4$	$x_{2,3} : e_2b_2b_1b_7b_5e_5$		$x_{4,3} : e_1b_1b_2b_7b_5e_5$
$x_{1,4} : e_2b_2b_7b_8b_4e_4$			$x_{4,4} : e_1b_1b_2b_8b_5e_5$
type 5 - $n_5 = 2$	type 6 - $n_6 = 3$	type 7 - $n_7 = 3$	type 8 - $n_8 = 2$
$x_{5,1} : e_3b_3b_8b_7b_6e_6$	$x_{6,1} : e_2b_2b_1b_7b_6e_6$	$x_{7,1} : e_1b_1b_2e_2$	$x_{8,1} : e_3b_3b_4e_4$
$x_{5,2} : e_3b_3b_4b_8b_5b_7b_6e_6$	$x_{6,2} : e_2b_2b_8b_7b_6e_6$	$x_{7,2} : e_1b_1b_7b_2e_2$	$x_{8,2} : e_3b_3b_8b_4e_4$
	$x_{6,3} : e_2b_2b_7b_6e_6$	$x_{7,3} : e_1b_1b_7b_8b_2e_2$	

0.65^t). Finally, discretization of the continuous was done using a backward rule approximation: Let $\dot{x}_{i,j} = g_{i,j}(\mathbf{x},t)$ denote the continuous time laws derived in Section V. Then the discrete-time counterpart is

$$x^d_{i,j}\big[(k+1)t_d\big] = x^d[kt_d] + t_d g_{i,j}\big(\mathbf{x}(kt_d), kt_d\big);\ k = 0,1,\ldots,$$

The discretization step was chosen as $t_d = 5\,\text{ms}$ and the resetting interval as $T = 10\,\text{s}$.

The simulation results are shown in Fig. 3. It can be seen that the utility function converges to a value close to the optimal one, while satisfying the AF requirements imposed on calls of types $n = 3$ and $n = 4$. Calls of BE category of type $n = 2$ are also included as an example of the obtained behavior.

It can be seen that the trajectory of the data rates exhibits an oscillatory behavior. This phenomenon is due to non-ideal implementation factors such as delays and discretization (that were not considered in Section V). Furthermore, these factors prevent the algorithm from reaching the true optimum. Instead, convergence to a small neighborhood of the optimum is achieved. Section VIII-C provides some examples that show the sensitivity of these results to the choice of the parameters of the adaptation laws.

B. Robustness Against Link Failures

The control laws presented in this paper excel at re-routing traffic upon a failure in a node or link. In order to show this feature, the link connecting nodes b_7 and b_8 was opened at time $t_{fail} = 120\,\text{s}$. The behavior of the control laws is shown in Fig. 4 from time $t = 120\,\text{s}$ on. Note from Table I that both AF calls lose one of the

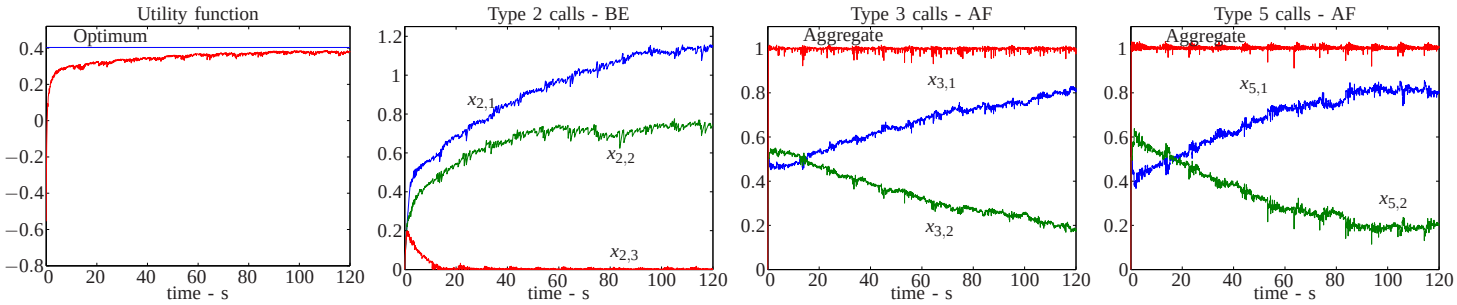

Fig. 3. Example of Simulation Results.

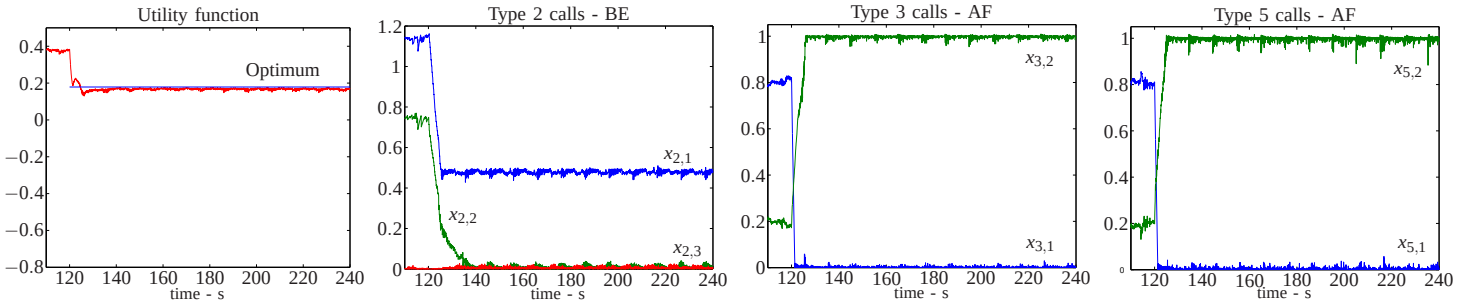

Fig. 4. Robustness Against Link/Node Failures.

two paths they have available so this can be considered to be an extreme situation. As an example, calls of type $i = 2$ have to "kill" all traffic on one of the available paths and greatly reduce another. Also, note that in the case when a source inferred congestion detection and notification is used for both congestion *and* failure detection and notification, the control laws implemented at the edge nodes are oblivious to the failure. They simply react to what they perceive as being congestion. In fact, these simulations do not attempt to detect link failure.

C. Sensitivity to the Design Parameters

In this section some relevant simulation are presented, showing the behavior of the algorithm under different choices of the design parameters.

1) Oscillation Reduction Functions: Perhaps one of the most important features of the adaptation laws presented in this paper is the adaptive oscillation reduction, since it has a big impact on performance. Fig. 5 show the behavior for a constant $z_{i,j} = \omega(0) = 2.25$; i.e., the maximum value allowed for the time-varying $z_{i,j}$. In comparison with Fig. 3 the observed oscillation is clearly larger in magnitude. Moreover, due to the larger oscillations, convergence to a larger neighborhood of the optimal is obtained and departures from the average

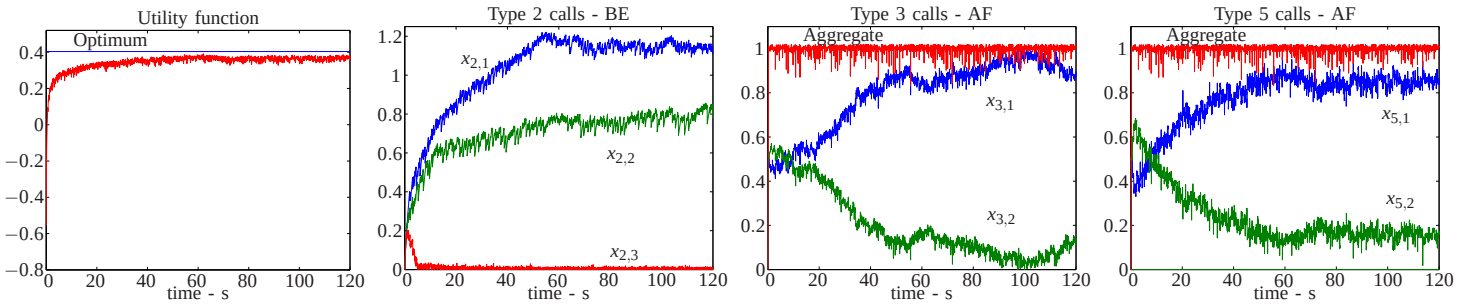

Fig. 5. Example of Constant $z_{i,j}$.

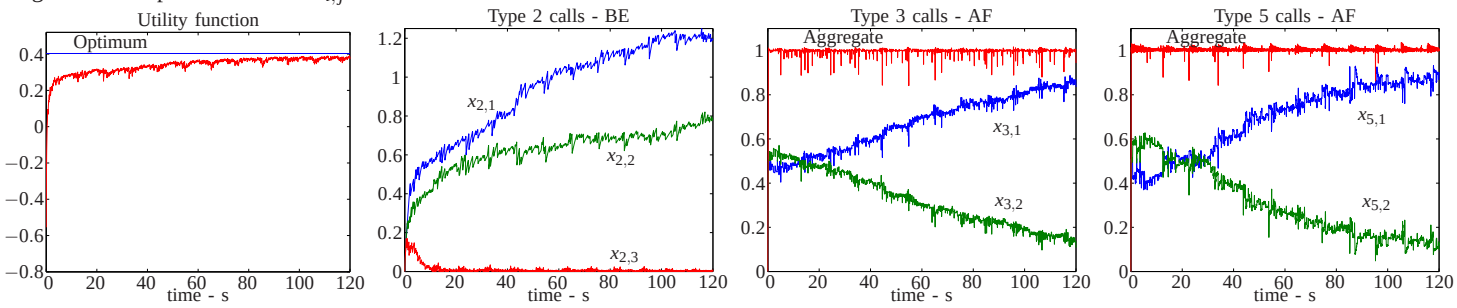

Fig. 6. Example of Larger t_d.

target rates for AF are also larger (providing a worse service to these users). On the other hand the transient response is faster due to larger data rate derivatives.

2) Discretization Step t_d: Another parameter that has a bearing in the performance of the algorithm is the discretization step. In order to show its influence, it was chosen as $t_d = 10\,\text{ms}$. Fig. 6 shows this scenario. Clearly, oscillations are also larger in this case. However, the response is still acceptable and a smaller $z_{i,j}$ could be used to limit the magnitude of the spikes.

3) Scaling of the Utility Function: The scaling of the utility function does not alter the solution of the optimization problem at hand. It does, however, change the bounds on the quantities r_i. Due to the exponential dependence on the gradient, it is advisable to choose a small value of α such that the resulting value of r_{max} is in the order of 1. Simulations have shown that the algorithm is very sensitive to α with the amplitude of the oscillations increasing substantially when one increases this parameter. However, convergence to a neighborhood of the optimal is still achieved as one can expect. Also, the AF constraints are satisfied in the average but large departures from the imposed average rate can happen for high values of α.

4) Propagation Delays: As mentioned before, delays in propagation of information result in an oscillating behavior. More precisely, an increase in the delays will result in a change of behavior similar to the one studied

on [17]. Depiction of this behavior is not presented here due to space constraints. The reader is referred to [17] for a more complete study of the influence of propagation delays.

IX. Conclusion

In this paper, a new family of distributed traffic control laws is obtained, which enables scalable quality of service, traffic engineering, and failure recovery features simultaneously, using only source inferrable information. More specifically, these features are enabled through fast timescale CoS-based, dynamic multipath load balancing and rate adaptation, performed by a set of control laws running at the edge nodes locally, independent of each other. Moreover, these control laws drive the network to a operation point where a global design objective is achieved; e.g., maximizing the network revenue. A salient feature of this family of control laws is that the input to each control law is whether a forwarding path in a multipath is congested or not. This feature allows a source node to infer the network congestion, without explicit feedback from the network core. This makes it possible to design a wide variety of highly scalable distributed traffic control protocols with proven optimality and stability.

Effort is now being put in the implementation of the control laws presented in this paper. In particular, these laws have several parameters for which only bounds are provided. Hence, criteria is now being developed for the determination of these "free parameters".

Appendix: Proof of Main Results

In this appendix, the proof of Theorem 1 is presented. We set the stage by introducing some additional notation. Due to space constraints, only the main steps of the proof are presented.

To simplify the exposition to follow, let the problem at hand be recast in the following form

$$\max_{\mathbf{x}} U(\mathbf{x})$$

subject to inequality constraints

$$h_k(\mathbf{x}) \leq 0; \quad k = 1, 2, \ldots, m$$

and the equality constraints

$$h_k(\mathbf{x}) = 0 \quad k = m+1, m+2, \ldots, L.$$

Now, let the admissible domain be defined as the set

$$\mathscr{C} = \left\{\mathbf{x} \in \mathbf{R}^N \colon h_k(\mathbf{x}) \leq 0 \text{ for } k \in \{1,2,\ldots,m\} \text{ and } h_k(\mathbf{x}) \text{ not a CoS constraint}\right\};$$

i.e., the set of data rates that can be admitted by the network without any further constraints. Also, let the feasible set be defined as

$$\mathscr{D} = \left\{\mathbf{x} \in \mathbf{R}^N \colon h_k(\mathbf{x}) \leq 0 \text{ for } k = 1,2,\ldots,m, \text{ and } h_k(\mathbf{x}) = 0 \text{ for } k = m+1, m+2, \ldots, L\right\};$$

i.e., the set of data rates $\mathbf{x} \in \mathscr{C}$ satisfying all the CoS constraints of the optimization problem.

The proof follows by first observing that the control laws in Section V converge to the admissible set in finite time. Then, once inside $\mathscr{C}$ the adaptation laws can be shown to be equivalent to the modified laws

$$\dot{\mathbf{x}} = \mathbf{Z}(\mathbf{x},t)\left[\nabla U(\mathbf{x}) - \mathbf{H}(\mathbf{x})\mathbf{v}(\mathbf{x})\right],$$

where $\mathbf{Z}(\mathbf{x},t)$ is a positive definite matrix and $\mathbf{H}(\mathbf{x}) = \left[\nabla h_1(\mathbf{x}), \nabla h_2(\mathbf{x}), \ldots, \nabla h_L(\mathbf{x})\right]$. In other words, they can be recast in the same form as that of the control laws developed in [17].

Lemma 1: Let r_i satisfy the conditions set forth in Theorem 1. Then vector $\mathbf{x}$ converges to the admissible domain $\mathscr{C}$ in finite time.

Proof: Let $x_{i,j} \geq 0$, for any given i and j, such that $\mathbf{x} \notin \mathscr{C}$ and let $\varepsilon > 0$ be an arbitrarily small constant. By construction of the control laws it holds that $\dot{x}_{i,j} \leq -\varepsilon < 0$. Hence, since the derivative is strictly negative outside $\mathscr{C}$, $x_{i,j}$ reaches the admissible region $\mathscr{C}$ in finite time. ■

The following Lemma, central to the proof of the results in this paper, provides an alternative representation of the proposed control laws.

Lemma 2: For all $\mathbf{x} \in \mathscr{C}$, the control laws above can be expressed as

$$\dot{\mathbf{x}} = \mathbf{Z}(\mathbf{x},t)\left[\nabla U(\mathbf{x}) - \mathbf{H}(\mathbf{x})\mathbf{v}(\mathbf{x})\right],$$

where $\mathbf{Z}(\mathbf{x},t)$ is a positive definite matrix and

$$\mathbf{H}(\mathbf{x}) = \left[\nabla h_1(\mathbf{x}), \nabla h_2(\mathbf{x}), \ldots, \nabla h_L(\mathbf{x})\right].$$

Proof: If at a given time $x_{i,j}$ is not sliding along the surface $x_{i,j} = 0$, the laws presented in Section V can be formulated as follows: Let $\mathscr{I}_{i,j}$ be the set of indices $k \in \{1,2,\ldots,m\}$ such that the capacity constraints

$h_k(\mathbf{x})$ involve the data rate $x_{i,j}$. Also, let $\mathscr{I}_{i,\mathrm{CoS}}$ be the set of indices $k \in \{1,2,\ldots,L\}$ such that the constraints $h_k(\mathbf{x})$, $k \in \mathscr{I}_{i,\mathrm{CoS}}$ impose CoS requirements on the data rate $x_{i,j}$. Note that this set is empty if calls of type i are of the BE class. Then,

$$\dot{x}_{i,j} = z_{i,j}\left[f_{i,j}(\mathbf{x}) - \left(1 - \prod_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,\mathrm{CoS}}} u_k\right)\right],$$

where the quantities u_k are defined as follows: For $k \in \mathscr{I}_i^{\mathrm{CoS}}$, $i = 1,2,\ldots,s_1$ (AF constraints) let $u_k \doteq r_i$. For $k \in \mathscr{I}_i^{\mathrm{CoS}}$, $i = s_1+1, s_1+2, \ldots, s_2$ (MGRS constraints) let $u_k \doteq r_i^m$. For $k \in \mathscr{I}_i^{\mathrm{CoS}}$, $i = s_2+1, s_2+2, \ldots, s_3$ (UBRS constraints) let $u_k \doteq r_i^M$. For $k \in \mathscr{I}_i^{\mathrm{CoS}}$, $i = s_3+1, s_3+2, \ldots, s_4$ (MRGUBS constraints) let $u_k \doteq r_i^m r_i^M$. Finally, for $k \in \mathscr{I}_{i,j}$, $i = s_4+1, s_4+2, \ldots, n$ (capacity constraints) let $u_k \doteq \overline{cg_{i,j}}$.

Given the formulation above and to prove that the control laws can be put in the form mentioned, two cases are considered: i) At the given time instant $x_{i,j}$ is sliding along the surface $x_{i,j} = 0$ and ii) At the given time instant, $x_{i,j}$ is not sliding along the surface $x_{i,j} = 0$.

Let us first consider case i). In this case, one has $\dot{x}_{i,j} = 0$ and the motion can be put in the form

$$\dot{x}_{i,j} = z_{i,j}(\mathbf{x},t)\left[\frac{\partial U}{\partial x_{i,j}} - \sum_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{\mathrm{CoS}}} \log\frac{1}{u_{k,\mathrm{eq}}} + \xi_{i,j,\mathrm{eq}}\right]$$

where $\xi_{i,j,\mathrm{eq}} \geq 0$ is such that

$$\xi_{i,j,\mathrm{eq}} = \sum_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{\mathrm{CoS}}} \log\frac{1}{u_{k,\mathrm{eq}}} - \frac{\partial U}{\partial x_{i,j}}$$

and $u_{k,\mathrm{eq}}$ are the equivalent controls (see [23]) corresponding to the constraints involving $x_{i,j}$. Note that, in this case, such $\xi_{i,j,\mathrm{eq}} \geq 0$ exists because one only has a sliding motion along $x_{i,j} = 0$ if the control laws presented would result in a non-positive derivative of $x_{i,j}$ and, hence,

$$\frac{\partial U}{\partial x_{i,j}} - \sum_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{\mathrm{CoS}}} \log\frac{1}{u_{k,\mathrm{eq}}} = -\log\big(1 - f_{i,j}(\mathbf{y})\big) + \log\left(\prod_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_i^{\mathrm{CoS}}} u_{k,\mathrm{eq}}\right) \leq 0$$

Now, let us consider case ii) where $x_{i,j}$ is not sliding along the surface $x_{i,j} = 0$. Now, since $f_{i,j}(\mathbf{x}_i) > 0$, when $\mathbf{x} \in \mathscr{C}$ either $\mathbf{x}$ is an inner point of $\mathscr{C}$ where by definition $u_k = 1$ for all $k \in \mathscr{I}_{i,j}$ or a sliding mode occurs on some surface $s(\mathbf{x}) = 0$, where $\mathbf{x} \in \partial\mathscr{C}$ (the boundary of $\mathscr{C}$). In the latter case, using the equivalent control method (see [23]) there exists $u_{k,\mathrm{eq}}$, such that

$$\dot{x}_{i,j}(t) = z_{i,j}(\mathbf{x},t)\left[-\big(1 - f_{i,j}(\mathbf{x})\big) + \prod_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{\mathrm{CoS}}} u_{k,\mathrm{eq}}\right].$$

Moreover, since $\max_{\mathbf{x}\in\mathscr{C}} f_{i,j}(\mathbf{x}_i) = \mu < 1$, then there exists a constant $\chi > 0$ such that $\chi < u_{k,\text{eq}}$, for all $\mathbf{x} \in \mathscr{C}$. For $k \in \mathscr{I}_{i,j}^{CoS}$ this is immediate since the lower bound on $u_{k,\text{eq}}$ is $r_{\min}^{M}$. If $k \in \mathscr{I}_{i,j}$, this is a consequence of the fact that $u_{k,\text{eq}} = 1$ if the constraint in not active. If the constraint is active, then one has to have

$$\prod_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{CoS}} u_{k,\text{eq}} \geq 1-\mu$$

for all i, j such that $k \in \mathscr{I}_{i,j}$ since, if this is not satisfied, $\dot{x}_{i,j} < 0$ for all i, j such that $k \in \mathscr{I}_{i,j}$ and one could not have a sliding mode along the boundary of constraint k.

Hence, given that the log function has a bounded derivative in the interval $\left[1-\mu, 1\right]$, the evolution of $x_{i,j}$ can be represented as

$$\begin{aligned}\dot{x}_{i,j} &= \hat{z}_{i,j}(\mathbf{x},t)\left[-\log\left(1-f_{i,j}(\mathbf{y})\right)+\log\left(\prod_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i}^{\text{CoS}}} u_{k,\text{eq}}\right)\right]\\ &= \hat{z}_{i,j}(\mathbf{x},t)\left[\log\frac{1}{1-f_{i,j}(\mathbf{y})}-\sum_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i}^{\text{CoS}}}\log\frac{1}{u_{k,\text{eq}}}\right] = \hat{z}_{i,j}(\mathbf{x},t)\left[\frac{\partial U}{\partial x_{i,j}}-\sum_{k\in\mathscr{I}_{i,j}\cup\mathscr{I}_{i,j}^{\text{CoS}}}\log\frac{1}{u_{k,\text{eq}}}\right],\end{aligned}$$

where $\hat{z}_{i,j}(\mathbf{x},t) = \gamma z_{i,j}(\mathbf{x},t) \geq \hat{\mu}$ and $\gamma \in \left[1-\mu, 1\right]$. This is a consequence of the fact that the Mean Value Theorem implies that $\log(a) - \log(b) = (a-b)/c$ for some $c \in [\min\{a,b\}, \max\{a,b\}]$.

Now, given the two cases addressed above, we note that $\xi_{i,j,\text{eq}}$ corresponds to a single value of k such that $h_k(\mathbf{x})$ imposes a non-negativity constraint on $x_{i,j}$ and, hence,

$$\dot{\mathbf{x}} = \mathbf{Z}(\mathbf{x})\left[\nabla U(\mathbf{x}) - \mathbf{H}(\mathbf{x})\mathbf{v}(\mathbf{x})\right],$$

where $\mathbf{v}(\mathbf{x})$ is a column vector containing the quantities $\log(1/u_{2,\text{eq}})$ and $\xi_{i,j,\text{eq}}$ ordered by k, and $\mathbf{Z}(\mathbf{x},t)$ is a positive definite diagonal matrix with elements $\hat{z}_{i,j}$. ■

We are now ready to prove convergence of the rate adaptation control laws. The line of reasoning is the same as in [17]. Hence, we refer the reader to [17] and [23] for proofs of the intermediate results presented below. Define the auxiliary function

$$\widehat{U}(\mathbf{x}) = U(\mathbf{x}) - \Xi(\mathbf{x}), \qquad \text{where} \qquad \Xi(\mathbf{x}) = \left[h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_L(\mathbf{x})\right]\mathbf{v}(\mathbf{x}).$$

Theorem 2: Let $\mathbf{v}^0$ be a vector whose entries are of the form

$$0 \leq v_k \leq \gamma_k\,; \quad k = 1,2,\ldots,m$$

$$-\psi_k \leq v_k \leq \psi_k\,; \quad k = m+1, m+2, \ldots, L\,,$$

where $v_k = 0$ for non-binding constraints. Then, the maximum of $\widehat{U}(\mathbf{x})$ coincides with the optimal $U(\mathbf{x}^*)$ if and only if there exists $\mathbf{x}^*$ such that $\nabla U(\mathbf{x}^*) = \mathbf{H}(\mathbf{x}^*)\mathbf{v}^0$.

Lemma 3: The function $\widehat{U}(\mathbf{x})$, for $\mathbf{x} \in \mathscr{C}$ does not decrease along the trajectories.

Lemma 4: The time derivative of $\widehat{U}(\mathbf{x})$, for $\mathbf{x} \in \mathscr{C}$, is zero only when $\dot{\mathbf{x}} = 0$.

Lemma 5: The stationary points of $\widehat{U}$ are the maximum points of $\widehat{U}$.

The results above imply the following.

Theorem 3: The control laws presented above converge to the set of maximum points of the utility function $U(\mathbf{x})$ if this set is bounded, the condition of Theorem 2 is satisfied and vector $\mathbf{v}^0$ is an inner point of the set defined in Theorem 2, except for the non-binding constraints.

We are now finally ready to address the proof of the main result in this paper.

A. Proof of Theorem 1

The definition of $f_{i,j}$ together with the conditions on r_{lower} and r_{upper} imply that the Lagrange multipliers at the KKT point of the optimization problem at hand lie in the convex hull generated by the set of all possible $\mathbf{v}$. Indeed, if each congested link is traversed by a BE call, then in the KKT conditions at the optimum $\mathbf{x}^*$

$$\nabla U(\mathbf{x}^*) = \mathbf{H}(\mathbf{x}^*)\mathbf{v}^0$$

the components of $\mathbf{v}^0$ associated with capacity constraints; i.e., v_k^0 for $k = 1,2,\ldots,\text{card}\,(\mathscr{L})$, appear in a set of equations decoupled from the remaining components of $\mathbf{v}^0$. Then, the worst case (larger) value of v_k^0, $k = 1,2,\ldots,\text{card}\,(\mathscr{L})$ is

$$v_{k,max}^0 = \max_{i,j,\mathbf{x}\in\mathscr{X}} \frac{\partial U(\mathbf{x})}{\partial x_{i,j}}$$

Now, using this information in the remaining equations, it is possible to solve for $v_{k,max}^0$, $k = \text{card}(\mathscr{L})+1,\ldots,L$. Since $U(\mathbf{x})$ is an increasing function in all its arguments $x_{i,j}$, the largest absolute value of v_k^0 associated with

CoS constraints is given by

$$v^0_{k,max} = \sum_{\kappa \in \mathscr{K}} v^0_{\kappa,max} = \max_{i,j} B_{i,j} \max_{i,j,\mathbf{x}\in\mathscr{X}} \frac{\partial U(\mathbf{x})}{\partial x_{i,j}},$$

where

$$\mathscr{K} \doteq \left\{ \kappa \colon \kappa \in \mathscr{I}_{i,j^*};\ j^* = \arg \max_{j=1,2,\ldots,n_i} \mathrm{card}(\mathscr{I}_{i,j});\ i \colon k \in \mathscr{I}_i^{\mathrm{CoS}} \right\}.$$

Finally, note that the multipliers for the non-negativity constraints appear in a single equation where all the others are already determined. Therefore, the worst case value is given by

$$v^0_{k,max} = 2 \max_{i,j} B_{i,j} \max_{i,j,\mathbf{x}\in\mathscr{X}} \frac{\partial U(\mathbf{x})}{\partial x_{i,j}}$$

Hence, it should hold that

$$\begin{aligned} &v_k \le v^0_{k,max} < \gamma_k; && k = 1,2,\ldots,m \\ &|v_k| \le \left| v^0_{k,max} \right| < \psi_k; && k = m+1,\ldots,L. \end{aligned}$$

That is: For capacity constraints $u_k < e^{-v_{k,max}}$, and for CoS constraints $u_k < e^{-v_{k,max}}$ and $u_k > e^{v_{k,max}}$. For the capacity constraints the condition is trivially satisfied with $u_k = 0$, while for COS constraints, these are the conditions imposed on r_{lower} and r_{upper}. Finally, for the positivity constraints, the condition is also satisfied since the equivalent control associated with these constraints $\xi_{i,j,\mathrm{eq}}$ can have arbitrarily high values (see proof of Lemma 2).

Therefore, Theorems 2 and 3 hold. Hence, the family of adaptation laws proposed in this paper converge to the maximum of the utility function $U(\mathbf{x})$ subject to $\mathbf{x} \in \mathscr{D}$. In other words, they converge to the optimum of the optimization problem at hand. ■

REFERENCES

[1] T. Alpcan and T. Basar, "Global stability analysis of an end-to-end congestion control scheme for general topology networks with delay," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 1, December 2003, pp. 1092–1097.

[2] F. Bonomi, D. Mitra, and J. B. Seery, "Adaptive algorithms for feedback-based flow control in high-speed, wide-area networks," *IEEE J. Select. Areas Commun.*, vol. 13, no. 7, pp. 1267–1283, Sept. 1995.

[3] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint admission control: Architectural issues and performance," in *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, Aug./Sept. 2000, pp. 57–69.

[4] D. Chiu and R. Jain, "Analysis of the Increase/Decrease algorithms for congestion avoidance in computer networks," *J. Comput. Networks and ISDN Syst.*, vol. 17, no. 1, pp. 1–14, June 1989.

[5] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "Mate: Mpls adaptive traffic engineering," in *Proceedings of IEEE INFOCOM'01*, vol. 3, Anchorage, AK, April 2001, pp. 1300–1309.

[6] A. F. Filippov, *Differential Equations with Discontinuous Righthand Sides*, ser. Mathematics and Its Applications. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1988.

[7] S. Floyd and T. Henderson, "The new Reno modification to TCP's fast recovery algorithm," IETF RFC 2582, Apr. 1999.

[8] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[9] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 118–124, Oct. 2002.

[10] S. J. Golestani and S. Bhattacharyya, "A class of end-to-end congestion control algorithms for the Internet," in *Proc. Int. Conf. Network Protocols, ICNP*, Austin, TX, USA, Oct. 1998, pp. 137–150.

[11] T. Guven, C. Kommareddy, R. La, M. Shayman, and B. Bhattacharjee, "Measurement based optimal multi-path routing," in *Proceedings of INFOCOM'04*, vol. 1, March 2004, pp. 187 – 196.

[12] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Overlay TCP for multi-path routing and congestion control," in *ENS-INRIA ARC-TCP Workshop*, Paris, France, Nov. 2003. [Online]. Available: http://comm.csl.uiuc.edu/~srikant/Papers/multipath_v3.pdf

[13] K. Kar, S. Sarkar, and L. Tassiulas, "A simple rate control algorithm for max total user utility," in *Proc. IEEE INFOCOM 2001*, vol. 1, Anchorage, AK, USA, Apr. 2001, pp. 133–141.

[14] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, Mar. 1998.

[15] S. K. Korovin and V. I. Utkin, "Using sliding modes in static optimization and nonlinear programming," *Automatica*, vol. 10, no. 5, pp.

525–532, Sept. 1974.

[16] R. J. La and V. Anantharam, "Charge-sensitive TCP and rate control in the Internet," in *Proc. IEEE INFOCOM 2000*, vol. 3, Tel Aviv, Israel, Mar. 2000, pp. 1166–1175.

[17] C. M. Lagoa, H. Che, and B. A. Movsichoff, "Adaptive control algorithms for decentralized optimal traffic engineering in the Internet," *IEEE/ACM Trans. Networking*, vol. 12, no. 3, pp. 415–428, June 2004.

[18] S. H. Low and D. E. Lapsley, "Optimization flow control, I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–874, Dec. 1999.

[19] D. Mitra and K. Ramakrishnan, "A case study of multiservice, multipriority traffic engineering design for data networks," in *Proc. GLOBECOM '99*, vol. 1b, Rio de Janeiro, Brazil, Dec. 1999, pp. 1077–1083.

[20] B. A. Movsichoff, C. M. Lagoa, and H. Che, "Decentralized optimal traffic engineering in connectionless networks," *IEEE J. Select. Areas Commun.*, vol. 23, no. 2, pp. 293–303, Feb. 2005.

[21] F. Paganini, Z. Wang, J. C. Doyle, and S. H. Low, "Congestion control for high performance, stability, and fairness in general networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 1, pp. 43–56, 2005.

[22] V. Sharma and F. Hellstrand, "Framework for Multi-Protocol Label Switching (MPLS)-based recovery," IETF RFC 3469, Feb. 2003, editors.

[23] V. I. Utkin, *Sliding Modes in Control and Optimization*, ser. Communications and Control Engineering Series. Berlin, Heidelberg: Spriger-Verlag, 1992, vol. 66, no. 1.

[24] L. Ying, G. E. Dullerud, and R. Srikant, "Global stability of internet congestion controllers with heterogeneous delays," in *Proceedings of the 2004 American Control Conference*, vol. 4, 2004, pp. 2948–2953.

A Unified, End-to-end Traffic Control Protocol for Elastic, Real-time Delay Adaptive and Real-time Rate Adaptive Services

Lei Ye[1], Zhijun Wang[1], Hao Che[2], and Constantino M. Lagoa[3]
[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[2]Department of Computer Science and Engineering, University of Texas at Arlington, TX, USA
[3] Department of Electrical Engineering, The Pennsylvania State University, PA, USA
Emails: {csyelei, cszjwang}@comp.polyu.edu.hk, hche@cse.uta.edu, lagoa@engr.psu.edu

***Abstract*—Existing end-to-end congestion control mechanisms used in TCP and variations of TCP were designed empirically, without provable properties, such as stability and optimality. The aim of this paper is to demonstrate, for the first time, that QoS-aware, end-to-end traffic control protocols can be designed based on theory, not empirically. In particular, using a utility-based design methodology developed in this paper, we were able to demonstrate that a unified, end-to-end traffic control protocol can be developed to support Real-time Delay Adaptive (RDA), Real-time Rate Adaptive (RRA), as well as Non-Real-time Elastic (NRE) applications. Moreover, it degenerates to the end-to-end TCP congestion control protocol when applying to NRE applications. This protocol possesses several provable properties, including "friendliness" to TCP, stability, and optimality. Both analytical and simulation studies of a window-based implementation of this protocol show that the protocol can provide soft minimum rate guarantee for real-time applications while being friendly to NRE applications and resilient to network resource shortages, caused by, e.g., link/node failures. As part of the protocol design, we also derived a utility function for TCP and the associated traffic control law, which, to the best of our knowledge, is the most accurate utility-based TCP model that captures major TCP behaviors.**

I. Introduction

The congestion control mechanism used in the Transmission Control Protocol (TCP) at the transport layer is a fully distributed, end-to-end traffic control mechanism. It relies solely on single-bit binary information feedback as input for the control, i.e., whether the forwarding path is congested or not. This binary information is acquired on the basis of source inferable information only, such as repetitive acknowledgements (ACKs) of the same segment, measured round-trip delay, and/or ACK timeout, without the assistance of the network nodes for the control and regardless of the link technology and queuing mechanism used in the network nodes. This has made the proliferation of the Internet applications at global scale possible. An excellent example is the swift, ubiquitous adoption of World Wide Web due to its use of TCP as its underlying transport.

However, as the Internet has evolved into a global commercial infrastructure, there has been a growing demand for the Internet to support various Hard Real-Time (HRT), Real-time Delay Adaptive (RDA) and Real-time Rate Adaptive (RRA) applications, such as voice over IP, IPTV (e.g., BBC iPlayer [20]) and multimedia streaming. Existing transport layer protocols, including TCP and User Datagram Protocol (UDP), are not well suited to support such applications.

A difficulty in designing end-to-end, transport layer traffic control protocols, in addition to TCP and UDP, is how to achieve desired performance in terms of user satisfaction, minimal impact on the existing TCP flows, and globally stable control, in the face of a highly interactive, end-to-end controlled mixture of traffic flows. The existing transport layer traffic control mechanisms, such as TCP, variations of TCP, and TCP-friendly protocols, were largely designed empirically. A consequence of such a design approach is that it is left unknown or difficult to reason whether or not the network will ever converge to a stable state and what kind of fairness/friendliness these protocols bring to themselves as well as TCP. Although the Internet has so far been more or less stable with the majority of applications running over TCP, things can easily get out of control if more and more empirically designed, end-to-end protocols are introduced into the Internet.

A promising approach, called utility-based approach in this paper, is to formulate the above problem as a distributed optimization problem aiming at achieving a given design objective, such as the maximization of the sum of individual user utilities (utility-maximization for short, e.g., [38]) or user utility max-min, e.g., [6]. The targeted solution is a set of end-to-end control laws governing the behaviors of individual user flows that drive the network to an operational state where the design objective is achieved.

Although optimal results exist, there are some fundamental issues concerning the traditional utility-based approach, which make it difficult to apply the existing results to the design of end-to-end protocols to support real-time applications. First, utility-based approach attempts to encode all the desired effects, such as QoS and fairness/TCP friendliness, implicitly in the utility function. This complicates the design of utility functions. Second, attempting to achieve a global design objective means that the achievable user utility is, in general, a complex function of the traffic load and pattern, making it difficult to quantify the QoS features of a given service, especially a real-time service. Third, as observed in this paper, the effective utility function for TCP (to be derived later) is a function of the current traffic load and pattern, making it even harder to design utility functions to enable new services, while being friendly to TCP, a dominant end-to-end transport layer protocol today. Finally, since the actual price charged to the user for using a given service will have an impact on

the user satisfaction of the service received, how the utility function design should be related to the pricing structure is still an open issue.

In this paper, we propose a new utility-based protocol design methodology. This methodology successfully addresses the above issues concerning the traditional utility-based approach. It leads to the design of a unified, end-to-end traffic control protocol to support RDA, RRA, as well as NRE applications/services, defined by Shenker [38]. Different services are enabled by a proper setting of a single parameter in the protocol only. This protocol is a generalization of the end-to-end TCP congestion control mechanism and degenerates to TCP when applied to NRE applications. This protocol enables real-time services similar to the controlled-load service defined in the IntServ architecture [21]. Namely, while behaving like TCP sessions when the traffic load is light, it outperforms the TCP sessions by maintaining a soft minimum guaranteed rate when the traffic load is heavy. Different services enabled by this protocol are fair/friendly to one another in the sense that they compete for network resources fairly, provided that the minimum guaranteed rates for real-time services are achieved. In the meantime, the protocol is also resilient to resource shortages, caused by, e.g., link/node failures or over commitment of network resources. In other words, it has the ability to reduce its flow rate to be lower than its soft minimum guaranteed rate, ensuring that the Internet will not experience partial or complete shutdown in the presence of resource shortages.

The proposed protocol is the first of its kind, in the sense that it is developed based on theory, i.e., a systematic design methodology and a well-grounded mathematical foundation. As a result, it enjoys some desirable and provable properties, including fairness, stability, and optimality. The design of the proposed protocol also leads to the finding of a utility function corresponding to the end-to-end TCP congestion control mechanism, which is by far the most accurate utility function that captures major TCP behaviors.

The rest of the paper is organized as follows: Section II reviews related work. Section III introduces the utility-based design methodology. Section IV derives and verifies the utility function of TCP. Section V develops a unified, end-to-end transport layer protocol based on the results in the previous two sections. Section VI provides the performance evaluation of the unified transport control protocol. Finally, Section VII concludes the paper.

II. Related Work

This section is not intended to provide an exhaustive survey of the literature on transport layer protocol design and research. Instead, it focuses on reviewing literature pertaining to the development of optimization-based approaches for distributed traffic control. We first review the literature on TCP related approaches and then cover the literature on optimization-based, QoS-aware, distributed protocol design.

The end-to-end TCP congestion control mechanism was developed empirically. There were quite a few existing mathematical techniques that can faithfully model TCP behaviors, notably, the models given in [28] [35]. However, these techniques are descriptive by design, meaning that while being able to faithfully mimic the TCP behaviors, they provide no knowledge about the underlying design objective of TCP and its convergence and stability properties. Researchers also made significant efforts to develop variations of TCP congestion control mechanisms (e.g., TCP Vegas [5], TFRC [19], DCCP [25]) and TCP-friendly protocols (e.g., see [23] and the references therein). However, similar to the congestion control mechanism of TCP, the congestion control mechanisms provided by these protocols were designed empirically without provable properties.

Significant research efforts [1,6-18,24,26-31,33,36] have been made on the development of optimization-based, distributed traffic control laws that achieve certain global design objectives (e.g., the utility-based approach, including utility-maximization, i.e., maximization of the sum of individual user utilities [38], and utility max-min [6]). Some work focuses on the understanding of TCP behaviors from the utility-maximization point of view [14][15][30][41]. In their seminal work, Kelly et al. [14] demonstrated that a utility function of $log(x)$ form (x is a flow data rate) leads to the so called proportional fairness, or proportional-fair allocation of the network resources, and a control law that exhibits additive-increase-and-multiplicative-decrease (AIMD) behavior, resembling the TCP behaviors. However, as pointed out in [14], the proportional-fair allocation may not always be in favor of flows with small round-trip times (RTTs) in terms of rate allocation, a typical characteristic of TCP behavior. This means that TCP does not implement proportional fairness. Nevertheless, the work in [14] has triggered significant research interests in an attempt to further understand the TCP behavior from an optimization point of view. In [41], by constructing a utility function inversely proportional to RTT, Vojnovic, et. al. was able to show that the control law for utility-maximization not only exhibits AIMD behavior but also is in favor of flows with smaller RTT values. In [15], Kunniyur, et al. showed that the TCP behavior without the slow start phase can be modeled using a framework based on a variation of the utility-maximization approach. By taking into account the randomness of packet loss, their model leads to a TCP utility function of the form $-1/x$ as x becomes large, similar to the one used in [41]. Low [30] studied various variations of TCP using a primal-dual nonlinear programming technique, that solves the utility-maximization problem. While the primal algorithms (i.e., control laws) capture the TCP window control behavior (without the slow start phase) at the TCP source node, the dual algorithms translate into given active queue management (AQM) algorithms running at the network nodes.

The fundamental design issues regarding the development of QoS services over the Internet was studied by Shenker [38], from a utility-based perspective. This utility-based approach associates each application-based flow with a user utility function $u(x)$, measuring the degree of satisfaction the user perceives at the allocated flow rate x. The design objective is to maximize the sum of individual user utilities, i.e., utility-

maximization. The rationale behind this approach is the fact that the user satisfaction should be the ultimate performance measure of QoS, rather than network centric performance metrics. In [38], Shenker identified four service classes or user utilities to meet various application needs, i.e., NRE, HRT, RDA, and RRA, as shown in Fig. 1. These utility functions reflect to what degree a user is satisfied with the service at a given allocated rate. A few results are available on the design of utility-based, QoS-aware congestion control protocols [6][17] [40]. In [17], Harks, et al. proposed a utility function construction method and used the second order utility optimization to do congestion control, which ensures that the real-time and best effort flows approach the utility proportional fairness. This framework, however, is tied to the AQM mechanism used in the routers and hence is not an end-to-end solution. In [40], Wang, et al. designed a distributed flow control algorithm to drive the network into the proportional (or max-min) fair state. This distributed flow control algorithm requires the feedback of the link price from the network nodes in the forwarding path. Hence again, it is not an end-to-end protocol. In [6], Cao, et al. found distributed control laws that achieve the utility max-min design objective. Again, the solution requires explicit information feedbacks from the network nodes and hence, is not an end-to-end solution.

A critical drawback of the existing solutions mentioned above is that they are point solutions with limited design scopes, such as modeling of TCP, modeling of variations of TCP tied to specific AQM mechanisms, QoS-aware control protocol that achieves a specific design objective, such as min-max or proportional fairness. Moreover, most of the existing solutions require significant involvement of network nodes for the control and hence are not end-to-end solutions. Very recently, based on the Sliding Mode technique in control theory, Movsichoff, et al, [34] found a large family of optimal, end-to-end, distributed control laws that maximize the sum of any user utility functions of concave form (i.e., a utility-maximization approach). Unlike the traditional utility-based approach attempting to encode all the QoS features implicitly into the utility functions, the solution in [34] allows some key QoS features to be expressed explicitly as flow-level constraints, in addition to the network resource constraints, in the utility-maximization problem. These constraints include target rate, minimum rate, and upper-bounded rate. This solution provides us with the much needed mathematical foundation underlying the proposed utility-based design methodology.

III. Utility-based Design Methodology

In this section, we first elaborate on the aforementioned fundamental issues concerning the traditional utility-based approach. Then we propose a new utility-based QoS design methodology aiming at addressing these issues, which leads to the design of the proposed protocol.

A. Issues Concerning Traditional Utility-Based Approach

As mentioned in the Section 1, there are four fundamental issues concerning the utility-based approach, including (a) how to design utility functions to enable desired service features; (b) how to quantify QoS features; (c) how the utility-based approach and the pricing structure are related; and (d) how to deal with the time-varying TCP utility function. In what follows, we elaborate more on these issues.

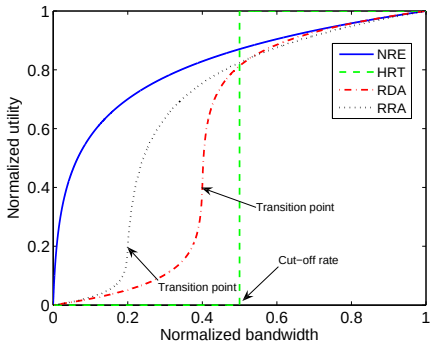

Fig. 1. Four types of utility functions

Utility Function Design and Quantification of QoS Features: By the traditional definition, a user utility function measures to what degree a user is satisfied with a given service provided to him/her. It does not provide any information about the relative degrees of user satisfaction across different service classes. This is not an issue when network resources are not shared by flows from different service classes. However, it becomes one when flows from different service classes have to compete with one another for the network resources. In such a case, the relative value ranges and shapes of different utility functions will have a strong effect on the final rate allocation. This makes the design of utility functions for different service classes difficult. Moreover, for the traditional utility-based approach, the final rate allocated to a flow, whether it is non-real-time or real-time, is dependent on the overall traffic load and pattern. This makes it difficult to quantify the QoS features for real-time service classes. Unfortunately, to the best of our knowledge, no literature has explicitly addressed these issues.

To understand the above issues better, let us take a look at the traditional NRT, RDA, RRA, and HRT utility functions and assume that they are all normalized, as given in Fig. 1, i.e., the degree of user satisfaction reaches one when the received rate goes to infinity and zero when the received rate is zero for all services. We argue that the utility-based approach using such utility functions may lead to undesirable resource allocations. For example, in the case of the utility-maximization, with the utility functions in Fig. 1, an HRT flow may not be allocated any link bandwidth, if it attempts to share the bandwidth of a link with a large number of NRE flows. This is simply because an NRE flow can achieve a rather high utility value at a rate much lower than the cut-off rate for an HRT flow and hence, the sum of the user utilities is maximized when the link bandwidth is allocated to NRE flows only. For the similar reason, both RDA and RRA flows may also perform poorly in the presence of a large number of NRE flows.

The situation can be even worse when the utility max-min

design objective which attempts to equalize the utility values for all the flows is used. With an HRT flow attempting to share a link bandwidth with a large number of NRE flows, a feasible solution may not even exist. On one hand, we note that an HRT flow receives either zero utility or full utility (i.e., value 1) depending on whether the allocated bandwidth is less or no less than its cut-off rate. As a result, either all the flows receive utility value 1, which is obviously impossible (note that a NRE flow achieves utility value 1 at the expense of an extremely high bandwidth allocation as evidenced in Fig. 1), or all the flows receive zero bandwidth. Similar problem occurs when a RDA or RRA flow coexists with a large number of NRE flows. In this case, the former is likely to experience poor performance (i.e., low utility) while consuming a sizable amount of link bandwidth, causing poor performance for NRE flows as well.

Relation to Pricing Structure: The above examples suggest that the traditional utility functions must, somehow, be re-designed to reflect the relative importance of different service classes. The relative importance between service classes must be related to the pricing structure in use. This makes sense from a user's perspective because a user generally will set his/her expectation of the service quality in proportion to the price he/she actually pays for the service. In other words, the user utility must be, to some extend, tied to the price the user pays for the service. This design objective also makes sense from the perspective of an Internet Service Provider (ISP) in the sense that the ISP would be willing to give better, higher priority service to users who pay more for the service. As a result, a utility-based solution should somehow account for such price-induced effects.

Based on the above understanding, a simple fix of the problem is to slightly modify the utility functions in Fig. 1 by allowing the value ranges of the utility functions to be proportional to the price paid by the user for the use of one unit of network resources (the exact functional relationship is not important for the current discussion). For example, if we magnify the value range of the RRA utility in Fig. 1 by, e.g., 2 times, the rate allocation will now be much more in favor of the RRA flows than the case with normalized utilities. Obviously, the need to incorporate the pricing effect in the utility function further complicates design of utility functions.

Interacting with TCP: Since TCP is a dominant transport layer protocol, the utility-based approach must take its impact on TCP performance into account, e.g., the rate allocation needs to be TCP friendly. An interesting observation made in this paper is that the effective TCP utility function is time-varying, i.e., it changes as traffic load fluctuates [1] (See next section for details). This observation suggests that to be TCP friendly, the time-varying components might need to be incorporated in the utility functions for new service classes as well. This however, makes it even harder for the development of new service classes.

[1]This may sound a bit odd since a utility function is meant to serve as a measure of the degree of user satisfaction and hence, it should not be dependent on the traffic load in the network. The reason for this being true is that the TCP congestion control mechanism was not designed with a user utility function in mind. As a result, it unintentionally leads to an effective TCP utility function whose value range and shape change from time to time.

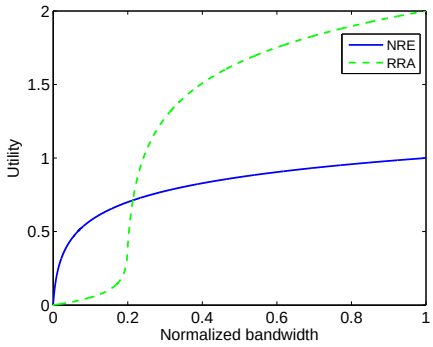

Fig. 2. Two types of utility functions with different scales

In summary, we conclude that for the utility-based approach to be practically useful for the design of new end-to-end transport layer protocols, the traditional utility function definition must be modified in some way to overcome the above difficulties. In the following subsection, we propose a new methodology for the utility-based approach aiming at overcoming these difficulties.

B. A New Methodology for Utility-based Approach

The root cause of the aforementioned difficulties can be mainly attributed to the idea of encoding all the desired service features into the utility functions, while attempting to achieve a global design objective involving all the user utilities. On one hand, achieving such a global design objective, whether it is utility-maximization or utility max-min, allows fair share of the network resources among flows, which is desirable. On the other hand, it means that the actual resources allocated to individual flows are a function of the overall traffic load and mixture. As a result, the desired service features of individual flows, including HRT features, may have to be compromised for the sake of fairness among all the flows, which is undesirable.

Basic Idea: A key idea in our methodology is to modify the traditional utility functions by encoding only the elastic part of the service features in them and letting the inelastic part of the service features be explicitly expressed as flow-level constraints. For example, the HRT utility function in Fig. 1 encodes only an inelastic service feature, i.e., totally satisfied or totally unsatisfied depending on whether the cut-off rate is achieved or not. In our approach, this inelastic feature is not encoded in the utility function but explicitly expressed as a flow-level constraint: $x =$ cut-off rate and the utility function encodes other effects if necessary (more on this later). As another example, for the service class corresponding to either RRA or RDA utility function in Fig. 1, the user satisfaction of the service grows slowly in the convex part of the utility and starts to increase faster in the concave part of the utility. Usually, for this kind of service, a minimum

guaranteed rate at the joint between the convex part and concave part of the utility function (i.e., the transition point in Fig. 1) needs to be allocated to a flow to ensure that the user is reasonably happy with the service. In other words, we can view the RRA/RDA utility function as composed of two parts: an inelastic part requiring a minimum guaranteed rate and an elastic part corresponding to the concave part of the utility function in Fig. 1. In our approach, the inelastic part is explicitly expressed as a flow-level constraint, i.e., x greater than the rate at the transition point and the utility function only encodes the concave part of the utility functions in Fig. 1 and other possible effects.

The implication of the above approach is significant. First, expressing inelastic features as flow-level constraints, or as the boundary conditions, in the problem space make these features explicitly quantifiable and not subject to traffic load and traffic mixture changes. This is also in better alignment with most existing pricing structures where the inelastic features are generally associated with some usage fees such as per unit guaranteed bandwidth usage charge. In return, the inelastic features must be protected from being compromised, which is enforced explicitly in our approach.

Second, involving only elastic components in the utility design makes it easier to justify and interpret the meaning of a specific choice of global design objective. For example, as we shall discuss shortly, in our protocol design, we choose to use the utility-maximization approach with all service classes having the same utility function as the effective TCP utility function. This can be easily interpreted as providing fair share of network resources for their elastic components. More specifically, all service classes are TCP friendly/fair, provided that the inelastic features are satisfied. Note that other effects, such as pricing effects corresponding to the elastic components can also be incorporated to further differentiate the services by e.g., adding a weight proportional to the monthly fee to the corresponding utility function.

Finally, as a byproduct of extracting the inelastic features from the utility functions, the flow rate corresponding to the convex part of the traditional utility functions in Fig. 1 is now outside the problem space. In other words, in the problem space constrained by the network resource and flow-level constraints, the utility functions are concave, freeing us from having to deal with a difficult non-convex optimization problem.

An Example Solution: Note that the HRT service cannot be supported by the protocol developed in this paper. Our position is that the end-to-end transport layer protocol alone can only provide soft performance guarantee (see Section V for definition). Any HRT performance guarantee calls for sophisticated call admission control and resource reservation involving network nodes.

We assume that TCP will continue to be a dominant transport layer protocol to support NRE applications. Our aim is then to design a unified, end-to-end transport layer protocol that generalizes TCP for the support of RDA and RRA applications. This protocol degenerates to TCP congestion control when applied to the NRE flows.

Here we note that a real-time service user who receives the minimum guaranteed rate is reasonably happy with the service already. This makes it less important as to how to design the utility functions (i.e., the elastic parts of the utility functions in Fig. 1) to further differentiate the services. Instead, we simply let all three services share the same utility function[2]. As we shall show later, this approach leads to an end-to-end traffic control protocol that *unifies* the controls for all three services. Service differentiation for the three services is achieved through a proper setting of a single control parameter in the protocol.

Since the NRE service has been supported by TCP with great success, we simply adopt the TCP utility function (to be derived shortly) to be shared by all three service classes. This solution unifies the existing end-to-end TCP congestion control with the control of the other two service classes, rendering the TCP congestion control a special case of the proposed protocol. Moreover, this solution addresses the difficult issue as to how to cope with a dynamically changing TCP utility function when designing a new protocol on the basis of the utility-based approach. With this solution, all three service classes are subject to the same dynamics of network conditions for the sharing of the network resources. As such, the term 'fairness/friendliness' is now well defined. Namely, a TCP flow receives fair share of the network resources with the elastic parts of the flows belonging to the other two service classes, which also share network resources fairly among themselves.

Note that the negative impact on TCP performance due to the introduction of minimum guaranteed rates for real-time services must, in principle, be avoided or mitigated by a call admission control mechanism at global scale. In other words, the minimum guaranteed rates should not be severely compromised in the name of having to be friendly/fair to TCP. Otherwise, any attempt to enable meaningful QoS features and pricing models is guaranteed to fail. Nevertheless, in this paper, we demonstrate that due to the softness of control, the current solution does not require call admission control to work properly.

Based on the above design methodology and the mathematical framework in [34], in the following two sections, the proposed protocol is designed.

IV. TCP Utility Function

Since all three service classes will share the same TCP utility function, in this section, we derive the TCP utility function based on the mathematical framework in [34]. Note that the existing utility-based models of TCP mentioned in Section II cannot capture the slow start phase (i.e., multiplicative-increase-and-multiplicative-decrease (MIMD) behavior of TCP) and the end-to-end nature of TCP. In this section, we aim at establishing a more accurate

[2]More elaborate utility can be used, by, e.g., adding a price-dependent weight to each utility in the utility-maximization objective

utility-based model for TCP that not only captures the AIMD behavior in the congestion avoidance phase, but also MIMD behavior in the slow start phase, as well as end-to-end nature of TCP, among other things.

A. Main Results From [34]

The problem can be stated as follows:

$$max \sum_{i=1}^{F} U_i(x_i) \tag{1}$$

subject to network constraints

$$\sum_{i:l\in Ł} x_i \leq B_l, \qquad l \in Ł \tag{2}$$

where $U_i(x_i)$ is the utility function for flow i; F is the total number of flows in the network; x_i is the rate of flow i; Ł is the set of links in the network; and B_l is the bandwidth of link l in Ł.

The distributed control law [34] that solves the above problem is given by (for simplicity, we omit the index i),

$$\dot{x} = (z(t,x)[f(x) - (1 - \bar{cg} \times r(x))])^{+}_{x=0} \tag{3}$$

with

$$f(x) = 1 - e^{-\partial U(x)/\partial x} \tag{4}$$

and

$$(y)^{+}_{x=0} = \begin{cases} max(y,0) & \text{if } x = 0 \\ y & \text{if } x \neq 0 \end{cases} \tag{5}$$

where $U(x)$ is a differentiable concave and strictly increasing function of x; $z(x,t)$ can be any strictly positive function, satisfying some loose condition (see [34]); cg is the binary congestion indicator (cg=1 if the packet forwarding path is congested and 0 otherwise); $\bar{cg}$ is the logical negation of cg; and $r(x)$ is a scalar factor, taking different values for different types of traffic. It is 1 for Best Effort (BE) flow. For flow with Minimum Rate Guarantee (MRG), i.e., a flow-level constraint: $x > \theta$ is added to the optimization problem in (1) and (2), $r(x)$ is given by:

$$r(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ r^m_{max} > 1 & \text{if } x < \theta \end{cases} \tag{6}$$

Here r^m_{max} is a tunable parameter. Note that the above family of control laws applies to any concave utility function; enable minimum rate guaranteed services (for how to use these control laws to enable target rate and upper bounded rate services, please refer to [34]); and allow flexible engineering of $z(x,t)$ function to realize various control behaviors. Moreover, for all the control laws in this family, the minimum non-local information required for the control is simply a discontinuous, aperiodic, single-bit binary indicator, cg, indicating whether the forwarding path is congested or not. This makes it possible to implement this family of control laws end-to-end. Also, this family of control laws is globally stable and optimal, as shown in [34]. It is also shown in [34] that the corresponding family of control laws in the discrete time domain with a finite time interval lead to stable and near optimal control. Clearly, this family of control laws sets the much needed theoretical foundation underlying proposed QoS design methodology.

B. Utility Function

On one hand, with any given concave user utility functions, a family of distributed control laws with different positive $z(x,t)$ functions can be readily derived from Eqns. (3)-(5), which will drive the network to an operational point where the sum of the utility functions is maximized. On the other hand, an empirically designed control protocol, such as the TCP congestion control protocol, can be *reversely engineered* by matching its behavior with the above family of control laws. This matching process, if successful, will lead to an "optimal control law" that best describes TCP behavior, and hence, to the discovery of its underlying utility function. In this subsection, we demonstrate how the utility function that underlies the TCP behavior can be identified through such a reverse engineering procedure.

Since the TCP congestion control mechanism is designed empirically with many detailed fine tunings, it is unlikely that there exists a utility based control law that can perfectly match all the details of TCP behavior. In this paper, we attempt to match two major TCP behaviors, i.e., the MIMD behavior in the slow start phase and the AIMD behavior in the congestion avoidance phase. The TCP behavior due to timeout (i.e., reducing the window size to one) is not matched. In other words, we assume that any congestion indications, such as timeout and three repetitive ACKs, have the same effect on the TCP behavior, i.e., causing the reduction of the window size by half. As a result, the resulting control law, known as TCP control law, tends to be more aggressive than the TCP congestion control protocol. This ensures that QoS-aware control law derived in this paper that are TCP control law friendly will also be friendly to the actual TCP congestion control protocol.

Assume that the increase rate is αx ($\alpha >$0) and the decrease rate is βx ($0 < \beta \leq 1$) in the slow start phase. In the absence of congestion, i.e., $cg = 0$, the control law in Eqn. (3) is given by

$$\dot{x} = z(t,x)f(x) = \alpha x \tag{7}$$

In the presence of congestion, i.e., $cg = 1$, we have

$$\dot{x} = z(t,x)[f(x) - 1] = -\beta x \tag{8}$$

Then we have the utility function

$$U(x) = xlog(1 + \frac{\alpha}{\beta}) \tag{9}$$

and

$$z(t,x) = \frac{\alpha x}{f(x)} = (\alpha + \beta)x \tag{10}$$

Since $U(x)$ is a differentiable, concave, and strictly increasing function, and $z(t,x)$ is positive for $x > 0$, the MIMD control law can achieve globally optimal rate allocation. The

coefficient of the utility depends on the ratio α/β. A flow with a larger increase factor (α) or smaller decrease factor (β) leads to higher utility. For the slow start phase in TCP, assume α=1 and β=1/2, then $U(x) = xlog3$.

For the congestion avoidance phase, assume the additive-increase rate is $\mu > 0$, and multiplicative-decrease rate is βx. In the absence of congestion, i.e., $cg = 0$, the control law is given by

$$\dot{x} = z(t,x)f(x) = \mu \tag{11}$$

In the presence of congestion, i.e., $cg = 1$, we have

$$\dot{x} = z(t,x)[f(x) - 1] = -\beta x \tag{12}$$

Then we have

$$U(x) = (\frac{\mu}{\beta} + x)[log(\mu + \beta x) - 1] - x[log(\beta x) - 1] + C \tag{13}$$

and

$$z(t,x) = \mu + \beta x \tag{14}$$

where C is a constant; $U(x)$ is a differentiable, concave, and increasing function; and $z(t,x)$ is positive. Hence the AIMD control law achieves globally optimal rate allocation maximizing the utility function described in Eqn. (13). In the TCP congestion avoidance phase, $\beta = 1/2$, then $U(x) = (2\mu + x)[log(\mu + x/2) - 1] - x[log(x/2) - 1] + C$. Note that the β value is found to be larger than 1/2 in [15] and [22]. However, our analysis in the next section shows that β value has limited effect on the equilibrium rate allocation and hence, we simply let $\beta = 1/2$.

Now, consider the steady state where the slow start phase ends at a constant rate x_s. We set C at a value such that $U(x)$ is continuous at this point. Then we have

$$C = x_s log[\frac{(\alpha + \beta)x_s}{\mu + \beta x_s}] - \frac{\mu}{\beta} log(\mu + \beta x_s) + \frac{\mu}{\beta} \tag{15}$$

C. Analysis

To the best of our knowledge, the TCP utility function derived in the previous section has been by far the only utility function that can capture both MIMD and AIMD behaviors of TCP. In this section, we provide detailed analysis of this utility function aiming at a better understanding of the end-to-end TCP congestion control mechanism and for the benefit of the proposed protocol design.

Dynamic Utility: We first note that in the above derivation of the TCP utility function, we implicitly assumed that the slow start threshold x_s is a constant. This is a steady state solution that holds only when the traffic load and pattern in the network stay unchanged. However, when the traffic load and pattern changes, x_s will change. Besides, x_s is also a function of RTT of a TCP session. This means the TCP utility function will change its value range and shape as traffic load and pattern changes, and also changes with RTT. Fig. 3 demonstrates this by plotting the TCP utility function at various x_s values. As

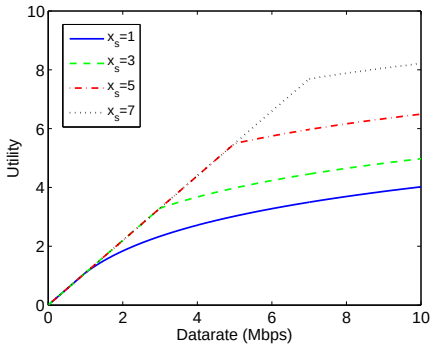

Fig. 3. Utility values with different slow start threshold

one sees from these plots that the TCP utility function is indeed concave and resemble the NRE utility function in Fig. 1.

Slow Start Phase: The result in Eqn. (9) shows that the underlying utility function for the slow start phase is a linear function with a system-parameter-dependent coefficient. It is well-known that the user utility of a linear form can easily lead to unfair share of the link bandwidth. In fact, it can be easily shown that when multiple flows sharing a single-hop link, the one with the largest coefficient will take over all the link bandwidth, starving all the rest of the flows.

The above observations explain why the slow start threshold in TCP must be dynamically adjusted to allow fair share of the network resources. In TCP, a flow always adjusts its slow start threshold to be half of the peak rate (the rate when congestion starts). When TCP is in the steady state, the peak rate is fixed and so is the slow start threshold. As we shall see shortly, the bandwidth allocation in the congestion avoidance phase is roughly inversely proportional to RTT and does not cause strong bias in favor of a particular flow. This ensures that no flow will be able to grab the entire link bandwidth, starving other flows.

Congestion Avoidance Phase: The utility function for the congestion avoidance phase is given in Eqn. (13). First, we note that when $x >> \mu/\beta$, $U(x) \simeq (\mu/\beta)log(x) + const$, reminiscent of the logarithm utility that leads to the well known proportional fairness in [14]. However, there is a key difference between the two. Namely, the utility function given in Eqn. (13) is explicitly proportional to the additive increase rate μ, whereas the one given in [14] is not dependent on μ. Since for the TCP window-based congestion control, the congestion window size is increased by one packet every RTT in the congestion avoidance phase, the additive increase rate μ is inversely proportional to RTT. In other words, our utility function is inversely proportional to RTT, whereas the one given in [14] is independent of RTT. The implication of this is significant in terms of steady state rate allocation. In what follows, we use an example similar to the one in [2] to explain why this difference is significant.

In [2], Chiu showed that with Kelly's $log(x)$ utility function, a TCP flow 0 traversing n-hops sharing the link bandwidth with some single-hop flows, one per link, receives a rate

allocation ratio $x_0/x = 1/n$, where x_0 and x are the rates allocated to flow 0 and each single-hop flow (note that the rates allocated to different single-hop flows are the same), respectively. This implies that the proportional fairness derived from Kelly $log(x)$ utility function does lead to rate allocations reflecting the TCP behavior, i.e., the flow traversing more number of hops tends to receive less resource. However, Chiu went on to show that this is true only when all the links are bottleneck links for flow 0. In the case where there is only one single-hop flow, e.g., at the first link, flow $x_0/x = 1$, independent of RTT, i.e., achieving min-max rate allocation. This is in contradiction with the intuition that a TCP flow with a larger RTT tends to achieve smaller rate allocation. As a result, Chiu concludes that TCP does not implement proportional fairness.

Now, what we are interested in is whether the TCP utility function given in Eqn. (13) can lead to a rate allocation that captures the dependence of TCP rate allocation on RTT. To this end, let us calculate x_0/x using the TCP utility function given in Eqn. (13). First, we consider the case where all the links are bottleneck links for flow 0. To be more general, we allow m single-hop flows running on each link. The flow rate x for each single-hop flow can be expressed in terms of x_0 as $x = (B - x_0)/m$, where B is the link bandwidth for all the links, and $\mu_i = \mu$ (i=1, 2, ..., m). Then at the maximum utility, we have $\partial U(\mathbf{x})/\partial x_0$=0, i.e., the flow value of x_0 can be given by

$$[\frac{\mu/\beta + (B - x_0)/m}{(B - x_0)/m}]^n = \frac{\mu_0/\beta + x_0}{x_0} \tag{16}$$

When $x = (B - x_0)/m >> \mu/\beta$, or equivalently, when the TCP utility function can be approximated by $U(x) \simeq (\mu/\beta)log(x) + const$, Eqn. (16) can be approximated by

$$\frac{x_0}{x} = \frac{x_0}{(B - x_0)/m} \simeq \frac{\mu_0}{n\mu} \tag{17}$$

What significant about this result is that the relative rate allocation for flow 0 here is not only dependent on n but also proportional to the ratio μ_0/μ, or equivalently, inversely proportional to the ratio RTT_0/RTT, in the case of window-based flow control, where RTT_0 and RTT are round-trip times for flow 0 and a single-hop flow, respectively. We also note that this result is independent of β. For that reason, we have simply set $\beta = 1/2$.

It can be easily shown that in the case where single-hop flows are only placed on one link, e.g., the first one, the relative rate allocation for flow 0 (without approximation) is then equal to μ_0/μ, or reversely proportional to RTT_0/RTT, in agreement with the intuition about the TCP rate allocation. This also explains why the utility function constructed in [41] has to be inversely proportional to RTT.

The above results clearly demonstrate that the utility function derived in the previous section can capture the essential aspects of the TCP behaviors qualitatively. We also note that the dependence of utility on RTT is derived from the dependence of utility on μ as a special case, i.e., in the context of the TCP window-based flow control. If the additive-increase μ rate is set to a constant value for all the flows, regardless of their respective RTT values, then all the flows have the same utility function and hence achieves proportional fairness as described in [14]. This explains why keeping a constant additive-increase rate in TCP leads to equal allocation of traffic rate to every flow under certain conditions, independent of RTT, as observed in [10] and [18]. Hence, our utility is more general than the ones in [14] [41] and it can lead to different rate allocations, depending on how μ is implemented.

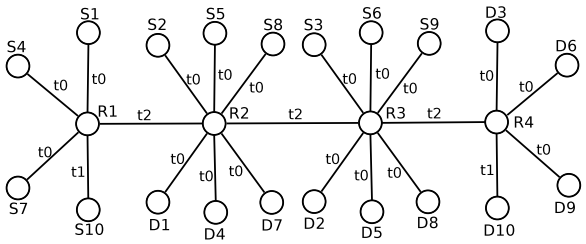

Fig. 4. Network topology I

The remaining question to be answered is how well the proposed TCP model captures the steady-state behaviors of TCP quantitatively.

D. Verification of Utility Function

Most existing utility-based TCP modeling solutions can only capture certain TCP behaviors qualitatively and they cannot capture the end-to-end nature of TCP. As a result, to the best of our knowledge, no existing solutions attempted to test their models against the end-to-end TCP congestion control mechanism quantitatively. Since in this work, the proposed protocol needs to be designed based on the TCP utility function, to ensure the protocol thus designed is truly TCP-friendly, the TCP utility function must be able to provide a rate allocation that reasonably matches the actual TCP rate allocation. In this section, we use NS-2 simulator to test it.

We consider the case of $n = 3$ and $m = 3$, i.e., a three link network with a total number of 10 flows, one traversing all three links and three single-hop flows on each link. The network topology is shown in Fig. 4. All the three links have the same link bandwidth, i.e., $B = 100\ Mbps$. The link propagation delays are given in Table I. Although not tightly coupled with the queuing mechanisms in use in network nodes, the family of control laws presented in the previous section does implicitly assume that different flows sharing the same congested link should have equal opportunity to sense the congestion. Hence we let all the flows share a single Random Early Detection (RED) first-in-first-out (FIFO) queue at each router. Flows are generated by TCP-Reno. Each simulation run is 300 seconds and the statistics are collected during the final 100 seconds.

To test the accuracy of the proposed TCP model, we compare the measured rate allocations of TCP flows against the theoretical one. Due to the use of a dynamic slow-start threshold, a TCP flow is always in the congestion avoidance

TABLE I
LINK PROPAGATION DELAY OF THREE CASES

	t_0	t_1	t_2
Case I	5 ms	5 ms	5 ms
Case II	2 ms	5 ms	5 ms
Case III	5 ms	5 ms	2 ms

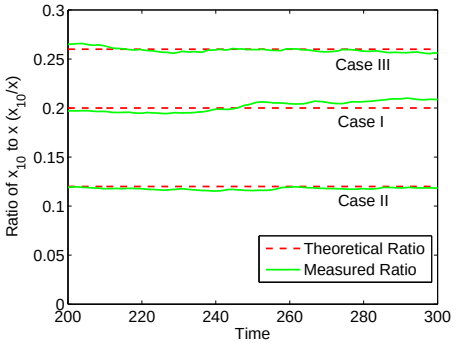

Fig. 5. Rate Ratio

phase in the steady state, assuming that there is no time-out events that put the TCP back to the slow start phase. Hence, the theoretical rate allocation is given by Eqn. (16). In the congestion avoidance phase, the congestion window is increased by 1 packet per RTT. If the packet size is w bits, the additive increase rate of a flow is w/RTT (in our simulation, $w = 8000$ for all packets). Hence, the ratio x_0/x is independent of w but dependent on RTT. The RTTs for flow 0 and other 9 flows are 50 ms and 30 ms in Case I; 50 ms and 18 ms in Case II; and 38 ms and 30 ms in Case III. Here the RTTs are evaluated based on the propagation delays only. Then, the theoretical rate ratios (x_0/x) based on Eqn. (16) for Case I, II and III are 0.2, 0.12 and 0.26, respectively.

Fig. 5 depicts the flow ratio versus time for the three cases, respectively. The flow rate x in the three cases is computed as the average rate of flows 1-9. Here, we observe that the measured TCP flow ratios closely match the theoretical ratios for all three cases. These results demonstrate that the TCP utility function and hence the TCP control law can capture the essential behaviors of TCP quantitatively.

V. A UNIFIED, END-TO-END TRANSPORT LAYER PROTOCOL

In this section, we derive the proposed control law/protocol, based on the TCP utility function derived in the previous section and test their performance based on a window-based implementation.

A. Control Law for Minimum Rate Guaranteed Service

Plugging the TCP utility function, i.e., Eqns. (9) and (13), into Eqn. (3), we arrive at the following control law:

Without congestion,

$$\dot{x} = \begin{cases} (3r(x) - 1)x/2 & \text{if } x < x_s \\ (r(x) - 1)x/2 + \mu r(x) & \text{otherwise} \end{cases} \quad (18)$$

where $r(x)$ is given in Eqn. (6).

With congestion, the QoS-aware control law acts the same as the TCP control law, i.e., the decreasing rate is $x/2$.

Note that if $r(x) = 1$, the control law in Eqn. (18) degenerates to the TCP control law. Hence, this control law unifies the TCP control with the control of the RDA and RRA flows.

To understand how this control law behaves under different traffic load conditions, let us consider two different cases, i.e., $x_s < \theta$ and $x_s \geq \theta$. For $x_s < \theta$, i.e., the traffic load is relatively heavy, the above control law can be written as follows:

Without congestion,

$$\dot{x} = \begin{cases} (3r^m_{max} - 1)x/2 & \text{if } x < x_s \\ (r^m_{max} - 1)x/2 + \mu r^m_{max} & \text{if } x_s < x < \theta \\ \mu & \text{otherwise} \end{cases} \quad (19)$$

With congestion, the control law acts the same as the TCP control law.

First, we consider the case without congestion. Assuming $r^m_{max} = 3$, the flow rate acceleration in the slow start phase is four times faster than that of the TCP control law. Moreover, the rate keeps increasing exponentially in the congestion avoidance phase (i.e., $\theta >$x$\geq x_s$) until the minimum guaranteed rate is reached, although at a reduced acceleration rate. Beyond that, the acceleration rate becomes equal to that of TCP. This confirms that our design methodology is friendly to TCP for the rate belonging to the elastic part of the flow. For the inelastic part of the rate, i.e., the minimum guaranteed rate, the flow acts more aggressively than TCP. Hence, under relatively heavy traffic load condition, a RDA/RRA flow is expected to outperform TCP. The value of the parameter r^m_{max} determines how much more aggressive the flow will be with respect to TCP. It is a tunable parameter, which will be discussed more later.

In the presence of congestion, it behaves the same as TCP in the case of window-based flow control, i.e., reducing the window size by half every RTT. Note that the rate may drop below the minimum guaranteed rate θ. But the flow rate can come back much faster than TCP as explained above. This is the reason that we say that this protocol provide *soft* minimum rate guarantee. This is a desirable feature for end-to-end control such as ours. In an end-to-end control scheme, the congestion feedback information can only be single-bit binary and there is no way for the control law to distinguish between congestion due to elastic traffic overloading or inelastic traffic overloading. If, for example, the congestion is triggered by inelastic traffic overloading, due to resource shortage caused by e.g., over commitment of network resources or link/node failures, maintaining a hard minimum guaranteed rate for the RDA and RRA flows can cause persistent congestions, resulting in partial or even complete shutdown of the Internet. The ability to reduce the flow rate below its soft minimum guaranteed rate makes this control law a viable solution to provide QoS features in the Internet of global reach, where call admission control at global scale may not be possible.

Our simulation studies in Section VI demonstrates that it is indeed a viable solution without call admission control.

For $x_s > \theta$, i.e., under relatively light traffic load condition, we have: (1) without congestion,

$$\dot{x} = \begin{cases} (3r^m_{max} - 1)x/2 & \text{if } x < \theta \\ x/2 & \text{if } \theta < x < x_s \\ \mu & \text{otherwise} \end{cases} \quad (20)$$

and (2) With congestion, the control law acts the same as the TCP control law.

At $r^m_{max} = 3$, the flow rate accelerates at 4 times the TCP slow start rate when $x < \theta$. However, as soon as $x \geq \theta$, it starts behaving the same as TCP in the slow start phase. In other words, the flow with the minimum guaranteed rate θ behaves very much like TCP and imposes minimum impact on TCP under relatively light traffic load condition.

In summary, the proposed protocol enables real-time services similar to the controlled-load service defined in the Intserv architecture. In relatively light traffic load condition, it behaves like TCP, while in the relatively heavy traffic load condition, it outperforms TCP, achieving a soft minimum guaranteed rate.

Since the TCP congestion control mechanism is window based, it makes sense to also implement the above control law as window-based traffic control protocol. Hence, in the following discussion, we consider only window-based implementation.

B. Theoretical Upper Bounded Guaranteed Minimum Rate

The control laws derived in the previous section allow the sending rates to fall below their minimum guaranteed rates. This raises the concern whether, on average, the minimum guaranteed rate for a real-time application using this protocol can be achieved or not. In this section, we derive a upper bound on the achievable minimum guaranteed rate for the above MRG control protocol based on the fluid model. We consider a network with one bottleneck link as shown in Fig. 6. Assume N_t TCP flows labeled as $1, \ldots, N_t$ and N_m MRG flow labeled as $N_t + 1, ..., N$ (here $N = N_t + N_m$) sharing a bottleneck link with bandwidth B as shown in Fig. 6.

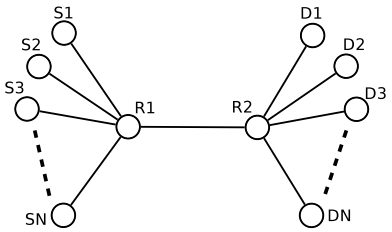

Fig. 6. Topology II

In fluid model [32] [39], the window size of TCP flow i can be modeled as:

$$\dot{W}_i(t) = \frac{(1 - p(q(t - R_i(t)))}{R_i(t)} - \frac{W_i(t)W_i(t - R_i(t))}{2R_i(t - R_i(t))} p(q(t - R_i(t))) \quad (21)$$

where $W_i(t)$ is the window size of TCP flow i at time t, $R_i(t)$ is the RTT of flow i at time t, $p(q)$ is the packet drop probability of a router at queue length q.

For an MRG flow, when the congestion occurs, its rate is reduced by half, then it will go back to the targeted minimum rate after one RTT if r^m_{max} is over 3. In the case that the target rate is greater than the slow start threshold, the control law can be approximated as the rate dropped from the peak rate to the targeted rate. Hence the window size of MRG flow j with targeted minimum rate θ_j can be approximately modeled as:

$$\dot{W}_j(t) \simeq \frac{(1 - p(q(t - R_j(t)))}{R_j(t)} - \frac{(W_j(t) - \theta_j R_j(t))W_j(t - R_j(t))}{R_j(t - R_j(t))} p(q(t - R_j(t))) \quad (22)$$

The packet drop probability $p(q)$ of RED takes the form,

$$p(q) = \begin{cases} 0 & 0 \leq q < q^{min} \\ \frac{q - q^{min}}{q^{max} - q^{min}} p^{max} & q^{min} \leq q \leq q^{max} \\ \frac{q - q^{max}}{q^{max}}(1 - p^{max}) + p^{max} & q^{max} < q \leq 2q^{max} \\ 1 & 2q^{max} < q \end{cases} \quad (23)$$

The RTT of flow k (k=1,2,..., N) $R_k(t)$ can be expressed as follows,

$$R_k(t) = a_k + \frac{q(t)}{B} \quad (24)$$

where a_k is the fixed propagation delay of flow k.

For the queue length $q(t)$,

$$\dot{q}(t) = -B + \sum_{k=1}^{N} \frac{W_k(t)}{R_k(t)} \quad (25)$$

By taking the expectation of each side of the Eqns. (21), (22) and (25), and following the approximated method used in [32], we get

$$E[\dot{W}_i(t)] = \frac{1 - p(E[q(t - R_i(t))])}{E[R_i(t)]} - \frac{E[W_i(t)]E[W_i(t - R_i(t))]}{2E[R_i(t - R_i(t))]} p(E[q(t - R_i(t))]) \quad (26)$$

for i=1, 2, ..., N_t, and

$$E[\dot{W}_j(t)] = \frac{1 - p(E[q(t - R_j(t))]}{E[R_j(t)]} - \frac{E[W_j(t - R_j(t))]}{E[R_j(t - R_j(t))]} \cdot (E[W_j(t)] - \theta_j E[R_j(t)])p(E[q(t - R_j(t))]) \quad (27)$$

for $j = N_t + 1$, ..., N, and

$$E[\dot{q}(t)] = -B + \sum_{k=1}^{N} \frac{E[W_k(t)]}{E[R_k(t)]} \quad (28)$$

The system stable point can be achieved at $E[\dot{W}_k] = 0$ and $E[\dot{q}] = 0$.

For the N_t TCP flows,

$$E[\dot{W}_i(t)] = 0 \Longrightarrow$$
$$E[W_i(t)]E[W_i(t - R_i(t))] = \frac{2(1 - p(E[q(t - R_i(t))]))}{p(E[q(t - R_i(t))])} \quad (29)$$

In the stable point, we can assume that $E[W_i(t)] = E[W_i(t - R_i(t))]$. That means that the average peak rate of TCP flow i at the stable state should be fixed. So we have

$$E[W_i] = \sqrt{\frac{2(1 - p(E[q]))}{p(E[q])}} \quad (30)$$

Similarly, the average peak rate of MRG flow j at the stable state is calculated as

$$E[\dot{W}_j(t)] = 0 \Longrightarrow$$
$$E[W_j]^2 - \theta_j E[R_j]E[W_j] - \frac{(1 - p(E[q]))}{p(E[q])} = 0 \quad (31)$$

Then we have

$$E[W_j] = \frac{\theta_j E[R_j] + \sqrt{(\theta_j E[R_j])^2 + \frac{4(1-p(E[q]))}{p(E[q])}}}{2} \quad (32)$$

The average queue length at the steady state can be obtained as

$$E[\dot{q}] = 0 \Longrightarrow \sum_{k=1}^{N} \frac{E[W_k]}{a_k + \frac{E[q]}{B}} = B \quad (33)$$

The average congestion window size of each flow can be solved through Eqns. (30), (32) and (33)). Then the average peak rate (the rate at the congestion time) of flow k is $E[W_k]/E[R_k]$.

Recall the MRG control law, i.e., the flow increases exponentially when its rate is smaller than its target rate; and increases additively after it reaches the target rate. The number of RTTs (n_j) of MRG flow j with additive increase rate μ_j in its additive increase phase is calculated as

$$n_j = \frac{E[W_j]/E[R_j] - \theta_j}{\mu_j} \quad (34)$$

Now let us compute the average flow rate of an MRG flow. Assume MRG flow j senses a congestion when its flow rate is at its average congestion rate $E[W_j]/E[R_j]$ (assume this rate is above its target rate θ_j). In the next RTT, the flow rate decreases to $E[W_j]/2E[R_j]$, and then increases back to its target rate (θ_j) in the following RTT by assuming r^m_{max} is no less than 3. In the following n_j RTTs, the flow rate increases additively to the congestion rate $E[W_j]/E[R_j]$. After it reaches the congestion rate, the packets in the following RTT may be dropped in the following RTT. We assume the effective flow rate is 0 in that RTT to calculate the upper bounded minimum target rate. So to guarantee the target rate θ_j, n_j has to satisfy the following equation

$$\frac{n_j(n_j + 1)}{2} \cdot \mu_j \geq (\theta_j - \frac{E[W_j]}{2E[R_j]}) + \theta_j \quad (35)$$

or

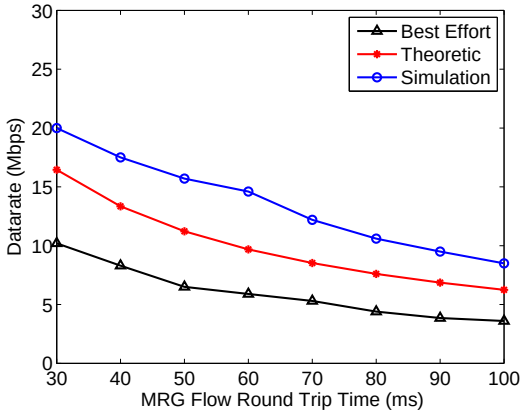

Fig. 7. The maximum guaranteed target Rate of MRG flow, RTTs of TCP flows vary from 20ms to 56ms

$$\theta_j \leq \frac{E[W_j]}{E[R_j]} - \frac{\sqrt{12\mu_j \frac{E[W_j]}{E[R_j]} + 25\mu_j^2} - 5\mu_j}{2} \quad (36)$$

Eqn. (36) shows the maximum target rate can be guaranteed for MRG flow j under certain network conditions. It means that all the target rates no larger than the value calculated in Eqn. (36) can be guaranteed using the MRG control law. It is verified through the following ns-2 simulation study.

In our simulation, we set $N = 11$ in the network. Among these flows, flows 1-10 are TCP flows, and flow 11 is an MRG flow. The bandwidth for the bottleneck link is 100 Mbps. Fig. 7 show the theoretical minimum guaranteed rate, the actual minimum guaranteed rate in the simulation, and the rate of flow 11 using the TCP control law. Here, RTTs of the TCP flows 1-10 linearly increases from 20 ms to 56 ms, i.e., the RTT of flow 1 is 20, of flow 2 is 24 ms and so on. The results show that the theoretical result in Eqn. (36) gives a little bit lower upper bound of the guaranteed target rate for the MRG flow, because we assume all the packets are dropped after congestion. The results also show that the minimum guaranteed rate of MRG is much greater than the average rate of TCP flow.

Now let us exam the rate allocation of link shared by multiple MRG flows and multiple TCP flows. In the simulation, we set 100 flows, the shared bandwidth is 500 Mbps. All flows have the same RTT 60 ms. From Fig. 8, we know that the average rate is about 4.5 Mbps when all flows are TCP flow. Then we set N_m flows to be MRG flows, and the remaining $100 - N_m$ flows are TCP flows, and change N_m from 10 to 100. We can see that the target rate can be guaranteed varies from about 9 Mbps to 4.5 Mbps which are about 1 Mbps above the theoretical bound. The average rate of TCP flows drops from about 4.5 Mbps to 2.5 Mbps. This shows that the MGR flow can significant improve the average target rate. Even if all the flows are MRG flows, the average guaranteed rate is just the same as TCP flows.

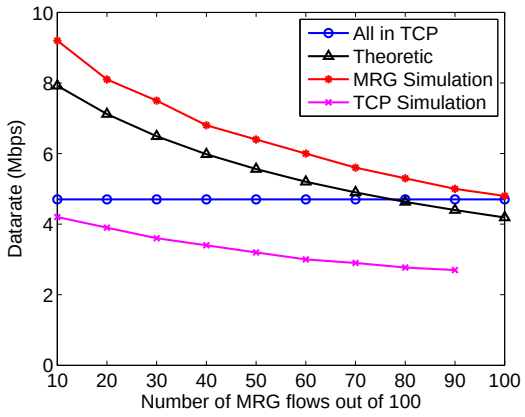

Fig. 8. The maximum guaranteed target rates of MRG flows

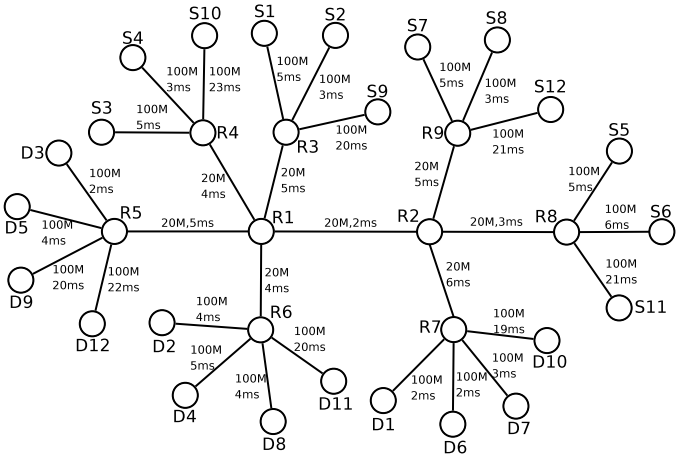

Fig. 9. Network topology

VI. Simulation Testing

The unified control law derived in the previous section is rate based. It is shown in [34] that the corresponding control law in discrete time domain with finite control epochs and granularity lead to stable and near optimal control. Hence we map the continuous, rate based control law into a window-based packet control protocol, in line with the TCP congestion control.

To translate the rate based control law into a window-based packet control protocol, we need to measure the transmission rate at the sender. A round trip time (RTT) based rate calculation is used. The rate is estimated as follows:

$$x(t) = w(t) \cdot s/T(t) \tag{37}$$

where $x(t)$ is the transmission rate at time t; $w(t)$ is the sliding window size at time t; s is the packet size; and $T(t)$ is the average measured RTT at time t. The average measured RTT at time t can be calculated as follows:

$$T(t) = \eta \cdot T(t-1) + (1-\eta) \cdot MT(t) \tag{38}$$

with $0 < \eta \leq 1$; $T(t-1)$ is the last calculated average RTT; $MT(t)$ is the measured RTT at time t; and η is a tunable parameter, i.e., the relative weight of the historical RTT estimation on the estimation of the current RTT. The sender uses the measured rate as well as the targeted minimum guaranteed rate to do congestion control.

We consider a network with 33 nodes, as shown in Fig. 9. There are 12 flows running in this network. Flow i starts at the source S_i and ends at the destination D_i. The RTTs excluding queuing delay for flows 1 to 12 are 40, 32, 32, 32, 38, 34, 38, 36, 100, 108, 100 and 110 ms, respectively.

In the simulations, we first run all the 12 flows using TCP-Reno. Fig. 10 shows the average rate of all the 12 flows. The rates of flows 1 to 12 are around 5.02, 5.89, 7.06, 5.84, 5.87, 5.92, 5.42, 4.8, 2.32, 2.12, 2.05 and 2.30 Mbps, respectively.

Now we run our protocol for flow 9 with the minimum guaranteed rate of 4.5 Mbps and TCP-Reno for the rest of flows. We found that the the minimum guaranteed rate 4.5

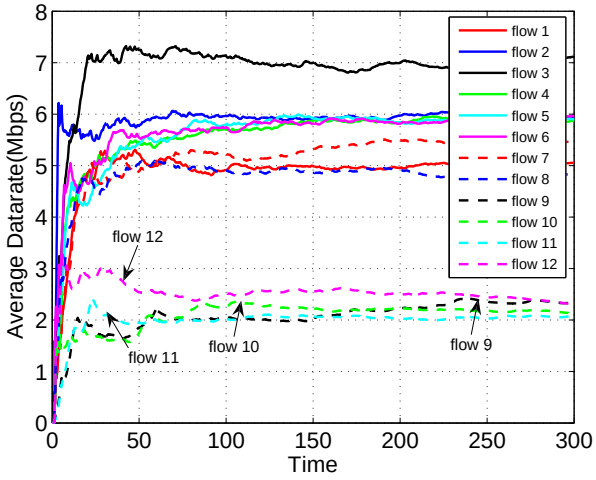

Fig. 10. Flow Rate Allocation, TCP-Reno for all flows

Mbps (it is 2.32 Mbps by using TCP) can be achieved when r^m_{max} is set at 3 or above. The simulation result for $r^m_{max} = 3$ is given in Fig. 11.

The above case study indicates that when the majority of the flows are TCP and the committed resource constitutes a small portion of the overall network resource (4.5/20), the minimum rates for QoS-based flows can be guaranteed, while the other TCP flows just have a little bit lower rate. For example, the rate of flow 12 is still above 2 Mbps.

Now we run flows 9, 10, 11 and 12 with minimum guaranteed rate of 3 Mbps for each flow and TCP-Reno for the rest of the flows. Again, $r^m_{max} = 3$ for flows 9, 10, 11 and 12. In this case, flows 10, 11, and 12 share the middle link in Fig. 9, meaning that about half of the link resource (9/20) needs to be committed to flows with minimum rate guarantee. Fig. 12 depicts the simulation results. One notes that all four QoS-aware flows reach their minimum guaranteed rate. In the meantime, the TCP flows do not suffer too much rate loses, for example, the rate of flow 3 from 7.06 Mbps to 6.5 Mbps,

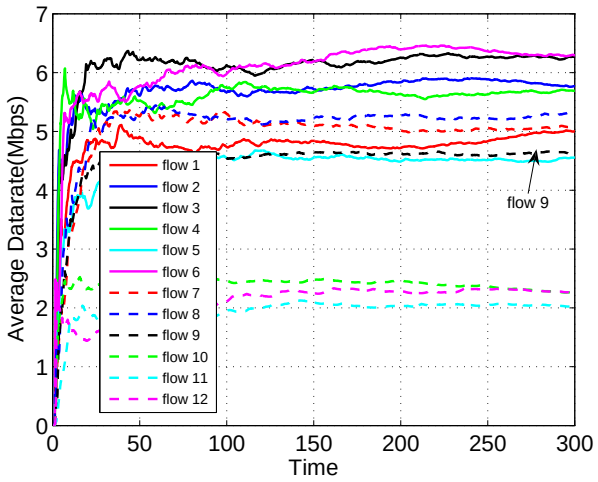

Fig. 11. Flow Rate Allocation, MRG for flow 9

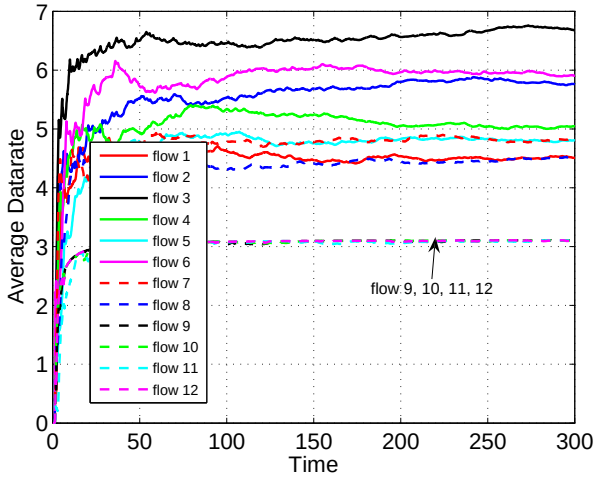

Fig. 12. Flow Rate Allocation, MRG for flows 9, 10, 11 and 12

and other flows with rate close to 5 Mbps are reduced to close to 4.5 Mbps.

Finally, we set flows 9 and 10 with minimum guaranteed rate of 3 Mbps and flows 11 and 12 with minimum guaranteed rate of 5 Mbps for each flow and TCP-Reno for the rest of the flows. Again, $r^m_{max} = 3$ for flows 9, 10, 11 and 12. Fig. 13 shows the rate allocation. Flows 9 and 10 can achieves their target rate while flows 11 and 12 cannot, but they have improved rate compared to TCP flows. This indicates that the proposed protocol can improve the flow rate even if in the heavy congestion conditions.

VII. Conclusions

In this paper, we proposed a unified, end-to-end transport layer traffic control protocol to support RDA, RRD, and NRE services. This protocol is the first of its kind, in the sense that it is design based on THEORY, i.e., a systematic, utility based design methodology and a well-grounded mathematical

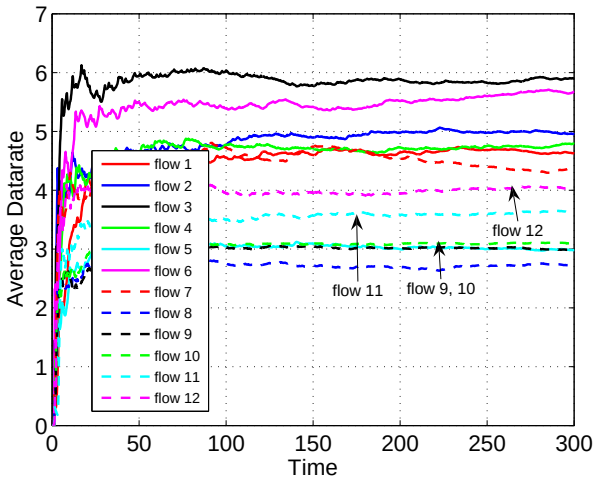

Fig. 13. Flow Rate Allocation, MRG for flows 9 and 10 with target rate 3 Mbps and flows 11 and 12 with target rate 5 Mbps

foundation. As a result, it possesses some desirable, provable properties, including fairness, stability, and optimality. Both analysis and simulation results demonstrated that the proposed protocol can provide soft minimum rate guarantee for real-time applications, while being friendly to TCP flows. As part of the protocol design, this paper also makes contributions to a better understanding of the utility-based approach and utility-based interpretation of TCP behaviors.

References

[1] T. Alpcan and T. Basar, "A Utility-Based Congestion Control Scheme for Internet-Style Networks with Delay", *IEEE INFOCOM*, 2003.

[2] D. M. Chiu, "Some Observations on Fairness of Bandwidth Sharing", *Fifth IEEE Symposium on Computers and Communications*, 2000.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", *RFC 2475, IETF*, December 1998.

[4] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin "Resourse ReSerVation Protocol (RSVP) – Version 1 Functional Specification", *RFC 2205, IETF*, September 1997.

[5] L. Brakmo, L. Peterson, "TCP Vegas: end to end congestion avoidance on a global Internet", *IEEE Journal on Selected Areas in Communications*, v13(8), pp 1465-1480, 1995.

[6] A. Cao and E. W. Zegura, "Utility Max-Min: An Application-Oriented Bandwidth Allocation Scheme", *IEEE INFOCOM*, 1999.

[7] Z. Duan, Z. Zhang, and Y. T. Hou, "Service Overlay Networks: Slas, QoS and Bandwidth Provisioning", *IEEE/ACM Transactions on Networking*, v13(6), pp 870-883, 2003.

[8] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "Mate: Mpls adaptive traffic engineering", *IEEE INFOCOM*, 2001.

[9] M. Fazel, Mung Chiang, "Network Utility Maximization With Nonconcave Utilities Using Sum-of-Squares Method Decision and Control", *2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*. 2005.

[10] S. Floyd. "Connections with Multiple Congested Gateways in packet-Switched Networks, Part 1: one-way Traffic", *ACM Computer Communications Review*, v21(5), pp30-37, 1991.

[11] S. J. Golestani and S. Bhattacharyya, "A class of end-to-end congestion control algorithms for the internet", *IEEE International conference on Network Protocols (ICNP)*, 1998.

[12] S. Kandula, D. Katabi, B. Davie, and A. Charney, "Texcp: Responsive yet stable traffic engineering", *ACM SIGCOMM*, 2005.

[13] K. Kar, S. Sarkar and L. Tassiulas, "A Simple Rate Control Algorithm for MAximizing Total User Utility", *IEEE INFOCOM*, 2001.

[14] F. Kelly, A. Maulloo, and D. Tan, “Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability”, *Journal of the Operational Research Society*, v49, pp237-252, 1998.

[15] S. Kunniyur and A. Srikant, “End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks”, *IEEE/ACM Transactions on Networking*, 11(5), 2003.

[16] T. Harks and T. Poschwatta, “Priority Pricing in Utility Fair Networks”, *IEEE ICNP*, 2005.

[17] T. Harks and T. Poschwatta, “Utility Fair Congestion Control for Real-Time Traffic”, *IEEE INFOCOM*, 2005.

[18] T. R. Henderson, E. Sahouria, S. McCanne and R. H. Katz, “On Improving the Fairness of TCP Congestion Avoidance”, *IEEE GLOBECOM*, 1999.

[19] http://www.rfc-archive.org/getrfc.php?rfc=3448

[20] http://www.bbc.co.uk/iplayer .

[21] http://www.ietf.org/rfc/rfc2998.txt

[22] P. Hurley, J. L. Boudec, and P. Thiran, “A Note on the Fairness of Additive Increase and Multiplicative Decrease”, *16th International Teletraffic Congress*, 1999.

[23] S. Jin, L. Guo, I. Matta, and A. Bestavros, “A Spectrum of TCP-friendly Window-Based Congestion Control Algorithms”, *IEEE/ACM Transactions on Networking*, 11(3), June 2003.

[24] V. Jocobson, “Congestion Avoidance and Control”, *ACM SIGCOMM*, 1998.

[25] E. Kohler, M. Handley, and S. Floyd, “Designing DCCP: Congestion Control Without Reliability”, *ACM SIGCOMM*, 2006

[26] R. La, V. Anantharam, “Utility-Based Rate Control in the Internet for Elastic Traffic”, *IEEE Transactions on Networking*,v10(2), pp272-286, 2002.

[27] C. M. Lagoa, H. Che, and B. A. Movsichoff,“Adaptive control algorithms for decentralized optimal traffic engineering in the internet”, *IEEE/ACM Transactions on Networking*, v12(3), pp. 415-428, 2004.

[28] T. V. Lakshman, U. Madhow, “The Performance of TCP/IP for Networks with High Bandwidth Delay Products and Random Loss”, *IEEE/ACM Transactions on Networking*, 5(3), pp. 336-350, 1997.

[29] J. Lee, R. Mazumdar and M. Shroff, “Non-convexity Issues for Internet Rate Control with Multi-class Services: Stability and Optimality”, *IEEE Infocom*, 2004

[30] S. H. Low, “A Duality Model of TCP and Queue Managemnet Algorithms”, *IEEE/ACM Transactions on Networking*, 11(4), pp525-536, 2003.

[31] S. Low and D. Lapsley,“Optimization flow control, i: Basic algorithm and convergence”, *IEEE/ACM Transactions on Networking*, v7(6), pp. 861-874, 1999.

[32] V. Misra, W. Gong and D. Towsley, “A Fluid-Based Analysis of a Network of AQM routers supporting TCP Flows with Application to RED”, *ACM SIGCOMM*, 2000.

[33] B. Movsichoff, C. Lagoa, and H. Che,“Decentralized optimal traffic engineering in connectionless networks”, *IEEE Journal on Selected Areas in Communications*, v23(2), pp 293-303, 2005.

[34] B. A. Movsichoff, C. M. Lagoa and H. Che, “End-to-End Optimal Algorithms for Integrated QoS, Traffic Engineering, and Failure Recovery”, *IEEE/ACM Transactions on Networking*, v15(4), pp813-823, 2007

[35] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. “Modeling TCP”, *ACM SIGCOMM*, 1998.

[36] J. Pongsajapan and S. Low, “Reverse Engineering TCP/IP-like Networks Using Delay-Sensitive Utility Functions”, *IEEE INFOCOM*, 2007.

[37] E. Rosen, A. Visvanathan, and R. Callon, “Multiprotocol Label Switching Architecture”, *RFC 3031, IETF*, January 2001.

[38] S. Shenker, “Fundamental Design Issues for the Future Internet”, *IEEE Journal on Selected Areas in Communications*, Vol. 13(7), pp 1176-1188, 1995

[39] R. Srikant, “Models and Methords for Analyzing Internet Congestion Control Algorithms”, *In Advances in Communication Control Networks in the series ”Lecture Notes in Control and Information Sciences(LCNCIS)”*, Springer-Verlag, 2004.

[40] W. H. Wang, M. Palaniswami, and S. H. Low, “Application-Oriented Flow Control: Fundamentals, Algorithms and Fairness”, *IEEE/ACM Transaction on Networking*, 14(6), December 2006.

[41] M. Vojnovic, J. Boudec, and C. Boutremans, “Global Fairness of Additive-increase and Multiplicative-decrease with Heterogeneous Round-trip Times”, *IEEE INFOCOM*, 2000.