# Enhance & Explore: an Adaptive Algorithm to Maximize the Utility of Wireless Networks

Adel Aziz†, Julien Herzen†, Ruben Merz‡, Seva Shneer⋆, Patrick Thiran†
† School of Computer and Communication Sciences, EPFL
‡ Deutsche Telekom Laboratories/TU Berlin
⋆ Heriot-Watt University
{adel.aziz, julien.herzen, patrick.thiran}@epfl.ch, ruben.merz@telekom.de, v.shneer@hw.ac.uk

## ABSTRACT

The goal of jointly providing efficiency and fairness in wireless networks can be seen as the problem of maximizing a given utility function. In contrast with wired networks, the capacity of wireless networks is typically time-varying and not known explicitly. Hence, as the capacity region is impossible to know or measure exactly, existing scheduling schemes either under-estimate it and are too conservative, or they over-estimate it and suffer from congestion collapse. We propose a new adaptive algorithm, called *Enhance & Explore* (E&E). It maximizes the utility of the network without requiring any explicit characterization of the capacity region. E&E works above the MAC layer and it does not demand any modification to the existing networking stack.

We first evaluate our algorithm theoretically and we prove that it converges to a state of optimal utility. We then evaluate the performance of the algorithm in a WLAN setting, using both simulations and real measurements on a testbed composed of IEEE 802.11 wireless routers.

Finally, we investigate a wireless mesh network setting and we find that, when coupled with an efficient mechanism for congestion-control, the E&E algorithm greatly increases the utility achieved by multi-hop networks as well.

## Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

## General Terms

Algorithms, Design, Measurement, Performance, Theory

## 1. INTRODUCTION

Two key objectives in networking are to design systems that (i) deliver high throughput to the users and that (ii) achieve a certain level of fairness between the different flows. Previous work shows that the tradeoff between fairness and efficiency can be captured mathematically by a utility function [6], where the best possible operating point of the system is found by solving a maximization problem. However, the techniques for solving this maximization problem usually require that the network capacity is known and time-invariant. Although these two assumptions may hold true in wired networks, the situation completely differs in practical wireless environments where the network capacity is not only time-varying, but also difficult to estimate on-the-fly [20].

In order to circumvent the difficult evaluation of capacity, this paper proposes a new practical and adaptive algorithm that provides fairness and efficiency (i.e., maximizes a given utility function) *without* knowing the capacity region. We show that our utility maximization algorithm, called *Enhance & Explore* (E&E), finds applications both in single-hop settings (e.g., WLANs) and multi-hop settings (e.g., wireless mesh networks).

In single-hop setups, the poor performance of the widely used IEEE 802.11 protocol is already visible in a simple 2-flow scenario. Figure 1 presents experimental evidence of a serious issue of the protocol known as the rate anomaly problem [12]. In this setting, two nodes (nodes 1 and 2) send traffic to the gateway (GW) using physical layer rates of respectively 1 Mb/s and 11 Mb/s. Interestingly, the node with the highest physical rate receives the smallest throughput. Although the situation slightly improves when using TCP (as compared to saturated UDP), it still remains far from efficient: for instance, in the sense of proportional fairness, where both flows should share the air time equally (i.e., each flow should get approximately half of the throughput that it obtains when transmitting in isolation). To overcome this problem, a possible direction is

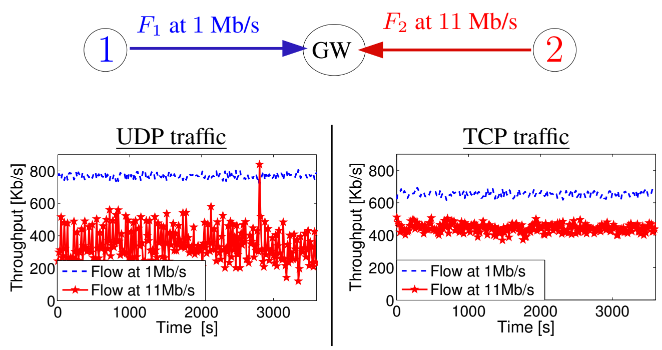


**Figure 1: Illustration of the fairness problem occurring when different rates are used within a WLAN. Even though flow $F_1$ operates at 11 Mb/s and flow $F_2$ at 1 Mb/s, flows $F_1$ obtains the smallest throughput. Our measurements show that this rate anomaly problem occurs both with UDP (left) and TCP (right).**

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*MobiCom'11,* September 19–23, 2011, Las Vegas, Nevada, USA.
Copyright 2011 ACM 978-1-4503-0492-4/11/09 ...$10.00.

to design algorithms that appropriately rate-limit the sources so that the system operates at its optimal point according to a utility function. Given that the traffic demand at the sources and the rate region (namely, the set of rates that the system can sustain simultaneously) are generally unknown and time-varying, it is impossible to compute *a priori* this optimal rate allocation. Instead, we propose an adaptive algorithm that (i) provably converges to the optimal allocation, (ii) naturally adapts to time-varying conditions, and (iii) does not require any changes in the networking stack. We focus on networks with one gateway (although a trivial extension to multiple gateways is discussed in Section 3.4). In this case, our algorithm uses the fact that all the traffic goes through the gateway. The gateway is therefore aware of the utility achieved by the system at any point in time, and it can try new rate allocations and test whether they are feasible (i.e., whether the achieved throughput corresponds to the allocated rate). Based on this observation, we design the *Enhance & Explore* algorithm that alternates between (i) *enhance phases* that try to improve the utility of the network, and (ii) *explore phases* that find rate allocations, within the unknown capacity region, that achieve this targeted utility.

In multi-hop scenarios the inter-flow fairness problem persists and it is even coupled with an intra-flow congestion-control problem. Addressing congestion and ensuring stability with distributed access mechanisms in multi-hop networks has received much attention since the seminal work of Tassiulas et al. [23]. Recent solid analytical works [5, 10, 17, 22, 27, 29] on network stability require the knowledge of the capacity region of the network in order for sources to send at a rate within this region and avoid congestion collapse. In addition, their practical evaluation remains a hard problem due to their requirements for changes that are incompatible with existing wireless network interface controllers (NIC). Recent efforts that implement hop-by-hop congestion control schemes [3, 26] use the contention window parameter $CW_{min}$ of IEEE 802.11. However, this interferes with the regular operation of the MAC. Instead, we follow a passive approach to perform congestion control at layer 2.5 (i.e., no interactions with the MAC) without any form of message passing.

Our main contributions in this paper are:

- a utility maximization algorithm for WLANs;
- a combined mechanism for joint utility maximization and congestion control in wireless mesh networks.

The novelties in our mechanism are that (i) it does not require knowledge of the rate region, (ii) it uses almost no message passing (and no piggy-backing) regardless of the size of the multi-hop network, (iii) it is completely transparent to the rest of the networking stack and (iv) it performs reliably for any traffic input rate at the sources.

We show analytically that the E&E algorithm converges to a point of optimal utility in the single-hop network topology. In addition, we implement our schemes both in the ns-3 simulator and in a 9-node testbed. We evaluate several scenarios that cover both the utility and congestion-control aspects, and our experiments show excellent performance both in single-hop and multi-hop networks.

The remainder of the paper is organized as follows. The problem statement and necessary background is contained in Section 2. The E&E algorithm is presented in Section 3 along with a formal convergence proof. The extension to multi-hop scenarios with the additional congestion-control problem is discussed in Section 4. The practical evaluations of the E&E algorithm and the congestion-control mechanism are performed using both experiments in Section 5 and simulations in Section 6. Finally, after discussing related work in Section 7, we conclude by summarizing our findings in Section 8.

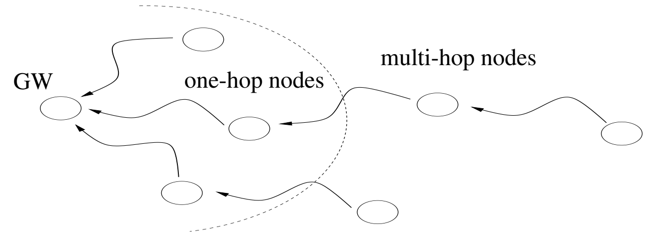


**Figure 2: We consider two topologies for our evaluation: a single-hop WLAN topology and a multi-hop mesh topology.**

## 2. BACKGROUND AND MODEL

We consider scenarios where several wireless nodes communicate with one gateway (an extension to scenarios with multiple gateways is discussed in Section 3.4). The nodes can typically be home wireless equipments, or access points serving clients in a Wireless Mesh Network (WMN). In the WMN setting, not all the nodes directly communicate with the gateway, as some of them may be several hops away. We assume that the topology is a tree that is rooted at the gateway, and that the routes are known by the nodes. A typical setup is depicted in Figure 2.

We consider a single-antenna setup with all the nodes set to the same channel. In WMNs, the access points may possess one extra wireless interface to serve clients in an orthogonal channel, but all the access points use the same channel to communicate together (i.e., in the backbone of the mesh).

Finding an optimal schedule for the downstream traffic (from the gateway to the nodes) is a well-known problem that can be directly solved by using weighted fair queuing at the gateway [7]. As the solution to the downstream problem already exists, we focus on upstream traffic. In this case, the distributed coordination of several sources is challenging, because they are not aware of each other's state.

Before describing the algorithm, we introduce a few definitions:

- A *flow* is the set of all packets exchanged between a source and a destination in the wireless network. Here, the destination is always the gateway and the source can be any wireless node of the network. Therefore, a flow is uniquely identified by the IP address of the wireless node from which it emanates. In WMNs, we take the source of a flow to be the access point that serves this flow, and not the end-client from which it emanates. This ensures scalability as the number of flows remains bounded by the number of access points in the mesh. In WLANs (i.e., single-hop networks) the number of flows is upper-bounded by the number of clients connected to the gateway. We denote by $F$ the number of flows present in the system.
- The gateway uses *time slots* that are indexed by $n \in \mathbb{N}$ and that correspond to the time duration during which (i) the throughputs of the flows are measured and averaged by the gateway, and (ii) a given rate allocation is enforced (the rates are chosen by the gateway at the beginning of each slot $n$).
- The *rate allocation vector* is denoted by $\vec{\rho}[n] \in \mathbb{R}_+^F$ and its entries $\rho_i[n]$ are the rates given to the rate limiters for each of the $F$ flows of the system.

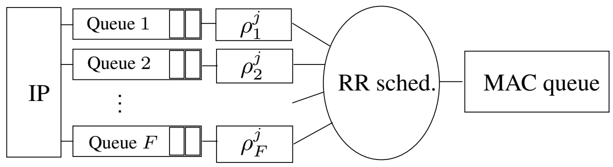


**Figure 3: Rate control mechanism at node $j$. Each flow $i$ has a dedicated queue and a rate limiter $\rho_i^j$ that limits its rate (note that, for the sake of the example, all the $F$ flows are going through node $j$. In the general setting, only a subset of the $F$ flows go through each node). A round-robin (RR) scheduler connects the rate limiters to the MAC queue.**

- The *throughput vector* is denoted by $\vec{x}[n] \in \mathbb{R}_+^F$ and its entries $x_i[n]$ are the throughputs actually achieved by the flows.

- The *rate region* is denoted by $\Lambda[n]$ and it is the set of all throughputs $\vec{x}[n]$ that are achievable by the network.

- The *flow utility function* $u_i(\cdot) : \mathbb{R}_+ \to \mathbb{R}_+$ captures the utility of the $i$-th flow. We assume that for all $i$, $u_i(\cdot)$ is continuously differentiable, strictly increasing and non-negative (i.e., $u_i(x_i[n]) \geq 0$).

- The *network utility function* $U(\cdot) : \mathbb{R}_+^F \to \mathbb{R}_+$ captures the utility of the entire system (i.e., of all $F$ flows);

  $$U(\vec{x}[n]) = \sum_{i=1}^{F} u_i(x_i[n]).$$

  Two typical examples of utility functions are proportional fairness:

  $$U_{prop}(\vec{x}[n]) = \sum_{i=1}^{F} \ln(x_i[n] + 1), \tag{1}$$

  and throughput maximization:

  $$U_{max}(\vec{x}[n]) = \sum_{i=1}^{F} x_i[n]. \tag{2}$$

  Note that maximizing proportional fairness is equivalent to maximizing the products of the throughputs achieved by the flows.

- The *level set* $L(\mu[n]) : \mathbb{R}_+ \to \mathbb{R}_+^F$ is the set of all rates having utility $\mu[n]$, that is,

  $$L(\mu[n]) = \{\vec{x}[n] : \vec{x}[n] \in \mathbb{R}_+^F, U(\vec{x}[n]) = \mu[n]\}.$$

Our rate-control mechanism operates between the layer 2 (MAC) and the layer 3 (IP). As shown in Figure 3, each node $j$ (except the gateway) maintains one IP queue per flow $i$. The output of each queue $i$ is attached to a *rate limiter* $\rho_i^j$ that throttles the output of the queue at a given rate[1]. Finally, the rate limiters are scheduled to the MAC queue in a round-robin fashion.

[1]We use the word *rate* to denote any kind of bitrate. In our implementation, we represent the rates in packets per second for convenience. This should not be confused with the underlying modulation rate used by the physical layer.

# 3. E&E ALGORITHM IN SINGLE-HOP

In this section, we first describe the E&E algorithm in a single-hop setup, where several nodes directly send traffic to one single gateway. We then discuss an extension to multiple gateways and we defer the discussion of multi-hop networks to Section 4 where we introduce the congestion problem.

## 3.1 High-Level Description

In the WLAN setting, the gateway is the end-point of all traffic, it thus knows the throughput achieved by each flow at every point in time. The global throughput information present at the gateway makes it the ideal location to run a utility maximization algorithm. The E&E algorithm maximizes the network utility without *a priori* knowing the capacity region by using the following iterative approach:

1. The gateway selects a rate allocation $\vec{\rho}[n]$ and it broadcasts its decision to the one-hop nodes.

2. The one-hop nodes apply the allocation $\vec{\rho}[n]$ to their rate limiters.

3. The gateway measures the throughput $\vec{x}[n]$ that is actually achieved by the flows.

4. From this observation, the gateway discovers if the rate $\vec{\rho}[n]$ is feasible ($\vec{\rho}[n] \in \Lambda$) and selects a new, hopefully better, allocation. Then the process repeats.

In this process, the main challenge for the gateway is to find rate allocations that increase the achieved utility and that are feasible. To this end, the algorithm consists in a succession of so-called *enhance* and *explore* phases:

- During an **enhance phase**, the algorithm chooses the next targeted utility $\mu[n]$ that it will aim to achieve. To do so, it computes a gradient ascent of the utility function calculated from the current allocation point. This gives the next rate allocation $\vec{\rho}[n]$ (such that $\mu[n] = U(\vec{\rho}[n])$). From our definition of the level set $L(\mu[n])$, the utility $\mu[n]$ determines a subset of rate allocations. Note that the targeted utility $\mu[n]$ is always higher than the utility of the current allocation (due to the gradient ascent). In case no feasible allocation achieving the utility $\mu[n]$ is found, the size of the gradient ascent is gradually decreased. At the end of the enhance phase, an explore phase follows.

- During an **explore phase**, the algorithm measures the throughput $\vec{x}[n-1]$ achieved during the previous slot in order to discover whether the allocation $\vec{\rho}[n-1]$ is feasible or not. To do so, it tests if $\vec{x}[n-1] = \vec{\rho}[n-1]$.
  If the test succeeds, the algorithm goes to the enhance phase (where the size of the next gradient ascent is $\alpha$).
  Otherwise, the gateway randomly selects a new allocation $\vec{\rho}[n]$ in the level set $L(\mu[n])$ (with the constraint that all the points in $L(\mu[n])$ have a positive Lebesgue measure). It then performs a new explore phase in the next time slot. Finally, if no feasible allocation is discovered after at most $T$ explore phases, the algorithm goes back to the enhance phase and it halves the size of the next gradient ascent.

In the next subsections, we describe E&E in detail and provide a proof of its convergence to a rate allocation that achieves the optimal utility.

## 3.2 E&E Algorithm

The pseudo-code of the Enhance & Explore algorithm is specified in Algorithm 1. It uses the following parameters:

- The *last stable assignment vector* $\vec{r}[n] \in \mathbb{R}_+^F$ records the best feasible allocation discovered by the gateway at time $n$. More precisely, at any time slot $n$, we have $\vec{r}[n] = \vec{\rho}[n - \overline{k}]$, with $\overline{k} = \min\{k \mid \vec{\rho}[n-k] = \vec{x}[n-k]\}$.

- The *step size* $\alpha \in \mathbb{R}_+$ is the size of the gradient ascent performed in the enhance phase directly after the discovery of a new feasible allocation.

- The *trial limit* $T \in \mathbb{N}^*$ is the maximal number of allocations drawn from a given level set $L(\mu)$ (i.e., the maximal number of successive explore phases).

**Algorithm 1** E&E algorithm at the gateway

1: **At time slot** $0$**:**
2: measure $\vec{x}[0]$, the throughput delivered by the MAC
3: start the algorithm from the allocation $\vec{\rho}[0] = \vec{x}[0]$
4: store the *last stable assignment* $\vec{r}[0] = \vec{\rho}[0]$
5: store the gradient ascent $\vec{g}[0] = \vec{\rho}[0]$
6: store the achieved utility $\mu[0] = U(\vec{\rho}[0])$
7: wait for next time slot $(0 \rightarrow 1)$
8: go to **Enhance phase**
9:
10: **Enhance phase:** (*find the next targeted level set*)
11: **if** $\vec{x}[n-1] = \vec{\rho}[n-1]$ **then**
12: $\quad \vec{g}[n] = \vec{r}[n-1] + \alpha \frac{\overrightarrow{\nabla U(\vec{r}[n-1])}}{\|\overrightarrow{\nabla U(\vec{r}[n-1])}\|}$
13: **else**
14: $\quad \vec{g}[n] = (\vec{g}[n-1] + \vec{r}[n-1])/2$
15: **end if**
16: $\mu[n] = U(\vec{g}[n])$
17: $\vec{\rho}[n] = \vec{g}[n]$
18: broadcast one packet containing the allocation $\vec{\rho}[n]$
19: wait for next time slot $(n \rightarrow n+1)$
20: go to **Explore phase**
21:
22: **Explore phase:** (*find new allocation in the level set*)
23: measure the throughput $\vec{x}[n]$ obtained during slot $n$
24: **if** $\vec{x}[n-1] = \vec{\rho}[n-1]$ **then**
25: $\quad$ update the *last stable assignment* $\vec{r}[n] = \vec{\rho}[n-1]$
26: $\quad$ go to **Enhance phase**
27: **else**
28: $\quad$ keep previous *last stable assignment*: $\vec{r}[n] = \vec{r}[n-1]$
29: $\quad$ keep previous gradient ascent: $\vec{g}[n] = \vec{g}[n-1]$
30: $\quad$ keep previous targeted level set: $\mu[n] = \mu[n-1]$
31: $\quad$ pick allocation $\vec{\rho}[n]$ randomly in level set $L(\mu[n])$
32: $\quad$ broadcast one packet containing the allocation $\vec{\rho}[n]$
33: $\quad$ wait for next time slot $(n \rightarrow n+1)$
34: $\quad$ repeat **Explore phase** at most $T$ times, then move to **Enhance phase**.
35: **end if**

The algorithm starts from the throughput allocation given by the underlying MAC (lines 2-6). It then moves to an enhance phase that selects the next targeted utility $\mu[n]$ and this utility completely determines the next targeted level set $L(\mu[n])$. We note that the determination of the next utility depends on whether the previous rate allocation is feasible ($\vec{x}[n-1] = \vec{\rho}[n-1]$) or not. In case it is feasible, the utility $\mu[n]$ is obtained by performing a full-size gradient ascent (line 12). Otherwise, the size of the gradient ascent, starting from the last stable allocation $\vec{r}[n]$, is halved (line 14). Finally, after having determined the level set $L(\mu[n])$, the algorithm moves to an explore phase.

In the explore phase, the algorithm starts by evaluating the feasibility of the rate allocation $\vec{\rho}[n-1]$ (lines 23-24). We emphasize that our model does not capture the packet losses due to the link-variability of the wireless medium. Therefore, in our analytical study, the condition $\vec{x}[n-1] = \vec{\rho}[n-1]$ (lines 11 and 24) is true if and only if $\vec{\rho}[n-1] \in \Lambda$. In the practical implementation, however, we need to cope with the possible packet losses due to channel variability and we do so by allowing for a margin $\delta$ in our feasibility-test (i.e., testing for $|\rho_i[n-1] - x_i[n-1]| < \delta$ for all $i$ instead).
If the test succeeds, it means that the algorithm found a feasible allocation in the level set $L(\mu[n-1])$. Hence, the last stable allocation $\vec{r}[n]$ is updated (line 25) and the algorithm moves to an enhance phase to select a new target utility $\mu[n]$.
If the feasibility-test of line 24 fails, the algorithm randomly selects a new rate allocation in the level set $L(\mu[n])$ and it repeats an explore phase in the next time slot (lines 28-33). In case the explore phases are repeated $T$ times successively (i.e., no new feasible allocation is discovered), the algorithm does not repeat an additional explore phase. Instead it moves to the enhance phase and decreases the size of the gradient ascent.

There are multiple distributions of probabilities that can be used in the random selection process (line 31) in order to satisfy the requirement of a positive Lebesgue measure. In our implementation, we use a pseudo-Gaussian distribution of standard deviation $\sigma$ (the exact procedure is described in the Appendix). We follow this approach because it allows to control the desired level of *risk aversion* taken during the explore phase. A small $\sigma$ (high risk aversion) means that the future allocations $\vec{\rho}[n]$ are more likely to remain close to $\vec{r}[n]$, the best known allocation at time $n$. On the contrary, a large $\sigma$ (low risk aversion) allows to explore more often allocations $\vec{\rho}[n]$ that are further away from $\vec{r}[n]$. The advantage of the first strategy is that it is more likely to result in an actual throughput $\vec{x}[n]$ close to $\vec{r}[n]$. However, it is also more likely to remain locked in a local optimum for a longer (albeit finite) period of time. The second strategy does not suffer this drawback, but it may result in larger variations of throughput $\vec{x}[n]$ by trying allocations far from $\vec{r}[n]$ that are not feasible.

## 3.3 Convergence Analysis

In this section, we show that the E&E algorithm converges to an optimal allocation. We formally prove that it yields an allocation whose utility is the best possible given the underlying MAC layer.

For the sake of analysis, we impose the two following constraints on the rate region $\Lambda[n]$:

- $\Lambda[n]$ is constant with time (i.e., $\Lambda[n] = \Lambda$). This hypothesis is necessary for the convergence analysis, because there cannot be convergence to an optimum that is always changing in time. We show that given a fixed but unknown rate region, the algorithm finds an allocation delivering the best possible utility. However, the result holds *for any initial condition* and therefore, in real systems, the algorithm keeps adapting to any change in the rate region.

- $\Lambda$ is coordinate-convex[2]. This is an extremely reasonable

[2] A set $S \in \mathbb{R}_+^F$ is coordinate-convex when the following is true: if $\vec{b} \in S$, then for all $\vec{a} : \vec{0} \leq \vec{a} \leq \vec{b}$, $\vec{a} \in S$, with $\leq$ denoting the component-wise comparison.

assumption for a rate region: given a feasible allocation, decreasing one or several rates still yields a feasible allocation.

The convergence of the E&E algorithm follows from the two following lemmas. Lemma 3.1 first states that the intersection of any level set involved in line 31 of the algorithm with the rate region has a non zero probability to be visited by an explore phase. Lemma 3.2 states that such a region of the rate space is eventually almost surely discovered by the explore phase. Finally, Theorem 3.1 wraps these two facts together and establishes the convergence of the algorithm to a point of optimal utility.

LEMMA 3.1. *For any $\mu < U^*$, the intersection $L(\mu) \cap \Lambda$ has a non-zero $(F-1)$-dimensional Lebesgue measure.*

**Proof:** Take a point $\vec{x}^* \in \Lambda$ such that the utility function at this point is equal to the optimal utility, $U(\vec{x}^*) = U^*$, and consider the set

$$C = \{\vec{y} : y_i < x_i^* \quad \text{for all} \quad i = 1, \dots, F\}.$$

It is clear that due to the continuity of the functions $u_i$, the function $U$ on the set $C$ takes every value between 0 and $U^*$. Indeed, assume that it does not hold, i.e. $\sup_{\vec{y} \in C} U(\vec{y}) < U^*$. Take a sequence $\{\vec{y}_n\}$ such that $\vec{y}_n \in C$ for all $n$ and $\vec{y}_n \to \vec{x}^*$ as $n \to \infty$. Then, due to the continuity of the functions $u_i$ (and hence, of $U$), it should hold that

$$U(\vec{y}_n) \to U^*.$$

Whereas,

$$\lim_{n \to \infty} U(\vec{y}_n) \le \sup_{\vec{y} \in C} U(\vec{y}) < U^*,$$

which leads to a contradiction.

Note also that any point from the set $C$ is not on the boundary of $\Lambda$. Indeed, as $\Lambda$ is coordinate-convex, we have

$$\overline{C} = \{\vec{y} : y_i \le x_i^* \quad \text{for all} \quad i = 1, \dots, F\} \subseteq \Lambda,$$

and, clearly, for any point $\vec{y} \in C$, there exists an $\varepsilon > 0$ such that

$$B(\vec{y}, \varepsilon) \subseteq \overline{C},$$

where $B(\vec{y}, \varepsilon)$ is a ball with the center at $\vec{y}$ and radius $\varepsilon$

Take any $\mu < U^*$. There exists a point $\vec{y} \in C$ and a positive number $\varepsilon$ such that $U(\vec{y}) = \mu$ and

$$B(\vec{y}, \varepsilon) \subseteq \overline{C} \subseteq \Lambda.$$

As the level set $L(\mu)$ contains the point $\vec{y}$ and the function $U$ is continuous, clearly, the intersection of the level set $L(\mu)$ with $B(\vec{y}, \varepsilon)$ has a positive $(F-1)$-dimensional Lebesgue measure.

∎

LEMMA 3.2. *Suppose that at slot $n$ the allocation is $\vec{r}[n]$, and let $\mu' > U(\vec{r}[n])$. If $L(\mu') \cap \Lambda$ has a non-zero $(F-1)$-dimensional Lebesgue measure, then, with probability one, there exists a $k < \infty$ such that $U(\vec{r}[n+k]) > U(\vec{r}[n])$. In other words, the algorithm eventually discovers an allocation with higher utility, if such an allocation exists.*

**Proof:** This is obvious due to the construction of the explore phase of the algorithm and to the previous lemma: the $(F-1)$-dimensional Lebesgue measure of $L(\mu') \cap \Lambda$ is positive, and the algorithm makes an unbounded number of attempts to discover it. Assuming the converse to the statement of the lemma would lead to a trivial contradiction.

∎

THEOREM 3.1. *Assume that the capacity region $\Lambda$ is fixed and coordinate-convex. Then the E&E algorithm guarantees that, for any initial rate allocation $\vec{r}[0]$, the utility of the last stable allocation $\vec{r}[n]$ converges to the maximal utility for $n \to \infty$.*

**Proof:** Note that, by construction of the algorithm,

$$U(\vec{r}[n+1]) \ge U(\vec{r}[n])$$

and that

$$U(\vec{r}[n]) \le U^*$$

for every $n$. Hence, the sequence $\{U(\vec{r}[n])\}$ is non-decreasing and bounded from above, which yields the existence of $U^{**}$ such that $U(\vec{r}[n]) \to U^{**}$. Clearly, $U^{**} \le U^*$. Assuming that $U^{**} < U^*$ would immediately contradict Lemma 3.2, hence $U^{**} = U^*$.

∎

## 3.4 Extension to Multiple Gateways

When more than one gateway is used to serve flows that have dependent rate regions, the E&E algorithm can still be applied distributively by the gateways. If all the gateways are connected to a wired network (which is the case if they belong to the same administrative domain, or if they are connected to the Internet), they can collaborate, elect one of them as a leader, and run the E&E algorithm unmodified at the leader. In this case, all the other gateways report their throughputs $\vec{x}[n-1]$ to the leader at the beginning of time slot $n$, and the leader replies with the rate allocation $\vec{\rho}[n]$ that is in turn broadcasted by the gateways on the wireless network.

# 4. EXTENSION TO MULTI-HOP NETWORKS

Up to this point in the paper, we have focused on single-hop topologies where all the nodes are in the transmission range of the gateway. In this setting the gateway runs the E&E algorithm and it only needs one single broadcast message per time slot to inform all sources of the new rate allocation $\vec{\rho}[n]$. In multi-hop scenarios the situation differs and rate-limiting solutions generally require network-wide message passing [9].

We propose to avoid the overhead of network-wide message passing by taking advantage of the congestion phenomena that naturally appears in multi-hop topologies. To this end, we briefly discuss the congestion problem and we introduce a mechanism to efficiently adapt the rate limiters of the relay nodes (i.e., to perform congestion-control) without any form of message passing or piggybacking. Finally, we explain how the coupling of this congestion-control mechanism with E&E enables us to avoid network-wide message passing when propagating the fair rate allocations to the sources.

## 4.1 Problem Description

Wireless networks typically use distributed MAC protocols (e.g. CSMA/CA) that have been proven to suffer from congestion when no counter-measure is applied [2]. In wired networks, the queuing policy is the key factor for unfairness. A well-known solution is to use *fair queuing*: one queue per-flow is maintained and the queues are scheduled in a round-robin fashion [7, 16]. In wireless networks, fair queuing is required but not sufficient by itself to ensure fairness among flows. Indeed, ensuring fairness depends both on the MAC and the queuing policy [9]. Figure 4 shows experimental results that illustrate how fair queuing fails to achieve fairness when a 1-hop flow competes with a 3-hop flow even if it is max-min fair when a 1-hop flow competes with a 2-hop flow [3].

[3]Demo at: http://icawww1.epfl.ch/NetController/ (Video 1)

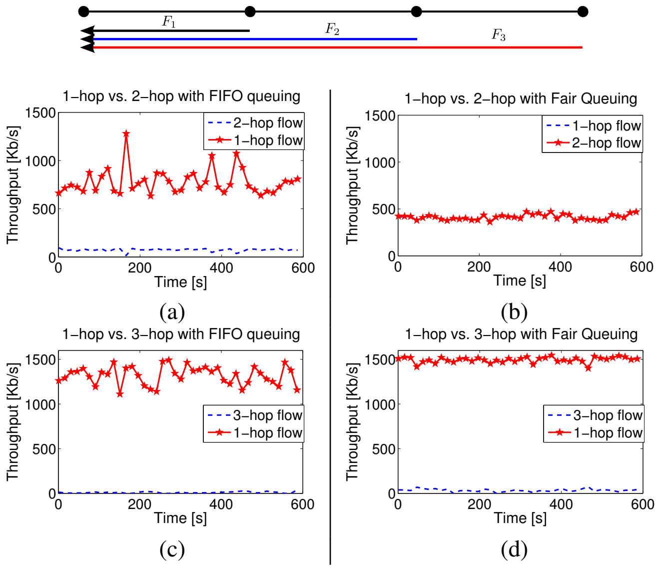


**Figure 4: End-to-end throughput in a line topology (shown above) with a single-hop and a multi-hop UDP flow (2- or 3-hop) with the standard FIFO policy (left) and with fair queuing (right). Fair queuing achieves max-min fairness for the 2-hop case, but fails to prevent starvation in the 3-hop case. Results in the top-right picture (b) look identical, but it is only an artifact of the scale of the picture. Smaller scale plots show differences.**

In order to solve this flow starvation problem without requiring network-wide message passing, we propose to use the E&E algorithm and to combine it with a passive hop-by-hop congestion control algorithm that we describe in the next section.

## 4.2 Intra-Flow Congestion Control

Considering a single multi-hop flow, the goal of intra-flow congestion control is to create a smooth packet flow with low end-to-end delays and high throughput. We use a hop-by-hop approach where the rates of the source and of its relay nodes are adapted to maintain a small (but non-zero) number of packets in the relay queues. Obviously, the last node just before the gateway is not rate limited by this mechanism.

We denote by $q_i^j$ the number of packets that are contained in the queue of flow $i$ at node $j$ and by $s_i(j)$ the index of the next-hop (or successor) of node $j$ for flow $i$. We will drop the index $i$ if it is clear from the context. For instance, $q_i^{s(j)}$ is the size of the queue for flow $i$ at the next-hop of node $j$.

For each flow $i$ traversing node $j$, our algorithm sets the rate $\rho_i^j$ of the rate limiter (see Figure 3) according to the size $q_i^{s(j)}$ of the next-hop queue. Hence, our solution comprises two phases: (i) a passive estimation of $q_i^{s(j)}$, without message-passing, and (ii) the adaptation of $\rho_i^j$.

The first phase follows a methodology similar to our previous work on EZ-flow [3], but with the advantage of not requiring two wireless interfaces. For each flow $i$, each node $j$ maintains a list of packet-identifiers (e.g., UDP or TCP checksums) of the last $P$ successfully transmitted packets (typically, $P = 100$). In addition, each node runs in promiscuous mode and attempts to overhear packets forwarded by the *next-hop* node $s_i(j)$. Whenever a forwarded packet is overheard, $j$ can use the packet-identifier list to compute an estimate of the occupancy $q_i^{s(j)}$ of the next-hop queue. Node $j$ simply counts how many packet identifiers have been added to the list since the identifier of the last overheard packet was added. This method assumes that the next-hop node uses the standard FIFO queuing policy.

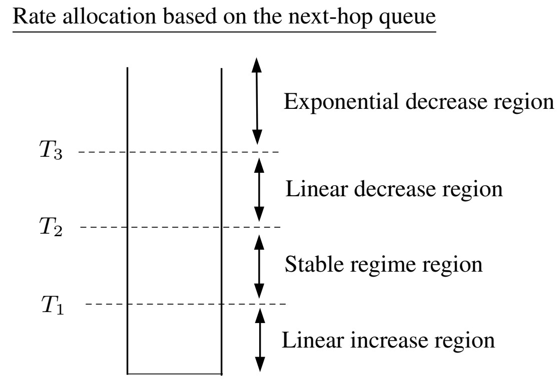


**Figure 5: Mechanism used by the network-layer congestion-control scheme to adapt the rate $\rho_i^j$ of the rate limiter based on the next-hop queue $\bar{q}_{s(i)}^j$.**

For each flow $i$ at node $j$, the second phase uses the estimates of $q_i^{s(j)}$ to adapt $\rho_i^j$. For a flow $i$, an update of the rate limiter parameter $\rho_i^j$ is performed every $R$ packets that are overheard from $s_i(j)$. Let $T_1 < T_2 < T_3$ denote queue thresholds and $\bar{q}_i^{s(j)}$ the per-flow time-averaged occupancy of $q_i^{s(j)}$ computed over the last $R$ overheard packets. Whenever an update occurs, *one* of these four cases takes place:

1. $\bar{q}_i^{s(j)} \leq T_1$**:** The queue at the next-hop is under-utilized and should be increased (positive expected drift). Thus, if the node has packets in its own queue, $\rho_i^j$ is linearly increased.

2. $T_1 < \bar{q}_i^{s(j)} < T_2$**:** The queue at the next-hop is neither empty nor overflowing. This is a desirable situation and $\rho_i^j$ should remain unchanged (zero expected drift).

3. $T_2 \leq \bar{q}_i^{s(j)} \leq T_3$**:** The queue at the next-hop builds up and $\rho_i^j$ should be decreased (negative expected drift). As $\bar{q}_i^{s(j)} \leq T_3$, a small decrease of $\rho_i^j$ might be enough to maintain a reasonable number of packets at node $i$: $\rho_i^j$ is linearly decreased.

4. $T_3 > \bar{q}_i^{s(j)}$**:** The next-hop queue is close to overflowing (e.g., due to a sudden environmental change) and $\rho_i^j$ should be quickly decreased to avoid packet losses (large negative expected drift): $\rho_i^j$ is multiplicatively decreased.

The role of $T_1$, $T_2$ and $T_3$ is to describe the number of packets that need to be maintained in the queues. They depend only on the buffer size of these queues, which are fixed and known in general.

Finally, in order to avoid the complete starvation of a flow, we do not allow $\rho_i^j$ to go below the minimal value of 1 packet per second. This is necessary for the nodes to estimate the next-hop queue occupancies at any time. The parameter $R$ represents a tradeoff between reactivity and stability: a large $R$ fits a highly stable environment and smoothes short-term variations. On the contrary, smaller $R$ values are better for highly time-varying environments that require a quick reactivity of the protocol. In our experiments, we set $R = 40$.

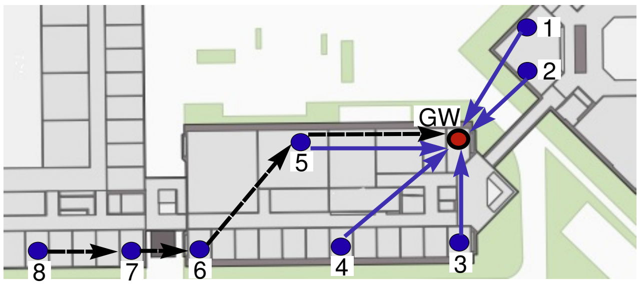


**Figure 6: Map of our $9$-node testbed that is composed of Asus WL-500gP wireless routers.**

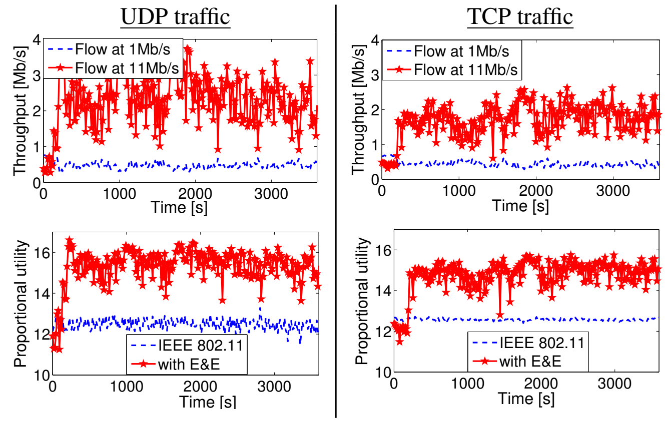


**Figure 7: Illustration of the gain in utility achieved by the E&E algorithm in the scenario of Figure 1. The top figures show the new throughput allocation achieved with E&E. The bottom figures show the achieved utility either with E&E or without E&E (i.e., IEEE 802.11 without rate limitation).**

## 4.3 Complete Solution: Congestion Control with Utility Maximization

We propose to couple the mechanism of Section 4.2 with the E&E algorithm in order to create a complete solution that performs congestion control and utility maximization *without* requiring network-wide message passing.

When coupling these mechanisms together, the rates $\vec{\rho}^j$ of the rate-limiters at node $j$ are set either: (i) remotely (via a broadcast message) by the gateway running E&E (if node $j$ is a one-hop node), or (ii) locally by the congestion-control mechanism based on the queue size of the next-hop (if node $j$ is more than 1 hop away from the gateway). Hence, when an allocation is enforced to the one-hop nodes by E&E it affects the dynamic of their queues. This change in queue occupancy is then detected by the upstream nodes running our passive congestion-control mechanism. As a result, these nodes will also adapt their rate-limiters and through this process the *congestion information* (artificially created by the rates set by E&E) propagates up to the source.

# 5. EXPERIMENTAL EVALUATION

In this section, we extensively evaluate our solution on real wireless networks using IEEE 802.11 nodes. We begin by studying in isolation the performance of both the E&E algorithm (in a WLAN setting) and the congestion control mechanism (in a multi-hop setting). We then evaluate both mechanisms together as a mean to improve the utility of WMNs when several flows are present in the system.

## 5.1 Hardware and Software Description

As depicted in Figure 6, we use 9 IEEE 802.11 nodes of the multi-hop testbed deployed on the EPFL campus [1]. Each node is an off-the-shelf Asus WL-500gP wireless router equipped with a single omni-directional antenna. Each router runs the version *8.09.2* of the openWRT firmware[4] with the *Click* modular router [14] used in user-level mode. We implemented our mechanisms in C code by creating five new *Click* elements that use the MultiFlowDispatcher [21] functionalities in order to create a new queue at run time only when the corresponding flow appears at a node. Additionally, we set the size of the MAC interface queue to 10 packets and the size of the per-flow buffers to 100 packets. We set the parameters of the intra-flow mechanism accordingly to maintain a small amount of packets in the per-flow queues. We use $T_1 = 20$ (a little larger than the MAC queue size to maintain some packets in the per-flow queue), $T_2 = 40$ and $T_3 = 80$ (close to the buffer size limit to avoid overflows). We use $\alpha = 5$, $T = 2$ and time slot durations of 5 seconds for the implementation of the E&E algorithm. The margin $\delta$ used to tolerate errors in the comparison of $\vec{\rho}[n-1]$ with the obtained throughput $\vec{x}[n-1]$ is set to 8 packets/second. In addition, we handle the time variability of the capacity region by testing the last stable assignment $\vec{r}[n]$ after 10 unsuccessful explore phases, to check if it remains feasible. If $\vec{r}[n]$ is not feasible anymore, the algorithm starts from the new allocation $\vec{r}[n] = \vec{x}[n-1]$. We also limit the possible loss of the control messages sent by the gateway by using a pseudo-broadcast packet (addressed to the neighbor with the weakest link and overheard by the other nodes), instead of a pure broadcast one. In Sections 5.2 and 5.4, we focus on the proportional fairness utility function $U_{prop}$ (given by Eq. 1). Some results using $U_{max}$ (given by Eq. 2) are presented in Section 6.

[4]http://openwrt.org/

## 5.2 WLAN Configuration

We evaluate experimentally the performance of E&E in the WLAN setting by looking at scenarios with respectively 2, 3 and 4 flows both for the case of UDP and TCP traffic. All node indexes correspond to the topology of Figure 6.

### 5.2.1 $2$-Flow Setting

This experiment corresponds to the setting introduced in Figure 1, where IEEE 802.11 suffers from the rate anomaly problem (i.e., the flow with the highest physical rate receives the smallest throughput). In our testbed the two flows are:

- $F_1$: from node 5 to the gateway (physical rate = 1 Mb/s);
- $F_2$: from node 3 to the gateway (physical rate = 11 Mb/s).

We illustrate in Figure 7 the throughput achieved by the system when the E&E algorithm is used and the corresponding gain in utility. We use the proportional utility in this scenario. From our 1-hour experiment, we note that, both for UDP and TCP traffic, the system starts from the throughput allocation achieved by IEEE 802.11 and rapidly solves the rate anomaly problem by correctly rate-limiting the flow with the smallest physical rate. This avoids the congestion collapse observed in Figure 1 and enables the faster flow to get a larger share of the airtime, thus significantly increasing its throughput. We see that thanks to proper rate-allocation,

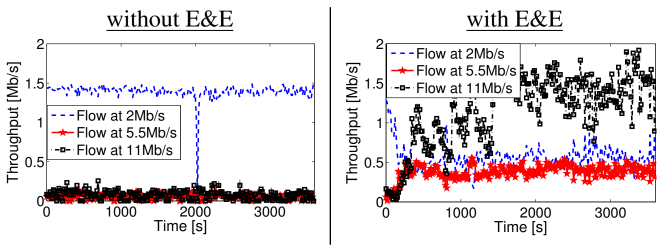


**Figure 8: Illustration of the temporal throughput evolution when $3$ single-hop UDP flows compete for channel access. The use of E&E allows for a significant gain in utility compared to IEEE 802.11 alone.**

| | Utility without E&E | Utility with E&E |
|---|---|---|
| $25^{th}$ percentile | 14.03 | 18.92 |
| median | 15.56 | 19.50 |
| $75^{th}$ percentile | 16.28 | 19.62 |

**Table 1: Proportional utility corresponding to the throughput depicted in Figure 8 for the $3$-flow scenario.**

E&E enables the system to increase its proportional utility function, roughly from 12 to 15. This increase might at first look small, but we remind the reader that the proportional fairness utility is expressed in log-scale. This increase from 12 to 15 is actually obtained by dividing the throughput of flow $F_1$ by a factor 2 and by multiplying the throughput of flow $F_2$ by a factor $4-7$ (hence multiplying their product by a factor more than two).

### 5.2.2 $3$-*Flow Setting*

In order to extend our study to a 3-flow scenario, we consider the following flows:

- $F_1$: from node 5 to the gateway (physical rate = 2 Mb/s);
- $F_2$: from node 1 to the gateway (physical rate = 5.5 Mb/s);
- $F_3$: from node 2 to the gateway (physical rate = 11 Mb/s).

We study this topology with fully saturated UDP traffic and we activate the RTS/CTS mechanism, because of the hidden-node configuration between node 5 and the other sources. We repeat our 1-hour experiments both for the case with the E&E algorithm and without E&E (i.e., IEEE 802.11).

Figure 8 shows that the rate anomaly problem occurs again with standard IEEE 802.11 in this setting, with the flow with the lowest physical rate ($F_1$) capturing almost all the airtime. We also note that the use of the E&E algorithm rapidly improves the performance by reaching a higher utility. The corresponding gain in utility (whose statistics are given in Table 1) is obtained by (i) dividing the throughput of $F_1$ by a factor 2, (ii) multiplying the throughput of $F_2$ by a factor 6, and (iii) multiplying the throughput of $F_3$ by a factor 13.

### 5.2.3 $4$-*Flow Setting*

Finally, we study a 4-flow WLAN scenario with the flows:

- $F_1$: from node 5 to the gateway (physical rate = 1 Mb/s);
- $F_2$: from node 4 to the gateway (physical rate = 5.5 Mb/s);
- $F_3$: from node 3 to the gateway (physical rate = 11 Mb/s);

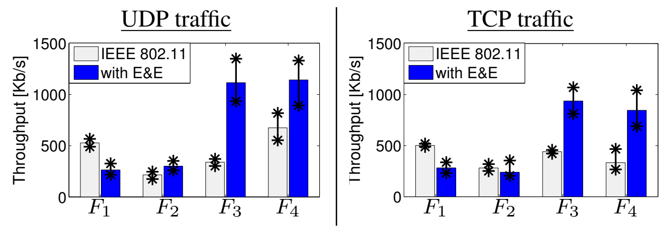


**Figure 9: Illustration of the median throughput (with its $25^{th}$ and $75^{th}$ percentile) achieved in a $4$-flow topology when the E&E algorithm is turned on or off. The left figure shows the effect on UDP traffic and the right figure shows the effect on TCP traffic.**

| | UDP w/o E&E | UDP with E&E |
|---|---|---|
| $25^{th}$ percentile | 23.59 | 25.16 |
| median | 23.93 | 25.34 |
| $75^{th}$ percentile | 24.27 | 25.43 |
| | **TCP w/o E&E** | **TCP with E&E** |
| $25^{th}$ percentile | 23.58 | 24.57 |
| median | 23.75 | 24.78 |
| $75^{th}$ percentile | 23.91 | 24.91 |

**Table 2: Proportional utility corresponding to the throughput depicted in Figure 9 for the $4$-flow scenario.**

- $F_4$: from node 2 to the gateway (physical rate = 11 Mb/s).

In this setting, node 2 is hidden from the other sources, we therefore activate the RTS mechanism to avoid packet collisions. Our results, from four 1-hour experiments, capture the effect of both TCP *vs.* UDP and E&E *vs.* IEEE 802.11. For readability purposes we depict our results as bar-plots in Figure 9, where we show for each flow its throughput statistics.

For the case of IEEE 802.11 (without E&E), our measurements show that the rate anomaly problem occurs again as the flow with the lowest physical rate ($F_1$) receives a large share of the throughput. Furthermore, it is interesting to note that the largest difference between UDP and TCP occurs for flows $F_3$ and $F_4$. Although both flows are set to the same physical rate, we see that with UDP traffic, flow $F_4$ achieves a higher throughput than $F_3$, but the opposite prevails for TCP traffic. This occurs because the source of $F_4$ (node 2) is hidden from the other sources. Therefore it senses the medium to be idle more often than the other sources (e.g., during the RTS transmissions from nodes 3, 4 and 5). As a result, it ends up (i) having more frequent access to the channel (because its backoff counter decrements faster) and (ii) having a larger probability of suffering from RTS collisions (because it cannot sense the concurrent transmission from the other nodes).

With E&E, we observe that the throughput of flow $F_1$ is decreased in favor of flows $F_3$ and $F_4$. From Table 2, we see that the rate allocation enforced by E&E leads to an increase in utility for both UDP and TCP traffic.

## 5.3 Congestion Control in WMN

Before considering the utility maximization problem in multi-hop networks, we first evaluate the efficiency of our intra-flow congestion control mechanism by considering one 4-hop flow ($F_4$) and one 5-hop flow ($F_5$). These scenarios are relevant because IEEE 802.11 is known to perform poorly and introduce much congestion

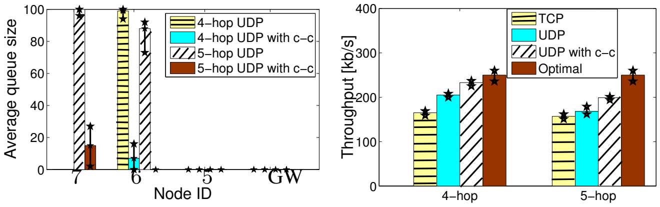


**Figure 10: Experimental results for a** $4$**-hop (**$F_4$**) and a** $5$**-hop flow (**$F_5$**). We show the effect of our intra-flow congestion control described in Section 4.2 (labeled "with c-c") on the queue size (left) and on the median end-to-end throughput (right) with the** $25^{th}$ **and** $75^{th}$ **percentile confidence intervals.**

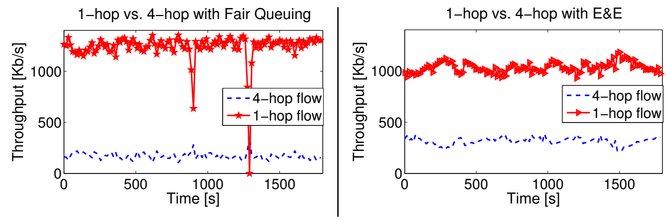


**Figure 12: Illustration of the throughput achieved when** $4$**-hop UDP flow competes with a** $1$**-hop flow with both flows having a separated last-hop (left). The E&E algorithm rate-limits the** $1$**-hop flow to deliver more throughput to the** $4$**-hop (right).**

in such configurations [2]. $F_4$ and $F_5$ consist in the following paths through the network:

- $F_4$: $7 \rightarrow 6 \rightarrow 5 \rightarrow GW \rightarrow 2$;
- $F_5$: $8 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow GW \rightarrow 2$.

Note that for the evaluation of our congestion-control mechanism, we study the performance of each flow when it transmits alone. In order to fit a 5-hop line in our testbed, the destination of both flows is node 2 and the node $GW$ is just a regular relay node. To assess the level of congestion, we look at the queue occupancies at the relay nodes (the source is fully-backlogged and its queue is full at all time). Additionally, we look at the end-to-end throughputs. We run each experiment for 90 minutes and every minute we measure the queue size and average throughput. Figure 10 shows the median value with the $25^{th}$ and $75^{th}$ percentiles.

We observe that, when UDP is used alone, the queue size of the first relay of flow $F_4$ (node 6) and of the first and second relays of flow $F_5$ (nodes 7 and 6) are very close to their size limit and often overflow. This creates packet losses, waste of efficiency, and high end-to-end delays.

In contrast, the intra-flow mechanism efficiently performs congestion control by keeping a small queue at every relay node. This translates into a significant reduction of end-to-end delays, because both the queuing delay and the traveling delay (i.e., higher end-to-end throughput) are reduced.

Other solutions exist to achieve congestion-control, the most common one is TCP. Although TCP may behave relatively well when only one flow is present, the end-to-end throughput is reduced due to the explicit ACK messages. Our mechanism performs congestion-control without suffering from this drawback. It improves the end-to-end throughput and maintains small queue sizes.

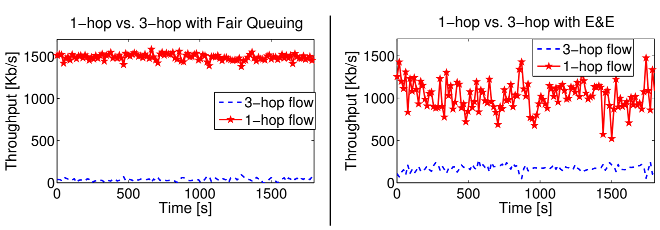


**Figure 11: Starvation problem for Fair Queuing when a** $3$**-hop UDP flow competes with a** $1$**-hop flow with both flows sharing the same last-hop (left). The E&E algorithm rate-limits the** $1$**-hop flow to prevent the starvation of the** $3$**-hop flow (right).**

In an attempt to assess how close to optimal the obtained throughputs are, we compute an *optimal* value as follows: we divide by 3 the capacity of the bottleneck link when transmitting in isolation (assuming a 2-hop interference model). This represents the best throughput that one could expect (i.e., it does not consider losses due to collisions).

We stress that our congestion-control scheme is not intended to be a replacement of TCP, as we do not focus on reliable in-order delivery. In fact, our results give us insight into studying mechanisms that decouple the goals of congestion-control and reliable in-order delivery. Indeed our intra-flow mechanism ensures that packet losses are, to a large extent, not due to buffer overflow (congestion) anymore.

## 5.4 Utility Maximization in WMN

### *5.4.1 3-Hop vs. 1-Hop*

We begin our evaluation of utility maximization in multi-hop networks by revisiting the starvation problem presented in Figure 4 by using the following concurrent flows.

- $F_1$: $5 \rightarrow GW$;
- $F_3$: $7 \rightarrow 6 \rightarrow 5 \rightarrow GW$.

In this setting, the physical rate of all nodes is set to 2 Mb/s and node 5 is the last-hop of both flows. Figure 11 shows the throughput achieved by each flow when Fair Queuing is used alone (left) and when the E&E algorithm is run at the gateway (right). Our results indicate that E&E increases the achieved the utility by (i) dividing the throughput of flow $F_1$ by a factor 1.5 and (ii) multiplying the throughput of flow $F_3$ by a factor 4.6.

### *5.4.2 4-Hop vs. 1-Hop*

Finally, we end our experimental study by considering the following concurrent flows.

- $F_1$: $3 \rightarrow GW$;
- $F_4$: $8 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow GW$.

In this setting the physical rate is again set to 2 Mb/s for all nodes, but this time the two flows do not have any node in common. Figure 12 shows the throughputs achieved when the E&E algorithm is turned off (left) and on (right). Interestingly, we see that the fact that the two flows go through disjoint paths improves the throughput of the multi-hop flow. Yet, the use of E&E further improves the utility performance by (i) dividing the throughput of flow $F_1$ by a factor 1.2 and (ii) multiplying the throughput of flow $F_4$ by a factor 1.8.

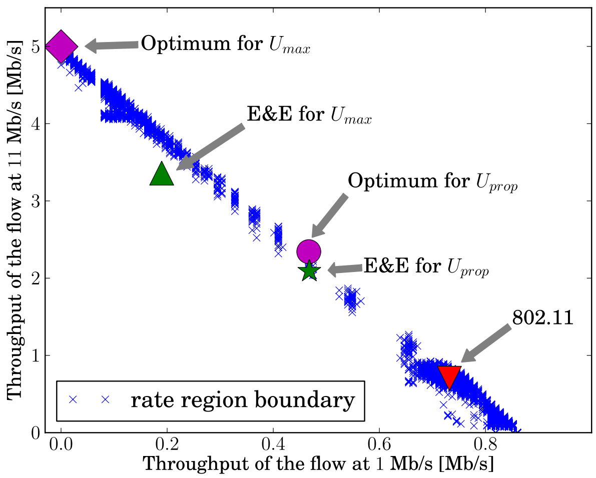


**Figure 13: Simulation-based measurements of the rate region, along with optimum for the two considered utility functions ($U_{prop}$ and $U_{max}$). Also shown are the average throughputs obtained by E&E and 802.11, respectively.**

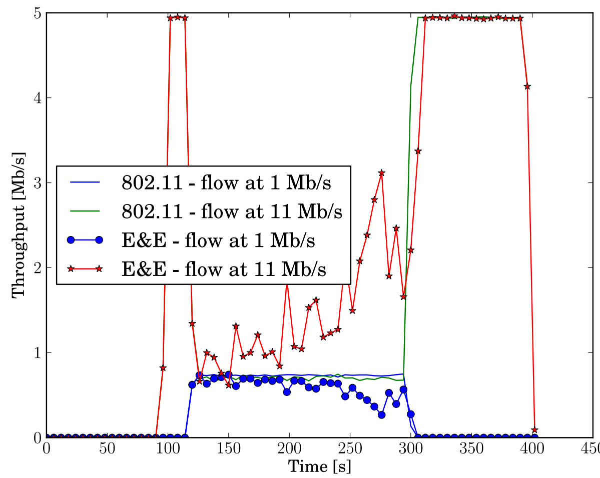


**Figure 14: Dynamic scenario. The flow at 1 Mb/s is started after the flow at 11 Mb/s (at 120 s) It is stopped at 300 s. While 802.11 gives the same throughputs to the two flows, E&E converges to a proportionally fair allocation.**

# 6. SIMULATION RESULTS

To corroborate our testbed measurements with results on a more controlled environment, we perform additional simulations. We focus on a single-hop scenario with two sources.

## 6.1 Simulation Settings

The simulations are performed with the ns-3 simulator. We use the exact same implementation of the E&E algorithm thanks to the ability of ns-3 to be used with the Click modular router [18]. Our topology is identical to the scenario of Figure 1, with one source with a physical rate of 1 Mb/s and one source with a physical rate of 11 Mb/s.

## 6.2 Results

In order to benchmark our scheme, we use the fact that the rate region $\Lambda$ is significantly less time-varying in simulation than in real deployments, making it possible to measure $\Lambda$. We empirically sample points on the boundary of the feasible rate region $\Lambda$ by using different combinations of source rates. From the rate region $\Lambda$, we find the allocations that provide the optimal utility for the two functions that we consider, namely proportional fairness ($U_{prop}$) and sum of the throughputs ($U_{max}$). Figure 13 depicts the measured capacity region $\Lambda$ and the computed optima. It also shows the average throughputs achieved when E&E is running (for 200 s), for the two utility functions. We see that E&E adapts extremely well to the chosen utility and that it manages to obtain average throughputs close to the optima.

The next experiment is designed to observe the reactivity of E&E when flows join or leave the system. We performed 10 runs with different random seeds, and the results turned out to be very similar. For one of these runs, Figure 14 shows the temporal evolution of the throughputs obtained by the flows when such events occur, both for E&E and IEEE 802.11. As expected, we see that IEEE 802.11 reacts immediately to a change in input rate. However, when the two flows transmit concurrently the performance is relatively poor as the system suffers from the rate anomaly problem. We note that contrary to the experimental results of Figure 1, the two flows obtain exactly the same throughput in our ns-3 simulation results. The explanation for this behavior is that ns-3 does not take into account the capture effect that significantly impacts the results in practice. When the 1 Mb/s flow joins the network, E&E starts from an allocation that is close to the one achieved by 802.11 and gradually improves the situation. When this flow leaves the network, E&E quickly recovers its previous allocation.

# 7. RELATED WORK

The problem of unfairness between flows occurs in both single-hop topologies, for example due to the rate anomaly problem [12], or in multi-hop settings [11] and a significant research effort has been devoted to each scenario.

Different approaches have been followed to tackle unfairness in single-hop networks (WLAN). For example, the Distributed Fair Scheduling (DFS) algorithm is proposed and evaluated with simulations [24]. As opposed to DFS, our approach does not modify IEEE 802.11 and our evaluation is conducted with both ns-3 simulations and testbed experiments. More complex WLAN scenarios have been studied such as the multi-AP WLAN setting [4] and the multi-hop WLAN case [8]. The evaluations of both algorithms show interesting performance. However, they are solely evaluated in simulations without real measurements.

Several practical solutions have also been proposed to address the unfairness and starvation problems in multi-hop networks. Recently, [15] followed a similar rate limiting approach. However, their method requires determining the optimal rate allocation offline using analytical models, and they provide a proof only for local convergence. In [9], the Inter-TAP Fairness Algorithm (IFA) achieves a fair allocation, but this solution requires a large amount of network-wide message passing (particularly in dynamic scenarios). Indeed, all nodes compute their offered load and capacity, and they transmit this information to all the other nodes in the network. In [11], the authors identify a starvation problem that occurs when a 2-hop TCP flow competes with one or more 1-hop

TCP flows. Their proposed counter-starvation policy consists in "*increasing the contention window of all the nodes directly connected to the gateway*". This policy definitively provides fairness improvements but it is topology-dependent and relies on TCP to deal with congestion (i.e, it does not directly apply to UDP traffic). Another approach for overcoming this unfairness of TCP in wireless networks is to replace TCP with another transport protocol. In [19], the WCP protocol is proposed as a replacement for TCP. Similarly, [20] introduces a rate-control protocol at the network layer, which estimates the network capacity and adapts the transmission rate accordingly to avoid congestion. Nevertheless, this end-to-end methodology is better suited for a static network than a dynamic one and hop-by-hop approaches have been shown to potentially provide better performance [28].

Hop-by-hop approaches for congestion control have been studied both theoretically [5, 10, 17, 22, 27, 29] and experimentally [3, 26]. The analytical approach is to guarantee that any rate in the capacity region is achievable by using variants of the MaxWeight scheduling algorithm [17,22]. It is important to point out that these approaches rely on three fundamental conditions: (i) the set of active flows in the network must be static and no flow can appear (or leave) [25], (ii) all sources should know a priori the capacity region and (iii) no source can transmit at a rate above capacity. These strong assumptions are one of the key reasons why these analytical approaches have not yet been implemented and tested experimentally. Our work fundamentally differs because we do not assume that the sources know the capacity region nor that they rate-limit themselves at the capacity of the network. Our mechanisms do not rely on any assumption about the source behavior and provide both congestion control and fairness in a distributed manner. There has been some recent efforts to combine throughput optimal scheduling with congestion control in a way that maximizes a given utility function [13]. However, their approach remains very theoretical and, in particular, relies on the assumption that the interference between the nodes is symmetric and well defined. We take a different approach, we use the fact that the gateway can naturally act as a central controller, in order to devise a measurement-based algorithm that maximizes the utility of the network without relying on any sort of interference model. Two practical approaches that exist for congestion control are DiffQ [26] and EZ-flow [3] and they both operate at the MAC layer. Our approach differs as we do not interact with any parameter of the MAC layer or of the upper layers.

# 8. CONCLUSION

We have presented and evaluated the *Enhance & Explore* algorithm (E&E). It is a practical and adaptive algorithm to maximize the utility of wireless networks. E&E consists in a succession of enhance and explore phases. These phases, respectively, (i) try to improve the utility of the system, and (ii) find allocations, within the unknown rate region, that achieve the targeted utility. We have formally proved that E&E converges to an optimal allocation when the rate region is fixed. When the rate region is time-varying, E&E keeps adapting the rate vector in the direction of the best allocation currently sustainable by the system. A notable feature of E&E is that it starts from the allocation given by the underlying MAC layer, and continues to improve the utility from this point. This ensures that, even if an optimal allocation might be reached only for long-lived flows whose duration exceeds the transient phase of the adaptive algorithm, it is not at the expense of short-lived flows. In addition, it incurs very small overhead (one message per time unit) and it is transparent to the rest of the networking stack.

To handle the case of multi-hop networks and propagate the rate allocations without incurring network-wide message passing, we have also proposed an extension of E&E that consists in a congestion-control scheme. This scheme is passive and only requires the nodes to listen to transmissions of their next-hop neighbors in order to infer their queue occupancy. These nodes then adapt their flow rates accordingly to prevent congestion.

We have evaluated our schemes on a testbed of IEEE 802.11 nodes and also using ns-3 simulations. The testbed measurements show that E&E manages to rate-limit the flows in a way that is neither too conservative, nor subject to congestion collapse. We considered two different utility functions, namely proportional fairness and the sum of the throughputs, in a variety of scenarios. In each case, E&E significantly increases the network utility. The simulation results show that E&E steadily converges to optimal rate allocations.

# Acknowledgement

We are very grateful to Cedric Westphal for his helpful comments on a preliminary version of this paper.

This work was supported (in part) by Deutsche Telekom Laboratories, in the framework of the BOWL project, and by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

# APPENDIX

We describe the SAMPLE_RANDOM_POINT($\vec{g}[n]$) procedure used by the E&E algorithm at line 31 to return a randomly chosen point in the level set $L(U(\vec{g}[n]))$. The main requirement for this procedure is that any non-empty subset of the (bounded) level set $L(U(\vec{g}[n]))$ has a non-zero Lebesgue measure. To this end, we propose an algorithm that returns a point having utility $U(\vec{g}[n])$ (hence, a point in $L(U(\vec{g}[n]))$), with $F-1$ coordinates being chosen according to a Gaussian distribution centered at the corresponding coordinates of $\vec{g}[n]$, and one coordinate chosen deterministically. The details of SAMPLE_RANDOM_POINT($\vec{g}[n]$) are presented in Algorithm 2. The procedure SAMPLE_GAUSSIAN$(a, b, \mu, \sigma)$ returns a value in the interval $[a, b]$ randomly picked according to a (truncated) Gaussian distribution with mean $\mu$ and standard deviation $\sigma$. Note that $\sigma$ is a parameter that allows to tune the extent with which the explore phases pick points that are far away from the last attempted gradient ascent $\vec{g}[n]$. In our implementation, we use $\sigma = \mu/3$.

**Algorithm 2** SAMPLE_RANDOM_POINT($\vec{g}[n]$)

pick $d$ uniformly in $\{1, \ldots, F\}$
**for** $i = 1$ to $F$ **do**
    **if** $i \neq d$ **then**
        $b \leftarrow u_i^{-1}\left(U(\vec{g}[n]) - \sum_{j \in \{1,\ldots,i-1\} \setminus d} u_j(p_j)\right)$
        $p_i \leftarrow$ SAMPLE_GAUSSIAN$(0, b, g_i, \sigma)$
    **end if**
**end for**
$p_d \leftarrow u_d^{-1}\left(U(\vec{g}[n]) - \sum_{j \in \{1,\ldots,F\} \setminus d} u_j(p_j)\right)$
**return** $\vec{p}$

# A. REFERENCES


[1] A. Aziz, A. E. Fawal, J.-Y. L. Boudec, and P. Thiran. Aziala-net: Deploying a scalable multi-hop wireless testbed platform for research purposes. In *Mobihoc Sˆ 3 2009*, http://icawww1.epfl.ch/aziala/.

[2] A. Aziz, D. Starobinski, and P. Thiran. Elucidating the instability of random access wireless mesh networks. In *Proc. of SECON*, 2009.

[3] A. Aziz, D. Starobinski, P. Thiran, and A. El Fawal. Ez-flow: Removing turbulence in ieee 802.11 wireless mesh networks without message passing. In *Proc. of CoNEXT*, Rome, Italy, Dec. 2009.

[4] Y. Bejerano, S.-J. Han, and L. Li. Fairness and load balancing in wireless lans using association control. *IEEE/ACM Transactions on Networking*, 15:560–573, June 2007.

[5] P. Chapokar, H. Kar, X. Luo, and S. Sarkar. Throughput and fairness guarantees through maximal scheduling in wireless networks. *IEEE Transactions on Information Theory*, 54(2):572–594, Feb. 2008.

[6] M. Chiang, S. Low, A. Calderbank, and J. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255 –312, jan. 2007.

[7] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Proc. of Sigcomm*, 1989.

[8] Q. Dong, S. Banerjee, and B. Liu. Throughput optimization and fair bandwidth allocation in multi-hop wireless lans. In *Infocom*, 2006.

[9] V. Gambiroza, B. Sadeghi, and E. Knightly. End-to-end performance and fairness in multihop wireless backhaul networks. In *Proceedings of ACM MobiCom*, Philadelphia, PA, Sept. 2004.

[10] A. Gupta, X. Lin, and R. Srikant. Low-complexity distributed scheduling algorithms for wireless networks. In *Infocom*, 2007.

[11] O. Gurewitz, V. Mancuso, J. Shi, and E. W. Knightly. Measurement and modeling of the origins of starvation of congestion-controlled flows in wireless mesh networks. *IEEE/ACM Transactions on Networking*, 17(6):1832–1845, 2009.

[12] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance anomaly of 802.11b. In *Proc. of Infocom*, 2003.

[13] L. Jiang, D. Shah, J. Shin, and J. Walrand. Distributed random access algorithm: Scheduling and congestion control. *Information Theory, IEEE Transactions on*, 56(12):6182 –6207, dec. 2010.

[14] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ACM Trans. Comput. Syst.*, 18(3):263–297, Aug 2000

.

[15] V. Mancuso, O. Gurewitz, A. Khattab, and E. W. Knightly. Elastic rate limiting for spatially biased wireless mesh networks. In *Proc. of Infocom*, pages 1720–1728, Piscataway, NJ, USA, 2010. IEEE Press.

[16] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *Networking, IEEE/ACM Transactions on*, 1(3):344–357, Jun 1993.

[17] A. Proutière, Y. Yi, and M. Chiang. Throughput of random access without message passing. In *Proc. of CISS*, Princeton, NJ, Mar. 2008.

[18] L. S. Puthalath and R. Merz. NS-3-Click: Click modular router integration for NS-3. In *Proceedings of the 3rd workshop on ns-3 (WNS3)*, Barcelona, Spain, March 2011.

[19] S. Rangwala, A. Jindal, K.-Y. Jang, and K. Psounis. Understanding congestion control in multi-hop wireless mesh networks. In *Proc. of MobiCom*, San Francisco, CA, May 2008.

[20] T. Salonidis, G. Sotiropoulos, R. Guerin, and R. Govindan. Online optimization of 802.11 mesh networks. In *CoNEXT*, 2009.

[21] H. Schiöberg and D. Levin. Multiflowdispatcher and TCPSpeaker. SyClick!, Nov. 2009.

[22] J. Shin, D. Shah, and S. Rajagopalan. Network adiabatic theorem: An efficient randomized protocol for contention resolution. In *Proc. of SIGMETRICS*, Seattle, WA, June 2009.

[23] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, Dec. 1992.

[24] N. Vaidya, A. Dugar, S. Gupta, and P. Bahl. Distributed fair scheduling in a wireless lan. *IEEE Trans. on Mobile Computing*, 4:616–629, Nov. 2005.

[25] P. van de Ven, S. Borst, and S. Shneer. Instability of maxweight scheduling algorithms. In *Proc. of INFOCOM*, Brazil, 2009.

[26] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee. Diffq: Practical differential backlog congestion control for wireless networks. In *Proc. of INFOCOM*, Rio de Janeiro, Brazil, 2009.

[27] Y. Yi, A. Proutière, and M. Chiang. Complexity in wireless scheduling: Impact and tradeoffs. In *Proc. of MobiHoc*, 2008.

[28] Y. Yi and S. Shakkottai. Hop-by-hop congestion control over a wireless multi-hop network. *IEEE/ACM Trans. on Net.*, 15(1):133–144, 2007.

[29] L. Ying, R. Srikant, and D. Towsley. Cluster-based back-pressure routing algorithm. In *Proc. of INFOCOM*, Phoenix, AZ, Apr. 2008.