In-network Congestion Control for Multirate Multicast

Georgios S. Paschos†, Chih-Ping Li¶, Eytan Modiano*, Kostas Choumas§‡ and Thanasis Korakis§

†Mathematical and Algorithmic Sciences Lab, France Research Center, Huawei Technologies Co., Ltd.
¶Qualcomm Research, San Diego, CA, USA
*LIDS, Massachusetts Institute of Technology, Cambridge, MA, USA
§Dept. of ECE, University of Thessaly, Volos, Greece
‡Informatics & Telematics Institute, CERTH, Greece

***Abstract*—We present a novel control scheme that dynamically optimizes multirate multicast. By computing the differential backlog at every node, our scheme adaptively allocates transmission rates per session/user pair in order to maximize throughput. An important feature of the proposed scheme is that it does not require source cooperation or centralized calculations. This methodology leads to efficient and distributed algorithms that scale gracefully and can be embraced by low-cost wireless devices. Additionally, it is shown that maximization of sum utility is possible by the addition of a virtual queue at each destination node of the multicast groups. The virtual queue captures the desire of the individual user and helps in making the correct resource allocation to optimize total utility. Under the operation of the proposed schemes backlog sizes are deterministically bounded, which provides delay guarantees on delivered packets. To illustrate its practicality, we present a prototype implementation in the NITOS wireless testbed. The experimental results verify that the proposed schemes achieve maximum performance while maintaining low complexity.**

***Index Terms*—Congestion control, multicast, network utility maximization, stability, throughput.**

I. Introduction

The increasing demand for multimedia applications, such as real-time conferencing, multiview video and video streaming, pushes data networks to their operational limits and motivates efficient resource allocation schemes. Multicast is a candidate method for delivering multimedia streams to multiple users across a network. To optimize individual user experience, it is desired to employ *multi-rate multicast* transmissions and use layered multimedia coding schemes to adapt users' perceived quality to allowable data rates, see [2], [3]. Since different receivers may require different data rates, we study the problem of *per-receiver Network Utility Maximization* (NUM) in multi-rate multicast, where each receiver is assigned a potentially different utility function.

Controlling multicast streams is challenging; the optimal network resource allocation and stream rate adaptation depends on the network state, which includes channel quality, network congestion, user demand and device capabilities. Current approaches collect network state information at the source and calculate the achievable stream rate per receiver, e.g. [4], [5]. Such a process can be overwhelming for the source, since a multicast stream may have thousands of receivers. In this work we develop a solution to per-receiver NUM in multi-rate multicast without source cooperation. This solution is highly desirable in situations where there is no coordination between stream inputs, and the network is utilized as a decoupled, neutral resource.

Our solution combines scheduling with intelligent packet dropping at intermediate nodes. Packets corresponding to all stream layers are initially injected into the network without any calculations. Progressively, some packets are dropped according to a dropping scheme which bases its decisions on local information. We show that the original stream is stripped of unnecessary packets so that each receiver obtains the exact amount of information that corresponds to maximum throughput. Moreover, we combine the above mechanism with receiver-end congestion control to produce a scheme that maximizes utility without source cooperation.

The proposed policies have the following attractive features. *First*, they dynamically track the optimal solution without explicitly exchanging information about time-varying system parameters such as stream rate, link quality and network congestion. *Second*, they are based on neighbor backlog information which is found locally. Thus the policies are amenable to distributed implementation for wireless and heterogeneous network technologies. *Third*, they do not require source cooperation, i.e. the sources transmit stream packets without calculating the achievable receiver rates–this simplifies multi-rate multicast for networks with a large number of receivers. *Last*, they yield deterministic bounds for the queue backlogs, which provides delay guarantees and facilitates implementation on systems with finite buffers. Our contribution is summarized in the following points:

- We present the Maximum Multicast Throughput (MMT) policy, which achieves near optimal throughput for multi-rate multicast. MMT uses backpressure-type scheduling and threshold-based packet dropping.

A part of this work is published in IEEE INFOCOM 2014 [1].

This work makes use of results produced by the SmartFIRE project, which is supported by the International Research & Development Program of the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (MSIP, Korea) (Grant number: K2013078191) and the Seventh Framework Programme (FP7) funded by the European Commission (Grant number: 611165).

This work was supported by NSF grants CNS-0915988 and CNS-1116209, and ONR grant N00014-12-1-0064. The work of Chih-Ping Li was done when he was at MIT. The work of G. Paschos was done when he was at MIT and it was supported in part by the WiNC project of the Action: Supporting Postdoctoral Researchers, funded by national and Community funds (European Social Fund). Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government, nor of the companies Huawei, and Qualcomm.

- We develop the Maximum Multicast Utility (MMU) policy, which additionally includes a utility-based congestion controller at the receivers. MMU is shown to solve the per-receiver NUM problem.
- We propose MMU-W, a heuristic modification for operation on IEEE 802.11-based wireless devices. We implement MMU-W in a wireless testbed and perform experiments. The results demonstrate the efficiency of the proposed schemes both in terms of achieved utility and computation complexity.

A. Related Work

The problem of Network Utility Maximization (NUM) has been extensively studied for the case of unicast sessions [6]. For multicast sessions, [4] provides a fair-utility offline solution. Maximizing network utility by solving offline optimization problems is less desirable in practice since variability in the network renders this approach ineffective. Every change requires re-solving the problem and enforcing new rules. A preferable methodology is to achieve the long-term goals by making adaptive real-time decisions, cf. [7].

In [8], a dynamic approach balances multicast streams across a selection of multicast trees. This approach provides maximum throughput, which for the special case of feasible arrivals coincides with the solution to the NUM problem. More generally to address the infeasible arrivals we need to design a congestion controller, which works jointly with multicast routing and admits the precise amount of traffic from each multicast session that solves the NUM problem.

Many existing congestion control approaches have the users estimate their maximally allowable throughput and convey this information to the source, [5], [9]. Utilizing such information, the source computes a set of stream layers allocating the optimal rate per layer and then implements a virtual multicast session for each layer. In a time-varying setting, this approach may be cumbersome requiring frequent reports and changes of virtual sessions. Moreover, the complexity of source computations becomes prohibitive in large networks. Our approach differs because it is based on local dropping and it does not require end-to-end signaling or computations at the source.

An in-network congestion control approach is proposed in [10], where a credit-based flow controller is shown to achieve *max-min* fairness, i.e. it solves the NUM problem for a specific choice of utility functions. We generalize in-network flow control for per-receiver multirate multicast NUM. To maximize the per-receiver utility, we utilize a virtual queue at each receiver, inspired by the unicast method proposed in [11]. The core challenge of generalizing [11] to multirate multicast lies in identifying the correct Lyapunov function that uses the information of receiver location in the tree to correctly balance the queue lengths.

A related recent work [12] proposed a dynamic policy for the stochastic NUM problem using the concept of the *shadow backpressure* routing. Virtual packets travel in the reverse direction in order to discover congestion and help route the actual data packets. Each receiver is equipped with a source-type flow controller, inspired by [13]. The combination of these elements is shown to maximize the sum of all receiver utilities under the assumption of infinite demand.

II. System Model

Let $G=(V,E)$ be the graph, serving a set C of multicast sessions. Session $c\in C$ consists of a source node $c\in V$[1] and a set of receivers $U^{(c)}$, and is delivered over a given tree $G^{(c)}=(V^{(c)},E^{(c)})\subset G$. We consider a wireline network in which all links in E can be used simultaneously (we discuss wireless networks in section V).

Time is slotted and in slot t, $A^{(c)}(t)$ packets arrive at the source of session c. We assume that $A^{(c)}(t)$ are i.i.d. over time slots with mean $\lambda^{(c)}$ and take finite values, i.e. $A^{(c)}(t)\le A_{\text{max}}$.

A. Queueing Structure

Each node maintains one *transmission queue* for every outgoing link l and session c, let $Q_l^{(c)}(t)$ denote its backlog at the beginning of slot t. Define $p(l)\in E^{(c)}$ to be the incoming (parent) link to that link[2] and let $L_{\text{out}}(c)\subset E^{(c)}$ be the set of outgoing links of the source node c. Queue $Q_l^{(c)}(t)$ evolves across slots according to

$$Q_l^{(c)}(t+1)\le\left[\left(Q_l^{(c)}(t)-\mu_l^{(c)}(t)\right)^+-d_l^{(c)}(t)\right]^+ \\ +A^{(c)}(t)1_{[l\in L_{\text{out}}(c)]}+\mu_{p(l)}^{(c)}(t),\ l\in E^{(c)},\quad (1)$$

where $\mu_l^{(c)}(t)$ is the allocated transmission rate[3] and $d_l^{(c)}(t)$ is the number of packets that are dropped from $Q_l^{(c)}(t)$. The inequality is due to the fact that the actual transmitted packets over $p(l)$ can be less than $\mu_{p(l)}^{(c)}(t)$ if there are not enough packets to be transmitted in $Q_{p(l)}^{(c)}(t)$. Let $\widetilde{\mu}_l^{(c)}(t)=\min[Q_l^{(c)}(t),\mu_l^{(c)}(t)]$ be the actual packets transmitted over link l in slot t. In addition, let μ_l^{max} denote the capacity of link l. The capacity constraint $\sum_{c\in C}\mu_l^{(c)}(t)\le\mu_l^{\text{max}}$ must be satisfied in every slot. Also, we impose $d_l^{(c)}(t)\in[0,d_{\text{max}}]$, where d_{max} is a system-defined parameter. Throughout the paper, we assume $d_{\text{max}}\ge A_{\text{max}}+\mu_{\text{max}}$, where $\mu_{\text{max}}\triangleq\max_{l\in E}\mu_l^{\text{max}}$ is the maximum link capacity. The value $A_{\text{max}}+\mu_{\text{max}}$ is an upper bound to the incoming data rate to a node, and our choice of d_{max} ensures that the packet dropping rate is large enough so that all transmission queues can always be stabilized. Let $\widetilde{d}_l^{(c)}(t)=\min[Q_l^{(c)}(t),d_l^{(c)}(t)]$ be the actual packets dropped from $Q_l^{(c)}(t)$ in slot t, which can be smaller than $d_l^{(c)}(t)$ if there are not enough packets to be dropped. To provide direct control over the amount of dropped packets, we set up a *drop queue* $D_l^{(c)}(t)$ associated with each transmission queue $Q_l^{(c)}(t)$. Before leaving the system, the dropped packets are "moved" to the drop queue from which they are later

[1]To simplify the notation we do not allow different sessions to have the same source. This limitation can be waived without affecting the results.

[2]Only one such incoming link $p(l)$ exists since $G^{(c)}$ is a tree. Note that the value of $p(l)$ depends on the multicast session under consideration, and we abuse the notation to simplify exposition.

[3]We assume that $\mu_{p(l)}^{(c)}=0$ if $l\in L_{\text{out}}(c)$.

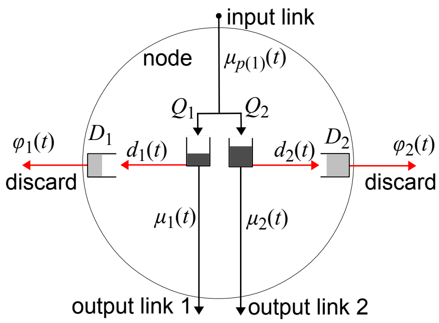


Fig. 1. The proposed queue structure on an node with one incoming and two outgoing links (we show one session and we omit the session notation). Each link $l = 1, 2$ is associated with a transmission queue $Q_l(t)$ and a dropping queue $D_l(t)$.

discarded according to the control policy. The drop queue $D_l^{(c)}(t)$ evolves across slots according to

$$D_l^{(c)}(t+1) = \left[D_l^{(c)}(t) - \varphi_l^{(c)}(t)\right]^+ + \widetilde{d}_l^{(c)}(t),\ l \in E^{(c)}, \quad (2)$$

where $\varphi_l^{(c)}(t) \in [0, d_{\max}]$ is a decision variable that determines the number of packets that are eventually removed from the network in slot t. Note that the packets in drop queues $D_l^{(c)}(t)$ are not going to be transmitted, and therefore it suffices to keep track of the values of $D_l^{(c)}(t)$ only as counters.[4]

Focusing on a network node, our queueing mechanism works as follows. All arriving packets are replicated to each transmission queue $Q_l^{(c)}(t)$, for example see Fig. 1. In a slot t, $\mu_l^{(c)}(t)$ determines the number of session c packets transmitted on link l, $d_l^{(c)}(t)$ decides the number of packets that are internally moved from $Q_l^{(c)}(t)$ to the drop queue $D_l^{(c)}(t)$, and $\varphi_l^{(c)}(t)$ is the number of packets that are discarded from queue $D_l^{(c)}(t)$ and permanently removed from the network. A control policy chooses the values of the decision variables, $\mu_l^{(c)}(t)$, $d_l^{(c)}(t)$ and $\varphi_l^{(c)}(t)$ at each slot.

III. Throughput Maximization

In order to introduce the concepts and the notation, we begin by considering the problem of maximizing the sum throughput of all receivers in multiple multicast sessions. In Section IV, we study the more general problem of utility maximization.

A. Flow-Level Characterization

Before we develop the dynamic network control policy, it is useful to provide a flow-level characterization of the optimal throughput vector and the optimal packet dropping rates, presented as solutions to linear optimization problems. These flow-level solutions are useful for proving the optimality of our control policies. However, the proposed policies solve these problems in a distributed and dynamic manner without the need to explicitly compute these solutions.

[4] The drop queues are essential to our mathematical analysis and they allow us to keep track of the actual number of dropped packets $\widetilde{d}_l^{(c)}(t)$. In Section V-E, we study by simulations a simplified version of our scheme without drop queues.

We define $f_l^{(c)}$ to be the average session c data flow rate over link l and $q_l^{(c)}$ the average packet dropping rate at queue $Q_l^{(c)}$. These flow variables must satisfy the flow conservation and link capacity constraints:

$$\lambda^{(c)} = f_l^{(c)} + q_l^{(c)}, \quad l \in L_{\text{out}}(c),\ \forall c \quad (3)$$

$$f_{p(l)}^{(c)} = f_l^{(c)} + q_l^{(c)}, \quad l \in E^{(c)} \setminus L_{\text{out}}(c),\ \forall c \quad (4)$$

$$\sum_{c \in C} f_l^{(c)} \le \mu_l^{\max},\ f_l^{(c)} = 0 \text{ if } l \notin G^{(c)}, \quad l \in E, \quad (5)$$

The packet dropping rate vector $(q_l^{(c)})$ is said to be feasible if there exist flow variables $f_l^{(c)}$ that satisfy (3)-(5).

Let $r_u^{(c)}$ denote the throughput of receiver $u \in U^{(c)}$. Let $\varLambda$ be the set of feasible throughput vectors $(r_u^{(c)})$. We have

$$\varLambda = \left\{ (r_u^{(c)}) \;\middle|\; \begin{array}{c} \text{(3)-(5) hold, } r_u^{(c)} = f_{l_u}^{(c)} \\ f_l^{(c)}, q_l^{(c)} \ge 0 \end{array} \right\}, \quad (6)$$

where l_u is the incoming link of the receiver u in session c (the session (c) index is omitted for simplicity). In (6), $r_u^{(c)} = f_{l_u}^{(c)}$ states that the throughput of a receiver is equal to its incoming flow rate.

The problem of maximizing the sum throughput of all receivers in the network is

$$\text{maximize} \sum_{c,u} r_u^{(c)}, \quad \text{subject to } (r_u^{(c)}) \in \varLambda. \quad (7)$$

It is useful to consider an equivalent optimization problem that minimizes packet dropping rates. Let $E_u^{(c)}$ denote the set of links that form the path from the source node c to a receiver u. Summing (4) over $l \in E_u^{(c)}$ and using $r_u^{(c)} = f_{l_u}^{(c)}$, we have

$$r_u^{(c)} = \lambda^{(c)} - \sum_{l \in E_u^{(c)}} q_l^{(c)}, \quad (8)$$

which states that the throughput $r_u^{(c)}$ of receiver u is equal to the exogenous data arrival rate less the sum of packet dropping rates along the path $E_u^{(c)}$ to u. Summing (8) over all receivers $u \in U^{(c)}$ in a session, the total session c throughput is

$$\sum_{u \in U^{(c)}} r_u^{(c)} = \left|U^{(c)}\right| \lambda^{(c)} - \sum_{l \in E} m_l^{(c)} q_l^{(c)}, \quad (9)$$

where $m_l^{(c)}$ is the number of session c receivers connected to their source via link l.[5] From (9) we see that maximizing the total throughput of session c is equivalent to minimizing the weighted packet dropping rate $\sum_{l \in E} m_l^{(c)} q_l^{(c)}$. Consequently, the throughput maximization problem (7) is equivalent to the minimization problem

$$\text{minimize} \sum_{c,l} m_l^{(c)} q_l^{(c)}, \quad \text{subject to } (q_l^{(c)}) \text{ feasible.} \quad (10)$$

Next, we design a control policy that stabilizes all queues in the network and achieves optimal packet dropping rates; from the equivalence of (7) and (10), our policy achieves the maximum total throughput as well.

[5] We assume that $m_l^{(c)} = 0$ if $l \notin E^{(c)}$.

B. Intuition for Packet-Level Control

To measure the degree of congestion in the network, we construct a strictly increasing function of the queue backlogs $Q_l^{(c)}(t)$ and $D_l^{(c)}(t)$, i.e., we define the weighted quadratic Lyapunov function

$$L(t) = \frac{1}{2}\sum_c \sum_{l\in E} m_l^{(c)}([Q_l^{(c)}(t)]^2 + [D_l^{(c)}(t)]^2).$$

The quadratic terms are weighted by $m_l^{(c)}$ because the importance of a queue is proportional to the number of receivers connected to that queue. Let $H(t) = (Q_l^{(c)}(t); D_l^{(c)}(t))$ be the queue backlog vector in slot t. Define the Lyapunov drift

$$\varDelta(t) = \mathbb{E}\left[L(t+1) - L(t) \mid H(t)\right] \tag{11}$$

as the expected difference of the congestion measure $L(t)$ over a slot. A control policy that minimizes the Lyapunov drift in every slot suffices to stabilize the network and keep the queue backlogs bounded [7].

Recall from (10), that we also seek to minimize the weighted time-average packet dropping rate

$$\sum_{c,l} m_l^{(c)} q_l^{(c)} = \sum_{c,l} m_l^{(c)} \overline{d}_l^{(c)} \triangleq \sum_{c,l} m_l^{(c)} \lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[\widetilde{d}_l^{(c)}(t)]. \tag{12}$$

If a drop queue $D_l^{(c)}(t)$ is stable, then from queueing theory its arrival rate must be less than or equal to its time-average service rate, i.e., from (2) we have

$$\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[\widetilde{d}_l^{(c)}(t)] \leq \lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[\varphi_l^{(c)}(t)]. \tag{13}$$

Our approach forces this bound to be tight, and hence minimizing (12) can be achieved by minimizing its upper bound in (13), provided all $D_l^{(c)}(t)$ queues are stable. In fact, it suffices to minimize in every slot the sum $\sum_{c,l} m_l^{(c)}\mathbb{E}_H[\varphi_l^{(c)}(t)]$, where $\mathbb{E}_H[\cdot]$ is a compact notation for the conditional expectation $\mathbb{E}\left[\cdot \mid H(t)\right]$.

Minimizing both the Lyapunov drift $\varDelta(t)$ and the sum $\sum_{c,l} m_l^{(c)}\mathbb{E}_H[\varphi_l^{(c)}(t)]$ induces a conflict, because the network becomes more congested when less packets are dropped. It is therefore natural to consider minimizing a weighted sum of these two metrics,

$$\varDelta(t) + V\sum_{c,l} m_l^{(c)}\,\mathbb{E}_H[\varphi_l^{(c)}(t)], \tag{14}$$

where $V>0$ is predefined parameter that reflects the relative importance of minimum packet dropping to queue stability. As we will see, V also controls a tradeoff between the performance gap of our policy from optimality and the required finite buffer size in the transmission queues $Q_l^{(c)}(t)$.

C. The Proposed Policy

Our policy arises from the minimization of (14). In Appendix A we derive the following bound

$$\begin{aligned}(14) \leq{}& \underbrace{B_1 + \sum_{c,l\in L_{\text{out}}(c)} m_l^{(c)} Q_l^{(c)}(t)\,\lambda^{(c)}}_{\text{constant}} \\ & \underbrace{-\sum_{c,l} m_l^{(c)}\mathbb{E}_H[d_l^{(c)}(t)]\big(Q_l^{(c)}(t) - D_l^{(c)}(t)\big)}_{\text{dropping}} \\ & \underbrace{-\sum_{c,l} m_l^{(c)}\mathbb{E}_H[\varphi_l^{(c)}]\big(D_l^{(c)}(t) - V\big)}_{\text{discarding}} \\ & \underbrace{-\sum_{c,l}\mathbb{E}_H[\mu_l^{(c)}(t)]W_l^{(c)}(t)}_{\text{scheduling}}\end{aligned} \tag{15}$$

where $B_1>0$ is a finite constant given in the Appendix, and

$$W_l^{(c)}(t) \triangleq m_l^{(c)} Q_l^{(c)}(t) - \sum_{l':p(l')=l} m_{l'}^{(c)} Q_{l'}^{(c)}(t) \tag{16}$$

is the weighted differential backlog. Fig. 2 gives an example calculation of $W_l^{(c)}(t)$. Next, we propose a throughput-optimal policy that is designed to minimize the RHS of (15) at each slot.

Maximum Multicast Throughput (MMT) **Policy**

Packet Dropping: Each transmission queue $Q_l^{(c)}(t)$ moves $\min\{d_l^{(c)}(t), Q_l^{(c)}(t)\}$ packets to its drop queue $D_l^{(c)}(t)$ at the end of slot t, where

$$d_l^{(c)}(t) = \begin{cases} d_{\max} & \text{if } Q_l^{(c)}(t) > D_l^{(c)}(t) \\ 0 & \text{otherwise.}\end{cases} \tag{17}$$

The drop queue $D_l^{(c)}(t)$ removes $\min\{\varphi_l^{(c)}(t), D_l^{(c)}(t)\}$ packets from the network according to

$$\varphi_l^{(c)}(t) = \begin{cases} d_{\max} & \text{if } D_l^{(c)}(t) > V \\ 0 & \text{otherwise.}\end{cases} \tag{18}$$

Scheduling: Let C_l be the set of multicast sessions that use link l. Define $W_l^*(t) = \max_{c\in C_l} W_l^{(c)}(t)$ and let c_l^* be a maximizer session (ties are broken arbitrarily). We allocate the link rate

$$\mu_l^{(c_l^*)}(t) = \begin{cases} \mu_l^{\max} & \text{if } W_l^*(t) > 0 \\ 0 & \text{otherwise.}\end{cases} \tag{19}$$

Let $\mu_l^{(c)}(t) = 0$ for all the other sessions $c\in C_l\setminus\{c_l^*\}$.

Observe that (17) minimizes the dropping term of (15), (18) minimizes the discarding term and (19) minimizes the scheduling term. Since the first two terms in (15) are constant, we conclude that MMT minimizes the RHS of (15).

We note that the policy operates in a distributed manner using only locally available information. For the computation of $W_l^{(c)}(t)$, we require knowledge of the neighbor backlogs. As shown in prior work, this is not restrictive for practical applications, e.g. see [14]. Also, delayed backlog information is sufficient for throughput optimality, see [15, §4.7].

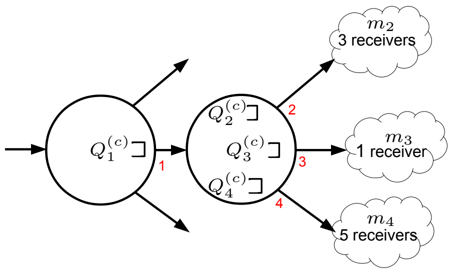


Fig. 2. Illustration of the differential backlog calculation in MMT policy; $W_1^{(c)} = 9Q_1^{(c)} - 3Q_2^{(c)} - Q_3^{(c)} - 5Q_4^{(c)}$.

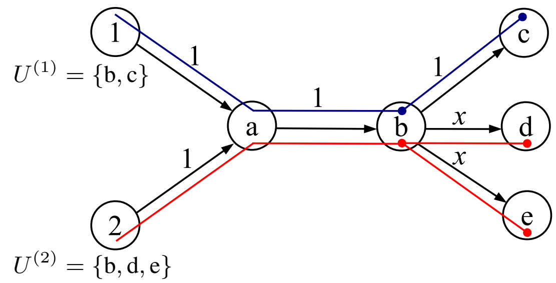


Fig. 3. An example of multirate multicast with two sessions. Session 1 uses the link set $E^{(1)} = \{(1, \mathrm{a}), (\mathrm{a}, \mathrm{b}), (\mathrm{b}, \mathrm{c})\}$ and session 2 uses the link set $E^{(2)} = \{(2, \mathrm{a}), (\mathrm{a}, \mathrm{b}), (\mathrm{b}, \mathrm{d}), (\mathrm{b}, \mathrm{e})\}$. The set of receivers are denoted with $U^{(1)}, U^{(2)}$. Numbers on links indicate capacities.

D. *Performance Evaluation of* MMT

Due to the dropping mechanism in (17)-(18), $Q_l^{(c)}(t)$ and $D_l^{(c)}(t)$ are deterministically bounded. Applying the approach of [11] we have the following result.

Lemma 1. All queues $Q_l^{(c)}(t)$ and $D_l^{(c)}(t)$ are deterministically bounded by

$$Q_l^{(c)}(t) \leq V + 2d_{\max}, \; D_l^{(c)}(t) \leq V + d_{\max}, \; \forall l, c, t.$$

Hence, a buffer size of $V+2d_{\max}$ is sufficient to avoid unexpected queue overflow at $Q_l^{(c)}(t)$. The MMT policy achieves near-optimal total throughput as the following theorem asserts.

Theorem 1 (Optimality of MMT)**.** The MMT policy yields the total throughput satisfying

$$\sum_{c,u} \overline{r}_u^{(c)} \geq \sum_{c,u} r_u^{(c)*} - \frac{B_1}{V}.$$

Where

$$\overline{r}_u^{(c)} \triangleq \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left[\widetilde{\mu}_{l_u}^{(c)}(t)\right]$$

is the throughput of receiver u in multicast session c and $(r_u^{(c)*})$ is a solution to (7). The performance gap B_1/V can be made arbitrarily small by choosing a sufficiently large $V > 0$. Appendix B provides the proof of Theorem 1.

E. *Simulation of* MMT

We illustrate how MMT adapts to changing conditions via simulations. Consider the multicast scenario of Fig. 3. Two multicast sessions share link (a, b). The set of multicast receivers are $U^{(1)} = \{\mathrm{b}, \mathrm{c}\}$ for session 1 and $U^{(2)} = \{\mathrm{b}, \mathrm{d}, \mathrm{e}\}$ for session 2. Links (b, d) and (b, e) have capacity x, while the rest links have unit capacities, i.e. 1packet/slot. Both sessions have unit arrival rate. We seek to maximize total throughput.

Observe that throughput maximization depends crucially on the value of x. For example, if $x = 1$, then maximum throughput is achieved by allocating all the resources of link (a, b) to session 2, since session 2 has three receivers and session 1 has two. If on the other hand $x = 0$, then maximum throughput is achieved by allocating all the resources of link (a, b) to session 1. In general, for $x \in [0, 1]$, throughput is maximized if the allocation on link (a, b) is x to session 2 and $1-x$ to session 1. Note, that the packet dropping decision of node $\{\mathrm{a}\}$ depends on the quality of links (b, d) and (b, e), information which is not directly available at $\{\mathrm{a}\}$.

In the simulation we vary the value x. Initially $x = 1$ and gradually x reduces in steps of 0.1. Fig. 4 (left) shows the receiver c throughput. According to the above discussion, the optimal average throughput is equal to $1-x$, showcased in the Figure with gray line. The simulations showed that the average throughput of MMT is equal to the optimal. Hence, we show the instantaneous throughput averaged in moving windows of 100 slots. At each interval, the throughput converges quickly to the optimal, which shows how MMT adapts to changing conditions.

In Fig. 4 (right), we showcase the backlog at node b with packets destined to node c, for the same sample path. In this simulation we have used $V = 25$ and $d_{\max} = 5$ and by Lemma 1, the backlog is upper bounded by 35 packets. In the simulations, the backlog never exceeds 25 packets despite the link quality variations and the randomness of the arrivals.

IV. Utility maximization

Next we consider the per-receiver NUM problem. Solving this general problem allows to use different utility functions to achieve several objectives such as maximum throughput (studied separately in the previous section), $\alpha-$fairness which includes proportional fairness and max-min fairness as special cases, user priority, and satisfying user-specific quality requirements.

A. *Per-Receiver NUM Problem Formulation*

In multicast session c, a receiver u has a utility function $g_u^{(c)}$, which is assumed to be concave, increasing and continuously differentiable with bounded derivatives.[6] Consider the per-receiver NUM problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{c,u} g_u^{(c)}(r_u^{(c)}) \qquad (20) \\ \text{subject to} \quad & (r_u^{(c)}) \in \Lambda. \end{aligned}$$

[6]We assume $[g_u^{(c)}]'(x) \leq [g_u^{(c)}]'(0) < \infty$. Utility functions that have unbounded derivatives as $x \to 0$, such as $\log(x)$, can be approximated by those with bounded derivatives. For example, we can approximate $\log(x)$ by $\log(x + \xi)$ for some small $\xi > 0$.

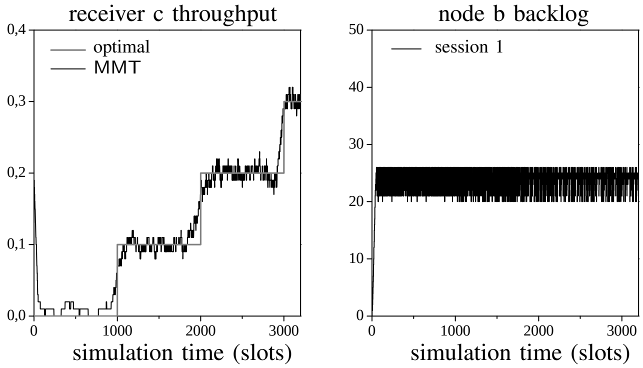


Fig. 4. Performance when varying quality for links (b, d), (b, e) in the topology of Fig. 3. The left Figure compares MMT to the optimal average throughput of receiver c. The right Figure shows the backlog of node b with packets for transmission to receiver c.

Define the auxiliary function

$$h_u^{(c)}(x) \triangleq g_u^{(c)}(x) - \theta x,$$

where $\theta > 0$ is a parameter decided later. Then, maximizing the total utility $\sum_{c,u} g_u^{(c)}(r_u^{(c)})$ is equivalent to maximizing

$$\begin{aligned} &\sum_{c,u}\big(h_u^{(c)}(r_u^{(c)}) + \theta\, r_u^{(c)}\big) \\ &\quad = \sum_{c,u} h_u^{(c)}(r_u^{(c)}) + \theta \sum_{c,u}\Big(\lambda^{(c)} - \sum_{l\in E_u^{(c)}} q_l^{(c)}\Big) \\ &\quad = \sum_{c,u} h_u^{(c)}(r_u^{(c)}) - \theta \sum_{c,l} m_l^{(c)} q_l^{(c)} + \theta \sum_{c,u}\lambda^{(c)}, \qquad (21) \end{aligned}$$

where the last sum is an (unknown) constant. In what follows, we modify our mechanism so that by controlling functions $\mu_l^{(c)}(t), d_l^{(c)}(t), \varphi_l^{(c)}(t)$ and a new virtual queue which will introduce next, the system is driven to the solution of (21).

B. Receiver virtual queue $Z_u^{(c)}(t)$

At each multicast receiver u, we set up the virtual queue $Z_u^{(c)}(t)$, which tracks the deficit/surplus of session c packets received at that user and evolves as

$$Z_u^{(c)}(t+1) = [Z_u^{(c)}(t) - \nu_u^{(c)}(t)]^+ + \widetilde{\mu}_{l_u}^{(c)}(t),$$

where $l_u \in E_u^{(c)}$ is the incoming link of node u. The departures $\nu_u^{(c)}(t)$ are controlled by the policy and chosen in the interval $[0, \nu_{\max}]$, we choose $\nu_{\max}$ below. The functionality of this virtual queue is to track the urgency of a receiver to obtain more packets: if $Z_u^{(c)}(t)$ is small, receiver u must urgently obtain packets for the maximum utility to be preserved.

We also define the *virtual pressure* for each receiver u which is regulated by the virtual queue:

$$Y_u^{(c)}(t) \triangleq \begin{cases} w e^{w(Z_u^{(c)}(t)-\zeta)}, & \text{if } Z_u^{(c)}(t) \ge \zeta, \\ -w e^{w(\zeta - Z_u^{(c)}(t))}, & \text{otherwise}, \end{cases} \qquad (22)$$

where w, ζ are positive parameters whose value will be chosen later. Note, that in normal backpressure, the pressure of a destination node is zero, while in our policy $Y_u^{(c)}(t)$ can take positive or even negative values. The sign of $Y_u^{(c)}(t)$ indicates the urgency of the particular receiver to obtain more or less packets according to the requested objective. Indeed, the behavior of $Y_u^{(c)}(t)$ is controlled by $\nu_u^{(c)}(t)$, which as we will see shortly, is chosen according to the utility function.

C. The Proposed Policy

Let $H(t) = (Q_l^{(c)}(t); D_l^{(c)}(t); Z_u^{(c)}(t))$ be the joint queue backlog vector in slot t. Define the Lyapunov function

$$\begin{aligned} L(t) = \frac{1}{2}\sum_{c,l} m_l^{(c)}\Big([Q_l^{(c)}(t)]^2 + [D_l^{(c)}(t)]^2\Big) \\ + \frac{1}{2}\sum_{c,u}\Big(e^{w(Z_u^{(c)}(t)-\zeta)} + e^{w(\zeta - Z_u^{(c)}(t))}\Big). \end{aligned}$$

Note, that the Lyapunov function is composed of two terms, the quadratic term is identical to the Lyapunov function used in throughput maximization section, while the exponential term is identical to the one used for receiver-based congestion control for unicast sessions in [16]. By using this form of Lyapunov function, we are able to center the virtual queue $Z_u^{(c)}(t)$ around the value ζ.

Recall the definition of Lyapunov drift $\Delta(t)$ from (11). In order to solve the problem in (21) we define the weighted objective:

$$\Delta(t) + V\left[\theta \sum_{c,l} m_l^{(c)} \mathbb{E}_H\big[\varphi_l^{(c)}(t)\big] - \sum_{c,u}\mathbb{E}_H\big[h_u^{(c)}(\nu_u^{(c)}(t))\big]\right]. \qquad (23)$$

Using standard drift derivation techniques we obtain the following bound in the Appendix C

$$\begin{aligned} (23) \le\; & \underbrace{B_2 + \sum_{c,l\in L_{\text{out}}(c)} m_l^{(c)} Q_l^{(c)}(t)\,\lambda^{(c)} + \frac{\epsilon}{2}\sum_{uc} Y_u^{(c)}(t)}_{\text{(constant)}} \\ & \underbrace{- \sum_{c,l} m_l^{(c)} \mathbb{E}_H[d_l^{(c)}(t)](Q_l^{(c)}(t) - D_l^{(c)}(t))}_{\text{(dropping)}} \\ & \underbrace{- \sum_{c,l} m_l^{(c)} \mathbb{E}_H[\varphi_l^{(c)}(t)](D_l^{(c)}(t) - V\theta)}_{\text{(discarding)}} \\ & \underbrace{- \sum_{uc}\mathbb{E}_H\Big\{V h_u^{(c)}(\nu_u^{(c)}(t)) + Y_u^{(c)}(t)\nu_u^{(c)}(t)\Big\}}_{\text{(flow control)}} - \underbrace{\Psi(t)}_{\text{(scheduling)}} \end{aligned} \qquad (24)$$

where $\epsilon > 0$ is a parameter, B_2 is a large constant defined in the Appendix and

$$\begin{aligned} \Psi(t) \triangleq & -\sum_{uc}\mathbb{E}_H[\mu_{l_u}^{(c)}(t)]\, Y_u^{(c)}(t) \\ & + \sum_{c,l} m_l^{(c)} Q_l^{(c)}(t)\,\mathbb{E}_H\big[\mu_l^{(c)}(t) - \mu_{p(l)}^{(c)}(t)\big]. \end{aligned}$$

Let $1_{[l,u]}^{(c)}$ be the indicator function on the event that the tail node of l on $G^{(c)}$ is a receiver $u \in U^{(c)}$. Then, define the weighted differential backlog as

$$W_l^{(c)}(t) = m_l^{(c)} Q_l^{(c)}(t) - \sum_{l' : p(l')=l} m_{l'}^{(c)} Q_{l'}^{(c)}(t) - 1_{[l,u]}^{(c)} Y_u^{(c)}(t). \tag{25}$$

Observe that the virtual pressure $Y_u^{(c)}(t)$ is applied only if the tail node of l is a receiver for this session. By rearranging terms, we have

$$\Psi(t) = \sum_{c,l} \mathbb{E}_H[\mu_l^{(c)}(t)]\, W_l^{(c)}(t).$$

We design our Maximum Multicast Utility (MMU) policy to minimize the RHS of (24). To achieve this, we add a receiver-end congestion controller, similar to the one used in [16].

Maximum Multicast Utility (MMU) Policy

Parameter Selection: Choose positive parameters V, $d_{\max}$, $\nu_{\max}$, w, ζ, and θ as summarized in the Table I. For a discussion on these parameter choices see [11, §V-C]. Initialize the queues with $Q_l^{(c)}(0) = 0$, $Z_u^{(c)}(0) = \zeta + \frac{1}{w}\log\left(\frac{V\theta}{w}\right)$ and $D_l^{(c)}(0) = V\theta$.

Packet Dropping: Same as in MMT policy.

Receiver-End Congestion Control: Choose $\nu_u^{(c)}(t)$ to be the solution to

$$\begin{aligned} \text{maximize} \quad & V h_u^{(c)}(x) + Y_u^{(c)}(t)\, x && (26)\\ \text{subject to} \quad & 0 \le x \le \nu_{\max}, && (27) \end{aligned}$$

where $Y_u^{(c)}(t)$ is given in (22).

Scheduling: Same as in MMT policy, except that we use (25) as the value of $W_l^{(c)}(t)$, instead of (16).

D. Performance Evaluation of MMU

Lemma 2. Under the MMU policy, all queues $Q_l^{(c)}(t)$, $D_l^{(c)}(t)$, and $Z_u^{(c)}(t)$ are deterministically bounded by

$$Q_l^{(c)}(t) \le V\theta + 2d_{\max},\ D_l^{(c)}(t) \le V\theta + d_{\max},\ \forall c, l, t,$$
$$Z_u^{(c)}(t) \le \zeta + \frac{1}{w}\log\left(\frac{V\theta + 2d_{\max}}{w}\right) + \mu_{\max},\ \forall c, u, t.$$

Theorem 2 (Optimality of MMU)**.** The MMU policy achieves the long-term utility satisfying

$$\sum_{c,u} g_u^{(c)}(\overline{r}_u^{(c)}) \ge \sum_{uc} g_u^{(c)}(r_u^{(c)*}) - \frac{B_2}{V} - \frac{3\epsilon}{2}\sum_{c,u}\left([g_u^{(c)}]'(0) + \theta\right), \tag{28}$$

where $(r_u^{(c)*})$ is the utility-optimal throughput vector.

E. Achieving Throughput Requirements

We show how to use the MMU policy to deliver a video stream to users with strict throughput requirements. Consider the optimization problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{c,u\in U^{(c)}} g_u^{(c)}(r_u^{(c)}) && (29)\\ \text{subject to} \quad & (r_u^{(c)}) \in \Lambda, \\ & (r_u^{(c)}) \ge (\xi_u^{(c)}), \end{aligned}$$

where the inequality is element-wise and $\xi_u^{(c)}$ denotes the throughput requirement of session c receiver u. We assume problem (29) admits a feasible solution. In order to solve (29) using MMU, we use the penalty method, see [17, §4.2]. Define the penalty function

$$\pi\left[(x_u^{(c)})\right] \triangleq K \sum_{c,u\in U^{(c)}} \left(\xi_u^{(c)} - x_u^{(c)}\right)^+,$$

where $(x_u^{(c)})$ is a vector with one element for every receiver-session pair. If all requirements are satisfied (i.e. $r_u^{(c)} \ge \xi_u^{(c)}$, $\forall u$) then $\pi\left[(r_u^{(c)})\right] = 0$. If some requirement is violated, then $\pi\left[(r_u^{(c)})\right]$ increases proportionally to K and to the norm-1 distance of $(r_u^{(c)})$ from the feasible set. Also note that π is convex and thus $-\pi$ is concave. Next, consider a convex optimization problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{c,u\in\mathcal{U}^{(c)}} \left[g_u^{(c)}(r_u^{(c)}) - K\left(\xi_u^{(c)} - r_u^{(c)}\right)^+\right] && (30)\\ \text{subject to} \quad & (r_u^{(c)}) \in \Lambda, \end{aligned}$$

By letting $K \to \infty$, the solution of (30) converges to the solution of (29) [17]. A practical approach is to pick a "large" finite value for K.

TABLE I
PARAMETER SELECTION GUIDELINES.

Parameter	Explanation	Suggested values
$m_l^{(c)}$	number of session c users connected to source c through link l	problem defined
$\mu_l^{\max}$	capacity of link l	problem defined
$\mu_{\max}$	maximum link capacity	$\max_l \mu_l^{\max}$
$d_{\max}$	drop batch size	$d_{\max} \ge A_{\max} + \mu_{\max}$
ϵ	utility gap parameter	> 0
$\nu_{\max}$	maximum value for $\nu_u^{(c)}(t)$	$\mu_{\max} + \epsilon/2$
$\delta_{\max}$	bound on $\lvert\nu_u^{(c)}(t) - \mu_{l_u}^{(c)}(t)\rvert$	$\max[\nu_{\max}, \mu_{\max}]$
w	multiplier in (22)	$\frac{\epsilon}{\delta_{\max}^2} e^{-\epsilon/\delta_{\max}}$
ζ	central value for $Z_u^{(c)}(t)$	$\ge \nu_{\max}$
$g_u^{(c)}(x)$	user utility function	objective specific
θ	upper bound on $[g_u^{(c)}]'(x)$, $x \ge \epsilon$	$\max_{u,c}[g_u^{(c)}]'(0)$
$h_u^{(c)}(x)$	auxiliary function	$g_u^{(c)}(x) - \theta x$
V	utility gap/backlog size tradeoff	$V\theta + 2d_{\max} \ge w$

F. Simulations: Prioritizing Base Layer Packets

In multimedia streaming with layer coding, the stream reconstruction requires the reception of specific data packets, belonging to the *base layer*. Then, the reception of additional *enhancement layer* packets improves the quality of the stream. Therefore, a reasonable strategy is to maximize the number of enhancement layer packets subject to the correct reception of base layer packets at each receiver. We show next how to tune MMU to have such a behavior.

We revisit the example of Fig. 3 and set $x = 1$ so that all links have unit capacities. Next, we tag the packets belonging to the base layer video to distinguish them from the enhancement layer packets. The video stream of each session is modeled by a superposition of two Poisson processes with $\lambda_{\text{base}} = 0.2$ and $\lambda_{\text{enh}} = 0.8$.

TABLE II
MMU SIMULATION RESULTS FOR PRIORITIZING BASE LAYER PACKETS.

	Session 1		Session 2		
receivers	b	c	b	d	e
stream rate	0.996		0.998		
$\xi_u^{(c)}$	0.2	0.2	0.2	0.2	0.2
$\overline{r}_u^{(c)}$	0.1948	0.1948	0.805	0.805	0.805

base layer packets breakdown					
stream rate	0.1997		0.199		
received rate	0.1944	0.1944	0.199	0.199	0.199
delivery ratio	97.35%	97.35%	100%	100%	100%

enhancement layer packets breakdown					
stream rate	0.7963		0.799		
received rate	0.0003	0.0003	0.606	0.606	0.606
delivery ratio	0.037%	0.037%	75.84%	75.84%	75.84%

Using the penalty approach explained in the previous subsection, it is possible to achieve throughput maximization subject to rate 0.2 at each receiver. We choose $g_u^{(1)}(x) = g_u^{(2)}(x) = x$, $\xi_u^{(1)} = 0.2$ and $\xi_u^{(2)} = 0.2$ for all $u \in U^{(1)}, U^{(2)}$ respectively. However, additionally to achieving a specific throughput rate requirement, we require the reception of specific packets. To cope with this added constraint, we impose a strict priority rule at all transmission queues $Q_l^{(c)}$: *enhancement layer packets are dropped first and only served if there are no base layer packets left in the queue.*

The resulting experiments for this scenario are shown in Table II. The combination of MMU with the priority rule provides delivery ratio of base layer packets very close to the ideal 100%. The small loss is attributed to randomness of the arrivals. Moreover, when the base layer packet delivery ratio is less than 100%, the enhancement layer counterpart is very small. Conclusively, our policy achieves the high-level goal to combine guaranteed delivery with optimal performance.

V. EXPERIMENTATION IN WIRELESS TESTBED

To demonstrate the practicality of the MMU policy, we develop a prototype implementation in NITOS testbed [18]. NITOS is a heterogeneous outdoor testbed, where two types of networks are used: a wireless network with IEEE 802.11a/b/g/n protocol and a wired network using Gbit Ethernet. Being partly deployed in a building roof, NITOS is a non-RF-isolated wireless testbed. To eliminate interference we employed 802.11a, which is not used by commercial 802.11 products in Greece. The NITOS nodes feature a 3.4GHz Intel i7 processor and two Atheros wireless cards. The main hardware and software specifications of the nodes are depicted in Table III.

A. Implementation Framework

The implementation is based on the Click Modular router framework [19]. *Click* facilitates experimentation and evaluation of scheduling and congestion control algorithms in real systems. It runs as a user-level daemon at each node and via the libpcap library it provides full control on packet transmission. Our implemented framework includes mechanisms for estimating channel quality, forming a queue structure, exchanging queue backlog information, and splitting time into virtual slots.

TABLE III
BASIC CONFIGURATION OF NITOS NODES

Model	Icarus nodes
CPU	Intel i7-2600 Proc., 8M Cache, at 3.40GHz
RAM	Kingston 4 GB HYPERX BLU DDR3
Storage	Solid State Drive 60GB
WiFi cards	two Atheros 802.11a/b/g/n (MIMO)
OS	3.2.0-31-generic Ubuntu precise
Driver	compat-wireless version 3.6.6-1-snpc

Estimating Channel Quality. To evaluate channel quality, we adopted the ETT estimation algorithm of Roofnet [20]. Nodes periodically broadcast probes which are used to estimate the successful transmission probability. With this process every node periodically obtains a table with the qualities for each channel rate/neighbor pair. Using this table, the $\mu_l^{\max}$ parameters are determined. This mechanism is known to incur negligible throughput overhead [20], [21].

Queue Structure. We implement the transmission queues $Q_l^{(c)}$ on each node and we create a counter for each $D_l^{(c)}, Z_l^{(c)}$ virtual queue. The counter $Z_l^{(c)}$ may take non-integer values. Each of these internal queues/counters is created upon the arrival of the first packet of a new session. This allows session generation "on the fly". The queues are removed after a period of inactivity.

Exchanging Queue Backlog Information. To compute $W_l^{(c)}$ from (25), each node broadcasts periodically the backlog size of all its transmission queues $Q_l^{(c)}$. If a node u is a receiver for some session c, it broadcasts $Q_l^{(c)} + Y_u^{(c)}$ instead. The broadcast messaging is repeated once every second. Prior experiments suggest that more frequent broadcasts incur visible throughput overhead, while rarer broadcasts may affect the delay performance due to obsolete queue information.

In the proposed schemes, the routing is based on fixed multicast trees. Thus $m_l^{(c)}$ parameters are predefined and known. However, in our implementation, it is possible to use the backlog exchange mechanism to transport information about $m_l^{(c)}$, should these be time-varying.

Virtual Slots. In order to simplify the implementation we use the concept of the virtual slot. Each node keeps an internal timer that expires once every slot. Upon counter expiration the policy selects the next queue to be served and for the duration of the next slot the decision remains fixed. The slot duration is set to 100msecs, equal to $1/10$ of the broadcasts period. Small values for the slot duration improve delay and reduce throughput fluctuations but burden the CPU of the device. We leave the investigation of optimum slot duration for future work. We note that the implementation of MMU is not tied to the idea of the virtual slot.

B. Policy Implementation

We modify our proposed policy so that it can operate on a network with wireless channels. Due to interference, some wireless links cannot be activated simultaneously. A well known link activation policy is the *maxweight* policy, proposed in [22] for stabilizing mobile packet networks. Maxweight activates at each slot the set of links that maximize the sum products $\sum_l \mu_l^{\max} W_l^{(c)}(t)$, effectively preferring links with higher capacity. In our setting, the activation of the transmitting nodes is automatically selected by the IEEE 802.11 protocol. Thus, it remains to choose the activation of a session and a receiving link, subject to the activated nodes. Using intuition from the maxweight policy we propose the following heuristic.

Maximum Multicast Utility for Wireless (MMU-W) **Policy**

Parameter Selection, Packet Dropping, Receiver-End Congestion Control, Scheduling on Wired Links: same as in MMU.

Scheduling on Wireless Links: Calculate $W_l^{(c)}(t)$ using (25). On a wireless node, choose the link-session pair

$$(l^*, c^*) \in \operatorname{argmax}_{(l,c)} \mu_l^{\max} W_l^{(c)}(t) 1_{[W_l^{(c)}(t)>0]}$$

ties broken arbitrarily. Then, allocate the rate

$$\mu_{l^*}^{(c^*)}(t) = \begin{cases} \mu_{l^*}^{\max} & \text{if } W_{l^*}^{(c^*)}(t) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mu_l^{(c)}(t) = 0$ for all the other link-session pairs.

C. Throughput Experiments and Results

We conduct experiments on the specific topology of Figure 5. Five NITOS nodes are used: Alice and Bob are connected via Ethernet while Bob is connected to the other three nodes via wireless. The nodes are configured to run the MMU-W policy. The wireless links use fixed physical rates instead of the 802.11 rate adaptation scheme. In particular we set the physical rates to 18Mb/s, 6Mb/s and 6Mb/s for the links to Carol, Dave, and Erin respectively. The physical rate of the wired connection is 1Gb/s.

We consider two sessions, A and B, each with traffic rate 14Mb/s. The source node for both sessions is Alice and the multicast receivers are {Bob, Carol} for A, and {Dave, Erin} for B, see Fig. 5. To generate packets we use two UDP streams created with the *iperf* tool [23]. We run *iperf* on external nodes to avoid polluting the CPU measurements. The receiver rate requirements are 4.5Mb/s for Bob, ξ_{C}Mb/s for Carol, 1.7Mb/s for Dave and ξ_{E}Mb/s for Erin, where the values $\xi_{\text{C}}, \xi_{\text{E}}$ are chosen differently per experiment. The objective is to satisfy all receiver rate requirements as well as achieve maximum throughput.

We show the measured instantaneous and average throughput for two scenarios. The instantaneous throughput is computed as the average over 1sec periods. In the first scenario we choose $(\xi_{\text{C}}, \xi_{\text{E}}) = (2.8, 1.7)$, see Fig. 6(a). The objective is achieved because all receiver requirements are satisfied and the excess wireless resource is allocated to the receiver with the highest capacity, i.e. Carol. We observed that the wireless medium was fully utilized. In the second scenario, we reverse the requirements of Carol and Erin, $(\xi_{\text{C}}, \xi_{\text{E}}) = (1.7, 2.8)$, see Fig. 6(b). The theoretical total throughput is smaller in this case due to Erin's low physical rate and high requirement.

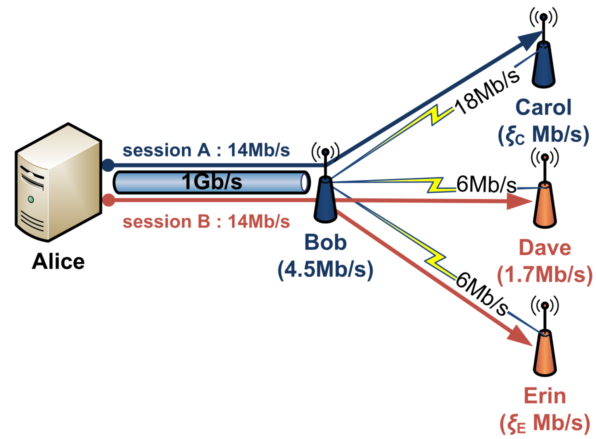
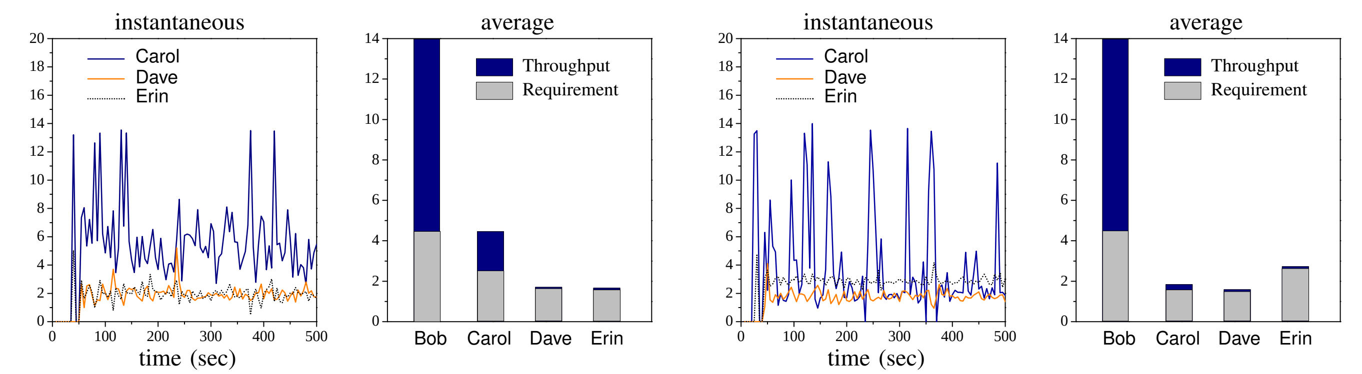


Fig. 5. Experiment topology with five NITOS nodes. Two sessions A and B are generated at Alice, forwarded to Bob via a wired connection, and then distributed to Carol, Dave, and Erin through wireless. The Figure shows the rate requirement per receiver (in parentheses) and the physical rate per link.

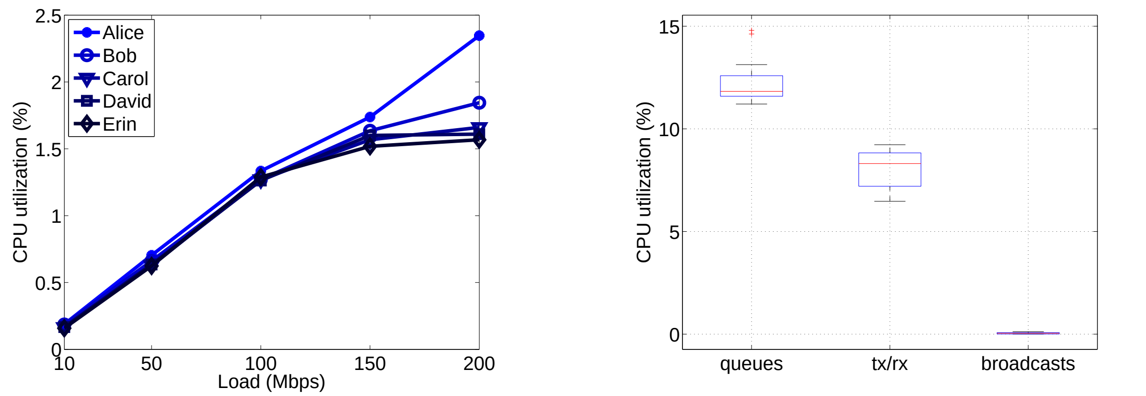
D. CPU Occupancy Experiments and Results

Our framework is implemented on user-level click. We observed the user-level CPU occupancy using the *sigar* library incorporated into the *click* framework, as well as the *valgrind* tool for analytical profiling. The CPU occupancy at every node remained below 3% even when we increased the traffic rate of the two sessions, A and B, up to 200Mb/s. We note, that a kernel-level implementation can improve this figure further. As we see in Fig. 6(c), the CPU usage remains the same for each node in most cases, indicating that our policy does not incur extra burden on the sources. A slight difference between the source and the other nodes is attributed mainly to high load and the fact that at optimality this specific node is charged to drop a lot of packets. However, we note that this difference is not comparable to the utilization variance caused by a congestion control based on source cooperation. Additionally, the CPU utilization was largely independent of data rates used, since it is slightly increased by almost 2.5% when the traffic rate of the sessions is extremely increased up to 200Mb/s and more than 180Mb/s are dropped.

This implies that packet operations and queue maintenance have a minor contribution to the CPU occupancy. This is also illustrated in Fig. 6(d), where we see that the percentage of the CPU occupancy of *click* that occurs due to the queues maintenance, reception/transmission operations and message broadcasting is low enough (almost 15%). Instead most of the CPU occupancy is due to operations related to the underlying routing/forwarding scheme, which is only used as a demonstrator vehicle here and it is not necessary for our scheme.



(a) Instantaneous and average throughput (Mb/s) for $(\xi_{\text{C}}, \xi_{\text{E}}) = (2.8, 1.7)$. (b) Instantaneous and average throughput (Mb/s) for $(\xi_{\text{C}}, \xi_{\text{E}}) = (1.7, 2.8)$.



(c) CPU utilization (%) for various traffic rates of the two sessions, A and B. (d) CPU utilization (%) of the main processes, queues maintenance, tx/rx operations and broadcast messaging.

Fig. 6. Results of the experimentation in wireless testbed, in the five nodes topology of Fig. 5.

E. Elimination of Dropping Queues

We use the drop queues to prove mathematically the performance of our MMU scheme. Here, we consider an extension of the MMU policy such that the packets are directly discarded from the packet queues without ever visiting the drop queues.

MMU without Drop Queues (MMU-Q) Policy

Parameter Selection, Receiver-End Congestion Control, Scheduling on Wired Links: same as in MMU.

Packet Dropping: Each transmission queue $Q_l^{(c)}(t)$ removes $\min\{d_l^{(c)}(t), Q_l^{(c)}(t)\}$ packets at the end of slot t, where

$$d_l^{(c)}(t) = \begin{cases} d_{\max} & \text{if } Q_l^{(c)}(t) > V \\ 0 & \text{otherwise.} \end{cases}$$

In Fig. 7(a)-7(b) we compare MMU and MMU-Q using an one-hop path. From the experiments we observe that both policies achieve the same throughput, while the data queues have a similar evolution yielding the same average delay.

VI. Conclusion

We proposed a distributed control scheme that maximizes utility in multirate multicast. The performance is analyzed and shown to be near-optimal. Several enhancements of the policy are described including a priority rule for base layer packets, and a modification for 802.11 wireless devices. The scheme is implemented in a wireless testbed and its applicability is demonstrated. In future work, we plan to derive the optimal policy for general wireless networks and to experiment further in larger topologies, investigating delay and CPU occupancy.

References

[1] G. S. Paschos, C.-P. Li, E. Modiano, K. Choumas, and T. Korakis, "Multirate Multicast: Optimal Algorithms and Implementation," in *IEEE INFOCOM*, 2014.

[2] X. Li, S. Paul, and M. Ammar, "Layered Video Multicast with Retransmissions (LVMR): Evaluation of Hierarchical Rate Control," in *IEEE INFOCOM*, 1998.

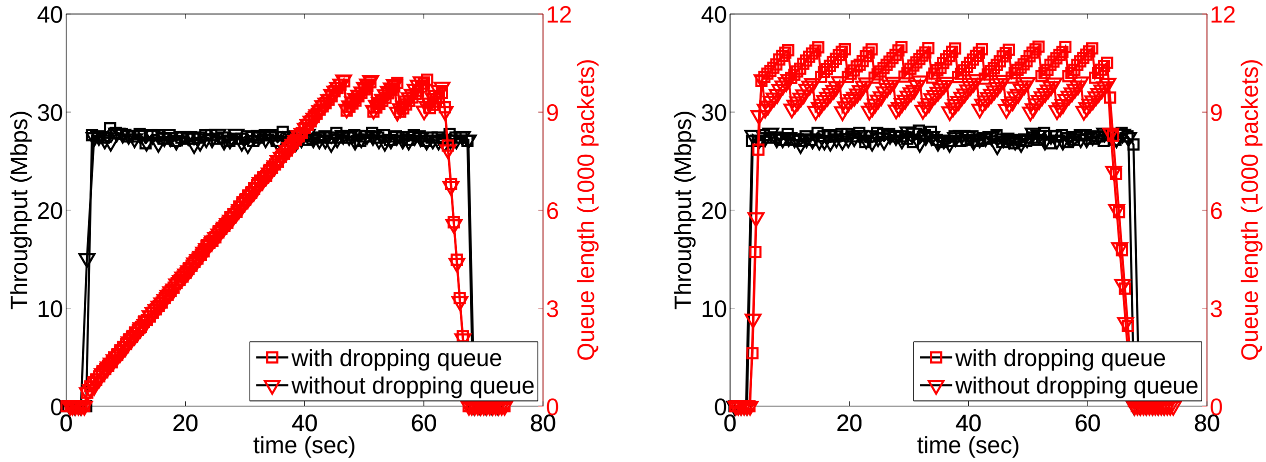
[3] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast," in *ACM SIGCOMM*, 1996.

[4] S. Sarkar and L. Tassiulas, "Fair Allocation of Utilities in Multirate Multicast Networks: A Framework for Unifying Diverse Fairness Objectives," *IEEE Trans. Autom. Control*, vol. 47, no. 6, pp. 931–944, Aug. 2002.

[5] S. Deb and R. Srikant, "Congestion Control for Fair Resource Allocation in Networks with Multicast Flows," *IEEE/ACM Trans. Netw.*, vol. 12, no. 2, pp. 274–285, Apr. 2004.

[6] S. Shakkottai and R. Srikant, "Network Optimization and Control," *Found. Trends Netw.*, vol. 2, no. 3, pp. 271–379, Jan. 2007.

[7] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*, Synthesis Lectures on Communication Networks, Ed. Morgan & Claypool, 2010.



(a) Critical loading

(b) Overloading

Fig. 7. Throughput and Data Queue Length experimentation of MMU and MMU-Q on a single link. Parameters used: $V = 10000$, $d_{max} = 1000$, $slot = 100ms$, the capacity of the link is set to 27Mbps. Two cases are examined: (left) The offered load is 30Mbps or $A^{(c)}(t) \approx 255$ for packet size 1470 bytes (critical loading), and (right) the offered load is 100Mbps or $A^{(c)}(t) \approx 842$ (overloading). The differences between the two policies are hardly visible.

[8] S. Sarkar and L. Tassiulas, "A Framework for Routing and Congestion Control for Multicast Information Flows," *IEEE Trans. Inf. Theory*, vol. 48, no. 10, pp. 2690 – 2708, Oct. 2002.

[9] K. Kar, S. Sarkar, and L. Tassiulas, "A Scalable Low-overhead Rate Control Algorithm for Multirate Multicast Sessions," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 8, pp. 1541–1557, Oct. 2002.

[10] S. Sarkar and L. Tassiulas, "Back Pressure Based Multicast Scheduling for Fair Bandwidth Allocation," *IEEE Trans. Neural Netw.*, vol. 16, no. 5, pp. 1279–1290, Sep. 2005.

[11] C.-P. Li and E. Modiano, "Receiver-Based Flow Control for Networks in Overload," *IEEE/ACM Trans. Netw.*, vol. 23, no. 2, pp. 616–630, April 2015.

[12] L. Bui, R. Srikant, and A. Stolyar, "Optimal Resource Allocation for Multicast Flows in Multihop Wireless Networks," in *IEEE CDC*, 2007.

[13] M. Neely, E. Modiano, and C.-P. Li, "Fairness and Optimal Stochastic Control for Heterogeneous Networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 396–409, 2008.

[14] R. Laufer, T. Salonidis, H. Lundgren, and P. Le Guyadec, "A Cross-Layer Backpressure Architecture for Wireless Multihop Networks," *IEEE/ACM Trans. Netw.*, vol. 22, no. 2, pp. 363–376, April 2014.

[15] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, Apr. 2006.

[16] C. Li and E. Modiano, "Receiver-Based Flow Control for Networks in Overload," in *IEEE INFOCOM*, 2013.

[17] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, Sep. 1999.

[18] "NITLab: Network Implementation Testbed Laboratory," http://nitlab.inf.uth.gr/NITlab/index.php/testbed.

[19] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click Modular Router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[20] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and Evaluation of an Unplanned 802.11b Mesh Network," in *MobiCom*, 2005.

[21] K. Choumas, T. Korakis, I. Koutsopoulos, and L. Tassiulas, "Implementation and End-to-end Throughput Evaluation of an IEEE 802.11 Compliant Version of the Enhanced-Backpressure Algorithm," in *Tridentcom*, 2012.

[22] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[23] "Iperf: The TCP/UDP Bandwidth Measurement Tool," http://sourceforge.net/projects/iperf/.

Appendix A
Derivation of Bound in (15)

From (1)-(2) and [11, Lemma 7], we obtain

$$\frac{1}{2}\left([Q_l^{(c)}(t+1)]^2 - [Q_l^{(c)}(t)]^2\right) \le B_Q - Q_l^{(c)}(t)\Big(\mu_l^{(c)}(t) + d_l^{(c)}(t) - \mu_{p(l)}^{(c)}(t) - A^{(c)}(t)1_{[l\in L_{\text{out}}(c)]}\Big), \tag{31}$$

$$\frac{1}{2}\left([D_l^{(c)}(t+1)]^2 - [D_l^{(c)}(t)]^2\right) \le B_D - D_l^{(c)}(t)\left(\varphi_l^{(c)}(t) - d_l^{(c)}(t)\right), \tag{32}$$

where

$$B_Q \triangleq (\mu_{\max}+d_{\max})^2+(\mu_{\max}+A_{\max})^2,\ B_D \triangleq 2(d_{\max})^2. \tag{33}$$

Multiplying (31) and (32) by $m_l^{(c)}$, summing over l and c, and taking conditional expectation, we have

$$\begin{aligned}
&\Delta(t) + V\sum_{c,l} m_l^{(c)}\, \mathbb{E}_H[\varphi_l^{(c)}(t)] \\
&\le B_1 + \sum_{c,l\in L_{\text{out}}(c)} m_l^{(c)} Q_l^{(c)}(t)\,\lambda^{(c)} \\
&- \sum_{c,l} m_l^{(c)} Q_l^{(c)}(t)\mathbb{E}_H[\mu_l^{(c)}(t) + d_l^{(c)}(t)] \\
&+ \sum_{c,l} m_l^{(c)} Q_l^{(c)}(t)\mathbb{E}_H[\mu_{p(l)}^{(c)}(t)] \\
&- \sum_{c,l} m_l^{(c)}\, D_l^{(c)}(t)\mathbb{E}_H[\varphi_l^{(c)}(t) - d_l^{(c)}(t)] \\
&+ V\sum_{c,l} m_l^{(c)}\, \mathbb{E}_H[\varphi_l^{(c)}(t)],
\end{aligned} \tag{34}$$

where $B_1 = \frac{1}{2}\sum_{c,l} m_l^{(c)}(B_Q+B_D)$ and $V > 0$ is a predefined parameter. Rearranging terms and using the equality, for each session c,

$$\sum_l m_l^{(c)} Q_l^{(c)}(t)\, \mathbb{E}_H[\mu_{p(l)}^{(c)}(t)] = \sum_l \mathbb{E}_H[\mu_l^{(c)}(t)] \sum_{l' : p(l')=l} m_{l'}^{(c)} Q_{l'}^{(c)}(t), \tag{35}$$

we obtain (15). ■

Appendix B
Proof of Theorem 1

We denote by $\varUpsilon(\pi, t)$ the RHS of the inequality (15) evaluated under a policy π in slot t, where $\varUpsilon(\pi, t)$ can be re-written as

$$\begin{aligned}\varUpsilon(\pi, t) &\triangleq B_1 + V \sum_{c,l} m_l^{(c)} \mathbb{E}_H[\varphi_l^{(c)}(t)] \\ &+ \sum_{c,l} m_l^{(c)} Q_l^{(c)}(t) \mathbb{E}_H\Big[\mu_{p(l)}^{(c)}(t) + \lambda^{(c)} 1_{[l \in L_{\text{out}}(c)]} \\ &- \mu_l^{(c)}(t) - d_l^{(c)}(t)\Big] \\ &- \sum_{c,l} m_l^{(c)} D_l^{(c)}(t) \mathbb{E}_H[\varphi_l^{(c)}(t) - d_l^{(c)}(t)]. \end{aligned} \tag{36}$$

It is clear that the decision variables in (36) are chosen according to policy π. Let $DPP(\pi, t)$ be the sum (14) evaluated under policy π in slot t. The inequality (15) under the MMT policy is

$$DPP(\mathsf{MMT}, t) \le \varUpsilon(\mathsf{MMT}, t).$$

Now, consider the optimal stationary policy, denoted by STAT, that chooses $\mathbb{E}\Big[\mu_l^{(c)}(t)\Big] = f_l^{(c)*}$ and $\mathbb{E}\Big[d_l^{(c)}(t)\Big] = \mathbb{E}\Big[\varphi_l^{(c)}(t)\Big] = q_l^{(c)*}$ for all t, where $q_l^{(c)*}$ and $f_l^{(c)*}$ are the optimal flow variables that solves (10). This is a feasible policy. Since the MMT policy minimizes the RHS of (36) in every slot, we have

$$DPP(\mathsf{MMT}, t) \le \varUpsilon(\mathsf{MMT}, t) \le \varUpsilon(\mathsf{STAT}, t). \tag{37}$$

Using the flow conservation (3)-(4), we have

$$\varUpsilon(\mathsf{STAT}, t) = B_1 + V \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}. \tag{38}$$

Plugging (38) into (37) yields, under the MMT policy,

$$\varDelta(t) + V \sum_{c,l} m_l^{(c)}\, \mathbb{E}_H[\varphi_l^{(c)}(t)] \le B_1 + V \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}.$$

Taking expectation, summing over slots $\{0, \dots, t-1\}$, dividing by Vt, and using $\mathbb{E}\,[L(0)] \ge 0$, we have

$$\frac{\mathbb{E}\,[L(t)]}{Vt} + \sum_{c,l} m_l^{(c)}\, \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[\varphi_l^{(c)}(\tau)] \le \frac{B_1}{V} + \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}.$$

Lemma 1 shows that $\mathbb{E}\,[L(t)]$ is finite for all t. Taking $t \to \infty$ yields

$$\sum_{c,l} m_l^{(c)}\, \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[\varphi_l^{(c)}(\tau)] \le \frac{B_1}{V} + \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}.$$

Since all $D_l^{(c)}(t)$ queues are bounded, we have

$$\hat{d}_l^{(c)} \le \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[\varphi_l^{(c)}(\tau)].$$

It follows that

$$\sum_{c,l} m_l^{(c)}\, \hat{d}_l^{(c)} \le \frac{B_1}{V} + \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}.$$

In a multicast session c, along the path $E_u^{(c)}$ from the source c to a receiver u, traffic that arrives at source c in an interval $[0, t]$ must be equal to the sum of total dropped packets and data delivered to receiver u in that interval, plus the total queue backlogs $\sum_{l \in E_u^{(c)}} Q_l^{(c)}(t)$ at time t. In other words,

$$\sum_{\tau=0}^{t-1} A^{(c)}(\tau) = \sum_{\tau=0}^{t-1} \sum_{l \in E_u^{(c)}} \widetilde{d}_l^{(c)}(\tau) + \sum_{\tau=0}^{t-1} \widetilde{\mu}_{l_u}^{(c)}(\tau) + \sum_{l \in E_u^{(c)}} Q_l^{(c)}(t).$$

Taking expectation and time average, and using the finiteness of queues $Q_l^{(c)}(t)$, we have as $t \to \infty$

$$\lambda^{(c)} = \overline{r}_u^{(c)} + \sum_{l \in E_u^{(c)}} \hat{d}_l^{(c)}, \ u \in U^{(c)}. \tag{39}$$

Summing (39) over $c \in C$ and $u \in U^{(c)}$ yields

$$\sum_c |U^{(c)}| \lambda^{(c)} = \sum_{c,u} \overline{r_u^{(c)}} + \sum_{c,l} m_l^{(c)} \hat{d}_l^{(c)} \tag{40}$$

$$\le \sum_{c,u} \overline{r}_u^{(c)} + \frac{B_1}{V} + \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}. \tag{41}$$

From (9), the maximum total throughput is

$$\sum_{c,u} r_u^{(c)*} = \sum_c |U^{(c)}| \lambda^{(c)} - \sum_{c,l} m_l^{(c)}\, q_l^{(c)*}.$$

As a result,

$$\sum_{c,u} \overline{r}_u^{(c)} \ge \sum_{c,u} r_u^{(c)*} - \frac{B_1}{V}.$$

The proof is complete. ■

Appendix C
Derivation of Bound in (24)

Define the indicator function $1_{uc}^R(t) = 1$ if $Z_u^{(c)}(t) \ge Q$, and 0 otherwise. Let $1_{uc}^L(t) = 1 - 1_{uc}^R(t)$. Define $\delta_u^{(c)}(t) = \nu_u^{(c)}(t) - \mu_{l_u}^{(c)}(t)$ and $\delta_{\max} = \max[\nu_{\max}, \mu_{\max}]$. Then we have $|\delta_u^{(c)}(t)| \le \delta_{\max}$ and $\nu_u^{(c)}(t) \le \nu_{\max}$. For each link $l \in E$, we obtain from [11, Lemma 7] that

$$\begin{aligned}&\frac{1}{2}\big([Q_l^{(c)}(t+1)]^2 - [Q_l^{(c)}(t)]^2\big) \le B_Q \\ &- Q_l^{(c)}(t)\Big(\mu_l^{(c)}(t) + d_l^{(c)}(t) - \mu_{p(l)}^{(c)}(t) - A^{(c)}(t) 1_{[l \in L_{\text{out}}(c)]}\Big), \\ &\frac{1}{2}\big([D_l^{(c)}(t+1)]^2 - [D_l^{(c)}(t)]^2\big) \le B_D \\ &- D_l^{(c)}(t)\big(\varphi_l^{(c)}(t) - d_l^{(c)}(t)\big),\end{aligned}$$

where B_Q and B_D are given in (33). Using the same analysis in [11, Lemma 5], we get

$$e^{w(Z_u^{(c)}(t+1)-Q)} - e^{w(Z_u^{(c)}(t)-Q)} \le e^{w(\nu_{\max}+\mu_{\max})}$$

$$+\, w\,(\delta_{\max} + \frac{\epsilon}{2}) - w\, 1^{R}_{uc}(t)\, e^{w(Z^{(c)}_{u}(t)-Q)} \big[\delta^{(c)}_{u}(t) - \frac{\epsilon}{2}\big],$$

$$e^{w(Q-Z^{(c)}_{u}(t+1))} - e^{w(Q-Z^{(c)}_{u}(t))} \le e^{wQ} + w\,(\delta_{\max} + \frac{\epsilon}{2})$$
$$+\, w\, 1^{L}_{uc}(t)\, e^{w(Q-Z^{(c)}_{u}(t))} \big[\delta^{(c)}_{u}(t) + \frac{\epsilon}{2}\big].$$

Combining the above inequalities yields

$$\begin{aligned}\Delta(t) \le B_2 &- \sum_{c,l} m^{(c)}_{l} D^{(c)}_{l}(t)\, \mathbb{E}_H \Big[\varphi^{(c)}_{l}(t) - d^{(c)}_{l}(t)\Big] \\ &- \sum_{c,l} m^{(c)}_{l} Q^{(c)}_{l}(t)\, \mathbb{E}_H \Big[\mu^{(c)}_{l}(t) + d^{(c)}_{l}(t) - \mu^{(c)}_{p(l)}(t) \\ &\quad + A^{(c)}(t) 1_{[l \in L_{\text{out}}(c)]}\Big] \\ &- w \sum_{uc} 1^{R}_{uc}(t)\, e^{w(Z^{(c)}_{u}(t)-Q)} \big[\mathbb{E}_H[\delta^{(c)}_{u}(t)] - \frac{\epsilon}{2}\big] \\ &+ w \sum_{uc} 1^{L}_{uc}(t)\, e^{w(Q-Z^{(c)}_{u}(t))} \big[\mathbb{E}_H[\delta^{(c)}_{u}(t)] + \frac{\epsilon}{2}\big],\end{aligned}$$

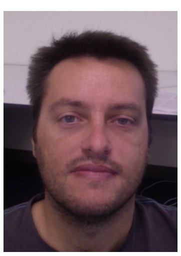
where

$$\begin{aligned}B_2 &= \frac{1}{2} \sum_{c,l} m^{(c)}_{l} (B_Q + B_D) \\ &+ \frac{1}{2} |U||C| \Big[e^{wQ} + e^{w(\nu_{\max}+\mu_{\max})} + w(2\delta_{\max} + \epsilon)\Big]. \qquad (42)\end{aligned}$$

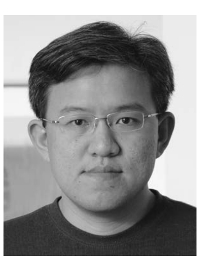
By definition of $Y^{(c)}_{u}(t)$ we have

$$Y^{(c)}_{u}(t) = w\, 1^{R}_{uc}(t)\, e^{w(Z^{(c)}_{u}(t)-Q)} - w\, 1^{L}_{uc}(t)\, e^{w(Q-Z^{(c)}_{u}(t))}$$

Using $\delta^{(c)}_{u}(t) = \nu^{(c)}_{u}(t) - \mu^{(c)}_{l_u}(t)$ and rearranging terms, we obtain the requested. ■



Georgios S. Paschos (S'01-M'06-SM'15) received his dipl. in Electrical and Computer Engineering '02 from Aristotle University of Thessaloniki, and Ph.D. degree '06 from ECE dept. University of Patras, both in Greece. He was an ERCIM postdoc fellow for one year in VTT, Finland. From '08 to '14, he was affiliated with The Center of Research and Technology Hellas–Informatics & Telematics Institute, CERTH-ITI, Greece. He was also teaching as an adjunct lecturer in the Department of Computer and Communication Engineering of the University of Thessaly, for the period '09-'11. From '12 to '14, he was a postdoctoral associate at LIDS, Massachusetts Institute of Technology, USA. Since '14, he is a Principal Researcher in the French Research Center of Huawei Technologies, Paris, and the head of the Network Control and Resource Allocation team. His main interests are in the area of wireless communications, networks and stochastic modeling. He has served as a Technical Program Committee member in INFOCOM, Mobihoc, and WiOPT conferences. Since Sep '15 He serves as an Associate Editor for the IEEE/ACM Trans. on Networking.



Chih-ping Li received his B.S. degree in Electrical Engineering from the National Taiwan University in 2001, and his M.S. and Ph.D. degrees in Electrical Engineering from the University of Southern California in 2005 and 2011, respectively. He was a Postdoctoral Associate in the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology between 2011 and 2014. Since 2014 he joined the Corporate R&D department at Qualcomm and has been working on 5G system design and analysis. His research interests include stochastic control, resource allocation, and QoS in communication networks, wireless networks, and queueing systems.

Eytan Modiano (S'90–M'93–SM'00–F'12) received the B.S. degree in electrical engineering and computer science from the University of Connecticut, Storrs, CT, USA, in 1986, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, MD, USA, in 1989 and 1992, respectively.

He was a Naval Research Laboratory Fellow between 1987 and 1992 and a National Research Council Post-Doctoral Fellow from 1992 to 1993. Between 1993 and 1999, he was with the Massachusetts Institute of Technology (MIT) Lincoln Laboratory, Lexington, MA, USA, where he was a project leader for MIT Lincoln Laboratory's Next Generation Internet (NGI) project. Since 1999, he has been on the faculty with MIT, Cambridge, MA, USA, where he is a Professor with the Department of Aeronautics and Astronautics and the Laboratory for Information and Decision Systems (LIDS). His research is on communication networks and protocols with emphasis on satellite, wireless, and optical networks.

Prof. Modiano is an Associate Fellow of the AIAA. He is an Editor-at-Large for the IEEE/ACM TRANSACTIONS ON NETWORKING and served as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY and IEEE/ACM TRANSACTIONS ON NETWORKING. He was the Technical Program Co-Chair for IEEE WiOpt 2006, IEEE INFOCOMM 2007, and ACM MobiHoc 2007.

Kostas Choumas received the BS and MS degrees from the Electrical and Computer Engineering Department, University of Thessaly, Greece, in 2007 and 2008 respectively and the PhD degree in the same department from University of Thessaly, in 2015, under the supervision of Prof. Leandros Tassiulas and assistant Prof. Thanasis Korakis. From 2009, he is a research scientist working on multiple projects with University of Thessaly and Centre for Research and Technology Hellas. His current research interests are in the field of designing and implementing enhanced routing and access layer protocols for wireless ad hoc networks, as well as in experimenting with use of testbed platforms.

Thanasis Korakis received the BS and MS degrees from the Informatics and Telecommunication department, University of Athens, in 1994 and 1997 respectively and the PhD degree in Computer and Communications Engineering from University of Thessaly, Greece in 2005, under the supervision of Prof. Leandros Tassiulas. In the summer of 2004, he was a visiting researcher in the Computer Science and Engineering Department, University of California, Riverside. From 2005 to 2006, he was a research scientist with the department of Electrical and Computer Engineering, Polytechnic University, NY. From 2006 to 2012 he was a research assistant professor in the same department. Currently, he is an assistant professor in the Electrical and Computer Engineering Department, University of Thessaly. His research interests are in the field of wireless networks with emphasis on access layer protocols, cooperative networks, directional antennas, quality-of-service provisioning and network management. From 2007 to 2012 he was a voting member of the IEEE 802.16 standardization group. He served as a publication chair for WiOpt 2007, TPC chair of WiNTECH 2010, TPC chair of Tridentcom 2011, general chair of Tridentcom 2012, chair of EMUTools 2013 and chair of WINMEE 2016.