Distributed Load Shedding with Minimum Energy

Kostas Choumas*, Georgios S. Paschos†, Thanasis Korakis* and Leandros Tassiulas‡
* Dept. of ECE, University of Thessaly, Volos, Greece
† Huawei Technologies, Paris, France
‡ Dept. of ECE, Yale University, New Haven, USA

***Abstract*—This paper proposes *distributed load shedding* policies for regulating excessive network load. Data packets are inserted into the network to be delivered to intended destinations. The intermediate network nodes may decide to forward or shed some packets depending on temporally available resources. It is possible for some packets to traverse several nodes in the network until they are finally dropped before reaching the destination, which exacerbates energy consumption. We define a multi-objective optimization problem where we aim to minimize the used energy subject to providing maximum sum throughput. For the case of single-path unicast sessions, we show that Energy-efficient Distributed Load Shedding (E-DLS), a simple shedding mechanism combined with pushback routing, solves this load shedding optimization. We implement E-DLS in a testbed and use the experiments to select policy parameter values that strike a good balance between energy and delay performance. We then propose a heuristic extension of E-DLS for multirate multicast routing, and showcase via testbed experiments its optimal performance.**

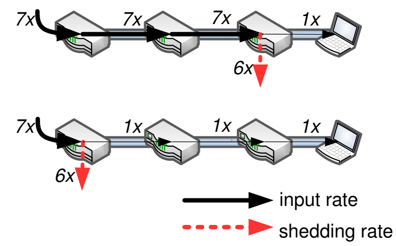


Fig. 1. An example of an overloaded network where distributed load shedding (top) may consume more energy than source-based congestion control (bottom). We propose a novel load shedding mechanism which minimizes energy usage by adjusting real-time shedding decisions.

I. Introduction

Networks employ congestion control in order to regulate the network load, prevent overload, and ultimately improve the Quality of Service delivered to users. In applications such as mobile ad hoc networks and datacenter networks, dynamic congestion controllers have been proposed to solve optimally the *Network Utility Maximization* (NUM) problem [1]. When the network resource allocation is decided by the *backpressure* mechanism [2], a simple queue-based congestion control mechanism can be implemented at every source [3].

More recently, it was shown that congestion control can also be performed without source cooperation, by a combination of routing, packet shedding, and receiver congestion feedback [4]. Data packets enter the network without regulation, while the network reacts by shedding excess load to maximize the utility of the ultimately received traffic. In such a distributed load shedding approach, the congestion control happens *inside* the network, and there are no specified nodes responsible for performing it. This approach is desirable for many applications since it provides network protection from misbehavior or malicious sources, and simplifies the congestion control decisions in a large network. Load shedding has been successfully applied to data stream systems such as very large databases and information collection and computation systems [5]. In the special case of throughput maximization (corresponds to linear utility functions), the receiver-based congestion control of [4] reduces to a simple mechanism where each node sheds packets if its backlog exceeds a certain threshold.

Although the load shedding approach yields the same throughput as the source-based congestion control, it may lead to excessive energy consumption. Some packets may traverse several network nodes before they are eventually dropped, and thus the energy used for their transmissions is wasted. In the example of Fig. 1, under the threshold shedding mechanism of [4] (shown on top), the excess load is shed at the last node. This approach is equivalent in terms of throughput to the source-based approach. However, in terms of consumed energy is clearly worse. This requires 15 transmissions per packet delivered on average, as opposed to the source-based congestion control solution (bottom) which only requires 3 transmissions. Since modern network devices turn on/off the network interface cards to save power [6], the consumed energy is roughly proportional to the time the cards need to be active. In an ad hoc network, nodes often spend most of their energy on communication [7]. Hence for this example, the system consumes roughly five times more energy when shedding the load in a distributed way, which is a heavy price for implementing load shedding in a distributed way. In this paper we propose a novel distributed load shedding scheme which minimizes the energy expenditure subject to achieving maximum sum throughput.

We study a multi-objective load shedding optimization problem, where we aim to minimize the energy usage measured in number of transmissions, subject to maintaining maximum sum throughput. First, we focus on single-path unicast sessions and show that E-DLS, a simple distributed shedding policy combined with pushback routing, approximately solves the above optimization. Similarly to [4], E-DLS performs threshold-based shedding, but in our case the threshold at a node depends on its hop count distance from the destination. We implement E-DLS in a testbed, and demonstrate by ex-

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).

periments how the internal parameters affect the energy usage and the average delay. It adapts to changing conditions, such as arrival rate and channel quality. It can be implemented in a distributed fashion, it is highly scalable since it relieves sources from heavy computations, and it yields deterministic bounds for the queue backlogs facilitating implementation on systems with finite buffers, and providing delay guarantees. Since every network node is capable of shedding excessive load, E-DLS provides a natural shielding against overload.

We extend our study to multirate multicast and propose heuristic extensions for E-DLS in this case. For multirate multicast, source-based congestion control is insufficient, hence our approach is the only solution that combines maximum throughput with minimum energy.

Our contributions can be summarized as follows:

- We formulate a multi-objective load shedding optimization problem whose solution yields minimum number of transmissions subject to maximum sum throughput.
- We prove that a simple online shedding mechanism combined with pushback routing approximates the above offline optimization. Unlike prior work [4], this approach does not use virtual queues and thus it is simpler to implement in any real system.
- We implement our proposed policy in a testbed and study how energy-delay tradeoffs depend on policy parameters.
- We propose heuristic extensions for multirate multicast.

A. Related Work

Congestion control is used in communication networks to regulate network load, avoid congestion phenomena such as unexpected buffer overflow and congestion collapse [8], and improve the utilization of network resources. The most commonly known congestion control mechanism is that of the Transmission Control Protocol (TCP) [9]. This mechanism adapts the rate of each session at its source by changing the number of packets to be injected into the network based on network congestion. Following its prevelence as the standard Internet congestion control mechanism, it was later theoretically understood that the TCP mechanism (under some simplified assumptions) provides fair per-session delays [10]. More generally, the congestion control can be formulated as an optimization problem, called NUM, where a concave function of packet transmission rates is maximized using regulators at the sources [11]. By choice of an appropriate concave function we may achieve different targets such as to maximize sum throughput or achieve a fairness condition such as max-min fairness [12], or proportional fairness [13].

More recently congestion control was studied jointly with dynamic routing and scheduling under stochastic arrivals [3]. It was shown that the long-term average rate optimization (called stochastic NUM) can be solved by deciding locally in time how many packets to inject at each source. The optimal network control policy combines backpressure routing and scheduling with a source-based congestion controller that decides on how many packets to inject by comparing the utility function of the particular source with the observed source queue backlog, an indication of network congestion. This is a highly desirable congestion control scheme since it can solve the NUM problem in an online, distributed, and adaptive way.

The previous works all refer to source-based congestion control, where the sources decide how many packets should be injected into each source in the network. There are several issues that source-based congestion control might face. The sources may misbehave in which case the network becomes susceptible to overloading. The same situation may arise in networks where regulating the sources is difficult, for example consider the flash crowd scenario. Then malicious attacks such as distributed denial of service attacks may occur. For these reasons [14] proposes congestion control at the receiver side. In this work all network nodes can decide to shed packets when needed, while utility maximization is enforced by signals sent by the receivers in the form of values of virtual queues. This was extended in [15] where a receiver-based congestion control mechanism is proposed for multicast sessions. The proposed scheme solves the multi-receiver NUM. A parallel line of research involves the *Distributed Load Shedding* (DLS) techniques [16]. Load shedding has been proposed in the database community as a means of rejecting excess load that has been already introduced into the system. An incredible achievement of receiver-based congestion control and DLS is that the network load is regulated in a distributed manner and yet the optimal behavior can be obtained.

A common feature of receiver-based congestion control and DLS is the fact that a packet or job may travel through the system and hence consume resources only to be dropped at latter stage. This phenomenon may exarcebate energy consumption, which is not taken into account in the formulation of congestion control optimization. As shown in the example of Figure 1, this can result in great amounts of wasted energy in the form of transmissions which are not useful. In this paper we extend the above works by considering a multi-objective optimization where we minimize the energy required to achieve maximum network utility. In our work, not only the NUM is solved without source cooperation, but additionally we make sure that the majority of packets/jobs are dropped in an economical way. Equivalently, our algorithms make sure that the wasted transmissions are minimized.

II. Network Model

We consider a network with prespecified single-path routing and multiple unicast sessions, which are defined as source-destination pairs. This network model demonstrates adequately the important notions involved in our problem, while in Section V we extend our model for multicast routing. Our network consists of nodes in the set $\mathcal{N}$ and directional links in the set $\mathcal{L}$, where each link $(n,m) \in \mathcal{L}$ represents a communication channel between nodes n,m. In the analysis we consider a wireline network in which all links in $\mathcal{L}$ can be used simultaneously. Packets belong to a set of sessions $\mathcal{C}$, where session $c \in \mathcal{C}$ packets are generated at the source s_c and follow a specified path $P^{(c)}$ to the destination δ_c. We assume slotted time; in slot t, $A^{(c)}(t)$ session c packets are generated at s_c, where $A^{(c)}(t)$ are random, bounded above by $A_{\max}$, and i.i.d. over time with mean $\lambda^{(c)}$.

With the exception of destinations, each node n maintains a queue for every session c, whose backlog at the beginning of slot t is denoted by $Q_n^{(c)}(t)$. Let $(\cdot)^+ \triangleq \max(\cdot, 0)$, we may express the evolution of $Q_n^{(c)}(t)$ as

$$Q_n^{(c)}(t+1) \leq \left[\left(Q_n^{(c)}(t) - \mu_n^{(c)}(t)\right)^+ - d_n^{(c)}(t)\right]^+ + \mu_{p(n)}^{(c)}(t), \quad \forall n \in P^{(c)} \cap \mathcal{N} \setminus \{\delta_c\}, c \in \mathcal{C}, \quad (1)$$

where $\mu_n^{(c)}(t)$ is the number of packets to be routed from n to the next node on path $P^{(c)}$[1], $d_n^{(c)}(t)$ is the number of packets to be dropped, and $p(n)$ is the incoming neighbor node (parent) of n. Hence, $\mu_{p(n)}^{(c)}(t)$ is an upper bound to the number of session c packets arriving at n. In the special case where $n = s_c$, we have by convention that $\mu_{p(n)}^{(c)}(t) = A^{(c)}(t)$. Moreover, for the simplicity of our notation, we assume that $Q_{\delta_c}(t) = 0$.

In slot t, our queueing mechanism at node n works as follows. Newly arrived session c packets are added to backlog $Q_n^{(c)}(t)$. A distributed control policy chooses the values of the decision variables $\mu_n^{(c)}(t)$ and $d_n^{(c)}(t)$ for all sessions. In particular, each node n determines the number $\mu_n^{(c)}(t)$ of class c packets transmitted on its outgoing link on path $P^{(c)}$, as well as the number $d_n^{(c)}(t)$ of packets that are dropped from queue $Q_n^{(c)}(t)$ at slot t. However, the real numbers of the transmitted and dropped packets are $\tilde{\mu}_n^{(c)}(t)$ and $\tilde{d}_n^{(c)}(t)$ respectively, since they depend on the backlog availability and they are less or equal to the decision variables $\mu_n^{(c)}(t)$ and $d_n^{(c)}(t)$. For a decision to be admissible, the capacity constraint over link (n, m) must be satisfied

$$\sum_{c \in \mathcal{C}:(n,m) \in P^{(c)}} \mu_n^{(c)}(t) \leq R_{nm}, \quad \forall (n, m) \in \mathcal{L},$$

where R_{nm} is the capacity of link (n, m), and the summation is over all sessions that may transmit over link (n, m). For technical reasons, we limit the maximum number of permissible drops to $d_n^{(c)}(t) \leq d_{\max}$, where $d_{\max}$ is a system-defined parameter. Throughout the paper, we assume $d_{\max} \geq A_{\max} + \max_{(n,m)} R_{nm}$. Since $A_{\max} + \max_{(n,m)} R_{nm}$ is an upper bound to the incoming data rate to a node, our choice of $d_{\max}$ ensures that the packet shedding rate is large enough to allow queue stabilization.

The purpose of this paper is to propose distributed policies for determining the variables $\mu_n^{(c)}(t)$ and $d_n^{(c)}(t)$ at every slot, so that the long-term average performance of the system, as measured by throughput and energy metrics, is optimized.

III. Minimizing Energy Consumption under Maximized sum Throughput

In order to introduce the concepts, we begin by considering the problem of minimizing the total energy consumption under maximum sum throughput of all receivers in multiple sessions.

[1]Path $P^{(c)}$ is a sequence of nodes and links, hence we write $n \in P^{(c)}$ and $(n, m) \in P^{(c)}$ to mean that node n and link (n, m) are in this path.

A. Flow-Level Analysis

To design an optimal online policy, it is useful to first characterize the flow-level system performance. We define $q_n^{(c)}$ to be the rate of dropped packets of session c at node n, and $f_n^{(c)}$ the rate of forwarded packets to the outgoing link. The rate of received session c packets by destination δ_c is called *throughput*, and it is denoted by $r^{(c)}$. Since packets follow a single path, we have

$$r^{(c)} = f_{p(\delta_c)}^{(c)}, \quad \forall c \in \mathcal{C}. \quad (2)$$

For an arrival rate vector $\boldsymbol{\lambda} = (\lambda^{(c)})$, we say that a throughput vector is feasible if and only if there exist non-negative variables $f_n^{(c)}$, $q_n^{(c)}$ such that conditions (3)-(4) are satisfied:

$$f_{p(n)}^{(c)} = q_n^{(c)} + f_n^{(c)}, \quad \forall n \in P^{(c)} \cap \mathcal{N} \setminus \{\delta_c\}, c \in \mathcal{C}, \quad (3)$$

$$\sum_{c:(n,m) \in P^{(c)}} f_n^{(c)} \leq R_{nm}, \quad \forall (n, m) \in \mathcal{L}, \quad (4)$$

where by convention $f_{p(s_c)} = \lambda^{(c)}$. Equations (3) capture flow conservation and inequalities (4) capture capacity constraints. Let $\Lambda_{\boldsymbol{\lambda}}$ be the set of all feasible throughput vectors $\boldsymbol{r} = (r^{(c)})$. We have $\Lambda_{\boldsymbol{\lambda}} = \{\boldsymbol{r} \mid (2) \text{ - } (4)\}$. We define T^* to be the maximum sum throughput of all sessions, which is given by the following problem.

<u>*Sum Throughput Maximization:*</u>

$$T^* \triangleq \max_{\boldsymbol{r} \in \Lambda_{\boldsymbol{\lambda}}} \sum_{c \in \mathcal{C}} r^{(c)}. \quad (5)$$

Summing up (2) over all sessions and using (3) we see that the sum throughput is equal to the total arrived traffic less the sum of shedding rates

$$\sum_c r^{(c)} = \sum_c \lambda^{(c)} - \sum_{c,n} q_n^{(c)}, \quad (6)$$

where the index in the summations hereinafter is a shorthand form of $c \in C$ and $n \in P^{(c)} \cap \mathcal{N} \setminus \{\delta_c\}$. Therefore, for any fixed $\boldsymbol{\lambda}$, problem (5) is equivalent to

<u>*Sum Shedding Minimization:*</u>

$$Q^* \triangleq \min_{\boldsymbol{f}, \boldsymbol{q} \geq \boldsymbol{0}} \sum_{c,n} q_n^{(c)}, \text{ subject to (3) - (4),}$$

where $\boldsymbol{f} = (f_n^{(c)})$ and $\boldsymbol{q} = (q_n^{(c)})$ are the flow and the shedding rates respectively.

B. Problem Statement

As illustrated in Fig. 1 using flows, problem (5) admits multiple optimal solutions, where each solution has potentially different performance with respect to the required number of transmissions and hence energy used. Motivated by this, we focus on *minimizing the average number of transmissions required to achieve maximum sum throughput*. Since the most energy-expensive process for a device is often the transmission/reception process, we effectively minimize the average consumed energy.

Minimizing the average number of transmissions can be achieved by minimizing the total flow across all links $\sum_{c,n} f_n^{(c)}$. Therefore our multi-objective can be expressed in flow-level by the following optimization.

Energy Minimization s.t. Maximum Sum Throughput:

$$E^* = \min_{\boldsymbol{f},\boldsymbol{q}\geq\mathbf{0}} \sum_{c,n} f_n^{(c)}, \text{ s.t. } \sum_{c} r^{(c)} = T^* \text{ and (2) - (4)}, \quad (7)$$

where E^* is the minimum energy measured in terms of rate of transmissions and T^* is the maximum value of (5).

Next, we exploit the assumption of single-path routing to simplify the problem and pose it in terms of shedding rates. Denote with $\mathcal{U}^{(c)}(n) \subset P^{(c)}$ the set of upstream nodes of n on path $P^{(c)}$, including n. Then, using the flow conservation constraints (3) we have

$$\sum_{c,n} f_n^{(c)} = \sum_{c,n}\left(\lambda^{(c)} - \sum_{k\in\mathcal{U}(n)} q_k^{(c)}\right). \quad (8)$$

Let $h_n^{(c)}$ be the hopcount from node n to the destination δ_c, e.g. $h_{p(\delta_c)}^{(c)} = 1$. By enumerating links in two different ways we have the following identity

$$\sum_n \sum_{k\in\mathcal{U}(n)} q_k^{(c)} = \sum_n h_n^{(c)} q_n^{(c)}, \quad \forall c \in \mathcal{C},$$

which combined with (8) leads to

$$\sum_{c,n} f_n^{(c)} = \sum_c N^{(c)}\lambda^{(c)} - \sum_{c,n} h_n^{(c)} q_n^{(c)}, \quad (9)$$

where $N^{(c)} \triangleq |P^{(c)} \cap \mathcal{N}| - 1$ is the number of nodes on the path $P^{(c)}$ except for the destination. Since the arrival term above is constant, using (6), we conclude that in the case of multiple unicasts with single path routing, (7) is equivalent to

Maximum Weighted Shedding s.t. Minimum Sum Shedding:

$$\max_{\boldsymbol{f},\boldsymbol{q}\geq\mathbf{0}} \sum_{c,n} h_n^{(c)} q_n^{(c)}, \text{ s.t. } \sum_{c,n} q_n^{(c)} = Q^* \text{ and (3) - (4)}. \quad (10)$$

We further relax the minimum sum shedding constraint of (10) to obtain

$$\max_{\boldsymbol{f},\boldsymbol{q}\geq\mathbf{0}} \left(\sum_{c,n} h_n^{(c)} q_n^{(c)} - K\left(\sum_{c,n} q_n^{(c)} - Q^*\right)\right), \text{s.t. (3) - (4)}, \quad (11)$$

where $K > \max_{(c,n)} h_n^{(c)}$ is a positive parameter. Consider the positive weights $\alpha_n^{(c)}(K) \triangleq K - h_n^{(c)}$ defined for nodes on paths $P^{(c)}$. We obtain the equivalent problem

Weighted Shedding Minimization:

$$\min_{\boldsymbol{f},\boldsymbol{q}\geq\mathbf{0}} \sum_{c,n} \alpha_n^{(c)}(K) q_n^{(c)}, \text{ s.t. (3) - (4)}. \quad (12)$$

By letting $K \to \infty$, any solution of (12) converges to some solution of (10) (see the *penalty method* in [17, §4.2]). This also provides an optimal solution for our desired problem (7), i.e., it minimizes the energy used to achieve maximum sum throughput. Summarizing our flow-level analysis, we conclude that the "energy minimization subject to maximum sum throughput" problem of interest can be turned into a weighted shedding minimization problem for the case of single path unicast. In the following section we use the queue shedding analysis to design an online policy.

C. Packet-Level Control

We develop an online control policy that chooses instantaneous routing $\mu_n^{(c)}(t)$ and shedding $d_n^{(c)}(t)$ to yield a good time average performance. Below we make precise what we mean by "good time average performance". We study the time averages of $\tilde{\mu}_n^{(c)}(t),\ \forall n \in P^{(c)} \setminus \delta_c,\ c \in \mathcal{C}$, and we assume that under our policy they converge. We define the average sum throughput of our scheme as

$$\hat{T} \triangleq \lim_{\tau\to\infty} \frac{1}{\tau} \sum_{t=0}^{\tau-1} \sum_c \mathbb{E}[\tilde{\mu}_{p(\delta_c)}^{(c)}(t)]. \quad (13)$$

In (13) we take the time average of the expected received packets at all receivers $\sum_c \mathbb{E}[\tilde{\mu}_{p(\delta_c)}^{(c)}(t)]$ over a time period of τ slots and then let τ grow large. In a similar fashion, the average number of total transmitted packets in the network is

$$\hat{E} \triangleq \lim_{\tau\to\infty} \frac{1}{\tau} \sum_{t=0}^{\tau-1} \sum_{c,n} \mathbb{E}[\tilde{\mu}_n^{(c)}(t)]. \quad (14)$$

We wish to develop a control policy that drives these limits to be equal to the flow-level variables T^*, E^* from (7).

We will use the drift-plus-penalty method from the theory of Stochastic Optimization [18]. This is a technique used for stabilizing a queueing network while also minimizing the time average penalty function. In every slot t, the queue backlogs are observed and control actions are taken to greedily minimize a bound on the drift-plus-penalty expression $\Delta(t) + V\pi(t)$. The drift $\Delta(t) \triangleq \mathbb{E}[L(t+1) - L(t)|\mathbf{Q}(t)]$ is determined using a quadratic Lyapunov function $L(t) = \frac{1}{2}\sum_{c,n}[Q_n^{(c)}(t)]^2$, which is a scalar measure of the queue backlog vector $\mathbf{Q}(t) = (Q_n^{(c)})$. Because of the squares, minimizing the drift requires reducing the large backlogs. For the penalty function, driven from (12) we choose

$$\pi(t) = \sum_{c,n} \alpha_n^{(c)}(K)\mathbb{E}[d_n^{(c)}(t)|\mathbf{Q}(t)].$$

Observe that the two goals are conflicting. We can choose a large $d_n^{(c)}(t)$ to shed many packets and reduce the backlog $Q_n^{(c)}$ hence reducing $\Delta(t)$, but this increases the penalty $\pi(t)$, and vice versa. Last, the positive parameter V controls the preference between reducing backlogs or reducing shedding penalties. We investigate the role of V in detail in Section IV.

Performing a standard derivation [18], [19], it can be shown that the drift-plus-penalty expression is bounded by the following quantity (B is a positive constant related to network capacities and characteristics of the arrival process)

$$\begin{aligned}\Delta(t) + V\pi(t) \leq B &+ \sum_c Q_{s_c}^{(c)}(t)\lambda^{(c)} \\ &\underbrace{- \sum_{c,n}\left(Q_n^{(c)}(t) - \alpha_n^{(c)}(K)V\right)\mathbb{E}[d_n^{(c)}(t)|\mathbf{Q}(t)]}_{\text{shedding}} \\ &\underbrace{- \sum_{c,(n,m)\in P^{(c)}}\left(Q_n^{(c)}(t) - Q_m^{(c)}(t)\right)\mathbb{E}[\mu_n^{(c)}(t)|\mathbf{Q}(t)]}_{\text{routing}}, \quad (15)\end{aligned}$$

Observe that the Right-Hand Side (RHS) of (15) has two controllable terms noted with under brackets, one term controlled by the shedding variable $d_n^{(c)}(t)$ and one term controlled by the routing variable $\mu_n^{(c)}(t)$. We propose E-DLS, an online policy which observes the current queue state $\mathbf{Q}(t)$ and minimizes the RHS of (15) at every slot. The policy is presented below.

E-DLS **Energy-efficient Distributed Load Shedding**

Pushback Routing: Let $\mathcal{C}_{nm} \subseteq \mathcal{C}$ be the set of sessions that compete for service on each link $(n, m) \in \mathcal{L}$. Choose $c^*_{nm} \in \arg\max_{c \in \mathcal{C}_{nm}} (Q^{(c)}_n(t) - Q^{(c)}_m(t))$ (ties broken arbitrarily). Over link (n, m) route

$$\mu^{(c^*_{nm})}_n(t) = \begin{cases} R_{nm} & \text{if } Q^{(c^*_{nm})}_n(t) - Q^{(c^*_{nm})}_m(t) > 0, \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

and $\mu^{(c)}_n(t) = 0$ for all the other sessions $c \in \mathcal{C}_{nm} \setminus \{c^*_{nm}\}$, where R_{nm} is the capacity of the link.

Packet Shedding: For each session $c \in \mathcal{C}$ and node $n \in P^{(c)} \cap \mathcal{N} \setminus \{\delta_c\}$ choose

$$d^{(c)}_n(t) = \begin{cases} d_{\max} & \text{if } Q^{(c)}_n(t) > \alpha^{(c)}_n(K)V, \\ 0 & \text{otherwise.} \end{cases} \quad (17)$$

Above, $\alpha^{(c)}_n(K) = K - h^{(c)}_n$, where $h^{(c)}_n$ is the hopcount from n to the destination δ_c.

Parameter Selection: $d_{\max}, V, K$ are given in Table I.

TABLE I
PARAMETER SELECTION FOR E-DLS

Param.	Explanation	Suggested values
$\mu^{(c)}_n(t)$	routing control variable	$[0, R_{nm}]$
$d^{(c)}_n(t)$	shedding control variable	$[0, d_{\max}]$
R_{nm}	capacity of link (n, m)	problem defined
$h^{(c)}_n$	hop count from destination along $P^{(c)}$	problem defined
$A_{\max}$	maximum arrivals in one slot	problem defined
$d_{\max}$	shedding batch size	$\geq A_{\max} + \max_{(n,m)} R_{nm}$
K	weight of throughput maximization	$\geq \max_{(n,c)} h^{(c)}_n + 1$
V	energy/delay tradeoff	> 0

E-DLS operates in a distributed manner, using only locally available information. For the computation of (16) we require knowledge of the neighbor backlogs. It is straightforward to collect this information using messaging that incurs minor throughput overhead [15]. Backlog information obtained in this way is often delayed, however prior findings show that this does not hurt throughput optimality [19, §4.7]. The computation of (17) is done locally at each node.

To save energy, E-DLS drops all unnecessary packets near the sources, hence with respect to vector $\boldsymbol{q}$ the obtained solution resembles the one from source-based congestion control. However, E-DLS additionally protects the network from denial of service attacks, or situations where a source is malfunctioning. By performing congestion control everywhere, the network is inherently shielded from overload. Moreover, as shown in Section V, the load shedding solution for multicast sessions differs fundamentally from the source-based congestion control.

D. Performance Evaluation of E-DLS

The performance of E-DLS is characterized by the following theorem.

THEOREM 1 [PERFORMANCE OF E-DLS]: *Consider a sequence of systems operating under* E-DLS *parameterized by* K, V. *Suppose that each system starts empty. For each system, denote the sum throughput achieved by* E-DLS *with* $\hat{T}(K)$ *(defined in* (13)*) and the consumed energy with* $\hat{E}(K, V)$ *(defined in* (14)*).*

(1) ***Bounded backlogs.*** *For any* K, V *we have*

$$Q^{(c)}_n(t) \leq KV + d_{\max}, \quad \text{for all } c, n, t. \quad (18)$$

(2) ***Near optimal throughput.*** *The sum throughput achieved satisfies*

$$\lim_{K \to \infty} \hat{T}(K) = T^*, \quad (19)$$

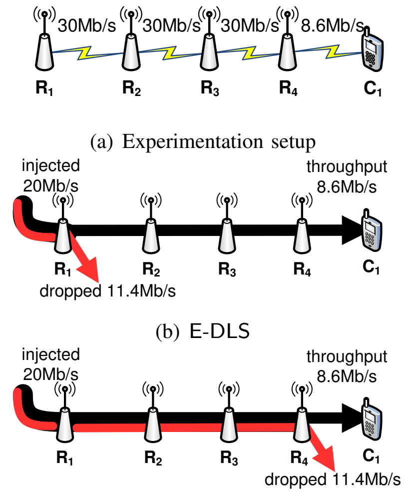
where T^* *is the maximum sum throughput from* (5).

(3) ***Near optimal number of transmissions.*** *The total energy consumed (measured in number of transmissions) satisfies*

$$\lim_{K \to \infty} \lim_{V \to \infty} \hat{E}(K, V) = E^*,$$

where E^* *is the minimum value of* (7).

Proof of Theorem 1. The proof is in Appendix A. □



(a) Experimentation setup

(b) E-DLS

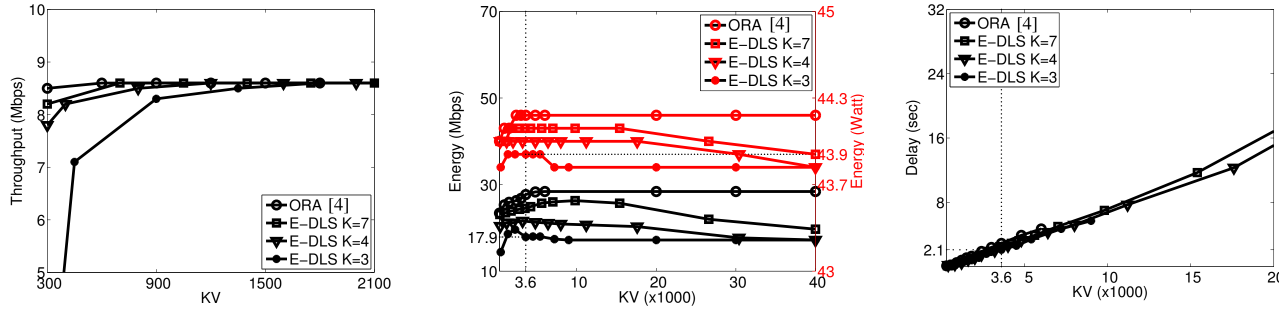
(c) ORA [4]

Fig. 2. Single-path experimentation setup (a) and experimental results (b)-(c).

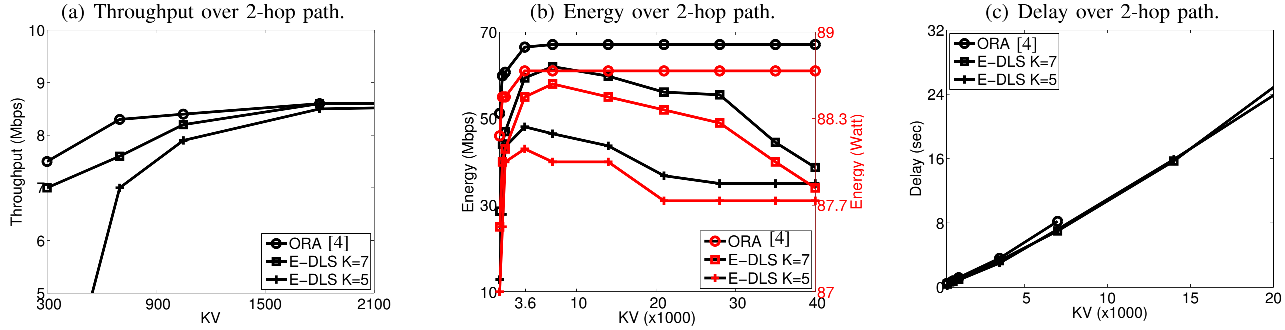
Theorem 1 shows that E-DLS approximately solves (7), i.e., achieves maximum sum throughput using the minimum required number of transmissions. As we see in the theorem, to accurately approximate the desired solution both K and V need to be chosen arbitrarily large. In practice we will choose finite values for K and V so that the solution is approximated in a satisfactory level but KV is kept as small as possible, which provides delay guarantees. We study this tradeoff in the next section.

IV. EXPERIMENTAL EVALUATION OF E-DLS

In this section we focus our study on a simple five-node network shown in Fig. 2(a). We conduct two sets of experiments, where in the first we inject 20Mb/s at node R_1 (4-hop topology shown) and in the second we inject the same load at node R_3 (2-hop topology, not shown). The data are always destined to client C_1. The experiments are run in NITOS [20], where network nodes implement our proposed policy E-DLS. Details about our implementation are given in [21]. We note that all wireless links operate in non interfered frequency channels, since each node is equipped with two



(a) Throughput over 2-hop path. (b) Energy over 2-hop path. (c) Delay over 2-hop path.



(d) Throughput over 4-hop path. (e) Energy over 4-hop path. (f) Delay over 4-hop path.

Fig. 3. Single-path experimentation results.

wireless interfaces, one used for the preceding and one for the following link on the path. As follows, the network model is the same as this of a wireline network, since the links are wireless but non-interfered and almost static (no mobility). To showcase our improvements, we compare our approach to an implemented version of the ORA policy from [4], which performs distributed load shedding for maximum sum throughput without any energy considerations. Fig. 2(b)-2(c) show the resulting queue shedding rates from the 4-hop set of experiments and clearly demonstrate the benefits of our proposed scheme. By shedding 11.4Mb/s early on, our scheme reduces the total transmissions from 68.6Mb/s to 34.4Mb/s, reducing the energy spent in transmissions by half.

In the remaining section we study tradeoffs between energy, throughput and delay, that naturally arise in stochastic optimization [22]. Fig. 3 shows the measured energy, throughput, and delay performance of E-DLS compared to ORA, for both 2-hop (top figures) and 4-hop (bottom figures) cases, where we vary K, KV. For ORA we vary V and plot the corresponding point assuming $K = 1$. From Figures 3(a)-3(d) we observe that the throughput of E-DLS improves when K, or KV increase, as expected. The maximum throughput is 8.6Mb/s, due to the bottleneck of the last hop in Figure 2, and it is achieved in all cases when $KV > 1500$. From Figures 3(b)-3(e) we derive a similar conclusion for energy performance: larger values for K and KV reduce energy. The minimum energy for the 2-hop topology measured in rate of transmissions and power consumption is 17.2Mb/s and 43.8W respectively, while for the 4-hop topology is 34.4Mb/s and 87.7W. ORA has a gap from the optimal energy, while E-DLS achieves it as K, KV increase. For small values of KV, E-DLS consumes less energy because most of the packets are dropped early and throughput is small. Increasing KV, first we achieve maximum throughput, and then energy consumption gradually reduces to minimum. Last from Figures 3(c)-3(f), we observe that the average delay increases linearly to KV.

Combining the observations from Fig. 3, the best choice is to keep K as small as possible and increase V. In particular, we choose $K = \max_{(n,c)} h_n^{(c)} + 1$, which is the smallest integer that satisfies the condition given in the explanation under (11). Then V must be chosen accordingly to yield a good tradeoff between energy and average delay. For example, in the 2-hop topology, a good choice is $K = 3$ and $V = 1200$, which yields energy 17.9Mb/s and average delay 2.1sec, as it is illustrated in Figures 3(b) and 3(c) for $KV = 3600$. Moreover, the total power consumption in this case is reduced to 43.9W, that is 300mW less than the corresponding power consumption of ORA. As future work, we plan to research methods for reducing the backlog sizes, thus improving the experienced delay. Two promising methods are (i) the initialization of backlogs with low-priority virtual packets, and (ii) the use of learning methods from [23].

V. Multirate Multicast

In single-path unicast, the solution of our distributed load shedding optimization (7) with respect to the shedding vector $\boldsymbol{q}$ resembles the solution of source-based congestion control, although E-DLS solves (7) using a distributed scheme that operates in the entire network. In this section we consider multicast sessions delivered to receivers over multicast trees. Our objective remains to minimize energy subject to delivering maximum sum throughput to all receivers in the multicast group. We allow receivers to obtain the data stream at different rates, i.e., we consider multirate multicast. In multirate multicast, congestion control at the source is not efficient, [15], [24]. To maximize sum throughput, packets should be dropped at multiple nodes. In particular, each node should shed that many packets such that the remaining packet transmission rate is equal to the maximum reception rate by all receivers served through this node.

Multicast session c has a set of receivers $\Delta^{(c)}$ and utilizes a multicast tree with set of links $\mathcal{L}^{(c)}$. To perform multicast operations, a node with multiple outgoing links has multiple queues. For every link $l \in \mathcal{L}^{(c)}$, there is a queue backlog $Q_l^{(c)}(t)$ at the incident node holding packets for transmission over that link. Upon arriving to a node, packets are replicated to all queues $Q_l^{(c)}(t)$, where l are outgoing links from that node. The shedding rates at each queue may be different, denoted with $q_l^{(c)}$. Similarly, the data flow rate to each link is denoted with $f_l^{(c)}$. Following the same notation as before, the rate of received session c packets by all destinations $\Delta^{(c)}$ is expressed as $r^{(c)}$. The sum throughput received at all receivers of multicast sessions can be expressed in the same fashion with (6). The throughput of each session c receiver is the arrived traffic $\lambda^{(c)}$ less the shedding rates of the links connecting the source to this receiver.

As follows, the sum throughput is the total injected rate times multicast group size less a weighted sum of the shedding rates, where the weight of each shedding rate is the number of the receivers that are affected by this. In particular,

$$\sum_c r^{(c)} = \underbrace{\sum_c \left|\Delta^{(c)}\right| \lambda^{(c)}}_{\substack{\text{total throughput}\\ \text{without shedding}}} - \underbrace{\sum_{c,l} m_l^{(c)} q_l^{(c)}}_{\substack{\text{total shedding rate}\\ \text{times affected receivers}}}, \qquad (20)$$

where the index in the summations hereinafter is a shorthand form of $c \in \mathcal{C}$ and $l \in \mathcal{L}^{(c)}$, and $m_l^{(c)}$ is the number of receivers connected to source s_c via link l. When packets are dropped on this link, the throughput of all such receivers is reduced, hence the multiplier in (20). Moreover, the total number of transmissions can be expressed in the same fashion with (9) as

$$\sum_{c,l} f_l^{(c)} = \underbrace{\sum_c \left|\mathcal{L}^{(c)}\right| \lambda^{(c)}}_{\substack{\text{total trans/sion rate}\\ \text{without shedding}}} - \underbrace{\sum_{c,l} H_l^{(c)} q_l^{(c)}}_{\substack{\text{total shedding rate}\\ \text{times affected links}}}, \qquad (21)$$

where $H_l^{(c)}$ is the number of links in the subtree connected to the source via l (including l). When packets are dropped on l, the packet transmission rate of all the links in the remaining subtree is also reduced. Following the same approach of Section III-B, by replacing the total packet transmission rate and throughput of (7) with the RHS of (20) and (21), we may express our objective:

Weighted Shedding Minimization for Multicast:

$$\min_{\boldsymbol{f},\boldsymbol{q}\geq \mathbf{0}} \sum_{c,l} \beta_l^{(c)}(K) q_l^{(c)} \qquad (22)$$

$$\text{s.t. } f_{p(l)}^{(c)} = q_l^{(c)} + f_l^{(c)}, \quad \forall l \in \mathcal{L}^{(c)}, c \in \mathcal{C}, \qquad (23)$$

$$\sum_{c:l\in\mathcal{L}^{(c)}} f_l^{(c)} \leq R_l, \qquad \forall l \in \mathcal{L}^{(c)}, \qquad (24)$$

where $p(l)$ is the preceding link of l in the multicast tree of session c, and by convention we set $f_{p(l)}^{(c)} = \lambda^{(c)}$ for l outgoing from source. Also R_l is the capacity of link l and we define $\beta_l^{(c)}(K) \triangleq m_l^{(c)} K - H_l^{(c)}$ for all c, l. Above, equations (23) are flow conservation constraints per link, and inequalities (24) are capacity constraints.

The online control policy of *multicast* E-DLS is derived from the same analysis of Section III-C. The penalty function is now driven from (22), so we choose

$$\pi(t) = \sum_{c,l} \beta_l^{(c)}(K)\mathbb{E}[d_l^{(c)}(t)|\mathbf{Q}(t)],$$

where $d_l^{(c)}(t)$ is the number of packets to be dropped from $Q_l^{(c)}(t)$. Denoting the packet transmission rate over link l as $\mu_l^{(c)}(t)$ and assuming that $\mu_{p(l)}^{(c)}(t)$ are the source generated packets when l is an outgoing link of s_c, then each queue evolution is expressed as

$$Q_l^{(c)}(t+1) \leq \left[\left(Q_l^{(c)}(t) - \mu_l^{(c)}(t)\right)^+ - d_l^{(c)}(t)\right]^+ + \mu_{p(l)}^{(c)}(t),$$
$$\forall l \in \mathcal{L}^{(c)}, c \in \mathcal{C}$$

and the drift-plus-penalty expressions is bounded by

$$\Delta(t) + V\pi(t) \leq B_2 + \sum_{c,l=(s_c,m)\in\mathcal{L}^{(c)}} Q_l^{(c)}(t)\lambda^{(c)}$$
$$- \underbrace{\sum_{c,l} \left(Q_l^{(c)}(t) - \beta_l^{(c)}(K)V\right) \mathbb{E}[d_l^{(c)}(t)|\mathbf{Q}(t)]}_{\text{shedding}}$$
$$- \underbrace{\sum_{c,l} \left(Q_l^{(c)}(t) - \sum_{l':p(l')=l} Q_{l'}^{(c)}(t)\right) \mathbb{E}[\mu_l^{(c)}(t)|\mathbf{Q}(t)]}_{\text{routing}}.$$

The following policy minimizes the above RHS at every slot.

---

multicast E-DLS

Pushback routing: Similarly to E-DLS, choose $c_l^* \in \arg\max_{c\in\mathcal{C}_l}(Q_l^{(c)}(t) - \sum_{l':p(l')=l} Q_{l'}^{(c)}(t))$ (ties broken arbitrarily) and route

$$\mu_l^{(c_l^*)}(t) = \begin{cases} R_l & \text{if } Q_l^{(c)}(t) - \sum_{l':p(l')=l} Q_{l'}^{(c)}(t) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

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

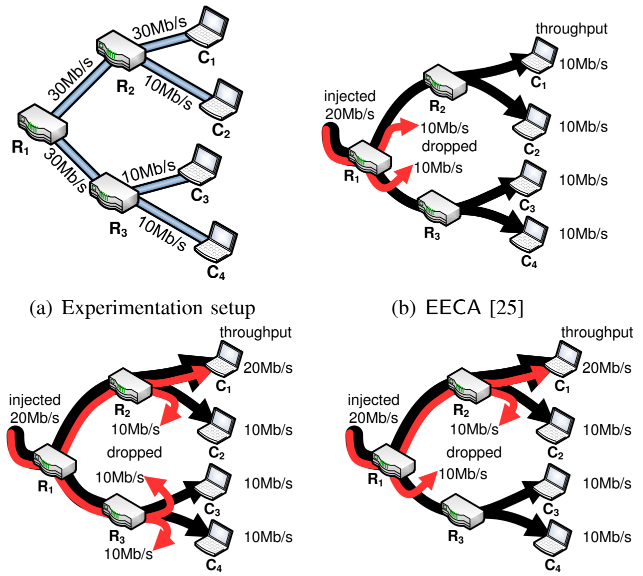
Packet Shedding per link: For each $c \in \mathcal{C}$ and $l \in \mathcal{L}^{(c)}$ choose

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

Above, $\beta_l^{(c)}(K) = m_l^{(c)}K - H_l^{(c)}$, where $m_l^{(c)}$ is the number of session c receivers downstream from link l and $H_l^{(c)}$ is the number of links in the subtree connected to s_c via l, with l.

Parameter Selection: $K \geq \lceil(\max_{c,l} H_l^{(c)} + 1)/m_l^{(c)}\rceil$ and $d_{\max}, V$ are given in Table I.

We experiment in NITOS with *multicast* E-DLS and we compare its performance to MMT [15] and EECA [25]. In Fig. 4(a) we consider a simple multicast tree topology that showcases the advantages of our distributed load shedding mechanism. First, observe that EECA (Fig. 4(b)), a source-based congestion control scheme designed for minimum en-



(a) Experimentation setup (b) EECA [25]

(c) MMT [15] (d) *multicast* E-DLS

Fig. 4. Multicast tree setup (top-left) and experimental results (rest).

ergy in unicast, drops all excess load at the source, and hence its sum throughput is 40Mb/s, which is less that the theoretically maximum 50Mb/s. This demonstrates that in multirate multicast, a source-based solution is not efficient. Then, observe that the distributed load shedding policy MMT from [15] (Fig. 4(c)), which is the multicast extension of ORA. It achieves maximum sum throughput, but uses unnecessarily more energy than *multicast* E-DLS (Fig. 4(d)), since it carries excess traffic of 10Mb/s from R_1 to R_3 which is later dropped. In conclusion, the proposed multicast extension *multicast* E-DLS achieves maximum sum throughput using only the necessary average number of transmissions.

VI. Conclusion & Future Work

We study distributed load shedding in a network, which amounts to initially admitting all extraneous load and gradually shedding part of the load at individual nodes in a distributed fashion. It is observed that such a methodology may lead to excessive energy consumption, since some packets may be forwarded several hops before they are dropped. We formulate a multi-objective load shedding optimization problem where we seek to minimize the energy used to achieve maximum sum throughput. We prove that a simple distributed online policy, using threshold-based shedding and pushback routing, solves this problem for the case of single-path unicast sessions. Then we propose a heuristic extension for multirate multicast routing. In [21] we also include a heuristic extension for multipath routing. Experiments in NITOS showcase that our policies achieve our objectives.

Our work can be extended in many ways. First, the consideration of multipath routing can be incorporated using methods similar to [24]. It is desirable to prove that our proposed policy succeeds in solving optimally the multirate multicast routing case as well. Moreover, we plan to further extend the proposed policy towards the maximization of the sum utility, instead of the sum throughput, assuming that each destination is assigned a potentially different utility function. Last but not least, we are willing also to include network modeling for multihop wireless with links operating on the same channel.

References

[1] S. Shakkottai and R. Srikant. Network Optimization and Control. *Found. Trends Netw.*, 2007.

[2] L. Tassiulas and A. Ephremides. Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks. *IEEE Trans. on Automatic Control*, 37(12):1936–1948, December 1992.

[3] M. J. Neely, E. Modiano, and C. Li. Fairness and Optimal Stochastic Control for Heterogeneous Networks. *IEEE/ACM Trans. on Networking*, 16(2):396–409, April 2008.

[4] C. Li and E. Modiano. Receiver-Based Flow Control for Networks in Overload. *IEEE/ACM Trans. on Networking*, 23(2):616–630, April 2015.

[5] N. Tatbul, U. Çetintemel, S. Zdonik, M. Cherniack, and M. Stonebraker. Load Shedding in a Data Stream Manager. In *Proc. International Conference on Very Large Data Bases (VLDB)*, 2003.

[6] AP. Bianzino, C. Chaudet, D. Rossi, and J. Rougier. A Survey of Green Networking Research. *IEEE Commun. Surveys Tuts.*, 2012.

[7] L. M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proc. IEEE INFOCOM*, 2001.

[8] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993.

[9] V. Jacobson. Congestion Avoidance and Control. *SIGCOMM Comput. Commun. Rev.*, 18(4):314–329, August 1988.

[10] S. H. Low and D. E. Lapsley. Optimization Flow Control&Mdash;I: Basic Algorithm and Convergence. *IEEE/ACM Trans. on Networking*, 7(6):861–874, December 1999.

[11] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures. *Proceedings of the IEEE*, 95(1):255–312, January 2007.

[12] J. Mo and J. Walrand. Fair End-to-end Window-based Congestion Control. *IEEE/ACM Trans. on Networking*, 8(5):556–567, October 2000.

[13] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Oper. Res.*, 49:237–252, 1998.

[14] C. Li and E. Modiano. Receiver-Based Flow Control for Networks in Overload. http://arxiv.org/abs/1207.6354.

[15] G. S. Paschos, C. Li, E. Modiano, K. Choumas, and T. Korakis. Multirate Multicast: Optimal Algorithms and Implementation. In *Proc. IEEE INFOCOM*, 2014.

[16] N. Tatbul, U. Çetintemel, and S. Zdonik. Staying FIT: Efficient Load Shedding Techniques for Distributed Stream Processing. In *Proc. International Conference on Very Large Data Bases (VLDB)*, 2007.

[17] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[18] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool Publishers, 2010.

[19] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource Allocation and Cross-layer Control in Wireless Networks. *Found. Trends Netw.*, 2006.

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

[21] Distributed Load Shedding with Minimum Energy. https://www.dropbox.com/s/vvoaonc751do60a/EDLS-report.pdf?dl=0.

[22] M. J. Neely. Super-Fast Delay Tradeoffs for Utility Optimal Fair Scheduling in Wireless Networks. *IEEE Journal on Sel. Areas in Comm.*, 24(8):1489–1501, August 2006.

[23] L. Huang and M. J. Neely. Delay Reduction via Lagrange Multipliers in Stochastic Network Optimization. *IEEE Trans. on Automatic Control*, 56(4):842–857, April 2011.

[24] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu. Joint asynchronous congestion control and distributed scheduling for wireless networks. In *Proc. IEEE INFOCOM*, 2006.

[25] M. J. Neely. Energy Optimal Control for Time Varying Wireless Networks. *IEEE Trans. on Information Theory*, 52(7):2915–2934, July 2006.

Appendix A
Proof of Theorem 1

Proof of Theorem 1, part (1). To show that the queue lengths are deterministically bounded by (18), we use induction. Since the system starts empty, (18) clearly holds for $t=0$. Suppose it also holds for t, we will show it is true for $t+1$. First, assume $Q_n^{(c)}(t) \leq KV$. Then from (1) we have

$$Q_n^{(c)}(t+1) \leq Q_n^{(c)}(t) + \mu_{p(n)}^{(c)}(t) \leq KV + A_{\max} + \max_{(n,m)} R_{nm} = KV + d_{\max}.$$

Else, if $Q_n^{(c)}(t) > KV$ then by the inductive step it has to be $KV < Q_n^{(c)}(t) \leq KV + d_{\max}$. From (17), E-DLS decides to shed $d_{\max}$ packets in this slot. Hence

$$Q_n^{(c)}(t+1) \leq \left(Q_n^{(c)}(t) - d_{\max}\right)^+ + \mu_{p(n)}^{(c)}(t) \leq KV + d_{\max}.$$

This completes the proof for (18). □

Proof of Theorem 1, part (2). In this part, we will show that the dynamic packet-level decisions converge in the limit to time-averages that support maximum throughput. The total number of session c packets generated up to time τ are split in the following way

$$\underbrace{\sum_{t=0}^{\tau-1} A^{(c)}(t)}_{\text{arrived}} = \underbrace{\sum_{t=0}^{\tau-1} \tilde{\mu}_{p(\delta_c)}^{(c)}(t)}_{\text{received}} + \underbrace{\sum_n \sum_{t=0}^{\tau-1} \tilde{d}_n^{(c)}(t)}_{\text{dropped}} + \underbrace{\sum_n Q_n^{(c)}(\tau)}_{\text{queued}},$$

where we denote with $\tilde{d}_n^{(c)}(t)$ the actual number of packets dropped from node n, since $\tilde{d}_n^{(c)}(t)$ depends on backlog availability and $\tilde{d}_n^{(c)}(t) \leq d_n^{(c)}(t)$. A consequence of the bounded backlogs from part (1) is that $\lim_{\tau\to\infty} \frac{1}{\tau}\sum_{c,n} \mathbb{E}[Q_n^{(c)}(\tau)] = 0$. Summing up over sessions, taking expectations, dividing by τ, taking limits, and using the Law of Large Numbers for $A^{(c)}(t)$, we get

$$\sum_c \lambda^{(c)} = \hat{T}(K) + \sum_{c,n} \hat{q}_n^{(c)}, \tag{25}$$

where $\hat{q}_n^{(c)} \triangleq \lim_{\tau\to\infty} \frac{1}{\tau}\sum_{t=0}^{\tau-1} \mathbb{E}[\tilde{d}_n^{(c)}(t)]$ is the limit of time average shedding for E-DLS.

We introduce RAN, a helpful randomized policy that uses an offline solution to (12) $(\boldsymbol{f}^*, \boldsymbol{q}^*)$ (the solution depends on K, V), and then chooses controls so that $\mathbb{E}[\mu_n^{(c)}(t)|\mathbf{Q}(t)] = f_n^{(c)*}$ and $\mathbb{E}[d_n^{(c)}(t)|\mathbf{Q}(t)] = q_n^{(c)*}$. This is an ω-only optimal policy [18]. Let us denote by $Y(t)$ the RHS of (15) evaluated under E-DLS and $Y^*(t)$ under RAN. Since E-DLS is designed to minimize the RHS of (15) at every slot, we have $Y(t) \leq Y^*(t)$, and combining with (15) we have

$$\Delta(t) + V\sum_{c,n} \alpha_n^{(c)}(K)\mathbb{E}[d_n^{(c)}(t)|\mathbf{Q}(t)] \leq Y^*(t). \tag{26}$$

Canceling out terms using the flow conservation (3) we find that $Y^*(t) = B + V\sum_{c,n} \alpha_n^{(c)}(K) q_n^{(c)*}$. So, summing up the inequality (26) over slots $\{0, ..., \tau-1\}$, dividing by $V\tau$, taking expectation with respect to backlogs, and using $L(0) = 0$,

$$\frac{\mathbb{E}L(\tau)}{V\tau} + \frac{1}{\tau}\sum_{t=0}^{\tau-1}\sum_{c,n} \alpha_n^{(c)}(K)\mathbb{E}[d_n^{(c)}(t)] \leq \frac{B}{V} + \sum_{c,n} \alpha_n^{(c)}(K) q_n^{(c)*}.$$

Taking limits, and using $\lim_{\tau\to\infty} \mathbb{E}L(\tau)/\tau \to 0$ from (18), the above becomes

$$\sum_{c,n} \alpha_n^{(c)}(K)\hat{q}_n^{(c)} \leq \frac{B}{V} + \sum_{c,n} \alpha_n^{(c)}(K) q_n^{(c)*}. \tag{27}$$

Also, by the constraint on maximum drops per slot we have $\sum_{c,n} h_n^{(c)}\hat{q}_n^{(c)} \leq CN^2 d_{\max}$. Thus,

$$\begin{aligned}
\hat{T}(K) &\overset{(25)}{=} \sum_c \lambda^{(c)} - \frac{1}{K}\sum_{c,n} \alpha_n^{(c)}(K)\hat{q}_n^{(c)} - \frac{1}{K}\sum_{c,n} h_n^{(c)}\hat{q}_n^{(c)} \\
&\overset{(27)}{\geq} \sum_c \lambda^{(c)} - \frac{B/V + \sum_{c,n}\alpha_n^{(c)}(K) q_n^{(c)*} + CN^2 d_{\max}}{K} \\
&\overset{(6)}{\geq} \sum_c r^{(c)*}(K) - \frac{B/V + CN^2 d_{\max}}{K},
\end{aligned}$$

where $\sum_c r^{(c)*}(K)$ is the max throughput from (12). Using proposition 4.2.1 from [17], any solution of (12) converges to some solution of (10), i.e., $\lim_{K\to\infty}\sum_c r^{(c)*}(K) = T^*$. The proof of (19) is completed by taking the limit $K \to \infty$. □

Proof of Theorem 1, part (3). From (7) we have

$$\begin{aligned}
E^* &= \min_{\boldsymbol{f},\boldsymbol{q}\geq \mathbf{0}} \sum_{c,n} f_n^{(c)} \overset{(9)}{=} \sum_c N^{(c)}\lambda^{(c)} - \max_{\boldsymbol{f},\boldsymbol{q}\geq \mathbf{0}} \sum_{c,n} h_n^{(c)} q_n^{(c)} \\
&= \sum_c N^{(c)}\lambda^{(c)} - \sum_{c,n} h_n^{(c)}\overline{q}_n^{(c)},
\end{aligned} \tag{28}$$

where $\overline{q}_n^{(c)}$ is the solution to (7), and $N^{(c)}$ is the number of nodes on path $P^{(c)}$ less the destination. The expected number of transmitted packets up to τ under E-DLS are

$$\sum_{t=0}^{\tau-1}\mathbb{E}[\tilde{\mu}_n^{(c)}(t)] = \sum_{t=0}^{\tau-1}\left(A^{(c)}(t) - \sum_{k\in\mathcal{U}(n)}\mathbb{E}[\tilde{d}_k^{(c)}(t)] + \mathbb{E}[Q_k^{(c)}(t)]\right).$$

Dividing by τ and taking limits we have

$$\hat{f}_n^{(c)} = \lambda^{(c)} - \sum_{k\in\mathcal{U}(n)} \hat{q}_n^{(c)}(K,V), \tag{29}$$

where $\hat{f}_n^{(c)} = \lim_{\tau\to\infty} 1/\tau \sum_{t=0}^{\tau-1}\mathbb{E}[\tilde{\mu}_n^{(c)}(t)]$. Now we express the total energy consumed under E-DLS as

$$\begin{aligned}
\hat{E}(K,V) &\overset{(14)}{=} \sum_{c,n}\hat{f}_n^{(c)} \overset{(29)}{=} \sum_{c,n}\left(\lambda^{(c)} - \sum_{k\in\mathcal{U}(n)}\hat{q}_n^{(c)}(K,V)\right) \\
&= \sum_c N^{(c)}\lambda^{(c)} - \sum_{c,n} h_n^{(c)}\hat{q}_n^{(c)}(K,V) \\
&\overset{(28)}{=} E^* + \sum_{c,n} h_n^{(c)}\left[\overline{q}_n^{(c)} - \hat{q}_n^{(c)}(K,V)\right].
\end{aligned} \tag{30}$$

Let $\hat{q}_n^{(c)}(K,\infty)$ denote the limit performance under E-DLS as we increase $V\to\infty$. From (27) we have

$$\sum_{c,n}\alpha_n^{(c)}(K)\hat{q}_n^{(c)}(K,\infty) \leq \sum_{c,n}\alpha_n^{(c)}(K) q_n^{(c)*}(K). \tag{31}$$

Since $(\hat{q}_n^{(c)}(K,V)), (\hat{f}_n^{(c)}(K,V))$ is a feasible solution of (12), then by (31) it follows that $\hat{q}_n^{(c)}(K,\infty)$ is also an optimal one. Therefore, we may revisit the design of the RAN policy and specifically select $q_n^{(c)*}(K) = \hat{q}_n^{(c)}(K,\infty)$ for all n. Finally the proof can be completed by taking the limit $V\to\infty, K\to\infty$ on (30) and noticing that $\lim_{K\to\infty} q_n^{(c)*}(K) = \overline{q}_n^{(c)}$. □