Experimental measurement of the capacity region of wireless networks

Y. Thomas, N. Smyrnioudis, S. Toumpis
MMlab, Department of Informatics, Athens University of Economics and Business (AUEB), Greece

***Abstract*—We present a method for experimentally measuring the capacity region of a wireless network, defined here as the set of all possible combinations of simultaneously achievable link transmission rates. Due to the inherent complexity of the problem, measuring the capacity region exactly typically involves a prohibitively large volume of experiments. Therefore, the method is based on a judicious, tunable algorithm, employing online machine learning, which aims at reducing the volume of experiments conducted. In order to evaluate the efficiency of the method, and also to demonstrate the usefulness of having the capacity region available, we apply our method to a network of Raspberry Pi nodes placed inside a building and communicating using IEEE 802.11. In particular, we compare the performance of an optimal, centrally organized communication scheme, computed using the measured capacity region within a network optimization formulation, with the performance of a simple communication scheme that uses multihop TCP flows.**

***Index Terms*—Capacity Region, Online Machine Learning, Experiment, Interference, Measurement, Wireless Network.**

I. Introduction

Wireless networks have been studied for many decades now [1], but research interest in them remains significant; indeed, such networks are currently deployed or considered for deployment in various settings [2], [3], [4], [5], [6]. A fundamental problem in wireless networks, and a watershed that sets them apart from wired ones, is the fact that transmissions interfere with each other. The capacity region is a multidimensional set of data rates that captures this interference; our topic is its experimental evaluation.

There are various, closely related definitions of the capacity region, capturing difference nuances of the real-life problem, based, notably, on information theory [7], queuing theory [8], network optimization [9], [10], [11], and optimal control and scheduling [12], [13]. In information theoretic settings, the capacity region is typically defined under the implicit assumption that the channel is specified, but the complete protocol stack is subject to optimization; in others, notably network optimization settings, the transmission and media access schemes are also specified, and the capacity region captures the performance of the network when the data flow is optimized; this is the approach adopted here.

An important problem in this field is that the capacity region is inherently very hard to describe, due to the fact that, as the number of nodes and links in the network increases, the number of ways with which the network can operate increases very fast. Another problem is that for the network to operate optimally, i.e., make full use of the envelope provided by the capacity region, nodes typically need to coordinate centrally.

Despite these problems, knowing the capacity region is very useful, for two reasons: firstly, it is always useful to compare the performance of distributed protocols with the theoretical optimum, even if this cannot be practicably achieved, as this comparison gives an absolute gauge of the suboptimality of these protocols. Secondly, finding how the network operates optimally under central coordination often provides insights on how distributed protocols can be improved.

Unfortunately, to the best of our knowledge, work so far on the capacity regions of wireless networks has been theoretical in nature, employing analysis and simulation. Typically, a model is adopted for the capacity region, based, in turn, on channel and physical layer models, without attempting to evaluate its accuracy experimentally. This is regrettable, as both the wireless channel and the technologies used cannot be captured accurately by tractable analytical models, as, e.g., studies of the MAC layer have demonstrated [14].

Machine Learning (ML) has been used extensively in the context of wireless networks research [15], [16], [17], [18], [19]. Here, we use ML to estimate the throughputs achieved *jointly* by multiple simultaneously transmitting links.

In Section II we present a network optimization model that features the capacity region. In Section III we present our method for measuring the capacity region, by conducting a sequence of experiments. In Section IV we showcase the usefulness of measuring the capacity region and evaluate the efficiency of the method using a small experimental network. We conclude in Section V.

II. Network and capacity region model

Here, we introduce a simple network optimization framework in which the capacity region appears as a key parameter of a network optimization problem. We note that, excepting our definition of the capacity region, similar frameworks have been applied in the past (e.g., in [10], [11]).

A. The Wireless Network Utility Maximization Problem

We consider a set $\mathcal{N}$ of N **nodes**. Each of the nodes is able to send information directly to some of the rest through point-to-point wireless **links**. The link from node i to node $j \neq i$ is denoted by $l = (i, j)$, and we refer to i as its **transmitter** and j as its **receiver**. Let $\mathcal{L}$ be the set of links and L their number. Note that the links are directional, so, for any i, j with $i \neq j$, (i, j) and (j, i) may both exist.

The data traffic is described in terms of two types of optimization variables, the flows and the divergences. Firstly, let x_{ij} be the **flow** through link (i, j), measured in units of data volume over units of time. Let $x \in \mathbb{R}^L$ be the **flow vector**, of length L, comprising all flows. We require that x belongs to the **capacity region (CR)** $\mathcal{C} \subset \mathbb{R}_+^L$, i.e., an L-dimensional set of vectors with non-negative components that describes the capabilities of the links to jointly convey traffic. We present our model for the CR in Section II-B. Secondly, let

$$s_i = \sum_{j:(i,j)\in\mathcal{L}} x_{ij} - \sum_{j:(j,i)\in\mathcal{L}} x_{ji}, \quad i = 1, \ldots, N, \tag{1}$$

be the **divergence** at node i. The divergence expresses the rate with which traffic is inserted into the network by node i. Let $s \in \mathbb{R}^N$ be the **divergence vector** comprising all divergences. As a means of more succinctly describing (1), we define the **adjacency matrix** A to be a matrix of size $N \times L$ such that

$$A_{i,l} = \begin{cases} 1, & \text{if } i \text{ is the transmitter of } l, \\ -1, & \text{if } i \text{ is the receiver of } l, \\ 0, & \text{otherwise.} \end{cases}$$

With this definition, (1) is equivalent to $Ax = s$.

We require the divergence vector s to be bounded as follows: $L \leq s \leq U$. The two vectors $L, U \in \mathbb{R}^N$ specify which nodes can insert traffic in the network and which nodes can extract traffic from the network. Finally, the objective function, which must be maximized, is the **utility function** $U(x, s)$.

Putting everything together, the resulting problem becomes:

Wireless Network Utility Maximization (WNUM)

$$\begin{array}{ll} \text{maximize:} & U(x,s) \\ \text{subject to:} & x \in \mathcal{C}, \quad Ax = s, \quad L \leq s \leq U. \end{array}$$

A number of comments can be made at this point. Firstly, observe that there is no node mobility, as, indeed, the parameters of the problem do not change with time. Secondly, there is only a single commodity in the optimization problem, meaning that any piece of data created at any node may be delivered to any other node, subject to the existing capacity and divergence constraints. This assumption often applies in, e.g., sensor and cellular networks, but not always. Thirdly, we have assumed that there is a single common channel available for communication, as there is a single CR. We refrain from considering more general versions of the WNUM Problem along, e.g., the above-mentioned lines, as in this work we focus on measuring the CR and the WNUM Problem is a straightforward problem in which the CR appears in a nontrivial manner.

B. A model for the capacity region of the network

We call each subset $\mathcal{T} \subset \mathcal{L}$ of links a **transmission mode (TM)**. We refer to the number of links in a TM as the **size** of the TM. Let $\mathcal{P}(\mathcal{L})$ be the set of all TMs, i.e., the power set of $\mathcal{L}$. Note that the size of $\mathcal{P}(\mathcal{L})$, i.e., the total number of distinct TMs, is $M = 2^L$; indeed, to construct one of the TMs, each link may be added to that TM or not.

We call a link (i, j) **active** when node i is transmitting data for node j, and j is attempting to receive them. We say that a TM is **active** when the links of that TM are active and no other link is.

We associate each TM $\mathcal{T}$ with a corresponding **primal rate vector (PRV)** $R(\mathcal{T}) \geq 0$ which is an L-dimensional vector (each dimension corresponding to a distinct link in the network) comprised of the rates achieved by each of the links that belong to $\mathcal{T}$ when $\mathcal{T}$ is active, in the respective dimensions, and zeros in all other dimensions.

Next, let $\{R_1, \ldots, R_K\}$, with $K \leq M = 2^L$ be any subset of PRVs and let the set

$$\mathcal{D}(\{R_1, \ldots, R_K\}) = \Big\{x \in \mathbb{R}^L : 0 \leq x \leq \sum_{k=1}^K a_k R_k, \\ \sum_{k=1}^K a_k \leq 1, \quad a_k \geq 0, \quad k = 1, \ldots K\Big\}. \tag{2}$$

Therefore, a given vector $x \in \mathcal{D}(\{R_1, \ldots, R_K\})$ if and only if there is a K-dimensional vector $a = (a_1, \ldots, a_k)$ such that

$$0 \leq x \leq \sum_{k=1}^K a_k R_k, \quad \sum_{k=1}^K a_k \leq 1, \quad a \geq 0, \tag{3}$$

and so finding if $x \in \mathcal{D}(\{R_1, \ldots, R_K\})$ is a feasibility linear program.

What makes definition (2) useful are the following observations, whose proofs we omit due to space constraints: $\mathcal{D}(R_1, \ldots, R_K)$ is the set of all possible combinations of *average* rates with which links transport information, if the network is constrained to use the K TMs corresponding to the PRVs $R_1, \ldots, R_K$. Furthermore, any combination of rates $x \in \mathcal{D}(\{R_1, \ldots, R_K\})$ is achievable by a time division of the TMs corresponding to $R_1, \ldots, R_K$ and padding transmissions with random bits. For this reason, we call $\mathcal{D}(\{R_1, \ldots, R_K\})$ the **time division set** of $\{R_1, \ldots, R_K\}$; we call its elements **time division rate vectors** of $\{R_1, \ldots, R_K\}$ and we refer to both PRVs and time division rate vectors as **rate vectors**. We also refer to any vector a with non-negative components that sum to at most 1 as a **time division (vector)**. Finally, for simplicity, if we are given a set of TMs $\mathcal{T}_1, \ldots, \mathcal{T}_K$, then we denote $\mathcal{D}(\{R(\mathcal{T}_1), \ldots, R(\mathcal{T}_K)\})$ also as $\mathcal{D}(\{\mathcal{T}_1, \ldots, \mathcal{T}_K\})$.

We can now define the capacity region $\mathcal{C}$ of the network as the time division set of *all* PRVs $R_1, R_2, \ldots, R_M$:

$$\mathcal{C} = \mathcal{D}(\{R_1, \ldots, R_M\}).$$

In other words, the CR is the set of all possible combinations of average rates with which links transport data, without any constraint on the TMs used.

C. The Reduction Algorithm

Given a set of PRVs $\{R_1, \ldots, R_K\}$, we do not expect *all* of them to contribute to the time division set $\mathcal{D}(\{R_1, \ldots, R_K\})$. Rather, if the transfer of data by a particular TM can also be achieved by some time division of the other TMs, removing its PRV from the set $\{R_1, \ldots, R_K\}$ will not change the time

Algorithm 1: Reduction Algorithm

```
Input: Set of $K$ PRVs, $\mathcal{R} = \{R_1, \dots, R_K\}$.
Output: Basis $\mathcal{B} \subseteq \mathcal{R}$
1 $\mathcal{B} = \mathcal{R}$;
2 for $k \in 1, \dots, K$ do
3     if $R_k \in \mathcal{D}(\mathcal{B} - \{R_k\})$ then
4         $\mathcal{B} = \mathcal{B} - \{R_k\}$;
5     end
6 end
```

division set. Indeed, it is straightforward to prove that if $\mathcal{R} = \{R_1, \dots, R_K\}$ is any set of PRVs and for some k, $R_k \in \mathcal{D}(\mathcal{R} - \{R_k\})$, then $\mathcal{D}(\mathcal{R}) = \mathcal{D}(\mathcal{R} - \{R_k\})$, but we omit the proof due to space constraints.

This property motivates us to introduce the following **Reduction Algorithm**, which can be applied to any set of PRVs $\mathcal{R}$: we go through all PRVs in the set and, for each, we check if that PRV is a time division rate vector of the rest. If it is, it is removed from the set, and is not included in the remaining checks. We refer to the set of those PRVs that are kept as a **basis** (the concept should not be confused with the bases of vector spaces). The pseudocode for this algorithm is Algorithm 1.

As a side note, we conjecture that, provided that no two input PRVs are equal, there is only *one* basis of their set, in the sense that the Reduction Algorithm arrives at the same basis irrespective of the order with which the PRVs are checked for removal. We make no use of this conjecture later on.

By the above property, it follows that if we execute the Reduction Algorithm on the complete set of $M = 2^L$ PRVs, then we will find a basis of PRVs, $\mathcal{B}_E$, for which

$$\mathcal{C} = \mathcal{D}(\mathcal{B}_E).$$

In the following, we will refer to the members of $\mathcal{B}_E$, as well as their respective TMs, as **efficient** and to all other PRVs, and their respective TMs, as **inefficient**. Intuitively, efficient PRVs contribute to the CR, and so the network has use for them, whereas inefficient PRVs do not contribute to the CR, and could be ignored when solving the WNUM Problem.

III. Method for measuring the capacity region

A. Sequential Algorithm

In principle, to establish the CR we need to measure each of the PRVs $R_1, \dots, R_M$, through experimentation, and then optionally execute the Reduction Algorithm, in order to arrive at the smaller, more manageable number of PRVs that comprise the basis. However, as the number M of PRVs increases very fast with the number of links, the time needed to run these experiments is prohibitive for all but the smallest networks. Motivated by this observation, we present an algorithm that aims to establish a manageable number of PRVs whose time division set, though not equal to the CR, approximates it well.

We start with a definition: if $b \geq 0$, then we refer to a TM, as well as its corresponding PRV, as b**-strong** if

1) its links do not share any common node and
2) all its links transmit with a rate at least equal to b.

Intuitively, these two conditions jointly attempt to ensure that b-strong TMs do not contain active links that suffer from excessive levels of interference or contention for media access, and so are expected to be more useful than the rest.

Next, we specify our **Sequential Algorithm**, which aims at discovering as many b-strong TMs as possible, in a sequential manner. In the first step, we find, through exhaustive measurements, the set $\mathcal{P}(\mathcal{L})_{b,1}$ of all b-strong TMs of size 1, i.e., single links that can support a rate at least equal to b when they are not interfered with. In the second step, we first construct all pairs of these links and then keep all of these that are also found to be b-strong, first by removing those where there are nodes shared by two links, and then by performing experiments on those TMs that remain to ensure that all their link rates simultaneously exceed b. In this manner, we find a set $\mathcal{P}(\mathcal{L})_{b,2}$ of b-strong TMs of size 2. More generally, in the i-th step, we take all combinations of b-strong TMs added in step $i-1$ with TMs added in the first step, i.e., single, b-strong links, and we keep, first by removing those with nodes that are shared by two links and then by performing measurements, a set $\mathcal{P}(\mathcal{L})_{b,i}$ of TMs of size i that are also b-strong. We continue this process until either the $\lfloor N/2 \rfloor$-th step has been concluded, so we have ran out of suitable links to add to our expanding b-strong TMs, or no new TMs were found in a step $i < \lfloor N/2 \rfloor$.

The pseudocode for this algorithm is Algorithm 2. There, the Cartesian operation $\mathcal{P}(\mathcal{L})_{\text{check}} = \mathcal{P}(\mathcal{L})_{b,i} \times \mathcal{P}(\mathcal{L})_{b,1}$ of line 11 creates all distinct TMs of size $i+1$ comprised of a single link from $\mathcal{P}(\mathcal{L})_{b,1}$ and i links from $\mathcal{P}(\mathcal{L})_{b,i}$, such that no node participates in more than one link. The Boolean function ***b*-strong**$(\cdot)$ is true if we find, through actual measurements, that the TM $\mathcal{T}$ is b-strong, and false otherwise. If $\mathcal{T}$ is b-strong, this function also stores its corresponding PRV, for later use in solving the WNUM Problem.

Having executed the Sequential Algorithm, we arrive at a set of b-strong TMs,

$$\mathcal{P}(\mathcal{L})_b = \cup_{i=1}^{\lfloor N/2 \rfloor} \mathcal{P}(\mathcal{L})_{b,i},$$

and we approximate the CR with their time division set:

$$\mathcal{C}_b \triangleq \mathcal{D}(\mathcal{P}(\mathcal{L})_b). \tag{4}$$

Observe that a basic tradeoff exists: we anticipate that increasing the value of b reduces the number of b-strong vectors we have to discover experimentally. Therefore, our measurements will be less time-consuming (so that our method remains applicable at higher levels of mobility) and solving the WNUM Problem will be less computationally intensive. On the other hand, it also leads to a worse approximation, $\mathcal{C}_b$, for the CR and a suboptimal performance of the network, if we use it for solving the WNUM Problem instead of the exact CR $\mathcal{C}$.

Also observe that the algorithm is not *guaranteed* to arrive at a good approximation of the capacity region. In fact, it is not guaranteed to even find all b-strong TMs. Indeed, it is possible

Algorithm 2: Sequential Algorithm

```
Input: $b$,
       Network of $N$ nodes, comprising set $\mathcal{N}$, and
       $L$ links, comprising set $\mathcal{L}$.
Output: Sets of $b$-strong TMs of length $i$:
       $\mathcal{P}(\mathcal{L})_{b,i}$, $i = 1, \dots, \lfloor N/2 \rfloor$
/* Initialization: */
for $i = 1$ to $\lfloor N/2 \rfloor$ do
    $\mathcal{P}(\mathcal{L})_{b,i} = \emptyset$;
end
/* We find all $b$-strong links: */
for $l \in \mathcal{L}$ do
    if $b$-strong($\{l\}$) then
        $\mathcal{P}(\mathcal{L})_{b,1} = \mathcal{P}(\mathcal{L})_{b,1} \cup \{\{l\}\}$;
    end
end
$i = 1$; /* Main iteration */
while $i < \lfloor N/2 \rfloor$ and $\mathcal{P}(\mathcal{L})_{b,i} \neq \emptyset$ do
    $\mathcal{P}(\mathcal{L})_{\text{check}} = \mathcal{P}(\mathcal{L})_{b,i} \times \mathcal{P}(\mathcal{L})_{b,1}$;
    $i$++;
    for $\mathcal{T} \in \mathcal{P}(\mathcal{L})_{\text{check}}$ do
        if $b$-strong($\mathcal{T}$) then
            $\mathcal{P}(\mathcal{L})_{b,i} = \mathcal{P}(\mathcal{L})_{b,i} \cup \{\mathcal{T}\}$;
        end
    end
end
```

that a b-strong TM of K links exists such that if any one link is removed, the remaining $K-1$ links fail to create a b-strong TM. However, as we show next, it is possible to arrive at a performance guarantee for the Sequential Algorithm, assuming a few modest assumptions.

B. Performance guarantee of the Sequential Algorithm

First, we specify three assumptions.

Assumption 1: *TMs that have multiple links that share a common node are inefficient (i.e., prohibiting the network to use them does not reduce the CR).*

The assumption is motivated by the fact that if such a TM is used, the nodes involved in multiple links might prefer to perform an impromptu time division, which will not be better than time divisions achieved using the rest of the TMs. Therefore, we expect that PRVs that correspond to such TMs do not contribute to the CR. For some choices of the MAC and PHY layers, such as those of IEEE 802.11, this is a reasonable assumption. In some other cases, notably involving spread spectrum techniques, this might not hold. Observe that adopting this assumption significantly reduces the number of TMs that can potentially contribute to the CR.

Assumption 2: *TMs in which some links are active but have zero data rate, due to excessive interference, are inefficient.*

This assumption is very reasonable. Indeed, if we shut all unsuccessfully transmitting links, we expect that there is a time division that the other links of that TM can use to maintain the performance they were experiencing when the unsuccessfully transmitting links were competing with them.

Assumption 3: *Let any b-strong TM of K transmitting links. At least one of the K TMs that are created by removing one link from that TM is also b-strong and, furthermore, the TM created by that single link that was removed is also b-strong.*

Indeed, if a set of links can all be active at the same time with their rates all exceeding some value b, we expect that there should be at least one link that can be removed from that set without any of the others dropping below b, and that link, by itself, should also be able to transmit with rate at least equal to b.

In the case of the hardware used in our experimental setting of Section IV, we expect these three assumptions to hold. In fact, our experiments revealed deviations, as we discuss there.

Using these three assumptions, we can show that there is some $b > 0$ such that, indeed, our Sequential Algorithm discovers all efficient TMs, and so

$$\mathcal{C}_b = \mathcal{C}. \tag{5}$$

Indeed, first note that, by induction, Assumption 3 ensures that for all i, the set $\mathcal{P}(\mathcal{L})_{b,i}$ that the algorithm finds in the i-th step comprises *all* b-strong TMs of length i. Indeed, the first step (lines 4-8) discovers all b-strong TMs of length 1 by exhaustively going through all L TMs of length 1. To prove the inductive step, assume that we have found all b-strong TMs in step $i-1$, using lines 10-18. Let any b-strong TM of length i. Then, by Assumption 3, it can be divided in a b-strong TM of length $i-1$, and a b-strong TM of length 1. As both of them have been discovered in the previous and first step, respectively, the given TM will belong to $\mathcal{P}(\mathcal{L})_{\text{check}}$ and will also be discovered, when these two are combined, in one of the checks performed in line 14. Concluding, when Assumption 3 holds the Sequential Algorithm finds all b-strong vectors. Furthermore, by Assumptions 1 and 2, the basis comprises only TMs whose active links do not share nodes and transmit with positive rates. Since these TMs are finite in number, it follows that there is a $b > 0$ such that all of them are b-strong, and so are all discovered by the algorithm. Therefore, for that b, (5) holds.

C. Accelerating experiments with a machine learning module

Here, we show how the experiments prescribed by the Sequential Algorithm can be accelerated using an online ML module.

The basic idea is to modify the b-***strong***$(\cdot)$ routine that is invoked by the Sequential Algorithm at line 14, so that it first performs a tentative check, using past measurements, of the input TM $\mathcal{T}$ under investigation. If $\mathcal{T}$ has a very low probability of being b-strong, no experiments are executed. On the other hand, those experiments that are executed are used to train the module further.

Deciding if the input TM $\mathcal{T}$ is b-strong or not is a *classification* problem with two classes. However, rather than solving this classification problem directly, we solve it indirectly, by first solving the *regression* problem of estimating the RV of

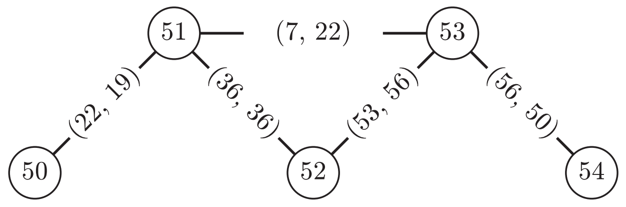

Fig. 1. The experimental network.

$\mathcal{T}$, and then deciding if $\mathcal{T}$ is b-strong or not, based on this estimation. The reason for this approach is that the regression approach can be applied in other related problems as well, and so is of independent interest.

Moving to the details of the module, first, regarding its state, the module stores, at any given time, all past TMs, $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$, for which experiments have been conducted, including those that did not turn out to be b-strong. For each $\mathcal{T}_k$ of these TMs, the module also stores the measured rate vector $R_k = (R_{k1}, \ldots, R_{kL})$, as well the estimate $\hat{R}_k = (\hat{R}_{k1}, \ldots, \hat{R}_{kL})$ of that rate vector that had been calculated, based on the previous $k-1$ measurements, right before the experiment for measuring R_k was executed.

Using the above information, the module selects its internal set of weights so that the following quantity is minimized:

$$\sum_{k=1}^{K}\sum_{\ell=1}^{L}(\bar{R}_{k\ell} - R_{k\ell})^2.$$

In the above, the vectors $\bar{R}_k = (\bar{R}_{k1}, \ldots, \bar{R}_{kL})$ are functions of the weights and are the best estimates of the corresponding rate vectors R_k, $k = 1, \ldots, K$. They are updated, together with the weights, whenever a new measurement is taken; they should not be confused with the aforementioned vectors $\hat{R}_k = (\hat{R}_{k1}, \ldots, \hat{R}_{kL})$, which are based on preceding measurements and are not updated once calculated. The minimization is subject to the intuitively clear constraint that an estimate $\bar{R}_k$ can have positive elements only in those dimensions that correspond to links that are active in the TM related to that estimate.

The module also keeps track of the following metric:

$$e_K = \begin{cases} \infty, & K < W, \\ \left(\frac{1}{N(K)} \sum\limits_{k=K-W+1}^{K} \sum\limits_{\ell=1}^{L} (\hat{R}_{k\ell} - R_{k\ell})^2 \right)^{\frac{1}{2}}, & K \geq W. \end{cases}$$

This metric is the sample standard deviation between the estimate of a link's data rate right before its measurement and its actual measurement, computed over a sliding window of the last W measurements preceding the $(K+1)$-th measurement. The quantity $N(K)$ is the number of all active links in the TMs included in the sliding window. If there have been fewer than W measurements, we deem all estimates inaccurate and, to indicate this, set e_K to infinity.

Whenever the ***b*-strong**$(\cdot)$ routine is invoked, the module is fed with a new candidate TM $\mathcal{T}_C$ and computes an estimate $\hat{R}_C = (\hat{R}_{C1}, \ldots, \hat{R}_{CL})$ of its rate vector. Based on this estimate, the module decides if the rate vector of that TM will be measured by experiment, or the estimate suggests such a suboptimal performance that it is safe to skip the experiment and promptly declare that $\mathcal{T}_C$ is not b-strong. In particular, let B be a marginally acceptable b-strong rate vector, i.e., a vector whose components are either zero (if they correspond to links not in $\mathcal{T}_C$) or b (for the rest of the components). Then, experiments are skipped when

$$\sum_{\ell=1}^{L} \max(B_\ell - \hat{R}_{C\ell}, 0) \geq Ge_K.$$

The sum expresses how much, in total, the estimated data rates should be increased so that the resulting rate vector marginally becomes b-strong; if it exceeds a multiple Ge_K of the standard deviation, e_K, experiments are skipped. The parameter $G > 0$ is chosen so that the proper tradeoff between the number of experiments that are executed and the number of b-strong TMs that are missed is achieved.

IV. Method evaluation

We now present a preliminary evaluation of our method based on measurements conducted on a testbed of 5 nodes.

A. Experimental testbed and measurements

Regarding the used hardware, each node comprises a Raspberry Pi 4 Model B with 4 GB of RAM equipped with a power bank and an external IEEE 802.11 card. The nodes form a wireless network using the internal IEEE 802.11 cards (with RTS/CTS handshakes deactivated, as is the default) in the 5 GHz band, but communicate with a central controller using the external IEEE 802.11 cards through an independent control channel in the 2.4 GHz band.

Regarding the network topology, the 5 nodes, with IDs 50 through 54, are located in successive floors of a multiple-floor residential building and can communicate using a total of 10 links, as shown in Fig. 1. In that figure, there are five lines connecting 5 node pairs, each line corresponding to two links. Each line is labeled with the two data rates of the respective links, when these links are not interfered with. The rate shown first in the label is the rate when the node closer to the beginning of the label transmits to the node closer to the end of the label. The rates are measured in Kbps and, in the figure, are rounded to integer values. We note that there were a few other links between other pairs of nodes, but these were weaker and more susceptible to competing transmissions, and so at this stage were disregarded, as we estimated that they would not have changed the CR, if included; formally, the CR computed is the CR in the network with only these 10 links available.

We conducted two sets of measurements. In the first set, we measured successively the data rates of single, fixed-duration multihop TCP flows for each of the 12 distinct source-destination pairs comprised of two nodes that do not have consecutive ids, using the best-performing route. We used the iperf [20] routine. These measurements were used to provide a baseline, as we discuss later on.

In the second set of measurements, we measured the PRVs of *all* TMs. Whenever a TM was tested, the states of all nodes were reset and concurrent single-hop TCP flows were initiated at each of its links, again using the iperf routine, for a fixed duration. Measurements were then collected at the central controller and stored for processing. As our network was comprised of 10 links, the total number of TMs was $2^{10} = 1024$, which was manageable. We used concurrent TCP flows with a duration of 60 seconds; together with the time intervals needed for switching, the duration of that part of the experiment lasted for approximately 18 hours.

There were two reasons for measuring all RVs beforehand. Firstly, we can emulate the execution of the Sequential Algorithm for any choice of the parameter b as well as the parameters of our ML model by running the algorithm and, whenever the RV of one of the 1023 non-empty TMs is needed, we retrieve it. This is the approach we adopted in the analysis of Section IV-B. Secondly, having all RVs available allows us to test the validity of the assumptions of Section III-B. We stress that in a practical setting measuring all RVs is neither practicable nor required by our method.

Regarding the quality of the measurements, we discovered that many of the measured throughputs were varying considerably with time, despite no mobility or apparent interference from other networks. We attribute this to the randomized manner with which IEEE 802.11 resolves contention.

Regarding the ML module implementation, we used the scikit-learn library [21] and the Python programming language. Scikit-learn is a library offering a wide variety of ML models and algorithms as well as different data processing utilities. We experimented with the multilayer perceptron (MLP), Random Forest and Linear Regression models for regression. In our investigations, we tuned the parameters G, W, the number of layers and the depth of the MLP by aiming to keep the $F1$ score of the predictions as large as possible. We note that

$$F1 = \frac{\mathrm{tp}}{\mathrm{tp} + \frac{1}{2}(\mathrm{fp} + \mathrm{fn})}$$

where tp is the number of true positive estimations (where a TM was deemed to be b-strong and indeed was), fp is the number of false positive estimations (where a TM was deemed to be b-strong but was actually not), and fn is the number of false negative estimations (where a TM was deemed to not be b-strong but actually was). For the results reported here we used $W = 300/b$, $G = 0.1$, and the Random Forest model with parameters such that for $b = 2$ Mbps the $F1$ is maximized.

Regarding the complexity of our online machine learning algorithm, we omit the details due to space constraints, however we briefly mention that it is dictated by the amount of training examples K that are used and the number L of links in the network. In particular, the amount of computations needed to train the model on a new measurement arrival is $\mathcal{O}(KL)$ for the case of the MLP, $\mathcal{O}(K^2L)$ for the case of the random forest, and $\mathcal{O}(L^2K + L^3)$ in the case of the linear regression.

B. Analysis of measurements

As a first comment, it is interesting to note that of the 1023 non-empty TMs, by executing the Reduction Algorithm only 19 were found to be efficient.

Regarding the extent to which the three assumptions of Section III-B hold, of the 19 efficient vectors 11 actually *had* links that shared nodes (thus violating Assumption 1) and 7 had active links that transmitted with 0 rate (thus violating Assumption 2). We attribute these findings to the very large number of measured TMs and the fact that measurements were not exact. However, as we show next, although the Sequential Algorithm fails to find these TMs, the resulting effects are limited. We also note that we found Assumption 3 to hold for all b-strong vectors, irrespective of the value of b.

To showcase the advantage of knowing the capacity region of the network, even approximately, we now use our measurements in order to compare two approaches for using the network in the scenario that one of the 12 source-destination pairs defined in Section IV-A must communicate.

The first approach is to discover the best route connecting the two nodes, and then set up a TCP connection. With this approach, using the results of the first set of experiments discussed in Section IV-A, it turns out that the sum of the throughputs over the 12 scenarios, rounded to an integer value, is 105 Mbps.

The second approach is first to compute an approximation of the capacity region, using our Sequential Algorithm of Section III-A, and then, for each of the 12 scenarios, solve a linear maximum-flow version of the WNUM problem of Section II-A in order to arrive at a centrally computed time-division scheme. In Fig. 2 we plot the sum of the 12 throughputs (as provided by a MATLAB solver) versus the value of b in three cases:

1) When the CR is found by the Sequential Algorithm without using the ML module (blue line).
2) When the CR is found by the Sequential Algorithm without using the ML module, but the constraint that b-strong TMs must have non-overlapping links is lifted (blue dashed line, on or very close to the blue line everywhere except for small values of b).
3) When the CR is constructed as follows: all 1023 nonempty TMs are sequentially checked by the ML module for having all their active links exceeding b and those accepted for experimentation are added in the CR irrespective of the outcome of the experiment, i.e., if all their active links indeed exceed b or not (red dash-dotted line).

Note that in the last case in particular, the number of PRVs comprising the CR equals the number of experiments conducted. For this case, we did not use the Sequential Algorithm out of necessity, in order to be able to feed the ML module with a volume of data sufficiently large for training and provide an indication of its performance in larger settings. We also draw the performance of the multihop TCP scheme

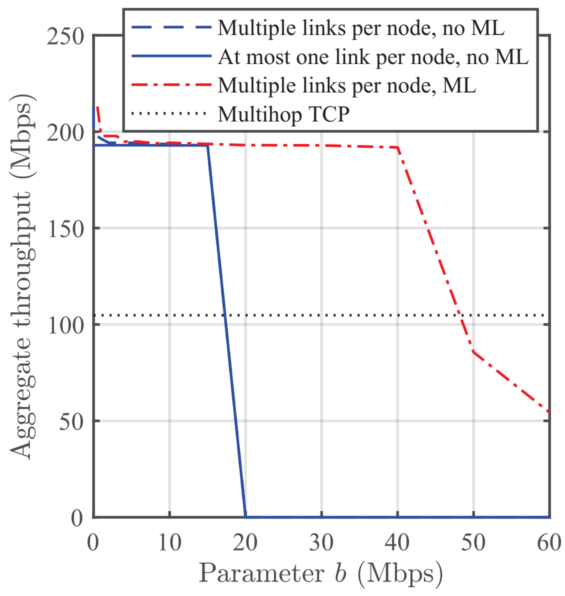

Fig. 2. Total throughput achieved versus the parameter b.

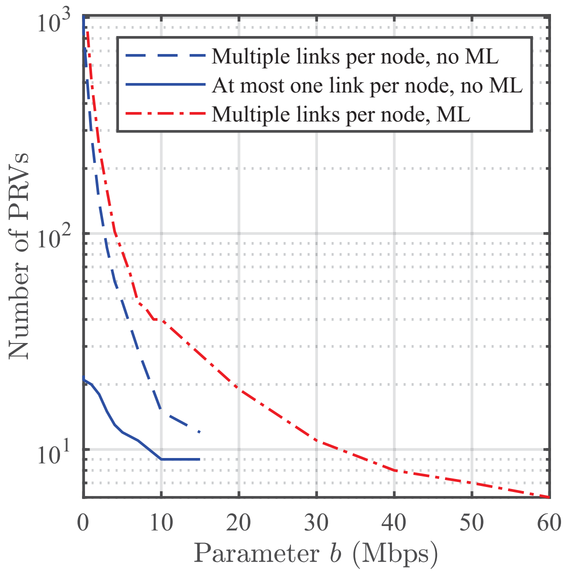

Fig. 3. Number of PRVs creating the approximate CR.

(horizontal black dotted line). The number of PRVs creating the approximate CR in the three cases is shown in Fig. 3.

The results of Figs. 2 and Fig. 3 are very encouraging.

Firstly, when the ML module is not enabled, despite the fact that the network can operate in 1023 distinct TMs, selecting any value for b in the reasonable range from 1 Mbps to 10 Mbps, and using only the b-strong TMs that are output by the Sequential Algorithm, it follows that the sum of throughputs remains greater than 192 Mbps. This is more than 90% of the absolute maximum sum of throughputs, achieved when all TMs are available, which is approximately 213 Mbps. However, the number of measured TMs for this range of b is very modest, as there is a total of 22 TMs with non-overlapping links. Furthermore, for all these values of b, the sum of throughputs is close to double the sum of throughputs achieved by multihop TCP. Therefore, by performing at most 22 measurements, a central optimization can almost double the performance of the network and bring it within 10% of the performance that would be achieved if 1023 measurements were conducted and so the complete CR was established.

Secondly, when the constraint that links must not overlap is lifted, the number of b-strong PRVs increases significantly, but the aggregate throughput does not change perceptibly. This results justifies the inclusion of this constraint in the definition of b-strong PRVs.

Thirdly, when the ML module is used, the tradeoff between the volume of measurements and the aggregate throughput is also excellent. Indeed, as long as $5 \text{ Mbps} < b < 40 \text{ Mbps}$, the specified algorithm achieves an aggregate throughput within 10% of the optimal with one to two orders of magnitude fewer measurements.

Finally, it is interesting to note that, even in this simple topology, the performance of multihop TCP is around 50% of the theoretical optimum. We are investigating this issue, and our current understanding is that the poor performance is due to the fact that multihop TCP fails to divide time efficiently between successive links of the same multihop route.

V. Conclusions

To the best of our knowledge, this is the first study to explore the capacity regions of wireless networks by actual measurements in a real-life testbed; previous work so far had been theoretical in nature, employing analysis and simulation. Notably, with respect to past works, we dispense with the need to model the interference and the operation of the MAC layer during the transmission of data and/or acknowledgment packets in terms of graph models, and depend on our measurements to accurately reveal their effects.

Our (very preliminary) results are encouraging, as they suggest that by measuring a small percentage of all TMs, we are able to determine the bulk of the CR. Furthermore, they reveal that the performance of the network can be improved significantly, with respect to plain TCP multihop routing, if its operation is prescribed by the solutions of the WNUM, which use the discovered CR. Developing practicable protocols along these lines is the subject of immediate future work.

On the other hand, our method has clear limitations. Notably, when there is high node mobility the CR cannot be discovered in time (we note, however, that if the mobility exhibits periodicity, multiple CRs, each corresponding to a different recurrent epoch, can be computed and employed in a suitably modified WNUM problem – this is the subject of future work). Also, using our method requires a central controller which will collect all measured PRVs, compile the traffic needs of the network, and solve the WNUM problem. Finally, our CR is calculated for the specific hardware used. Providing bounds for the gap between our CR and a more

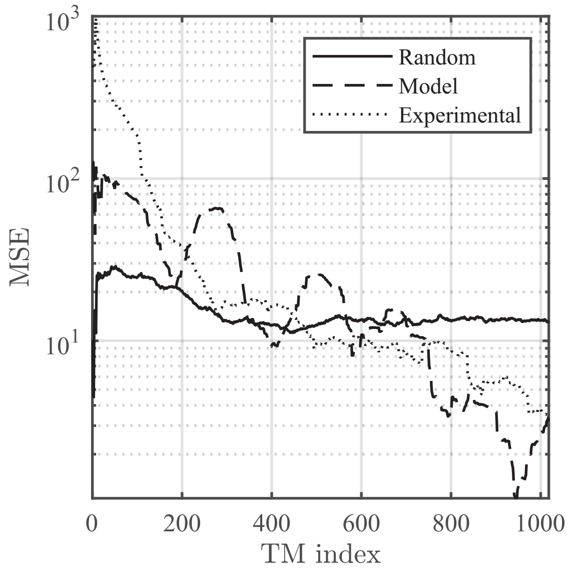

Fig. 4. Average MSE evolution for the three networks of Section V.

information-theoretic, hardware-independent CR could be the subject of future work.

Also as future work, we find the issue of the estimation of PRVs, irrespective of any CR work, to be of particular interest. For example, in Fig. 4 we present the results of the following experiment on three different networks:

1) Our experimental network.
2) A model network created using a simple model under which the rate at a receiver is an increasing function of its SINR, and such that the average value of PRV elements is the same as in the experimental network.
3) A random network, which has the PRVs of the experimental network, but with their positive values substituted by independent and uniform random numbers whose average value is the same is in the experimental network.

For each of these networks, we input the RVs sequentially into our ML module, in this case using an MLP of depth 5, train the network to minimize the mean square error (MSE) between the PRVs processed so far and their estimations, and measure, as a function of the input TM index, the active link MSE for new RV inputs, in units of $(\mathrm{Mbps})^2$, averaged over a window of size 100. It is interesting to note that the MSE of the experimental network exhibits a behavior much closer to that of the model network. Indeed, it is continuously reducing (as opposed to the random network, which flattens out after a few hundred measurements) as the MLP is able to glean structural information from the experimental network and train itself.

Acknowledgement

The research project was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "1st Call for H.F.R.I. Research Projects to support Faculty Members & Researchers and the Procurement of high-cost research equipment grant" (Project Number: HFRI-FM17-352, Project Title: Wireless Mobile Delay-Tolerant Network Analysis and Experimentation, Project acronym: LEMONADE).

References

[1] R. Kahn, "The organization of computer resources into a packet radio network," *IEEE Trans. on Communications*, vol. 25, no. 1, pp. 169–178, 1977.
[2] K. Thangaramya, K. Kulothungan, R. Logambigai, M. Selvi, S. Ganapathy, and A. Kannan, "Energy aware cluster and neuro-fuzzy based routing algorithm for wireless sensor networks in IoT," *Computer Networks*, vol. 151, pp. 211–223, 2019.
[3] D. Zhang, H. Ge, T. Zhang, Y.-Y. Cui, X. Liu, and G. Mao, "New multihop clustering algorithm for vehicular ad hoc networks," *IEEE Trans. on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1517–1530, 2018.
[4] B. Soret, S. Ravikanti, and P. Popovski, "Latency and timeliness in multi-hop satellite networks," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.
[5] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of things (IoT) communication protocols," in *2017 8th International conference on information technology (ICIT)*. IEEE, 2017, pp. 685–690.
[6] C. Huang, B. Zhai, A. Tang, and X. Wang, "Virtual mesh networking for achieving multi-hop D2D communications in 5G networks," *Ad Hoc Networks*, vol. 94, p. 101936, 2019.
[7] U. Pereg and Y. Steinberg, "The capacity region of the arbitrarily varying MAC: with and without constraints," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 445–449.
[8] M. Noori, S. Rahimian, and M. Ardakani, "Capacity region of aloha protocol for heterogeneous IoT networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8228–8236, 2019.
[9] S. Toumpis and A. J. Goldsmith, "Capacity regions for wireless ad hoc networks," *IEEE Trans. on Wireless Communications*, vol. 2, no. 4, pp. 736–748, 2003.
[10] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proceedings of the 11th annual international conference on Mobile computing and networking*, 2005, pp. 73–87.
[11] L. Liu, B. Yin, S. Zhang, X. Cao, and Y. Cheng, "Deep learning meets wireless network optimization: Identify critical links," *IEEE Trans. on Network Science and Engineering*, vol. 7, no. 1, pp. 167–180, 2020.
[12] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks," *IEEE/ACM Trans. on Networking*, vol. 17, no. 4, pp. 1132–1145, 2009.
[13] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with general traffic patterns: Capacity region and scheduling algorithms," *IEEE/ACM Trans. on Networking*, vol. 25, no. 6, pp. 3473–3486, 2017.
[14] K. Sanada, N. Komuro, Z. Li, T. Pei, Y.-J. Choi, and H. Sekiya, "Generalized analytical expressions for end-to-end throughput of IEEE 802.11 string-topology multi-hop networks," *Ad Hoc Networks*, vol. 70, pp. 135–148, 2018.
[15] A. Samba, Y. Busnel, A. Blanc, P. Dooze, and G. Simon, "Instantaneous throughput prediction in cellular networks: Which information is needed?" in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2017, pp. 624–627.
[16] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3039–3071, 2019.
[17] S. M. Srinivasan, T. Truong-Huu, and M. Gurusamy, "Machine learning-based link fault identification and localization in complex networks," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6556–6566, 2019.
[18] J. Zhang, R. Gardner, and I. Vukotic, "Anomaly detection in wide area network meshes using two machine learning algorithms," *Future Generation Computer Systems*, vol. 93, pp. 418–426, 2019.
[19] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1248–1261, 2019.
[20] https://iperf.fr.
[21] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.