Estimating an Additive Path Cost with Explicit Congestion Notification

Peyman Teymoori, David A. Hayes, Michael Welzl, Stein Gjessing

***Abstract*— Network Utility Maximization (NUM) is a well accepted theoretical framework that describes how congestion controls cooperate to achieve an ideal sending rate allocation, for given utility functions of senders and constraints of the network. These network constraints are expressed as a "cost" in the framework. In practice, most congestion control mechanisms obtain feedback that is different from the framework's "cost". This paper focuses on Explicit Congestion Notification (ECN), which has been shown to be advantageous when it is available, e.g. with the popular Datacenter TCP (DCTCP) mechanism. However, different from the framework's cost, ECN marks are not additive. We present a practical solution to this problem; it changes how end hosts interpret the ECN signal, while for routers, a special configuration of RED is used.**

***Index Terms*— Explicit Congestion Notification, Network Utility Maximization.**

I. Introduction

EXPLICIT Congestion Notification (ECN) has recently gained increased attention. It allows network entities to explicitly notify end-systems of congestion by setting a bit in the packet header. Datacenter TCP (DCTCP) [4] is a relatively recent mechanism that attains good performance with ECN. Rather than relying on an Active Queue Management (AQM) algorithm [2] to only give feedback about congestion when the *average*[1] queue length has exceeded a limit (an indication of long-lasting overload), DCTCP uses ECN feedback based on the instantaneous queue length. This allows end systems to learn about congestion earlier than with an average, and enables them to interpret the number of ECN marks that have occurred over a period of time as a multi-bit congestion signal. This has made DCTCP an attractive base for the design of other algorithms (e.g. [5], [14], [40], [46]).

The Network Utility Maximization (NUM) framework [22] facilitates a suitable allocation of source rates, given the network's constraints, by maximizing the utility of all sources with respect to a network "cost". Because packets from a sender incur a cost on all congested links they traverse, NUM requires the congestion signal to be additive (as if packets had a counter that could be increased by routers along the path). Because the "cost" represented by an ECN bit is not additive, it can only roughly approximate the true cost of a path if the marking probability is very low [25], rendering it unsuitable for direct application of NUM in protocol engineering.

This limitation of ECN is disappointing because, different from datacenters where customized mechanisms can be applied to satisfy NUM, ECN is a simple feedback mechanism of broad utility [15] and instantaneous queue marking could perhaps even be made to work for the Internet [37]. This paper[2] addresses this problem, showing how we can turn ECN into an additive signal and use it for NUM. Similar to DCTCP, the type of marking that we need can be attained by configuring a RED (Random Early Detection) [17] queue (see Section VIII-B), which means that a congestion control mechanism based on our suggested type of ECN usage can be deployed with commodity switch hardware, only needing to update code in end hosts. We focus on a practical case of using ECN as a better signal in end systems, and hence study how to deflate/inflate marking probabilities. This also allows us to obtain the utility function of controllers that work with high marking probabilities, such as DCTCP.

The paper is organized as follow. In Section II, we discuss our motivations. Section III surveys related work. In Section IV, we present a network model using NUM. We establish the theory on solving ECN limitations in NUM in Section V, and explain how to use it in Section VI. Section VII presents three different types of distributed algorithms based on the theory, and in Section VIII, we evaluate their correctness with simulation. Section IX discusses aspects and applications of our theory and algorithms, and Section X concludes.

II. Motivations

In the NUM framework, routers are assumed to be sending their *price*, a real non-negative number, to sources (senders). Sources calculate their send rate based on the price. In the following, we discuss the cases in which the network uses *packet marking* to send the price back to sources and *the marking probability is not small*.

Assume a network with more than one bottleneck link. Packets of source r are randomly marked at the links in its

The authors were part-funded by the Research Council of Norway under its "Toppforsk" programme through the "OCARINA" project (http://www.mn.uio.no/ifi/english/research/projects/ocarina/). The views expressed are solely those of the authors.

P. Teymoori, M. Welzl, and S. Gjessing are with the Department of Informatics, University of Oslo, Norway
(e-mail: {peymant|michawe|steing}@ifi.uio.no).

D. Hayes is with the Simula Metropolitan Center for Digital Engineering, Oslo, Norway (e-mail: davidh@simula.no).

[1]Definitions of "average" are varied—e.g.CoDel [30] requires the delay to be above a threshold for a certain fixed time before it gives a signal.

[2]An extended version of this paper has been made available online [41].

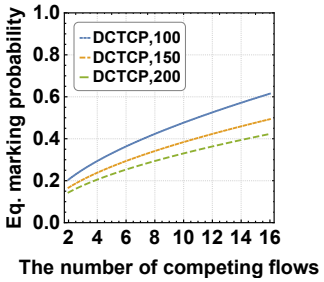


Fig. 1. Equilibrium marking probability of DCTCP (solving (6) in [4] for α): one bottleneck

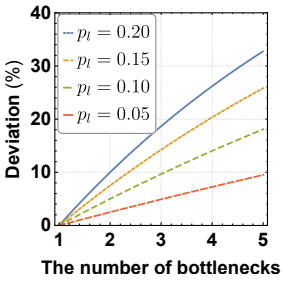


Fig. 2. Sum approximation error versus the number of bottlenecks.

path with a total marking probability of

$$p_r = 1 - \prod_{l \in L_r} (1 - p_l) \tag{1}$$

where p_l is the marking probability of link l on the path of source r to its destination. p_l is a function of link utilization at link l, and we follow the common assumption that p_l for different l are independent. What we focus on here is the equilibrium marking probability, which also depends on the number of competing flows. The NUM framework assumes that sources can obtain $q_r = \sum p_l$. If $p_l \ll 1$, then $1 - \prod(1 - p_l) \approx \sum p_l$ and p_r can be used as q_r [25]. This means that in this case, the NUM model closely matches an ECN-enabled network, or networks with binary signals such as packet drop. Fig. 1 illustrates the analytical marking probability of DCTCP for a single bottleneck network with different marking thresholds of 100, 150, and 200 packets. We see that as the number of competing flows increases, the marking probability quickly increases to values beyond 0.2. Fig. 2 shows the difference between q_r and p_r for different link marking probabilities. We see that as the number of bottleneck links increases, the difference also increases. It is, however, much less significant when p_l is at most 0.05, which is much smaller than the marking probability used in newer congestion controllers such as DCTCP. From Fig. 2, we can also infer that $\sum p_l \geq 1 - \prod(1 - p_l)$ and $1 - \prod(1 - p_l) \leq 1$, while $\sum p_l$ can exceed 1[3]. This indeed results in different throughputs for longer flows because they calculate their send rates based on an underestimate of the path cost. Therefore, the expected behavior of congestion controllers is affected significantly by the difference in q_r and p_r when the marking probability is not small. This paper provides solutions to this problem.

Having a higher equilibrium marking probability can lead to the following advantages: 1) a shorter queue length due to smaller and more frequent fluctuations, e.g. as in DCTCP vs. TCP (see Fig. 1 Section II.A of [41] for a detailed discussion) and 2) faster and smoother estimation of the marking probability of routers at sources (we will show this via simulation results in Section VIII-D). Therefore, it is not sufficient to approximate the sum of the per-router marking probabilities via a product by keeping the equilibrium marking probability low. We illustrate advantage 1) using a simple congestion controller, simulating a scenario in the INET framework of OMNeT++[4]. We used a Dual algorithm where link l's marking

[3]See Appendix A in [41] for the proof of the inequality.

[4]The simulation code is available at `http://tiny.cc/71utsz`

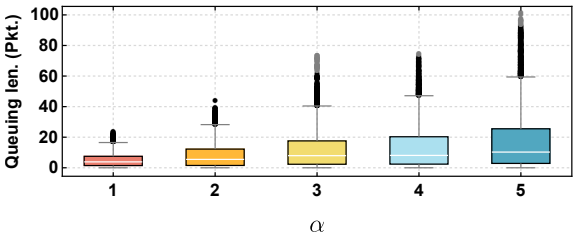


Fig. 3. Five sources, each transmitting to a different destination in a dumbbell network topology with link speeds of 100 Mbps and an RTT of 18 ms. The queue length distribution (in packets) at the bottleneck link is shown as a Box-Whisker plot for various α values. The box spans from the 0.25 quantile to the 0.75 quantile. Near and far outliers are shown with black and gray circles, respectively.

probability dynamic is $\dot{p_l} = \sigma[y_l - c_l]^+_{p_l}$. See Table I for notations. Source r adjusts its send rate using $x_r = U'^{-1}(p'_r)$ where $U(x_r) = \log(x_r)$. However, to force the equilibrium marking probability to be different, sources use $p'_r = \alpha p_r$ for $\alpha \geq 1$. Fig. 3 shows the results for α values of 1 to 5 where the equilibrium marking probabilities are 0.6, 0.3, 0.2, 0.15, and 0.12, respectively. We see that the smaller send-rate fluctuations at higher marking probabilities result in a shorter queue length. However, high making probabilities have the problem shown in Fig. 2, which this paper solves.

III. Related Work

Delay-based mechanisms such as TCP Vegas [9] or FAST TCP [19] have played a central role in the NUM literature as delay is an additive signal. However, it difficult to use effectively in practice since it can include various types of noise [36], though [46], [47] find ECN performs better.

An explicit additive signal, as modelled by NUM, could be a field in the packet header to additively record the path cost. However, there is no such field in the IP header. Two types of mechanisms have been proposed to convey cost using existing fields: 1) deterministic that uses existing bits in the IP header and 2) probabilistic markings. In deterministic marking, each router quantizes the cost of its outgoing link to a fixed number of bits, and then, with the use of the IP identification (IPid) field, data packets are mapped to different probe types where each probe type computes a partial sum of the path cost bits [33], [34]. However, these approaches are prone to calculation errors, and they also need changes in routers. Moreover, using the IPid field in this way conflicts with the specification [42].

Probabilistic marking mechanisms use the ECN bit in the IP header. The path cost is approximated by the end-to-end marking probability only if the marking probability is small [25], [38]. Random Early Marking (REM) uses a logarithmic marking function in routers and its inverse function at senders to estimate the actual path price [6]. However, REM needs changes in router functionalities. Random Additive Marking (RAM) [3] can also return an optimal cost estimate. It requires a router to know its position along the path taken by a packet, in addition to changes in functionalities.

Kelly has modeled MulTCP in [21], and showed how to estimate the path cost when the marking probability is not small. He uses the Barrier method, and a logarithmic function as the barrier function. In this paper, we present a number of general algorithms, and by utilizing the TCP

utility function obtained in [21], one of our algorithms (Primal) can reduce to Kelly's algorithm for MulTCP. In addition, we will present three different dual algorithms (and their corresponding primal-dual algorithms), show how RED (with a specific configuration) can be used as a dual algorithm, and also provide stability conditions for such a setup. RED as the dual method can be implemented by configuring the widely available RED AQM algorithm[5]. Yet, by changing the way the ECN signal is interpreted, we obtain an estimate of the additive path cost. This is achieved without requiring the marking probability to be extremely low—we do, however, require lower and upper bounds (e.g. the range $[0.05, 0.99]$).

IV. Network Optimization Model

The NUM framework assumes each source r has a utility function, $U_r(x_r)$, which determines the utility it achieves from sending at rate x_r. Resources in the network are constrained, and are able to "charge" sources based on their cumulative send rates with respect to the network's capacity limits. NUM models the network rate allocation problem as a set of greedy traffic sources that are limited by the capacity of links in the network the traffic traverses. The resulting optimization problem can be represented as follows:

$$\max_{x} \quad \sum_{r\in R} U_r(x_r) \tag{2a}$$

$$s.t. \quad y_l \le c_l, \qquad \forall l \in L \tag{2b}$$

where the objective is to maximize the sum of sources' utilities, constrained by the link capacities, and $y_l = \sum_{r\in R_l} x_r$ denotes the total rate crossing link l. Table I summarizes the notation used in the paper.

One way to solve (2) is via its Lagrangian function:

$$\mathcal{L}(x,\lambda) = \sum_{r\in R} U_r(x_r) - \sum_{l\in L} \lambda_l (y_l - c_l) \tag{3}$$

where $\lambda_l \ge 0$ is the Lagrange multiplier associated with link l or simply the shadow price (cost) of link l for violating its constraints. Rearranging the RHS of (3) yields

$$\mathcal{L}(x,\lambda) = \sum_{r\in R} \big(U_r(x_r) - x_r q_r\big) + \sum_{l\in L} (\lambda_l c_l) \tag{4}$$

where $q_r = \sum_{l\in L_r} \lambda_l$ is the sum of costs that source r accumulates while traversing the links towards its destination, i.e. L_r. At the optimum point we have $\partial\mathcal{L}/\partial x_r = 0$, which yields $x_r = U_r^{'-1}(q_r)$. This implies that the Lagrangian function can be optimized in a distributed way as each source r only needs q_r which is a real value, and q_r can be accumulated by the flow of source r on its path towards the destination using, for example, a long-enough field in the packet header. If such a field is not accommodated in the header, then cost should be carried via other methods such as ECN. However, before adopting ECN and to take advantage of higher marking probabilities, the optimization model needs to be extended in order to cope with the bias in the q_r estimate from ECN at higher marking probabilities (see section II).

[5]The "L4S" proposal [12] is deploying a DCTCP-like mechanism on the Internet, and it is currently considered for standardization in the IETF [11].

TABLE I
Notation

Symbol	Description
R	The set of sources
L	The set of links
x_r	Send rate of source r
λ_l	Lagrange multiplier of link l
p_l	Marking probability of link l
$U_r(.)$	Utility function of source r
R_l	The set of the sources crossing link l
L_r	The set of the links source r crosses
q_r	The total cost source r is charged by links L_r
c_l	Capacity of link l
y_l	The sum of rates of sources crossing link l
$.^*$	Optimum value of . that optimizes some function
$\varrho_r(x_r)$	Step size function of source r
$\sigma_l(p_l)$	Step size function of link l

V. Indirect Lagrange Multipliers

In the constrained optimization literature, Lagrange multipliers associated with equality/inequality constraints are defined as *constants* or *variables* associated with the problem [8], [39]. This also forms the usage basis of constrained optimization in the networking context (e.g. see [26], [32]). However, as we showed in section II, since ECN marks are not additive, they limit the applicability of this theory.

In this section, we present the theory which enables us to accurately estimate cost using ECN. This enables us to narrow the model–reality gap, making the NUM framework even more useful and practical; we extend the theory of Lagrange multipliers to encompass multipliers that are not *variables*, but are instead *functions* of some other variables in the form of $f(p_i)$. We more generally call these Lagrange multipliers *function multipliers* and their variables *indirect multipliers*. After establishing the necessary theory for the use of functions as Lagrange multipliers, we set feasibility conditions on these functions, and show how the choice of functions can affect the uniqueness and existence of the equilibrium.

A. Optimality Conditions with Indirect Multipliers

Consider the following constraint optimization problem

$$\max_{x} \quad U(x) \tag{5a}$$

$$s.t. \quad g_i(x) \le 0, \qquad i = 1,\dots,M_1 \tag{5b}$$

$$h_j(x) = 0, \qquad j = 1,\dots,M_2. \tag{5c}$$

with g_i inequality and h_j equality constraints and $x \in \mathbb{R}^n$.

Theorem 5.1 (Karush–Kuhn–Tucker optimality conditions with indirect multipliers (KKTi)): Considering problem (5), assume that the point $x^* \in C$ satisfies Abadie's constraint qualification [1]. If x^* is a local maximum of U over C, and functions f and k are defined such that $\mathbb{R}_+$ is in the range f, and the range of k is $\mathbb{R}$, then there exists a pair of vectors $(p, v) \in \mathbb{R}^{M_1} \times \mathbb{R}^{M_2}$ such that

$$\nabla U(x^*) - \sum_{i=1}^{M_1} f(p_i)\nabla g_i(x^*) - \sum_{j=1}^{M_2} k(v_j)\nabla h_j(x^*) = 0^n, \tag{6a}$$

$$f(p_i) g_i(x^*) = 0, \quad i = 1,\dots,M_1, \tag{6b}$$

$$f(p_i) \ge 0, \quad i = 1,\dots,M_1, \tag{6c}$$

$$g_i(x^*) \le 0, \quad i = 1,\dots,M_1, \tag{6d}$$

$$h_j(x^*) = 0, \quad j = 1,\dots,M_2. \tag{6e}$$

Proof: Here, we provide a sketch of the proof; Further details are found in Appendix B of [41]. The proof of the original KKT theorem is based on Farkas' Lemma [16] showing that a system of equations constructed from inequality and equality constraints has a solution, and this confirms the existence of KKT multipliers. However, we prove that instead of finding variables (as performed in the proof of the KKT theorem), we can find functions of some variables as multipliers; this is done by taking the inverse of those functions and setting those to the values obtained in the proof. ■

Using functions as multipliers affects the uniqueness of multipliers as follows:

Remark 1: If functions f and k are bijective, then for every ξ_i, unique p_i and v_i can be found; otherwise, there will be more than one value satisfying conditions (6).

Remark 2: If the optimization problem (6a) is concave, functions g_i are convex, and functions h_i are affine, and $x^* \in C$ satisfies (6), then x^* is a global maximum.

B. Lagrangian Function with Indirect Multipliers

Using (6a), we can now construct a Lagrangian function with function multipliers $f(p)$ and $k(v)$. Its general form is

$$\mathcal{L}(x,p,v) = U(x) - \sum_{i=1}^{M_1} f(p_i)g_i(x) - \sum_{j=1}^{M_2} k(v_j)h_j(x)\ . \quad (7)$$

The domain of $\mathcal{L}$ is now determined by the set of possible values of x, p, and v as *mapped* by $f(p)$ and $k(v)$. Comparing with the original Lagrangian function having multipliers λ and μ, we can *interpret* that p and v are respectively *mappings* of λ and μ to a new domain, which is the domain of f and k.

Linear translations in the Lagrangian function were proposed in [13][6]. However, by introducing a new set of variables (called p_l) and functions (called function multipliers) into the Lagrangian function, $\mathcal{L}$, allows the non-linear translation we need to solve the bias in q_r estimate.

VI. Optimization Using ECN

As discussed before, there are two problems with using the marking probability directly as the cost: 1) in the network utility maximization framework, the *sum* of path cost is required for sources to react properly, not the *product*, and 2) the cost is a non-negative real number but probability is limited to values between 0 and 1. We therefore need to be able to *map* marking probabilities to actual link costs. Choosing a logarithmic function as a function multiplier can help us solve both problems. We define a function multiplier of the form:

$$f(p_l) = -\log_\phi(1-p_l) \quad (8)$$

where $p_l \in [0,1]$ is the indirect multiplier, and the positive real number $\phi > 1$ is a system parameter. We use the marking probability of link l as p_l. Hence, instead of solving problem (2) using its Lagrangian function and Lagrange multipliers, i.e. (4), we solve it using its corresponding Lagrangian function with indirect multipliers. Incorporating (8) into (7) and ignoring equality constraints yields the following Lagrangian function

$$\mathcal{L}(x,p) = \sum_{r\in R} U_r(x_r) + \sum_{l\in L}\big(-\log_\phi(1-p_l)\big)\big(y_l - c_l\big)\ . \quad (9)$$

After arranging the terms, we get

$$\mathcal{L}(x,p) = \sum_{r\in R}\Big(U_r(x_r) - x_r\big(-\sum_{l\in L_r}\log_\phi(1-p_l)\big)\Big) + \sum_{l\in L}\Big(c_l\big(-\log_\phi(1-p_l)\big)\Big)\ . \quad (10)$$

From (1) the log of the compliment of the total end-to-end path marking probability is:

$$\log_\phi(1-p_r) = \sum_{l\in L_r}\log_\phi(1-p_l)\ . \quad (11)$$

Substituting (11) into (10) yields:

$$\mathcal{L}(x,p) = \sum_{r\in R}\Big(U_r(x_r) - x_r\big(-\log_\phi(1-p_r)\big)\Big) + \sum_{l\in L}\Big(c_l\big(-\log_\phi(1-p_l)\big)\Big)\ . \quad (12)$$

We see that the first problem of using the marking probability directly as cost can be solved by the property that $\log(a\cdot b) = \log(a)+\log(b)$. The second problem is solved by the property that the domain of (8) is $[0,1]$ while its range is $[0,+\infty)$. Therefore, it can map to any non-negative real-valued cost.

Lagrangian function (12) enables links to use cost values in the range $[0,1]$ and also allows the total path cost for each source to be the product of the costs at each link. In other words, this optimization function more accurately models the ECN-based costs the network feeds back. Each link l calculates p_l, and each source r uses p_r to calculate its optimum send rate. This can be seen as a *mapping* of the $[0,1]$ probability space to the $[0,\infty)$ cost space.[7]

VII. Optimization Algorithms

In this section, we discuss how (12) can be optimized in a distributed manner. The Primal, Dual, and Primal-Dual (e.g. see [39]) algorithms have been proposed in the literature, depending on how the controller logic is divided between sources and routers. However, these algorithms use Lagrangian function (3) as the optimization function and for their stability analysis. Since we empowered the Lagrangian function with indirect multipliers, as introduced by (7), we need to formulate these algorithms again based on our new Lagrangian function (12). We follow the same methods of dividing the controller logic and present three types of algorithms (called Primal, Dual, and Primal-Dual), and then evaluate their stability properties. These algorithms provide a general class of

[6] [13] proposes a generalized Lagrangian function, $\mathcal{L}(X,y,U,f)$, by introducing a set of continuous and non-decreasing functions, called f, which should satisfy the translation property: $f(y+\alpha g) = \alpha + f(y)$, and the penalty function is defined as $P(f(y),f(z)) = f(y) - f(z)$. In case f is linear, $\mathcal{L}$ would reduce to the standard Lagrangian.

[7] Parallels with the mirror gradient descent [29] are apparent. It also maps the optimization problem to a dual space that better matches the geometry of the problem, with the negative entropy version also using a logarithmic mapping. Mirror descent does this in order to more quickly iterate to the solution in the primal space. However, we do this because the correct solution exists in the mapped space. That said, one could use tools such as mirror gradient descent with our extended Lagrangian function.

algorithms working with ECN as a means to estimate the path cost. We also discuss how we can have a deployable solution over commodity hardware with minimal changes.

The stability analysis performed in this section focuses on comparatively showing the global stability properties of indirect multipliers, for the moment ignoring feedback delay for tractability. Feedback delay plays an important part in network stability and will be considered in future work.

A. Primal Algorithm

The main aim of the primal problem is to provide a deployable solution where the network is not modifiable and is only capable of packet marking. Therefore, the controller should be placed on the sender or receiver end. The primal algorithm design is similar to the following primal algorithm

$$\dot{x}_r = \varrho_r\Big(x_r U_r^{'}(x_r) - x_r \sum_{l\in L_r} \lambda_l\Big) \tag{13a}$$

$$\lambda_l = \rho_l\Big(\sum_{r\in R_l} x_r\Big) \tag{13b}$$

that was presented in [22]; ϱ_r is the step size, and $\rho_l(.)$ is the cost function of link l. Considering that a multiplier function in the form of (8) can map marking probability into cost, we define the primal algorithm as

$$\dot{x}_r = \varrho_r(x_r)\Big(U_r^{'}(x_r) + \sum_{l\in L_r} \log_\phi\big(1 - p_l(y_l)\big)\Big). \tag{14}$$

$p_l(y_l)$ with the range $[0, 1]$ is the *marking probability function* based on the aggregate send rate of the flows crossing link l, i.e. $y_l = \sum_{r\in R_l} x_r$. We also define the step size ϱ_r as a positive, increasing and continuous function of x_r. Substituting (11) in (14) yields

$$\dot{x}_r = \varrho_r(x_r)\Big(U_r^{'}(x_r) + \log_\phi(1 - p_r)\Big) \tag{15}$$

which we call the **Primal** algorithm. This implies that source r receives p_r instead of q_r, and it just needs p_r to calculate send rate. It has been proven that if U_r is strictly concave, the primal algorithm (13) is globally asymptotically stable [22]. The next theorem investigates the stability of our primal algorithm.

Theorem 7.1: If U_r is strictly concave, our new primal algorithm (15) is globally asymptotically stable.

Proof: See Appendix I-A. ∎

B. Dual Algorithm

The aim of devising a dual algorithm, as opposed to the primal algorithm, is to move the dynamics to the price setting part, i.e. routers. The dual problem is obtained by defining the dual objective function of (12). The dual function is

$$\mathcal{D}(p) = \max_{x_r} \ \mathcal{L}(x, p)\ . \tag{16}$$

Substituting the RHS of (16) with (10) yields

$$\mathcal{D}(p) = \sum_{r\in R}\Big(U_r\big(x_r(p_r)\big) + x_r(p_r)\sum_{l\in L_r}\log_\phi(1 - p_l)\Big) + \sum_{l\in L}\Big(c_l\big(-\log_\phi(1 - p_l)\big)\Big)\ . \tag{17}$$

The dual problem is now defined as

$$\min_{p} \quad \mathcal{D}(p) \tag{18a}$$

$$s.t. \quad 0 \le p_l \le 1, \quad \forall l \in L. \tag{18b}$$

Optimization problem (18) can be solved using different methods. To solve it using gradient descent, we need to take its partial derivative with respect to p:

$$\frac{\partial \mathcal{D}}{\partial p_l}(p) = (y_l - c_l)\frac{-1}{1 - p_l}\ . \tag{19}$$

We can obtain the following algorithm from (19)

$$\dot{p}_l = \sigma_l(p_l)\frac{1}{1 - p_l}[y_l - c_l]^+_{p_l}\ , \tag{20}$$

$$x_r = U_r^{'-1}\big(-\log_\phi(1 - p_r)\big) \tag{21}$$

where $\sigma_l(p_l)$ is a positive, increasing and continuous step size function, and

$$(.)^+_a = \begin{cases} . & : a > 0 \\ \max(., 0) & : a = 0 \end{cases}\ .$$

We call it **Dual1**. Every source r uses (21) to calculate its send rate for the next interval.

Theorem 7.2: The dual algorithm (20) is globally asymptotically stable.

Proof: See Appendix I-B. ∎

To deploy **Dual1**, routers need to implement (20), which might not be applicable on commodity hardware. We derived (20) to present a complete set of algorithms. Now, we present another dual algorithm, called **Dual2**, which can also solve (19); this shows that, as opposed to REM, which needs an exponential function at routers, we can use a linear function of the offered load to a link. This facilitates deployability over commodity hardware via RED. Assuming the utility function is concave, the following dual algorithm solves (19)

$$\dot{p}_l = \sigma_l(p_l)[y_l - c_l]^+_{p_l}\ . \tag{22}$$

We can see (20) as a special case of (22) where $\sigma_l(p_l) = \omega_l(p_l)^{1}/_{(1\,-\,p_l)}$ and $\omega_l(p_l)$ is some function of p_l. The solution of (22) is also unique because it is a compact set and the utility function is concave. This means that as long as the choice of $\omega_l(p_l)$ does not violate the conditions that we assumed about the step size function, i.e. positive, increasing and continuous, then (22) also solves (19) or equivalently (20).

Theorem 7.3: The dual algorithm (22) is globally asymptotically stable.

Proof: See Appendix I-C. ∎

Theorem 7.3 means that using the dual algorithm (22) at routers and the rate update (21) at sources converges to the same optimal rates and marking probabilities.[8] We now evaluate the possibility of using the RED AQM algorithm to

[8]In practice, the calculation of both (20) and (22) might yield values larger than 1. In this case, p_l is limited to 1, and if it becomes 1, p_r will also be 1. In a practical implementation, sources can slow down to the minimum send rate, e.g. 1 packet per RTT, when $p_r = 1$. If the minimum send rate of sources still overloads the buffer at the bottleneck and packets are dropped, this will be handled by the TCP's time-out mechanism. See our code for an actual implementation at `http://shorturl.at/ceiqr`

set the congestion price for the dual algorithm. In the discrete form, we define

$$p_l[n] = \left[\frac{b_l[n]}{\max_{\text{th}}}\right]_0^1 \tag{23}$$

where $b_l[n]$ and $\max_{\text{th}}$ are the backlog (queue length) of link l at time step n and some maximum threshold constant on the backlog size.

Theorem 7.4: Under the following conditions, C1–C2, the dual problem (23) solves (18) if the parameter $\max_{\text{th}} > \overline{\alpha}\,\overline{L}\,\overline{S}$ where $\overline{\alpha}$, $\overline{L}$, and $\overline{S}$ respectively denote the upper-bound on all $-1/U_r''(x_r)$, the length of the longest path used by sources, and the number of competing sources at the most congested link.

C1: The gradient of the dual objective function is Lipschitz.

C2: The queue service discipline guarantees that if the aggregate backlog is increased (decreased), then the backlog of no individual source is strictly decreased (increased).

Proof: See Appendix I-D. ∎

We call the dual algorithm (23) **Dual (RED)** because it can be *practically* implemented by RED with an instantaneous queue length by setting RED's averaging parameter (w_q) to 1; this means that via simple parameter tuning, we can use off-the-shelf router hardware[9], and sources use (21) to calculate their send rate. It should be noted that [24, Theorem 2] that we built upon to prove the above theorem does not consider feedback delay and noise in price estimation at sources. Employing RED in practice to implement (23) necessitates estimating the end-to-end marking probability using, for example, exponential smoothing because the price is fed back using a single bit. This indeed leads to some fluctuation around the equilibrium send rates and queue lengths. In the extensive simulation studies in section VIII-D, we did not observe any instability due to the noise produced by RED.

Condition C2 in the above theorem can be achieved by service disciplines such as round-robin or generalized processor sharing [24]. However, a FIFO queue does not guarantee that, but condition C2 can be met on average by RED stochastically marking packets in a FIFO queue.

C. Primal-Dual Algorithm

The primal-dual problem can lead to an algorithm running on both routers and sources. This class of algorithms can better represent many current algorithms that operate on both sides. They correspond to averaging over both cost and send rate at both routers and sources. Therefore, we take the algorithms (15) and (20) from the previous algorithms as below to present a primal-dual algorithm, which yields

$$\dot{x}_r = \varrho_r(x_r)\Big(U_r'(x_r) + \log_\phi\left(1 - p_r\right)\Big)\,, \tag{24a}$$

$$\dot{p}_l = \sigma_l(p_l)\frac{1}{1-p_l}[y_l - c_l]_{p_l}^+\,. \tag{24b}$$

In this algorithm, **Primal-Dual1**, sources use (24a) and routers use (24b) to adapt their send rate and cost, respectively.

[9]This ease of installation is one of the known benefits of DCTCP, which could also be deployed via an unusual way of configuring RED's parameters. Different from DCTCP, $\min_{\text{th}} = 1$ in our case, and $\max_{\text{p}} = 1$.

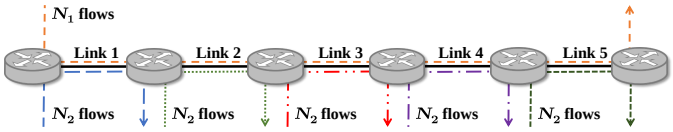


Fig. 4. A simple topology with N_1 five-hop flows crossing all links, and N_2 single-hop flows on each link

Theorem 7.5: The primal-dual algorithm (24) is globally asymptotically stable.

Proof: See Appendix I-E. ∎

Instead of (24b) in the above algorithm, it is also possible to use (22). The stability property of the new algorithm is proven to be the same as the above algorithm, but as it is straightforward and similar to the above, we do not provide a proof. We call the new algorithm using (22) **Primal-Dual2**.

VIII. Simulation Results

A. Configuration

Consider the parking lot topology shown in Fig. 4; there are N_1 flows crossing the five bottleneck links, and N_2 flows crossing each link. All the links are similar, and $x^{(1)}$ and $x^{(2)}$ denote the average send rate of the five-hop flows and single-hop flows[10], respectively. For simplicity we use $U(x_r) = \log_\phi(x_r)$. Optimizing (4) and (12) gives $x_r^* = \frac{1}{q_r^*}$ and $x_r^* = \frac{1}{-\log_\phi(1-p_r^*)}$, respectively. We set the link capacity to $c_l = 83$ packets per millisecond, which is close to 1 Gbps with 1500 B packets, and $\phi = 10$. Since, in this example, there is the same number of competing five and single-hop flows on each link l, $\lambda_l, \forall l$ should be equal at equilibrium, and $q_r = \sum_{l\in L_r}\lambda_l$ for five-hop flows should be 5 times that of single-hop flows. Hence, at equilibrium, $x^{(1)}/x^{(2)} = 0.2$.

We used Mathematica [44] to optimize (4), in which p_r is directly used as the approximation of q_r, and (12) which maps the marking probability to cost by (8). Fig. 5 illustrates the results. We see that in all the cases of (4), the ratio is higher than 0.2. As the number of competing flows increases, the ratio increases up to 0.3 for $N_1 = 30$ and $N_2 = 8$. However, the ratio remains as expected using (12). This confirms that the difference between *sum* and *product* becomes significant when there are more competing flows.

B. Model Validation

In this section, we evaluate the algorithms presented in section VII. Although our theoretical result is valid for every network topology, we again use the topology shown in Fig. 4 for the simulations in the INET framework of OMNeT++ [31]. All the link configurations are the same; their bandwidth is 1 Gbps and their propagation delay is 1 ms. The number of single-hop flows over each link, $N_2 = 8$, and it is fixed for all the simulation scenarios. However, the number of five-hop flows, N_1, varies from 1 to 30. The larger the value of N_1, the higher the marking probability, meaning that our algorithms are effective if the ratio $x^{(1)}/x^{(2)}$ remains constant for all N_1.

We ran each of the algorithms for different values of N_1 for 15 seconds, 10 repetitions, each with a unique random seed.

[10]This is the rate of each of the flows, not the aggregate rate.

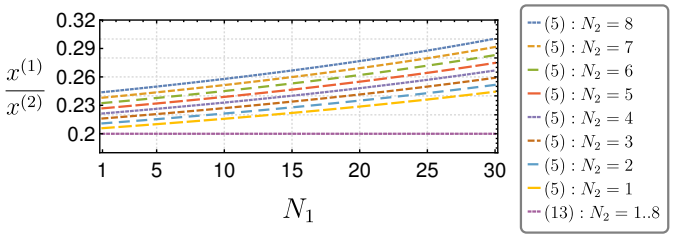


Fig. 5. The ratio of $x^{(1)}/x^{(2)}$ for the topology in Fig. 4 obtained from optimizing (4) and (12) for a varying number of nodes.

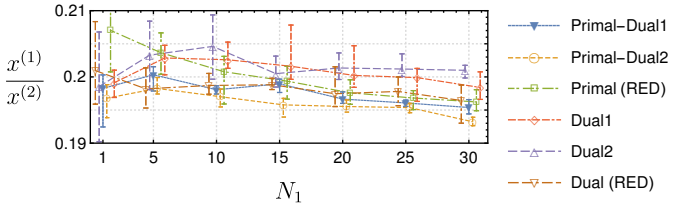


Fig. 6. Distribution of $x^{(1)}/x^{(2)}$. $N_2 = 8$ and N_1 varies from 1 to 30.

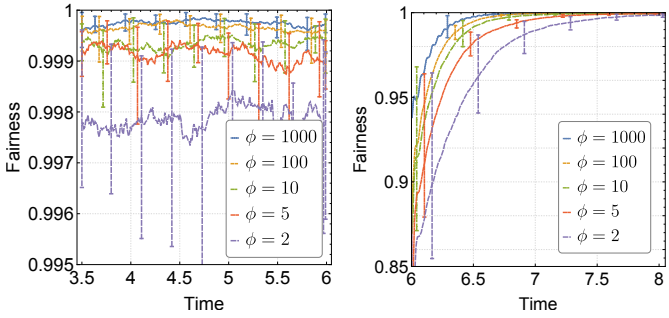
Sources randomly start to transmit during an interval shorter than one RTT and update their rate each RTT. Again we use the utility function $\log_\phi(x_r)$, where $\phi = 10$.

Fig. 6 illustrates the distribution of $x^{(1)}/x^{(2)}$ for various values of N_1 and different algorithms. The plots are displaced a little over x-axis values to increase their visibility. The error bars illustrate the range, and the lines show the average value over the runs. In Primal, $h_r = 1.5$. In Dual1, $h_l = 0.0002$, and in Dual2, $h_l = 0.00002$. In RED, $\min_{\text{th}} = 1\,\text{B}$, $\max_{\text{th}} = 220\,\text{KB}$, $\max_{\text{p}} = 1$, and $w_q = 1$ (see Appendix III for more discussions on these parameters). In all the dual algorithms, sources use an exponential smoothing average over the marks they receive, which is similar to (25) with a smoothing factor of 0.02. In Primal-Dual algorithms, the configurations of each of the primal or dual algorithms are the same as before. This also adds delay in the feedback loop that might have negative effects in terms of stability. However, with the above choice of parameters, we did not observe any instability. We started the measurement after 7 seconds of the transmission, measuring the steady state average send rate.

The end-to-end marking probabilities were between 0.07 to 0.45 in all the simulation scenarios. We observe that increasing the number of five-hop flows did not affect the $x^{(1)}/x^{(2)}$ ratio of ~ 0.2. In the implementation the congestion window size corresponds to send rate quantized by 1460 B segments. This quantization causes the ratio to decrease a little as $N_1 \to 30$.

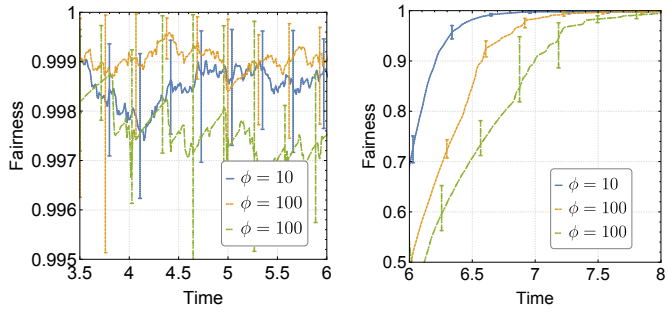
C. Effects of Marking Probability on Convergence Rate

In order to evaluate effects of different marking probabilities, we also simulated a scenario in which there are 8 senders in a dumbbell topology. The "Dual (RED)" algorithm is used, and senders use exponential smoothing with parameter 0.02 to estimate the marking probability. Parameter $\max_{\text{th}}$ is set such that the bottleneck link is fully utilized. Fig. 7(a) illustrates short-term fairness using Jain's fairness index over 10 different runs, calculated using a time window over the last 10 ms at any point in time. We observe that by increasing ϕ, the flows operate more smoothly, which also implies that fluctuations are shorter. In this scenario, the marking probabilities when $\phi = 2, 5, 10, 100, 1000$ are 0.031, 0.066, 0.087, 0.149, and



(a) Steady-state short-term fairness (b) A new flow joins

Fig. 7. Effects of marking probability on convergence: $c_l = 1$ Gbps



(a) Steady-state short-term fairness (b) A new flow joins

Fig. 8. Effects of marking probability on convergence: $c_l = 40$ Mbps

0.211, respectively. In the second scenario, we used different parameters for exponential smoothing, i.e. 0.007, 0.01, 0.013, 0.016, and 0.02 for the above ϕ values, to have similar short-term fairness. Then, at $t = 6\,\text{s}$, a new flow joins the network, and short-term fairness is measured. Results are shown in Fig. 7(b). We see that using smaller values of ϕ results in slower convergence rates. Therefore, it is more beneficial to have a higher equilibrium marking probability in this case.

We also decreased the capacity of the links to 40 Mbps in the above scenario, which consequently increases the equilibrium marking probability to very high values. We set $\phi = 10, 100, 1000$, which respectively result in marking probabilities of 0.84, 0.97, and 0.99. Fig. 8(a) illustrates that short-term fairness is affected, especially by $p_r = 0.99$. Fig. 8(b) shows that the rates still converge when the marking probability is very high, although it is slower. This means that our algorithm can operate even with $p_r = 0.99$. However, to make it faster and smooth, ϕ should be set such that p_r does not get very small (e.g. < 0.01) or very large (e.g. > 0.97). These results confirm that congestion controllers can benefit from a higher equilibrium marking probability to achieve 1) faster convergence and 2) smoother and more fair behavior. However, the marking probability should not be extremely high; again, it can be lowered using smaller ϕ_2 than ϕ_1.

D. Effects of Noise and Delay in ECN Signals

Although the evaluation presented in Section VII does not include feedback noise and delay for tractability, the results illustrated by Fig. 6, Fig. 7, and Fig. 8 confirm that even in the presence of noise and delay, which are set similarly to typical network topologies, the algorithms work well. Sources used exponential smoothing to estimate p_r. We did not observe instability in any scenario. In general, noise and delay make cost

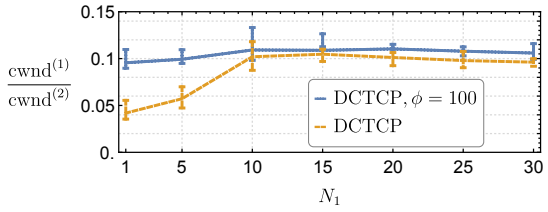


Fig. 9. DCTCP rate ratio

estimation using a binary signal like ECN difficult with slower convergence and more fluctuations, but as shown by Fig. 7, and Fig. 8, having a higher equilibrium marking probability is very beneficial, especially in the presence of these factors.

E. DCTCP

We simulated DCTCP under the topology shown in Fig. 4. Let $\text{cwnd}^{(1)}/\text{cwnd}^{(2)}$ denote the congestion window size of 5-hop flows divided by that of 1-hop flows. As we see in Fig. 9, due to the changes in marking probability when N_1 increases, the cwnd ratio changes. When $N_1 = 1$, the cwnd ratio is around 0.04, and it increases up to 0.1 by increasing N_1. We also applied our approach to DCTCP. The whole DCTCP operation is kept the same; however, when a sender obtains the marking probability of the last RTT, instead of p_r, it uses the value of $[-\log_\phi(1-p_r)]_0^1$ as the marking probability during the last RTT. We show the effect of this simple change in Fig. 9, denoted by "DCTCP, $\phi = 100$", and we can observe that the cwnd ratio remains almost the same regardless of N_1.

IX. Discussion

A. Obtaining Utility Functions

Theorem 5.1 enables us to obtain the utility function of the congestion controllers that directly use marking probability as cost in their rate update equation. Their utility function cannot be obtained by cost approximation because "cost" is not approximately equal to "marking probability" in them.

To obtain the utility, first we replace p_r with $1-\phi^{-q_r}$ in the rate update equation of the controllers that use marking probability. Then, using the fact that $q_r^* = U_r'(x_r^*)$, we can calculate the integral of q_r. In the following, we obtain the utility function of two congestion controls which have higher marking probabilities: DCTCP and LGC.

1) DCTCP [4]: In this method, source r increments the window by 1 if there is no marked ACK in the last transmitted window of packets. Otherwise the window is reduced in proportion to the number of received marks, i.e. $W_r = W_r(1-\frac{\alpha}{2})$ where W_r is the window size, and

$$\alpha = (1-g)\alpha + gF \tag{25}$$

is the exponentially-smoothed value of F with parameter g. F also denotes the ratio of marked ACKs during the last RTT. The window changes in each period according to $\frac{1}{R}(1-\hat{\alpha}_r) - \hat{\alpha}_r(1-\frac{\alpha_r}{2})\frac{W_r}{R}$ where R is RTT and $\hat{\alpha}_r$ is the probability of getting *at least* one mark during an RTT. We analyze the scenario in which there are several synchronized flows. As mentioned by [4], $\hat{\alpha}_r = \alpha_r$ in this case. Therefore, after dividing by R, the rate update equation is

$$x_r[n+1] = \Big[x_r[n] + \frac{1-\alpha_r}{R^2} - \frac{\alpha_r x_r[n]}{R}(1-\frac{\alpha_r}{2})\Big]^+.$$

Substituting $\alpha_r = 1-\phi^{-q_r}$ in the above and solving for q_r yields

$$q_r = -\log_\phi\left(\frac{-1+\sqrt{R^2x_r^2+1}}{Rx_r}\right), \tag{26}$$

and by solving $q_r^* = U'(x_r^*) = 0$, we get

$$U_r(x_r) = \frac{\sin^{-1}(Rx_r)}{R} - x_r\log_\phi\left(\frac{-1+\sqrt{1+R^2x_r^2}}{Rx_r}\right).$$

It can be easily verified that, for $x_r > 0$, the utility function is strictly concave because its second derivative, $-1/(x_r\sqrt{R^2x_r^2+1})$, is negative. It is also strictly increasing because $0 < (-1+\sqrt{R^2x_r^2+1})/(Rx_r) \le 1$ which makes $U'(x_r)$ (i.e. (26)) positive.

2) LGC [40]: As a congestion controller based on the logistic growth function, LGC was proposed for data centers with a similar usage of ECN as in DCTCP. In LGC, the marking probability is at least 0.5 for two flows with one bottleneck link; if there are S competing flows, the marking probability at equilibrium will be $(S-1)/S$. It will increase even more over a path with multiple bottlenecks. Since LGC was not originally defined as a NUM problem, the utility function was not defined. However, using Theorem 5.1 we can obtain its utility function. The rate of source r is updated by

$$x_r[n+1] = x_r[n]\gamma_r\big(1-x_r[n]-\hat{l}_r[n]\big) + x_r[n] \tag{27}$$

where γ_r is a constant value and $\hat{l}_r[t]$ is the measured marking probability at source r. The utility is obtained by considering that $\hat{l}_r$ corresponds to the cost value of $1-\phi^{-q_r}$. Then, using the fact that $q_r^* = U'(x_r^*)$, we get

$$U_r(x_r) = x_r - x_r\log_\phi(x_r). \tag{28}$$

This is a strictly concave function in the range $[0,1]$; LGC uses a normalized rate which always lies in this range.

B. Deflating/Inflating the Marking Probability

As a result of applying Theorem 5.1, we are also able to change the equilibrium marking probability; parameter ϕ was shown how can influence equilibrium marking probabilities. We now apply the same technique to LGC. Using LGC's utility function, (28), we can write its rate update in the form of a primal algorithm. Thus, using (14) and (8), we have

$$\dot{x}_r = x_r\gamma_r\Big(-\log_{\phi_1}(x_r) + \log_{\phi_2}(1-p_r)\Big). \tag{29}$$

We used different base parameters, ϕ_1 and ϕ_2, since (28) and (8) do not necessarily need to have the same base. However, all sources should use the same values for ϕ_1 and ϕ_2.

The equilibrium marking probability of (29) can be obtained using a similar approach as used for LGC in [40] when there is one bottleneck. In equilibrium, $\log_{\phi_1}(x_r^*) = \log_{\phi_2}(1-p_r^*)$. Since there is one bottleneck, p_r^* and consequently x_r^* are the same for all r. As sources are greedy, $\sum_{r\in R} x_r^* = 1$, which means that $x_r^* = 1/S$. This yields $\log_{\phi_1}\left(\frac{1}{S}\right) = \log_{\phi_2}(1-p_r^*)$. Solving the above equation for p_r^* results in

$$p_r^* = 1 - e^{\frac{\log(\phi_2)\log(\frac{1}{S})}{\log(\phi_1)}}. \tag{30}$$

We see that if $\phi_1 = \phi_2$, then (30) is reduced to $^{(S-1)}/_{S}$, which is equal to the original equilibrium marking probability of LGC. However, if $\phi_1 > \phi_2$, then p_r^* decreases. For example, if $\phi_1 = 10$, $\phi_2 = 1.2$, and $S = 10$, then $p_r^* \approx 0.17$, where it was originally $^{(10-1)}/_{10} = 0.9$.

We numerically solved (29) for different ϕ_2 values over the topology in Fig. 4 with $\phi_1 = 10$. The optimum send rates of both (27) and (29) are the same. We observed that where $N_1 = 30$, $N_2 = 8$, and $\phi_2 = 1.2$, the equilibrium probability was around 0.56, but it became very close to 1 for larger values, meaning that it was successfully deflated (further discussion in Appendix II).

C. Relationship With Other Unsaturated Signals

Since packet marking is probabilistic, it can be saturated when the marking probability is not small. This has been the purpose of finding an unsaturated (i.e. in the range $[0, +\infty)$) marking method in [10], [18], which suggests to use $^{p}/_{(1-p)}$ as the congestion signal in end hosts (p is the packet marking probability). Here, we show how it relates to our congestion signal, i.e. (8). For the sake of simplicity, we assume $\phi = \mathrm{e}$, and approximate (8) with its Taylor series:
$-\ln(1-p) = \ln(\frac{1}{1-p}) = \ln(1+\frac{p}{1-p}) = \sum_{n=1}^{\infty}(-1)^{n+1}\frac{1}{n}(\frac{p}{1-p})^n$.
This shows that $^{p}/_{(1-p)}$ is the first term on the RHS of the Taylor series, which can roughly approximate $-\ln(1-p)$, especially where p is small. Therefore, the more terms are included, the more accurate the signal becomes.

D. Future Directions

The demand for low latency communication has recently brought the issues of buffer sizing and AQM to the fore. Efficient early feedback was an important topic at a recent workshop on buffer sizing,[11] affirming our plan to investigate the applicability of our additive ECN signal for very early feedback, by congestion marking *before* a queue even grows. Mechanisms that work with such feedback—e.g., XCP [20], RCP [7] and MaxNet [45]—have been known to outperform implicit-feedback based congestion controls such as standard TCP or Cubic [35], which is the default selection in Linux and Microsoft Windows at the time of writing. However, such explicit feedback based schemes were found to be very hard to deploy due to i) the special calculations needed in routers, ii) the extra packet space needed for feedback, and iii) fairness problems between this kind of traffic and other Internet traffic.

Our signal is based on probabilistic marking, implementable using existing hardware (even for marking "below the queue": as shown in [5], this can be done with virtual (phantom) queues using the Cisco Nexus 5548p switch), with an adjustable marking probability range, and additive in nature. It therefore lends itself to such early marking without having problems i) or ii) above. If a non-additive ECN signal would be used to convey "half full" by ECN-marking proportional to the filling level, the signal would quickly be ruined when traversing multiple routers in sequence: if approximately half of the packets being marked would convey "half full", multiple "half full" routers would produce an "almost completely full" signal. With our ECN interpretation, each router would instead add a congestion cost in accordance with the NUM framework, enabling us to implement a useful control in end systems.

Others have shown that adding controller-specific calculations to routers is not necessary: the very recent HPCC [23] mechanism only requires switches or routers to add fine-grain load information to some packets and moves all the calculations to end systems, which should make deployment easier. A much older mechanism, CADPC/PTP [43], applies a similar model with logistic growth, which further motivates us to try such early marking with our ECN signal. Deployment problem iii) is currently being addressed by the "L4S" proposal [12], and there are other possibilities too (e.g., traffic could be segragated via a DiffServ Code Point (DSCP) value when the mechanism is used for shorter-range communication).

[11] http://buffer-workshop.stanford.edu/

X. Conclusion

In this paper, we presented a novel method to estimate the *additive* path cost required by the NUM framework. Previous work has approximated path cost using ECN, which is naturally a *product* value, and approximation is valid only when the marking probability is small. However, modern congestion controllers such as DCTCP suggest a higher marking probability, which violates the approximation. We solved this problem by extending the theory of Lagrange multipliers and introducing new multipliers which are functions of some other variables; we call the functions *function multipliers* and their variables *indirect multipliers*. We extended the KKT theorem, showed that indirect multipliers exist, and discussed how they can form a Lagrangian function. Then, using a logarithmic function as function multiplier, we showed how we can correctly map packet marks into path cost, irrespective of the marking probability.

Although our theory is general, we applied it to the NUM framework. We presented six different distributed algorithms based on the new Lagrangian function, and proved their stability. In particular, we discussed how the already deployed mechanism RED can be used by tuning its parameters to work with an instantaneous queue length; this expands the applicability of our results over commodity hardware. We implemented our algorithms in the INET framework of OMNeT++. Simulation results confirm that by incorporating indirect multipliers, the *additive* path cost can be estimated. Moreover, we showed how our new theory can effectively decrease the marking probability, and how to obtain the utility function of congestion controllers with higher marking probabilities, which could not be done with previous work.

We believe that the established theory in this paper can effectively help design new congestion controllers with a wide range of marking probabilities over ECN-enabled networks. It also strengthens the applications of ECN as a promising congestion signal. This opens the possibility for further work such as arbitrarily changing the equilibrium marking probability, designing new controllers, and stability analysis in the presence of delay, which we will consider in the future.

APPENDIX I
STABILITY ANALYSIS

A. Proof of Theorem 7.1

Proof: We relax the optimization problem (2) to obtain a function in the form of

$$\mathcal{W}(x) = \sum_{r\in R} U_r(x_r) - \sum_{l\in L} B_l\Big(\sum_{r\in R_l} x_r\Big) \tag{31}$$

where

$$B_l(y_l) = \int_0^{y_l} \rho_l(y)dy. \tag{32}$$

$B_l(.)$ is the barrier function, and $\rho_l(.)$ is called the *congestion price function* [38]. We substitute $\rho_l(y_l)$ with $-\log_\phi\big(1-p_l(y_l)\big)$ in (32), and since it is increasing and continuous, $B_l(y_l)$ is convex, and because U_r is strictly concave, $\mathcal{W}(x)$ is strictly concave. This way of defining $\mathcal{W}(x)$ yields (15) where each source r drives x_r towards the solution of ${}^{\partial\mathcal{W}}/_{\partial x_r} = 0$.

Now we define a Lyapunov function as $\mathcal{V}(x) = \mathcal{W}(x^*) - \mathcal{W}(x)$. Taking the time derivative of $\mathcal{V}$ at $x \neq x^*$ yields

$$\begin{aligned}
\frac{d\mathcal{V}}{dt} &= -\sum_{r\in R}\frac{\partial\mathcal{W}}{\partial x_r}\dot{x}_r \\
&= -\sum_{r\in R}\big(U_r'(x_r) + \sum_{l\in L_r}\log_\phi(1-p_l)\big)\dot{x}_r \\
&= -\sum_{r\in R}\big(U_r'(x_r) + \log_\phi(1-p_r)\big)\dot{x}_r \\
&= -\sum_{r\in R}\varrho_r(x_r)\big(U_r'(x_r) + \log_\phi(1-p_r)\big)^2 < 0.
\end{aligned} \tag{33}$$

At $x = x^*$, ${}^{d\mathcal{V}}/_{dt} = 0$. Consequently, algorithm (15) which is derived from (31) is globally asymptotically stable. ■

B. Proof of Theorem 7.2

Proof: We define a Lyapunov function as

$$\mathcal{V}(\lambda) = \sum_{l\in L}(c_l - y_l^*)\lambda_l \quad + \sum_{r\in R}\int_{q_r^*}^{q_r}\big(x_r^* - U_r^{'-1}(\varphi)\big)d\varphi. \tag{34}$$

where $\lambda_l = -\log_\phi(1-p_l)$ and $q_r = \sum_{l\in L_r}\lambda_l$. Taking the time derivative of $\mathcal{V}$ results in

$$\begin{aligned}
\frac{d\mathcal{V}}{dt} &= \sum_{l\in L}(c_l - y_l^*)\dot{\lambda}_l + \sum_{r\in R}\big(x_r^* - U_r^{'-1}(q_r)\big)\dot{q}_r \\
&= \sum_{l\in L}\big((c_l - y_l^*)\dot{\lambda}_l + (y_l^* - y_l)\dot{\lambda}_l\big) \\
&= \sum_{l\in L}(c_l - y_l)\dot{\lambda}_l.
\end{aligned} \tag{35}$$

Substituting the time derivative of λ_l, i.e. ${}^{\dot{p}_l}/_{(1-p_l)}$, in (35), and substituting $\dot{p}_l$ by (20) yields

$$\frac{d\mathcal{V}}{dt} = \sum_{l\in L}\sigma_l(p_l)(c_l - y_l)\Big[y_l - c_l\Big]_{p_l}^{+}\Big(\frac{1}{(1-p_l)^2}\Big) \leq 0.$$

We have ${}^{d\mathcal{V}}/_{dt} = 0$ only at (x^*, p^*), i.e. either $y_l^* = c_l$ or $y_l^* < c_l$ with $p_l = 0$. Otherwise, ${}^{d\mathcal{V}}/_{dt} < 0$ for all $p_l \in [0,1]$. Hence, the dual algorithm is globally asymptotically stable. ■

C. Proof of Theorem 7.3

Proof: The proof is similar to the proof of Theorem 7.2: the Lyapunov function (34) and its time derivative (35) are used, where λ_l and q_r are defined as before. However, we substitute $\dot{p}_l$ by (22), which is our **Dual2** algorithm. This yields

$$\frac{d\mathcal{V}}{dt} = \sum_{l\in L}\sigma_l(p_l)(c_l - y_l)\Big[y_l - c_l\Big]_{p_l}^{+}\Big(\frac{1}{1-p_l}\Big) \leq 0\ .$$

Since $p_l \in [0,1]$, $\frac{1}{1-p_l} \geq 0$, and we also have $\sigma_l(p_l) \geq 0$. The other two terms are proven to be less than or equal to zero similarly as we showed in the previous proof, which makes the Lyapunov function negative, and zero only at the equilibrium. Hence, **Dual2** is globally asymptotically stable. ■

D. Proof of Theorem 7.4

Proof: The proof follows similarly to the proof of Theorem 2 in [24]: it states that if the cost of link l, λ_l, is a fraction of the backlog, i.e. $\lambda_l[n] = \gamma b_l[n]$, under the above conditions and if $0 < \gamma < {}^1/_{(\overline{\alpha}\,\overline{L}\,\overline{S})}$, then the limit point generated by the sequence $(x[n], \lambda[n])$ is primal, dual optimal. However, we set $\gamma = 1/\max_{\text{th}}$, and then use the above inequality to obtain $\max_{\text{th}}$. In this way and according to Theorem 2 in [24], we should have $\max_{\text{th}} > \overline{\alpha}\,\overline{L}\,\overline{S}$, and hence, (23) solves the dual problem. ■

E. Proof of Theorem 7.5

Proof: We define the following Lyaponov function

$$\begin{aligned}
\mathcal{V}(x,\lambda) = &\sum_{r\in R}\int_{x_r^*}^{x_r}\frac{1}{\varrho_r(\varphi)}\big(\varphi - x_r^*\big)d\varphi \\
&+ \sum_{l\in L}\int_{p_l^*}^{p_l}\frac{1-\psi}{\sigma_l(\psi)}\big(\log_\phi(1-p_l^*) - \log_\phi(1-\psi)\big)d\psi
\end{aligned} \tag{36}$$

Taking the time derivative of the above, and substituting (24a) and (24b) yield

$$\begin{aligned}
\frac{d\mathcal{V}}{dt} &= \sum_{r\in R}\big(U_r'(x_r) + \log_\phi(1-p_r)\big)(x_r - x_r^*) \\
&\quad + \sum_{l\in L}\Big[y_l - c_l\Big]_{p_l}^{+}\big(\log_\phi(1-p_l^*) - \log_\phi(1-p_l)\big) \\
&\leq \\
&\quad \sum_{r\in R}\big(U_r'(x_r) + \log_\phi(1-p_r)\big)(x_r - x_r^*) \\
&\quad + \sum_{l\in L}(y_l - c_l)\big(\log_\phi(1-p_l^*) - \log_\phi(1-p_l)\big) \\
&= \sum_{r\in R}\big(\log_\phi(1-p_r) - \log_\phi(1-p_r^*)\big)(x_r - x_r^*) && \text{(37a)} \\
&\quad + \sum_{l\in L}\big(\log_\phi(1-p_l^*) - \log_\phi(1-p_l)\big)(y_l - y_l^*) && \text{(37b)} \\
&\quad + \sum_{r\in R}\big(U_r'(x_r) + \log_\phi(1-p_r^*)\big)(x_r - x_r^*) && \text{(37c)} \\
&\quad + \sum_{l\in L}(y_l^* - c_l)\big(\log_\phi(1-p_l^*) - \log_\phi(1-p_l)\big) && \text{(37d)} \\
&\leq 0.
\end{aligned}$$

According to (11), the sum of (37a) and (37b) is equal to zero. The term (37c) is less than or equal to zero because if $x_r \leq x_r^*$, then $U_r^{'}(x_r) \geq -\log_\phi\left(1-p_r^*\right)$. In other words, U_r' decreases as x_r increases, and vice versa. In the term (37d), if $y_l^* < c_l$, then $p_l^* = 0$ and the term is negative. If $y_l^* = c_l$, then it is zero. Therefore, this term is non-positive. Since the Lyaponov function is equal to zero at (x^*, λ^*), and it is less than or equal to zero at other points, algorithm (24) is globally asymptotically stable. ■

APPENDIX II
MARKING PROBABILITY RANGE

We observe that as ϕ grows, a wider range of cost values is mapped to a shorter range of high marking probabilities. For example, if $\phi = 2.0$, then all $\lambda_l \in [5, +\infty)$ are mapped to the probability range of $[0.95, 1]$. Thus, it is a design issue how to choose ϕ to keep p_l in an appropriate range because depending on the topology and the number of competing flows, p_l^* is different, and then, we need a different value of ϕ to keep p_l^* in the proper range. This parameter has the same effect as ϕ in REM [6]. However, as shown in [6], a *good* range can be found such that the price estimation is accurate; in normal cases, this range is $[0.2, 0.97]$, and if the source algorithm is smoothed, it can span up to $[0.05, 0.99]$.

Considering a total marking probability p_r, we can use (1) to obtain the per router marking probability. In a simple case where there are n bottleneck links in the path with equal p_l values, we have $p_l = \sqrt[n]{1-p_r}$. For example, if $n = 20$ and $p_r = 0.99$ (as an extreme case), then $p_l = 0.79$. We can also infer that if each router marks with a higher probability than 0.79, as shown before, it will be difficult for senders to estimate the cost unless it is smoothed accordingly.

We extended the topology in Fig. 4 from 5 to at most 50 bottleneck links with $N_1 = 30$ and $N_2 = 8$ to observe what happens when the link marking probability increases to high values. We calculated p_l^*, and assumed that sources get $p_r^* - \eta\, p_r^*$ where η represents some noise equal to 0.05 or 0.1. Then, we calculated the rate without error and the rate including error. We observed that in case of $\phi = 1.2$ where $p_l^* \leq 0.6$, there is at most 20% difference between the rates with and without error. However, in case of $\phi = 2.0$, when the marking probabilities grow beyond 0.7 and 0.9 for 10 and 5 percent error, respectively, the difference exceeds 20%.

APPENDIX III
CONFIGURATION OF RED PARAMETERS

RED marks (or drops; here we consider marking only) an arriving packet randomly with probability $p_a \in [0, \max_{\text{p}}]$ which is a function of the average queue size, avg. More specifically, if $\min_{\text{th}} \leq \text{avg} \leq \max_{\text{th}}$, then the packet is marked with probability p_a. If $\text{avg} \leq \min_{\text{th}}$, the packet is never marked, and in case $\text{avg} \geq \max_{\text{th}}$, it is always marked. To calculate avg, an exponential smoothing algorithm with parameter w_q is used. For example, if $w_q = 1$, then RED uses the instantaneous queue length. This is how DCTCP and also Dual (RED) are configured to send a faster signal to senders.

We set $\min_{\text{th}} = 1$ to start generating a cost as soon as possible, but $\max_{\text{th}}$ to larger values recommended by Theorem 7.4 to ensure stability. This is the main difference between DCTCP and our configuration; in DCTCP, $\min_{\text{th}} = \max_{\text{th}}$. We also set $\max_{\text{p}} = 1$; this allows us to utilize a wide range of marking probabilities and exploit their benefits.

Here, we elaborate more on how we set $\max_{\text{th}}$. By x_r, we mean the number of segments (in our case, 1500 Bytes) that source r sends in one millisecond. For example, to fully utilize a 1 Gbps link, we should have $x_r \geq 83$. To ensure stability, we should have $\overline{\alpha} \geq -1/U_r''(x_r) = x_r^2$, for all r, which yields $\overline{\alpha} = 83^2$. In our scenario, the longest path length is $\overline{L} = 5$ and the maximum number of competing flows is $\overline{S} = 38$. As a result, $\max_{\text{th}} \geq \overline{\alpha}\,\overline{L}\,\overline{S} = 1308910$ B. In the actual simulation, we used a smaller value, i.e. 220 KB to see how it would behave. However, we did not observe any instability. Such maximum queue lengths could occur under extreme circumstances, e.g. when many new flows begin to send at the same time, and cause delay outliers. This could motivate applying our method to virtual (phantom) queues instead of physical queues.

It is also possible to use different RED configurations as a dual algorithm. For example, a new configuration can be the simple threshold-based scheme used in DCTCP, where there is one threshold and packets are marked with probability 1 only if the instantaneous queue length exceeds the threshold. To make sure the link is not underutilized, the threshold needs to be higher than 1 (it should be around 30K B in the above example). However, this is a coarser-grain signal than the one produced by RED with the previous configuration because RED starts marking as the queue length exceeds 1, and the marking probability increases as the queue length grows, resulting in earlier feedback and a smoother behavior. In addition, using the simple threshold-based scheme, it takes longer for sources to get the first feedback. We also evaluated effects of queue length averaging in RED, but our proposed configuration outperformed all of these configurations.

REFERENCES

[1] J. Abadie. On the Kuhn-Tucker Theorem. *Nonlinear Programming*, pages 21–36, 1967.

[2] R. Adams. Active Queue Management: A Survey. *IEEE Communications Surveys and Tutorials*, 15(3):1425–1476, 2013.

[3] M. Adler, J.-Y. Cai, J. K. Shapiro, and D. Towsley. Estimation of congestion price using probabilistic packet marking. In *IEEE INFOCOM*, volume 3, pages 2068–2078. IEEE, 2003.

[4] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). In *Proc. ACM SIGCOMM*, pages 63–74, New Delhi, India, 2010.

[5] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, and M. Yasuda. Less is more: Trading a little bandwidth for ultra-low latency in the data center. In *USENIX NSDI'12*, 2012.

[6] S. Athuraliya, S. H. Low, and D. E. Lapsley. Random early marking. In *Proc. COST 263 International Workshop on Quality of Future Internet Services*, QofIS '00, London, UK, UK, 2000. Springer-Verlag.

[7] H. Balakrishnan, N. Dukkipati, N. Mckeown, and C. J. Tomlin. Stability analysis of explicit congestion control protocols. *IEEE Communications Letters*, 11(10):823–825, October 2007.

[8] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[9] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance. In *ACM SIGCOMM '94*, pages 24–35. ACM, 1994.

[10] B. Briscoe and K. De Schepper. Resolving Tensions between Congestion Control Scaling Requirements. Technical Report TR-CS-2016-001, Simula, Jul 2017.

[11] B. J. Briscoe, M. Kühlewind, and R. Scheffenegger. More Accurate ECN Feedback in TCP. Internet-Draft draft-ietf-tcpm-accurate-ecn-09, Internet Engineering Task Force, Jul 2019. Work in Progress.

[12] B. J. Briscoe, K. D. Schepper, M. B. Braun, and G. White. Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture. Internet-Draft draft-ietf-tsvwg-l4s-arch-04, Internet Engineering Task Force, Jul 2019. Work in Progress.

[13] J.-P. Chavas and W. Briec. On economic efficiency under non-convexity. *Economic Theory*, 50(3):671–701, 2012.

[14] L. Chen, S. Hu, K. Chen, H. Wu, and D. H. K. Tsang. Towards Minimal-delay Deadline-driven Data Center TCP. In *HotNets-XII*. ACM, 2013.

[15] G. Fairhurst and M. Welzl. The Benefits of Using Explicit Congestion Notification (ECN). RFC 8087 (Informational), Mar 2017.

[16] J. Farkas. Über die Theorie der Einfachen Ungleichungen. *Journal für die Reine und Angewandte Mathematik*, 1902(124), 1902.

[17] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Trans. Netw.*, 1(4), Aug 1993.

[18] R. Gibbens and P. Key. Distributed control and resource pricing. *ACM SIGCOMM Tutorial*, 2000.

[19] C. Jin, D. X. Wei, and S. H. Low. FAST TCP: motivation, architecture, algorithms, performance. In *IEEE INFOCOM 2004*, March 2004.

[20] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *ACM SIGCOMM*, 2002.

[21] F. Kelly. Mathematical modelling of the internet. In *Mathematics unlimited2001 and beyond*, pages 685–702. Springer, 2001.

[22] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3), 1998.

[23] Y. Li, R. Miao, H. H. Liu, Y. Zhuang, F. Feng, L. Tang, Z. Cao, M. Zhang, F. Kelly, M. Alizadeh, and et al. HPCC: high precision congestion control. In *ACM SIGCOMM '19*, 2019.

[24] S. Low. Optimization flow control with on-line measurement or multiple paths. In *Proc. ITC-16*, pages 237–249, 1999.

[25] S. H. Low. A duality model of TCP and queue management algorithms. *IEEE/ACM Transactions on Networking*, 11(4), Aug 2003.

[26] S. H. Low. Analytical methods for network congestion control. In *Synthesis Lectures on Communication Networks*, 2017.

[27] J. McCauley, Y. Harchol, A. Panda, B. Raghavan, and S. Shenker. Enabling a permanent revolution in internet architecture. In *ACM SIGCOMM '19*, New York, NY, USA, 2019.

[28] R. Mittal, V. T. Lam, N. Dukkipati, E. Blem, H. Wassel, M. Ghobadi, A. Vahdat, Y. Wang, D. Wetherall, and D. Zats. TIMELY: RTT-based Congestion Control for the Datacenter. In *SIGCOMM '15*. ACM, 2015.

[29] A. Nemirovsky and D. Yudin. *Problem complexity and method efficiency in optimization*. John Wiley & Sons, 1983.

[30] K. Nichols and V. Jacobson. Controlling Queue Delay. *Queue*, 10(5):20:20–20:34, May 2012.

[31] OMNeT++. The inet framework. `https://omnetpp.org/`, 2017.

[32] Q.-V. Pham and W.-J. Hwang. Network utility maximization-based congestion control over wireless networks: A survey and potential directives. *IEEE Comm. Surveys & Tutorials*, 19(2), 2017.

[33] I. A. Qazi, L. L. Andrew, and T. Znati. Congestion control with multipacket feedback. *IEEE/ACM Trans. Netw.*, 20(6):1721–1733, 2012.

[34] I. A. Qazi, L. L. H. Andrew, and T. Znati. Congestion control using efficient explicit feedback. In *IEEE INFOCOM 2009*, April 2009.

[35] I. Rhee, L. Xu, S. Ha, A. Zimmermann, L. Eggert, and R. Scheffenegger. CUBIC for Fast Long-Distance Networks. RFC 8312, Feb 2018.

[36] D. Ros and M. Welzl. Less-than-best-effort service: A survey of end-to-end approaches. *Comm. Surveys & Tutorials, IEEE*, 15(2), 2013.

[37] K. D. Schepper, O. Bondarenko, I.-J. Tsang, and B. Briscoe. 'Data Center to the Home': Ultra-Low Latency for All. Technical report, RITE Project, Jun 2015.

[38] S. Shakkottai and R. Srikant. Network optimization and control. *Foundations and Trends in Networking*, 2(3):271–379, 2008.

[39] R. Srikant. *The mathematics of Internet congestion control*. Springer Science & Business Media, 2012.

[40] P. Teymoori, D. Hayes, M. Welzl, and S. Gjessing. Even lower latency, even better fairness: Logistic growth congestion control in datacenters. In *IEEE LCN*, Nov 2016.

[41] P. Teymoori, D. A. Hayes, M. Welzl, and S. Gjessing. Estimating an additive path cost with explicit congestion notification. Technical Report ISBN 978-82-7368-452-3, University of Oslo, 2019. `http://tiny.cc/gvki8y`.

[42] J. Touch. Updated Specification of the IPv4 ID Field. RFC 6864 (Proposed Standard), Feb 2013.

[43] M. Welzl. *Scalable performance signalling and congestion avoidance*. Springer Science & Business Media, 2012.

[44] Wolfram Research, Inc. Mathematica 11.0.

[45] B. P. Wydrowski, L. L. Andrew, and I. M. Mareels. Maxnet: Faster flow control convergence. In *Proc. Networking*. Springer, 2004.

[46] Y. Zhu, H. Eran, D. Firestone, C. Guo, M. Lipshteyn, Y. Liron, J. Padhye, S. Raindel, M. H. Yahia, and M. Zhang. Congestion Control for Large-Scale RDMA Deployments. In *ACM SIGCOMM '15*, 2015.

[47] Y. Zhu, M. Ghobadi, V. Misra, and J. Padhye. ECN or Delay: Lessons Learnt from Analysis of DCQCN and TIMELY. In *CoNext*. ACM, 2016.

Peyman Teymoori received his Ph.D. degree in computer engineering from University of Tehran, in 2013. He was a visiting researcher at Gwangju Institute of Science and Technology, South Korea. Now, he is a researcher fellow in the Network and Distributed Systems group, Department of Informatics, University of Oslo, Norway. His research interests include computer network protocols, algorithmic aspects of wireless ad hoc networks, and performance evaluation.

David A. Hayes received his Ph.D. in Telecommunications Engineering from the University of Melbourne, Australia. His works mainly in the areas of network performance analysis and protocol engineering. Currently he is at Simula Metropolitan Center for Digital Engineering in the Mobile Systems and Analytics department where his work focuses on performance analysis of protocols and network devices in 5G networks and beyond.

Michael Welzl received the Ph.D. (with distinction) and the habilitation degrees from the University of Darmstadt, Germany, in 2002 and 2007, respectively, and was with the University of Linz and University of Innsbruck, Austria. He has been a Full Professor with the Department of Informatics,University of Oslo, since 2009. His habilitation thesis, the Wiley book Network Congestion Control: Managing Internet Traffic, is the only introductory book on network congestion control. He is active in the IETF and IRTF. He has also been participating in several European research projects, including roles such as a coordinator and a technical manager.

Stein Gjessing is a professor of Computer Science in Department of Informatics, University of Oslo. He received his the Cand. Real. degree in 1975 and his Dr. Philos. degree in 1985, both form the University of Oslo. He acted as head of the Department of Informatics for 4 years from 1987. From February 1996 to October 2001 he was the chairman of the national research program Distributed IT-System, founded by the Research Council of Norway. He has worked with computer interconnects and computer architecture for cache coherent shared memory, with DRAM organization, with ring based LANs (IEEE Standard 802.17) and with IP fast reroute. His current research interests are transport, routing and network resilience both in Internet-like networks and in sensor networks.