Radio Resource Allocation with QoS Constraints in Energy Harvesting and Hybrid Power Systems

Jair A. de Carvalho, F. Rafael M. Lima, Tarcisio F. Maciel, and F. Rodrigo P. Cavalcanti

***Abstract*—In this paper we formulate the radio resource allocation problem of maximizing throughput in an OFDMA (Orthogonal Frequency Division Multiple Access) downlink where the BS (Base Station) adapts the power allocation according to non-causal (offline) knowledge of the harvested energy and channel state. The offline case is important from a theoretical point of view since it provides a bound on the performance of the online problem (causal). Differently from previous works that consider a continuous mapping between SNR (Signal-to-Noise Ratio) and transmit data rate, we employ a discrete mapping that depends on the required MCSs (Modulation and Coding Schemes). Also, we propose a heuristic algorithm that provides near-optimal results and achieves a good complexity/performance trade-off. In addition, we analyze the online version of the problem, and we propose two novel solutions to solve the problem only with causal information. Furthermore, we reformulate our problem to satisfy QoS (Quality of Service) constraints for each user in a hybrid power system where the BS is powered by a fixed power source from the electric grid and by a stochastic power source from renewable energy sources. Lastly, we present an offline solution to this new problem that also achieves a considerable complexity/performance gain.**

***Index Terms*—Resource allocation, rate maximization, energy harvesting, OFDMA, MCS, hybrid power system, QoS.**

I. Introduction

Energy Harvesting (EH) communications are powered by renewable energy sources and can enjoy an unlimited lifetime [2]. However, such energy sources present a stochastic nature since it depends on environmental conditions that determine the amount of power available to perform transmissions. Differently from communication devices that work with fixed power supplies and have assurance of power availability, EH devices save any unused energy in a storage component such as a rechargeable battery. Then, because batteries have finite storage capacity, we subject our problem to the battery capacity constraints that limit the available energy to the maximum value supported by the battery. Moreover, we apply the energy consumption causality constraints that limit the used energy to the quantity available at the moment, despite the knowledge of future energy arrivals in the offline case. Both restrictions are known as energy harvesting constraints, and are present in several works [2]–[5].

Jair A. de Carvalho and F. Rafael M. Lima are with Federal University of Ceará (UFC), Campus Sobral, Brazil. F. Rafael M. Lima, Tarcisio F. Maciel, and F. Rodrigo P. Cavalcanti are with Wireless Communications Research Group (GTEL), UFC, Fortaleza. Emails: jair.alves.carvalho@gmail.com, {rafaelm, maciel, rodrigo}@gtel.ufc.br.

The authors wish to thank FUNCAP and the Government of the State of Ceará for the financial support.

A preliminary version of this paper was presented in XXXV Simpósio Brasileiro de Telecomunicações (SBrT'17), São Pedro, Brazil, Sep. 2017 [1].

Digital Object Identifier: 10.14209/jcis.2018.20

Many options of EH sources are available, for example: solar radiation, natural wind, radio frequency waves, vibration and thermal energy [2], [3]. Hence, in our study we considered a BS powered by a photovoltaic system similar to [6]. Also, our EH model follows a first-order stationary Markov chain, supported by studies on solar energy in [6]–[8]. The main advantages of EH systems include long term operation without stable power supplies, decreased need of cabling and component replacements, and consequently smaller cost per project. Motivated by those advantages we investigate EH scenarios related to wireless communications.

More specifically, our problem is in an offline setting, that means previous knowledge of the EH profile and full CSI (Channel State Information) at the BS. Even though previous knowledge of channel gain and harvested energy is very hard to obtain in practice, solutions for offline scenarios serve as performance benchmarks to more realistic situations and offer important insights on the development of solutions to the online scenario, where energy arrivals and channel gains are not known beforehand. Furthermore, RRA (Radio Resource Allocation) is an essential functionality in order to improve the use of the scarce radio resources such as power and spectrum when devices are powered by energy harvesters [3], since the main purpose of RRA consists in intelligently distribute the available resources on the goal to satisfy requirements or to optimize services. In this article we consider the RRA problem of maximizing the system throughput that is a typical and important problem studied in the literature [2]–[5].

We extend our analysis to the online case in order to study our problem in practical scenarios. In this case, all available information is causal and we have the difficulty to allocate power without exact knowledge of the next energy supplies. We also deepen our investigations by adding the hardship of ensuring QoS requirements per user. Nevertheless, the uncertain amount of energy provided by EH systems may be insufficient to fulfill the demands of each user [9]. Therefore, we add a non-renewable energy source to complement the energy provision. This extra supply is provided by the electric grid, as in [10]. Thus, we can overcome the limitations of energy availability in EH systems when ensuring QoS requirements.

This paper is divided as follows: section II shows a literature review on EH wireless communications and presents our main contributions; section III describes the system modeling for the problems developed in the next sections; section IV defines the data rate maximization problem in offline scenarios, proposes a heuristic solution and evaluates its performance in comparison to the optimal solution; section V analyzes the problem in section IV for online scenarios, proposes two online solutions

and compares the results for offline and online cases; section VI defines the rate maximization problem with hybrid energy source and QoS constraints, describes a heuristic algorithm to solve the problem and presents results for optimal and heuristic solutions; finally, section VII presents our conclusions.

II. State of the Art and Contributions

Generally, cooperative communications and relay networks have been studied in literature, and specially in EH scenarios [2], [11], [12]. The authors in [2] analyze a source, relay, destination network in offline (previous knowledge of EH profile) and online (only casual information) situations, and propose a solution for the online case through convex optimization. The studies in [11], [12] present, respectively, optimal and heuristic solutions for power allocation problems with relay selection. Moreover, OFDMA systems and EH technology are the main topics in [13], where the authors develop resource allocation algorithms for a hybrid EH base station, powered by fixed and renewable sources. Similarly, [14] introduces a self-sustainable approach for OFDM receivers in the context of green networks. In [9], the authors propose a solution that maximizes the number of users with satisfied QoS in a EH system. And [10] presents an algorithm that balances system throughput and grid energy consumption in OFDMA wireless networks with hybrid energy supplies.

Long-established studies in [3], [5] give the foundations of rate maximization in EH systems, and more recent work in [15], [16] consider cognitive radio networks (CRNs) powered by energy harvesters, where [15] optimize spectral efficiency and [16] focus on secrecy performance. Finally, the paper in [17] investigates RF (Radio Frequency) EH for mobile devices in cloud-based networks. Though the aforementioned works provide relevant contributions, all these studies support the unrealistic assumption of continuous mapping between SNR and transmit data rate, but, in real systems, the set of possible transmit data rate values is discrete and finite. Therefore, we implement a discrete mapping through MCSs, where the achievable transmit rates are determined by modulation levels and channel coding. Also, unlike the authors in [2]–[5] and [9]–[17], who employ a continuous mapping due to the ease of using convex optimization methods, our problem becomes entirely combinatorial with the use of discrete MCSs, which increases the computational complexity to obtain the optimal solution.

In addition, the papers in [2]–[4] assume an EH profile with discrete values from a finite set. However, empirical measurements in [6]–[8] demonstrate that the harvested energy assume continuous values in practice. The study in [7] proposes the design of a quantizer in order to work with discrete values of harvested energy. Such quantization process is acceptable because some harvesters collect an approximately constant amount of energy throughout the time [7], what justifies the use of low order quantizers. Since our problem is combinatorial, we prefer to enhance diversity by adopting an EH model that accepts continuous values over a finite range, thus, providing a more precise model because other approaches are prone to quantization error. Lastly, as far we know, the problem of rate maximization for multi-carrier and multi-user systems in EH communications with discrete MCSs has not been studied in the literature. Our contributions can be summarized to:

- Mathematical formulation of resource allocation problem with realistic model for link adaptation (discrete MCSs);
- More precise EH model based on Markov processes with states represented by continuous intervals;
- Heuristic solution (offline case) that provides close-to-optimal results at low computational cost;
- Online solutions that consider an empirical and a general Markov model to predict the harvested energy;
- New problem formulation that mathematically describes a hybrid power system with QoS constraints;
- Offline solution for the hybrid power system problem that achieves a considerable complexity/performance gain.

III. System Modeling

Our scenario consists of a BS located in the center of a circular cell of radius R that transmits to J users. Moreover, the system has a total of N OFDM subcarriers, and allows M possible MCS levels per subcarrier. In our problem, information is transfered through T transmission time intervals (TTIs), and for each TTI, the BS harvests H_i units of energy at TTI i. Assuming that γ_m is the needed SNR to achieve the m-th MCS level, and that $\alpha_{i,n,j}$ denotes the channel gain between user j and the BS on subcarrier n for the i-th TTI, we define

$$p^{\mathrm{r}}_{i,n,j,m} = \frac{\sigma^2 \gamma_m}{\alpha_{i,n,j}}, \tag{1}$$

as the required power to the BS transmit to user j on subcarrier n at TTI i with MCS m, where the superscript "r" stands for "required". Also, note that σ^2 is the noise power at the receiver in the bandwidth of a subcarrier. Other important variables are the maximum energy storage capacity of the battery, $B_{\max}$, and the transmit data rate of the m-th MCS level, r_m, necessary to define the discrete mapping $\gamma_m \longleftrightarrow r_m$ for the link adaptation, where $\gamma_m < \gamma_{m+1}$ and $r_m < r_{m+1}$.

On the matter of harvest dynamics we consider a Markov model for solar radiation, and according to [7], [8], we can assume a first-order stationary Markov chain. Basically, stationarity means that the states and transition probabilities do not vary over time. By first-order process we mean that the current state is the only and direct cause of the next one. These assumptions simplify the model, since we have a memoryless system with only one matrix of transition probabilities that works for any instant in time. This is not the case for wind energy [7] that needs a generalized Markov model.

Firstly, let S be the number of states in our model, and $\mathbf{P}$ the square matrix of transition probabilities of order S. Then, we define $P_{s,k}$ as the probability of transition from state s to state k. We also define the vector $\boldsymbol{v} = [v_1, \ldots, v_S]$ where v_k represents the probability of finding the system in state k after a large number of transitions, which is called marginal probability. Finally, following a model similar to [8], the harvested energy H_i is a random number from an uniform distribution belonging to a continuous interval

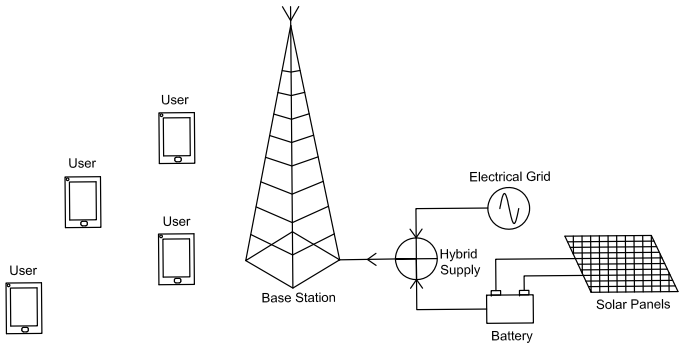

Fig. 1. BS in operation powered by a hybrid power system.

$[(k-1)(H_{\max}/S),\ k(H_{\max}/S)]$ for $k = 1, \ldots, S$, where $H_{\max}$ is the maximum energy that can be harvested, and each interval represents the S possible states of the Markov chain. This representation of states as continuous intervals simulates the actual behavior of the harvested energy, that presents a continuous nature in practice, thus, being an improvement over the studies in [2]–[4] that consider discrete values of energy. Figure 1 presents the scenario described in section VI: users served by a BS powered by hybrid energy supplies. In sections IV and V the BS is powered only by renewable energy sources.

IV. Data Rate Maximization with Renewable Energy Source - Offline Case

In this section, we study a radio resource allocation problem for rate maximization at the scenario described in section III. In this case, the BS is powered by a renewable energy source (solar radiation) and information regarding energy arrivals and channel gains is known non-causally (offline).

A. Problem Formulation and Optimal Solution

Before showing the problem formulation, we define $x_{i,n,j,m}$ as the binary decision variable that assumes 1 when subcarrier n with MCS m is assigned to user j at TTI i. Furthermore, our goal is to determine $p^{\mathrm{a}}_{i,n}$, that is the allocated power to subcarrier n at TTI i, and the subcarrier assignment $n \longleftrightarrow j$, with $\mathcal{T} = \{1, \ldots, T\}$, $\mathcal{N} = \{1, \ldots, N\}$, $\mathcal{J} = \{1, \ldots, J\}$ and $\mathcal{M} = \{1, \ldots, M\}$ being the set of all TTIs, subcarriers, users, and MCS levels, respectively, where $i \in \mathcal{T}$, $n \in \mathcal{N}$, $j \in \mathcal{J}$ and $m \in \mathcal{M}$. The superscript "a" in variable $p^{\mathrm{a}}_{i,n}$ stands for "allocated". The total data rate from T TTIs is given by

$$r_{\text{total}} = \sum_{i=1}^{T}\sum_{n=1}^{N}\sum_{j=1}^{J}\sum_{m=1}^{M} r_m x_{i,n,j,m}. \tag{2}$$

Since, for a certain TTI i, a subcarrier n can assume only one MCS m and be assigned to a single user j, we write this restriction as

$$\sum_{j=1}^{J}\sum_{m=1}^{M} x_{i,n,j,m} = 1,\ \forall i,\ \forall n. \tag{3}$$

Once we determined a feasible configuration for $x_{i,n,j,m}$, the power allocated to subcarrier n at TTI i can be written as

$$p^{\mathrm{a}}_{i,n} = \sum_{j=1}^{J}\sum_{m=1}^{M} p^{\mathrm{r}}_{i,n,j,m} x_{i,n,j,m},\ \forall i,\ \forall n, \tag{4}$$

and we can finally define the energy consumption causality constraints given by

$$\sum_{i=1}^{t}\sum_{n=1}^{N} p^{\mathrm{a}}_{i,n} \leq \frac{1}{\tau}\sum_{i=1}^{t} H_i,\ t = 1, \ldots, T. \tag{5}$$

The constraints in equation (5) mean that the BS cannot use energy packets which are yet to arrive, and that the used transmit power at TTI t cannot exceed the harvested power until that TTI. The constant τ consists in a factor to convert energy in/from power, i.e., the corresponding energy is obtained by multiplying τ by the power. Lastly, we need to consider that the harvested energy is being stored in the battery, and that a remaining amount of energy is always left over. In many situations, a considerable quantity of energy will be saved for next transmissions, and the next energy packet added to this saving can exceed the battery capacity. In order to assure that energy will not be wasted we define

$$\sum_{i=1}^{t+1} H_i - \tau\sum_{i=1}^{t}\sum_{n=1}^{N} p^{\mathrm{a}}_{i,n} \leq B_{\max},\ t = 1, \ldots, T-1, \tag{6}$$

as the battery capacity constraints. Now, we have to maximize the system throughput by solving the optimization problem:

$$\max_{\boldsymbol{x}} \sum_{i=1}^{T}\sum_{n=1}^{N}\sum_{j=1}^{J}\sum_{m=1}^{M} r_m x_{i,n,j,m}, \tag{7}$$

subject to

$$\text{Equations (3), (5) and (6),} \tag{8a}$$

$$x_{i,n,j,m} \in \{0, 1\},\ \forall i,\ \forall n,\ \forall j,\ \forall m. \tag{8b}$$

The problem described above is an ILP (Integer Linear Programming) mathematical optimization, that in general is an NP-hard problem. Algorithms based on BB (Branch and Bound) methods provide optimal results at the cost of high computational complexity, specially when the number of constraints and variables is increased. Hence, we obtained the optimal solution through the IBM ILOG CPLEX solver [18]. Due to the high computational complexity to obtain the optimal solution, we propose in the next section a low-complexity algorithm to solve the problem.

B. Heuristic Solution

A classical and optimal bit loading algorithm for point-to-point links with deterministic power supply is the HH (Hughes Hartogs) solution [19]. The main idea of this algorithm is to raise the MCS level of the subcarriers that need less power to reach the next level. This procedure is repeated in iterative manner while there is available transmit power or until all subcarriers achieve the maximum MCS level. Naturally, the HH algorithm could not be applied in an EH system since using all the available power on the battery at TTI i, represented by b_i, is not the best option. This is true because in EH systems we save energy to prepare for a poor harvest in the future. In this section, we propose a solution that limits to l_i the maximum used power per TTI i, and instead of starting to load the subcarriers that require the least power, the heuristic begins loading the subcarriers with smaller cost per bit, defined

Algorithm 1 HH-Based Heuristic Power Allocation

1: Calculate $p^{\mathrm{r}}_{i,n,j,m}$ using equation (1)
2: $q^{\mathrm{a}}_{i,n} = \mathbf{min}(p^{\mathrm{r}}_{i,n,j,M} : \forall j \in \mathcal{J}),\ \forall i,\ \forall n$
3: $u_{i,n} = \mathbf{argmin}(p^{\mathrm{r}}_{i,n,j,M} : \forall j \in \mathcal{J}),\ \forall i,\ \forall n$
4: $p^{\mathrm{s}}_{i,n,m} = p^{\mathrm{r}}_{i,n,j,m},\ \forall i,\ \forall n,\ \forall m,\ \text{for } j = u_{i,n}$
5: **for** $i \leftarrow 1$ **to** T **do**
6: **for** $n \leftarrow 1$ **to** N **do**
7: $m = M$
8: **while** $q^{\mathrm{a}}_{i,n} > \frac{1}{N}\sum_{k=1}^{N} q^{\mathrm{a}}_{i,k}$ **do**
9: $m = m - 1$
10: $q^{\mathrm{a}}_{i,n} = p^{\mathrm{s}}_{i,n,m}$
11: **end while**
12: $\lambda_{i,n} = m$
13: **end for**
14: **end for**
15: $q_i = \sum_{n=1}^{N} q^{\mathrm{a}}_{i,n},\ \forall i$
16: *AdjustReq*$(\boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{p}^{\mathbf{s}}, \boldsymbol{q}^{\mathbf{a}})$
17: $\Delta r_{m-1} = r_m - r_{m-1},\ \text{for } m = 2, \ldots, M$
18: $\Delta p^{\mathrm{s}}_{i,n,m-1} = p^{\mathrm{s}}_{i,n,m} - p^{\mathrm{s}}_{i,n,m-1},\ \forall i,\ \forall n,\ \text{for } m = 2, \ldots, M$
19: $c_{i,n,m} = \Delta p^{\mathrm{s}}_{i,n,m} / \Delta r_m,\ \forall i,\ \forall n,\ \text{for } m = 1, \ldots, M-1$
20: $c_{i,n,M} = \infty,\ \forall i,\ \forall n \qquad \Delta p^{\mathrm{s}}_{i,n,M} = \infty,\ \forall i,\ \forall n$
21: $b_1 = h_1 + b_0 \qquad p^{\mathrm{a}}_{i,n} = 0,\ \forall i,\ \forall n$
22: $\lambda_{i,n} = 1,\ \forall i,\ \forall n \qquad \lambda_{i,1} = 0,\ \forall i$
23: **for** $i \leftarrow 1$ **to** T **do**
24: $d_i = 0$
25: **if** $i < T$ **then**
26: $d_i = (b_i \sum_{t=i+1}^{T} q_t - q_i \sum_{t=i+1}^{T} h_t) / \sum_{t=i}^{T} q_t$
27: **if** $d_i < 0$ **then** $d_i = 0$
28: **end if**
29: **if** $h_{i+1} + d_i > b_{\max}$ **then** $d_i = b_{\max} - h_{i+1}$
30: **end if**
31: **end if**
32: $l_i = b_i - d_i \quad s_{\mathrm{pow}} = 0 \quad p = 0 \quad n = 1$
33: **while** $s_{\mathrm{pow}} < l_i$ **do**
34: $p^{\mathrm{a}}_{i,n} = p^{\mathrm{a}}_{i,n} + p$
35: $\lambda_{i,n} = \lambda_{i,n} + 1$
36: $n = \mathbf{argmin}(c_{i,k,1} : \forall k \in \mathcal{N})$
37: $p = \Delta p^{\mathrm{s}}_{i,n,1}$
38: $c_{i,n,m} = c_{i,n,m+1},\ \text{for } m = 1, \ldots, M-1$
39: $\Delta p^{\mathrm{s}}_{i,n,m} = \Delta p^{\mathrm{s}}_{i,n,m+1},\ \text{for } m = 1, \ldots, M-1$
40: $s_{\mathrm{pow}} = s_{\mathrm{pow}} + p$
41: **end while**
42: **if** $i < T$ **then**
43: $f = (h_{i+1} + b_i - \sum_{k=1}^{N} p^{\mathrm{a}}_{i,k}) - b_{\max}$
44: **if** $f > 0$ **and** $s_{\mathrm{pow}} < b_i$ **then** $p^{\mathrm{a}}_{i,n} = p^{\mathrm{a}}_{i,n} + p$
45: $\lambda_{i,n} = \lambda_{i,n} + 1$
46: **else if** $f > 0$ **then**
47: *AdjustPA*$(\boldsymbol{\lambda}, \boldsymbol{u}, \boldsymbol{p}^{\mathbf{a}}, \boldsymbol{p}^{\mathbf{r}}, i, f)$
48: **end if**
49: $b_{i+1} = h_{i+1} + b_i - \sum_{n=1}^{N} p^{\mathrm{a}}_{i,n}$
50: **end if**
51: **end for**
52: $r_{\mathrm{total}} = \sum_{i=1}^{T} \sum_{n=1}^{N} r_m,\ \text{for } m = \lambda_{i,n}$

by $c_{i,n,m}$. This variable stores the ratio between the increase in power and the increase in data rate for raising the MCS from m to $m+1$, for $m = 1, \ldots, M-1$, $\forall i$, $\forall n$. Now, we describe the first part of Algorithm 1, that goes from lines 1 to 16, where we set an initial subcarrier assignment $u_{i,n}$ and estimate an initial power allocation $q^{\mathrm{a}}_{i,n}$.

In Algorithm 1, we firstly calculate $p^{\mathrm{r}}_{i,n,j,m}$ by using its definition in equation (1), and in the next step we determine an initial power allocation $q^{\mathrm{a}}_{i,n}$ by taking the minimum required power to achieve the highest MCS level among all users. Consequently, we obtain an initial subcarrier assignment $u_{i,n}$, and define $p^{\mathrm{s}}_{i,n,m}$ as the required power for the users selected in this assignment. The *for* loop from lines 5 to 14 decreases the MCS level of a subcarrier n until its power becomes smaller than the average over all subcarriers in TTI i. Then, the resulting MCS is stored in $\lambda_{i,n}$, and after the loop ends we

Algorithm 2 Adjust Requirement

1: **procedure** *AdjustReq*$(\boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{p}^{\mathbf{s}}, \boldsymbol{q}^{\mathbf{a}})$
2: **for** $i \leftarrow 1$ **to** T **do**
3: **while** $q_i > \frac{1}{T}\sum_{t=1}^{T} q_t$ **do**
4: $n = \mathbf{argmax}(q^{\mathrm{a}}_{i,k} : \forall k \in \mathcal{N})$
5: $m = \lambda_{i,n} - 1$
6: $q^{\mathrm{a}}_{i,n} = p^{\mathrm{s}}_{i,n,m}$
7: $\lambda_{i,n} = m$
8: $q_i = \sum_{n=1}^{N} q^{\mathrm{a}}_{i,n}$
9: **end while**
10: **end for**
11: **end procedure**

Algorithm 3 Adjust Power Allocation

1: **procedure** *AdjustPA*$(\boldsymbol{\lambda}, \boldsymbol{u}, \boldsymbol{p}^{\mathbf{a}}, \boldsymbol{p}^{\mathbf{r}}, i, f)$
2: $\boldsymbol{g}_n = \mathbf{sortdesc}(\alpha_{i,n,j} : \forall j \in \mathcal{J}),\ \forall n,\ \text{where } \boldsymbol{g}_n = \{g_{n,j} : \forall j\}$
3: $\boldsymbol{\psi}_n = \mathbf{argsortdesc}(\alpha_{i,n,j} : \forall j \in \mathcal{J}),\ \forall n, \quad \boldsymbol{\psi}_n = \{\psi_{n,j} : \forall j\}$
4: **while** $f > 0$ **do**
5: $n = \mathbf{argmax}(g_{s,2} : \forall s \in \mathcal{N})$
6: $m = \lambda_{i,n}$
7: $k = u_{i,n}$
8: $j = \psi_{n,2}$
9: $\Delta p = p^{\mathrm{r}}_{i,n,j,m} - p^{\mathrm{r}}_{i,n,k,m}$
10: $p^{\mathrm{a}}_{i,n} = p^{\mathrm{a}}_{i,n} + \Delta p$
11: $u_{i,n} = j$
12: $\psi_{n,j} = \psi_{n,j+1},\ \text{for } j = 2, \ldots, J-1$
13: $g_{n,j} = g_{n,j+1},\ \text{for } j = 2, \ldots, J-1$
14: $g_{n,J} = 0$
15: $f = f - \Delta p$
16: **end while**
17: **end procedure**

calculate q_i, that is the total power requirement for each TTI i. In line 16 we call the procedure *AdjustReq*, that reduces the MCS level in overloaded carriers until the power requirement q_i gets smaller than the average over all transmissions. This procedure is shown in Algorithm 2. These adjustments give a more accurate estimation of the power to be used in practice, because it is not worthy to spend resources in subcarriers that require much power to achieve a higher modulation level. Alternatively, we prefer to save power for subcarriers that, in future transmissions, will experience more favorable channel conditions. After calculating q_i we start the second part of Algorithm 1, that goes from lines 17 to 22, where we set all variables needed to run the HH-based method.

Continuing Algorithm 1, we compute the data rate increase Δr_m and the incremental power matrix $\Delta p^{\mathrm{s}}_{i,n,m}$ in order to determine the efficiency ratio $c_{i,n,m}$, that measures the cost (in power) per bit for each MCS leap. Since the rate increase Δr_m is not uniform as m grows, the first subcarriers to be loaded are chosen by evaluating $c_{i,n,m}$, which is an improvement over performing a selection through $\Delta p^{\mathrm{s}}_{i,n,m}$ as the HH algorithm proposes [19]. Moreover, we set the last increment in $c_{i,n,M}$ and $\Delta p^{\mathrm{s}}_{i,n,M}$ to infinity for indicating that the final MCS level has been reached. Then, in line 21 we initialize the power allocation $p^{\mathrm{a}}_{i,n}$ to zero, and define the value of b_1 (available power at TTI 1) as the sum between h_1 (available power at TTI 1 resulting from the conversion to power of the harvested energy H_1) and b_0 (the initial power at the battery). Next, the MCSs $\lambda_{i,n}$ are set to one except for $\lambda_{i,1}$, that is set to zero because it always starts incremented by 1 in line 35. Lastly, we describe the third part of Algorithm 1, that goes from lines 23 to 52, where we compute the final power allocation $p^{\mathrm{a}}_{i,n}$, set the final subcarrier assignment in $u_{i,n}$ and determine the

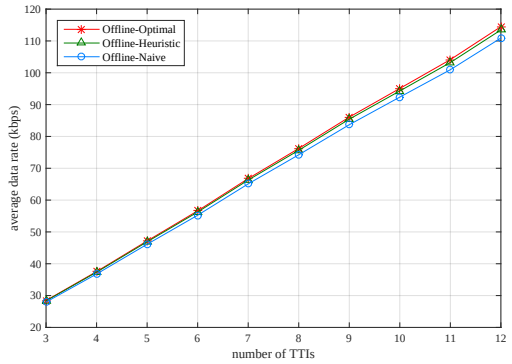

Fig. 2. Average data rate versus number of TTIs, for $M = 16$, $N = 15$, $J = 10$, $T = 3$ to 12, $h_{\max} = 112$ mW, $b_{\max} = 208$ mW, and $S = 7$.

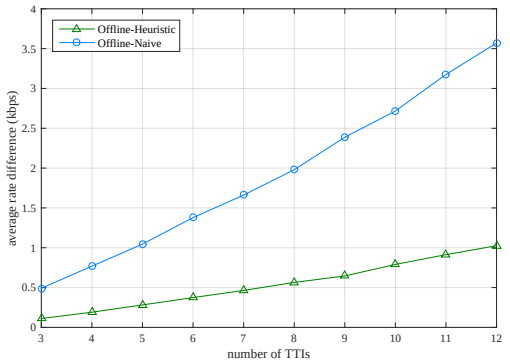

Fig. 3. Average rate difference versus number of TTIs, for $M = 16$, $N = 15$, $J = 10$, $T = 3$ to 12, $h_{\max} = 112$ mW, $b_{\max} = 208$ mW, and $S = 7$.

total data rate using the values of $\lambda_{i,n}$.

Hence, we finally run the *for* loop from lines 23 to 51, that calculates for every $i < T$ the power decrease d_i applied to b_i in order to compute the power limit l_i to be spent in the i-th TTI. The calculation of d_i in line 26 is an intuitive result, since the greater the available power b_i and the requirements for other transmissions (q_t for $t > i$) are, the greater is the decrease applied to b_i and lower will be l_i. On the other hand, the bigger the requirement q_i and the available power for future transmissions (h_t for $t > i$) are, the smaller is the value of d_i and higher will be l_i. In fact, the expression in line 26 is the solution of equation $(b_i - d_i)/q_i = (h_{i+1} + d_i - d_{i+1})/q_{i+1} = \cdots = (h_T + d_{T-1})/q_T$ for d_i. This equation means that the ratio between available power and required power must be the same for each TTI, thus, applying the principle of reserving more power to the TTI that needs it the most. The condition in line 27 is necessary to enforce the energy consumption causality constraints, because when $d_i < 0$, this would mean to take power from energy packets still not accessible. And the extra condition in line 29 adjusts d_i to prevent the violation of the battery capacity constraints.

Thereafter, the *while* loop from lines 33 to 41 applies the HH-based method described previously in the beginning of this section, and obtains the power allocation $p^{\mathrm{a}}_{i,n}$ for TTI i and the corresponding MCSs $\lambda_{i,n}$. Nevertheless, we need to calculate f that is the battery overflow in case the current power allocation is enforced. This becomes necessary because the power increase p is discrete, and the sum of the allocated power never equals l_i. If the overflow $f > 0$ and the power sum s_{pow} does not exceed what is available in the battery, we employ the last power increment p chosen previously. This ensures that $f \leq 0$ because of the condition in line 29, as s_{pow} surpasses l_i guarantees that the battery power limit $b_{\max}$ will be observed.

Our final option to eliminate the overflow is to call the procedure *AdjustPA*, that gradually increases the power consumption by changing the user to subcarrier assignment stored in $u_{i,n}$. Since the achieved MCS is kept the same when changing users, this procedure has the disadvantage of raising the consumed power without improving the data rate. However, this is necessary to always comply with the restrictions in equation (6). This raise in power is gradual because we choose the users with greater gains in the second column of $g_{n,j}$ (the first column contains values already selected). Given large enough values for N and J the loop in procedure *AdjustPA* is guaranteed to finish and the problem constraints are imposed. This procedure is presented in Algorithm 3, where *sortdesc* is a function that returns a vector sorted in descending order and *argsortdesc* returns the indexes corresponding to the sorted values. Lastly, the battery level is updated in line 49 by adding the remaining amount to the next harvested value, and after the *for* loop ends the total data rate is computed in line 52.

C. Performance Evaluation

On the goal to compare optimal and heuristic solutions, we performed simulations of the proposed model with M = 16, N = 15, J = 10, and T = 3 to 12. Our choice to vary T is justified by the fact that decisions related to power saving become more difficult as T increases. The results were obtained by running 3,000 instances for each set $\mathcal{A}_T = \{T, N, J, M\}$, counting to a total of 30,000 instances analyzed. The mapping through MCSs is $\mathcal{D}$ = {(γ,r): (0,0), (0.3,25), (0.4,39), (0.6,63), (0.9,101), (1.4,147), (2.1,197), (3.5,248), (5.1,321), (7.8,404), (12.3,458), (19.1,558), (28.8,655), (42.7,759), (79.4,859), (109.6,933)}, with r in bps. The SNR thresholds in γ were taken from [20], and the values of r are based on the channel quality indicators reported in [21]. Other important constants were defined as $h_{\max}$ = 112 mW, $b_{\max}$ = 208 mW, σ^2 = 4.74 $\times$ 10^{-16} W, with $\alpha_{i,n,j}$ ranging from 10^{-8} to 10^{-10}.

Also, the number of states of our Markov model is S = 7, with each state being an interval of length 16 mW. Initially, the simulation distributes uniformly J = 10 users inside a circular area of radius R = 467 m, and establishes a minimal distance of 70 m between user and BS in order to avoid near-field effects. Then, we define the Markov chain initial state s based on the marginal probabilities in $\boldsymbol{v}$, and determine the succession of states using the transition probabilities $P_{s,k}$ for obtaining all the values of h_i. After that, we build the matrices necessary to specify the ILP described in section IV, and solve the optimization problem through the CPLEX solver, that returns the values of $x_{i,n,j,m}$. The total throughput of the system is calculated according to equation (2). The sub-optimal solution is obtained next by running Algorithm 1, and by applying HH algorithm we obtain a result denoted as naive solution. The HH method receives this name because it does

not save any power for future TTIs and does not consider the rate increase Δr_m for raising the MCS levels.

Figure 2 shows the results for ten different values of T, where the performance metric is the average data rate over all the 3,000 instances evaluated by each solution. The CPLEX solver, the Algorithm 1 and the HH method generated the results displayed in the curves Offline-Optimal, Offline-Heuristic and Offline-Naive, respectively. We realize that as T increases the difference between optimal and heuristic grows, since the error of estimating q_i in procedure *AdjustReq* is greater as T becomes larger. However, our results remain very close to optimal as the plot indicates. Figure 3 presents the average rate difference from the optimal solution in order to better display the separation between each curve. In fact, the average difference between optimal and heuristic solutions exceeds 1 kbps only for T = 12, as indicated by Figure 3.

The best results are obtained for T = 3, with average difference of just 110 bps. The naive solution (HH) has poor performance when compared to optimal solution, as Figure 3 shows that the difference for T = 12 is greater than 3.5 kbps. Furthermore, the complexity/performance gain is high, since for T = 8 the average execution time of a single scenario is 1338 ms for optimal solution, and only 16.7 ms for heuristic, that is 80 times faster in this case. In order to complement our results, we compare solutions by evaluating a performance metric called NMSE (Normalized Mean Square Error). We computed this metric by using the definition found in [22]: $\text{NMSE} = \frac{1}{K}\sum_{k=1}^{K}(C_k - D_k)^2/\bar{C}\bar{D}$, with $\bar{C} = \frac{1}{K}\sum_{k=1}^{K} C_k$ and $\bar{D} = \frac{1}{K}\sum_{k=1}^{K} D_k$. C_k is the optimal result for sample k in Figure 2. D_k represents the result for sample k in Figure 2 obtained by the heuristic or naive solution. The goal of NMSE is to measure the average deviation between optimal solution and other solutions. Based on the $K = 10$ samples given by Figure 2, the NMSE for heuristic and naive solutions are $\text{NMSE}_{\text{heu}} = 0.074 \times 10^{-3}$ and $\text{NMSE}_{\text{naive}} = 0.939 \times 10^{-3}$, respectively. Thus, the error for the naive solution is 12.7 times greater than the error for the heuristic solution, what proves that Algorithm 1 achieves considerable improvement over the classical HH algorithm. All the results were obtained from a Windows 10 computer with a 2.4 GHz core i7 intel processor.

V. Data Rate Maximization with Renewable Energy Source - Online Case

This section is devoted to the study of the online version of the problem presented in section IV. Now, all available information is causal, in other words, at TTI i we only have knowledge of harvested energy and channel gains from the i-th TTI (present moment). These conditions provide a more realistic scenario and the solutions presented in this section can be applied to real systems.

A. Modified Problem Constraints

In regard to the problem formulation, all restrictions remain valid for the online case, meaning that the problem keeps constrained by the subcarrier and MCS assignment restrictions in (3), and that the energy harvesting constraints in (5) and (6) have to be enforced. However, since we do not know the harvested energy values beforehand, we cannot guarantee that the battery limit will always be observed and, therefore, energy overflow can happen in some instances of the online problem. Hence, for the online case, we rewrite the battery capacity restrictions (6) in the following manner:

$$B_1 = \min(B_{\max}, H_1 + B_0), \tag{9a}$$

$$B_i = \min\left(B_{\max}, H_i + R_{i-1}\right),\ i = 2, \ldots, T, \tag{9b}$$

$$R_i = B_i - \tau \sum_{n=1}^{N} p_{i,n}^{\text{a}},\ i = 1, \ldots, T-1. \tag{9c}$$

B_i represents the energy available in the battery at the beginning of the i-th TTI, and B_0 is the initial energy stored in the battery before the transmission starts. Equation (9a) defines the battery level for $i = 1$ as the minimum between $B_{\max}$ and $H_1 + B_0$. Since in our simulations $B_0 = 0$ and $H_1 < B_{\max}$, we have that $B_1 < B_{\max}$. Equation (9b) means that if the sum between the remaining energy and the harvested energy surpasses $B_{\max}$, then energy will be wasted and the battery will reach the maximum level. R_i represents the remaining energy in the battery at the end of the i-th TTI, as defined in equation (9c). Furthermore, once we redefined the battery capacity restrictions to (9a) and (9b), then we rewrite the energy consumption causality constraints (5) in terms of B_i:

$$\tau \sum_{n=1}^{N} p_{i,n}^{\text{a}} \leq B_i,\ i = 1, \ldots, T. \tag{10}$$

The above equation simply means that, for each TTI, it is possible to spend only the amount of power provided by the battery. Consequently, by replacing the expression for B_i into equation (10) we write the energy harvesting constraints as

$$\tau \sum_{n=1}^{N} p_{1,n}^{\text{a}} \leq \min(B_{\max}, H_1 + B_0), \tag{11a}$$

$$\tau \sum_{n=1}^{N} p_{i,n}^{\text{a}} \leq \min\left(B_{\max}, H_i + R_{i-1}\right),\ i = 2, \ldots, T. \tag{11b}$$

B. Online Solutions

Our goal in the online problem remains the same of the offline problem, that is, to maximize the total data rate experienced by all users throughout all TTIs, as expressed in equation (7). However, without knowledge of future channel states, we cannot estimate the power requirement q_i for each TTI as described in procedure *AdjustReq*. This is a consequence of not knowing all values of $p_{i,n,j,m}^{\text{r}}$ beforehand, because we only know the required power for the current TTI. Moreover, without knowledge of future energy arrivals, we can not calculate d_i as expressed in line 26 of Algorithm 1 neither the power limit l_i. Therefore, we need to find an expression for d_i that uses an estimation for the values of h_i.

In practice, this estimation can be done by using a Markov model developed through empiric data of the harvested energy, as shown in [6], [8]. Hence, in a similar way, the online solution in Algorithm 4 estimates h_i based on knowledge of the Markov model used to simulate the energy arrivals. In this case, knowing the model does not give precise knowledge of the harvested energy, because we cannot determine the state

Algorithm 4 Online Solution with Knowledge of the Markov Model

1: $\Delta r_{m-1} = r_m - r_{m-1},\ \text{for } m = 2, \ldots, M$
2: $b_{\text{rem}} = b_0 \qquad p^{\text{a}}_{i,n} = 0,\ \forall i,\ \forall n$
3: $\lambda_{i,n} = 1,\ \forall i,\ \forall n \qquad \lambda_{i,1} = 0,\ \forall i$
4: **for** $i \leftarrow 1$ **to** T **do**
5: Calculate $p^{\text{r}}_{i,n,j,m}$ for the current TTI
6: $u_{i,n} = \textbf{argmin}(p^{\text{r}}_{i,n,j,M} : \forall j \in \mathcal{J}),\ \forall n$
7: $p^{\text{s}}_{i,n,m} = p^{\text{r}}_{i,n,j,m},\ \forall n,\ \forall m,\ \text{for } j = u_{i,n}$
8: $\Delta p^{\text{s}}_{i,n,m-1} = p^{\text{s}}_{i,n,m} - p^{\text{s}}_{i,n,m-1},\ \forall n,\ \text{for } m = 2, \ldots, M$
9: $c_{i,n,m} = \Delta p^{\text{s}}_{i,n,m} / \Delta r_m,\ \forall n,\ \text{for } m = 1, \ldots, M-1$
10: $c_{i,n,M} = \infty,\ \forall n \qquad \Delta p^{\text{s}}_{i,n,M} = \infty,\ \forall n$
11: $d_i = 0 \qquad b_i = h_i + b_{\text{rem}}$
12: **if** $b_i > b_{\max}$ **then** $b_i = b_{\max}$
13: **end if**
14: **if** $i < T$ **then**
15: **for** $k \leftarrow 1$ **to** S **do**
16: **if** $h_i > \omega_k$ **and** $h_i \leq \omega_{k+1}$ **then** $s = k$
17: **break**
18: **end if**
19: **end for**
20: **for** $t \leftarrow i+1$ **to** T **do**
21: $s = \textbf{argmax}(P_{s,k} : k = 1, \ldots, S)$
22: $p^{\text{h}}_t = \omega_s + h_{\max}/2S$
23: **end for**
24: $d_i = (b_i(T-i) - \sum_{t=i+1}^{T} p^{\text{h}}_t)/(T-i+1)$
25: **if** $d_i < 0$ **then** $d_i = 0$
26: **end if**
27: **end if**
28: $l_i = b_i - d_i \quad s_{\text{pow}} = 0 \quad p = 0 \quad n = 1$
29: **while** $s_{\text{pow}} < l_i$ **do**
30: $p^{\text{a}}_{i,n} = p^{\text{a}}_{i,n} + p$
31: $\lambda_{i,n} = \lambda_{i,n} + 1$
32: $n = \textbf{argmin}(c_{i,k,1} : \forall k \in \mathcal{N})$
33: $p = \Delta p^{\text{s}}_{i,n,1}$
34: $c_{i,n,m} = c_{i,n,m+1},\ \text{for } m = 1, \ldots, M-1$
35: $\Delta p^{\text{s}}_{i,n,m} = \Delta p^{\text{s}}_{i,n,m+1},\ \text{for } m = 1, \ldots, M-1$
36: $s_{\text{pow}} = s_{\text{pow}} + p$
37: **end while**
38: $b_{\text{rem}} = b_i - \sum_{n=1}^{N} p^{\text{a}}_{i,n}$
39: **end for**
40: $r_{\text{total}} = \sum_{i=1}^{T} \sum_{n=1}^{N} r_m,\ \text{for } m = \lambda_{i,n}$

transitions neither the exact amount of energy obtained in a state. Thus, with knowledge of the current state, we guess the next state as indicated by the highest transition probability. Then, we guess that the harvested energy will be the average value of the predicted state.

The solution presented in Algorithm 4 is strongly based on the heuristic we described in Algorithm 1. Nevertheless, the online algorithm does not run the procedures *AdjustReq* and *AdjustPA*. The former procedure cannot run because we do not know $p^{\text{r}}_{i,n,j,m}$ beforehand. And the latter procedure cannot run since we know h_i only causally, what means that we cannot predict the battery overflow for the next TTI. Consequently, adjust the power allocation would be senseless because we are not sure if the battery will overflow. For this reason, the overflow is not always prevented, and we have to redefine the energy harvesting constraints to (11a) and (11b).

We start Algorithm 4 in line 1 by calculating Δr_m, and in line 2 we create a new variable, b_{rem}, that represents the power provided by the remaining energy in the battery. Initially, b_{rem} is set to b_0 and then it is continually updated in line 38. Next, we initialize $p^{\text{a}}_{i,n}$ and $\lambda_{i,n}$ in lines 2 and 3. Then, we run the *for* loop from lines 4 to 39, and in lines 5 to 10 we compute $p^{\text{r}}_{i,n,j,m}$, $u_{i,n}$, $p^{\text{s}}_{i,n,m}$, $\Delta p^{\text{s}}_{i,n,m}$ and $c_{i,n,m}$ only for the current TTI, since all available information is causal. Hence, in line 11, we set d_i to zero, and update the battery power b_i by adding the harvested power h_i to the remaining power b_{rem}. The condition in line 12 enforces the battery capacity restrictions in (9a) and (9b) by limiting b_i to $b_{\max}$. Thus, if $i < T$, we calculate d_i as described in lines 15 to 26. Firstly, we need to determine the current state of the Markov model by checking the power interval which h_i belongs. These intervals are defined by

$$\Omega_k = [(k-1)(h_{\max}/S),\ k(h_{\max}/S)],\ k = 1, \ldots, S. \quad (12)$$

Therefore, the limits of the power intervals are given by

$$\omega_k = (k-1)(h_{\max}/S),\ k = 1, \ldots, S+1. \quad (13)$$

In real systems, Ω_k represents the empirical Markov model, and $\mathbf{P}$ is the transition probability matrix obtained from empirical data. Continuing Algorithm 4, we run a loop, from lines 15 to 19, that iterates over each state k until we find the state s corresponding to h_i. This is done by checking if h_i is limited between ω_k and ω_{k+1}.

Once we determined the current state s, we can guess the next harvested power values, p^{h}_t for $t = i+1, \ldots, T$. This is shown in lines 20 to 23, where we predict the next state s by choosing the most likely state transition indicated by $\mathbf{P}$. Then, we set p^{h}_t to the average value of the predicted state s. This process continues until we guess all values for p^{h}_t, what means that we predict a chain with possible power values for the coming TTIs. Finally, we calculate d_i as expressed in line 24, which is a simplification of the equation in line 26 of Algorithm 1. In the online problem, we assume that all values of q_i are equivalent for each TTI, which is $q_i = q, \forall i \in \mathcal{T}$. This assumption cancels out all values of q_i and we obtain:

$$d_i = \left(b_i(T-i) - \sum_{t=i+1}^{T} p^{\text{h}}_t\right) / (T-i+1). \quad (14)$$

Furthermore, in equation (14), we replace h_t for p^{h}_t, which represents the estimated amount of power to be harvested in the next TTIs. The results obtained for the online problem heavily depend on a good estimation of p^{h}_t. In many occasions, a bad guess for p^{h}_t will lead to poor results. For example, if we predict a high value for p^{h}_t in the next TTI, then we will spend more power in the current TTI, since we have enough power for the next transmission. However, if the prediction fails and we harvest a small value of h_t, we may not load some cheap bits in the next TTI since we spent much power loading costly bits in the current TTI. This causes a less efficient resource allocation and decreases the total data rate.

Algorithm 5 Online Solution with General Markov Model

1: Repeat Algorithm 4 from lines 1 to 13
2: **if** $i < T$ **then**
3: **for** $k \leftarrow 1$ **to** 3 **do**
4: **if** $h_i > \phi_k$ **and** $h_i \leq \phi_{k+1}$ **then** $s = k$
5: **break**
6: **end if**
7: **end for**
8: $p^{\text{h}} = \phi_s + h_{\max}/6$
9: $d_i = (T-i)(b_i - p^{\text{h}})/(T-i+1)$
10: **if** $d_i < 0$ **then** $d_i = 0$
11: **end if**
12: **end if**
13: Repeat Algorithm 4 from lines 28 to 40

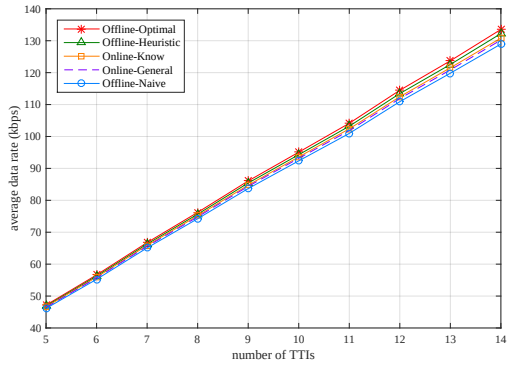

Fig. 4. Average data rate versus number of TTIs, for $M = 16$, $N = 15$, $J = 10$, $T = 5$ to 14, $h_{\max} = 112$ mW, $b_{\max} = 208$ mW and $S = 7$.

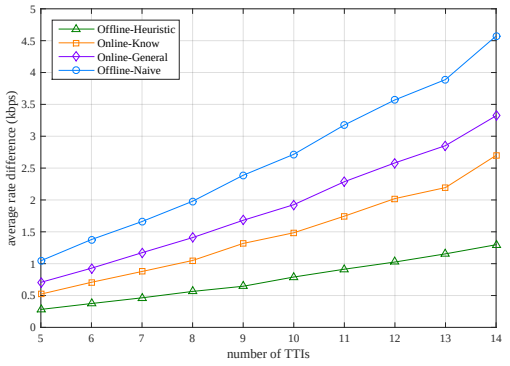

Fig. 5. Average rate difference versus number of TTIs, for $M = 16$, $N = 15$, $J = 10$, $T = 5$ to 14, $h_{\max} = 112$ mW, $b_{\max} = 208$ mW and $S = 7$.

After calculating d_i, we check if $d_i < 0$ in line 25, which happens if b_i is small when compared to p_t^{h}. Since $d_i < 0$ violates the restrictions in equation (10), in this case we set d_i to zero. Thereafter, in line 28, we compute the power limit l_i by applying the power decrease d_i to b_i. Then, we calculate the power allocation $p_{i,n}^{\mathrm{a}}$ by running the *while* loop from lines 29 to 37, exactly as in Algorithm 1. Lastly, in line 38, we update b_{rem} by subtracting the allocated power from b_i. And in line 40, we finally compute the total data rate with knowledge of the MCSs stored in $\lambda_{i,n}$.

Algorithm 4 shows a solution that requires knowledge of the Markov model that simulates the energy harvesting process. Nevertheless, this model may not always be available, mainly because of the difficulties in obtaining enough data to generate a precise model. For example, in [8] the authors generated a Markov model from data collected over 20 years. Moreover, in order to account for weather changes throughout the seasons, we may need a different model for each month [8]. Therefore, we propose an online solution that uses a general Markov model to predict the harvested energy. This general model is a special case of Ω_k, when $S = 3$. Thus, the model has only three states: low, middle and high energy states.

Algorithm 5 presents a similar solution to Algorithm 4. In line 1, we repeat the initial part of Algorithm 4 from lines 1 to 13. Thereafter, we rewrite the code snippet from lines 14 to 27 of Algorithm 4, that refers to the calculation of d_i. In fact, this is the only difference between both solutions, as described in lines 2 to 12 of Algorithm 5. Now, since we do not know the empirical Markov model, we use a general model with $S = 3$. Hence, the new limits for determining the current state s are:

$$\phi_k = (k-1)(h_{\max}/3),\ k = 1, 2, 3, 4. \tag{15}$$

Once we know the current state, after running lines 3 to 7, we simply predict that this state will preserve itself for all TTIs. This means that $p_t^{\mathrm{h}} = p^{\mathrm{h}},\ \text{for } t = i+1, \ldots, T$. Then, in line 8, we compute p^{h} as the average value of state s.

The assumption of state conservation comes from the nature of Markov models for harvested energy. As shown in [7], the energy harvesting environment typically changes slowly, which heavily contributes to state conservation. This becomes evident in [6], since the probabilities in $\mathbf{P}$ tend to concentrate on the main diagonal, what confirms the assumption of state preservation. In addition, [7] shows that the harvested energy varies considerably inside a state, what justifies the choice of p^{h} as the mean value in order to decrease the average error.

Continuing Algorithm 5, in line 9 we calculate d_i through a simplified version of equation (14). Since we assume that $p_t^{\mathrm{h}} = p^{\mathrm{h}}$, we reduce the computation of d_i to

$$d_i = (T-i)(b_i - p^{\mathrm{h}}) \,/\, (T-i+1). \tag{16}$$

This is an interesting result because equation (16) suggests to save less and less energy for future TTIs as the time passes. In order to illustrate this, consider the case for $T = 10$. In the first TTI, we have that $d_1 = (9/10)(b_1 - p^{\mathrm{h}})$, what means that we save 90% of the difference between b_1 and p^{h} to next TTIs. Then, for $i = 2$, we have that $d_2 = (8/9)(b_2 - p^{\mathrm{h}})$, and now we save 88.9% of the power difference. The process continues until we save only 50% of this difference when $i = 9$. Thus, it is worth noting that the series $t/(t+1)$, for $t = T-1, \ldots, 1$, appears as a natural pattern in the online solution of our problem. Finally, after we check for negative d_i in line 10, then we repeat Algorithm 4 from lines 28 to 40.

C. Performance Evaluation

Once we presented two online solutions for our problem, now we analyze the performance of the online case when compared to the offline case. We used the same simulation parameters chosen previously in section IV, with the exception of T that now varies from 5 to 14. This change aims to better display the progress of each solution. Similarly to Figure 2, Figure 4 shows the average data rate over 3,000 instances of the problem for each value of T. The curves for both online solutions are located between the curves for heuristic and naive solutions, as expected. The online solution with knowledge of the Markov model, shown in Algorithm 4, generated the results displayed in the curve Online-Know. The curve Online-General corresponds to the online solution with general Markov model, presented in Algorithm 5.

As we can see, the general Markov model provides a good estimation for the model used in our simulations, since the curves of the online solutions are very close to each other. Therefore, this general model offers a reasonable solution, given its simplicity and its considerable improvement over the HH algorithm (naive solution). Since it is not easy to build an empirical Markov model [8], the general model becomes very useful in practice because it only requires knowledge of

the maximum harvested power, $h_{\max}$. In Figure 5, we see that the rate difference between optimal and heuristic grows almost linearly as T increments. However, this is not the case for the other solutions, since the difference from the optimal solution grows more and more as T increases. This is clear when the curves become steeper in the interval between $T = 13$ and $T = 14$. In order to compare the results, we analyze the problem for $T = 14$, when the differences from the optimal result are maximum.

Firstly, the offline heuristic solution reaches 1.3 kbps of difference when $T = 14$. Now, without knowledge of future energy arrivals and future channel gains, the difference more than doubled to 2.7 kbps in the Online-Know solution. This proves how the results are inferior in practical systems with realistic conditions. Then, the difference increases to 3.3 kbps in the Online-General solution, which is still better than the naive solution for $T = 12$, that reaches 3.6 kbps. Lastly, far from the other solutions, we have the naive solution with 4.6 kbps of difference when $T = 14$. This shows the importance of saving energy for next TTIs as we limit the spent power to l_i, and consequently, we see how essential is the calculation of d_i in order to improve results. Additionally, by taking the samples in Figure 4, we calculated the NMSE for curves Offline-Heuristic, Online-Know, Online-General and Offline-Naive: $\text{NMSE}_{\text{heu}} = 0.083 \times 10^{-3}$, $\text{NMSE}_{\text{know}} = 0.321 \times 10^{-3}$, $\text{NMSE}_{\text{gen}} = 0.529 \times 10^{-3}$ and $\text{NMSE}_{\text{naive}} = 1.029 \times 10^{-3}$. Thus, the results clearly show that the average deviation from the optimal solution is gradually increased as the other solution has less available information about the problem. Also, we see how poor the HH algorithm performs even with full knowledge of the problem variables.

VI. Data Rate Maximization with Hybrid Energy Source and QoS Constraints

In the previous problem, our goal was simply to maximize the total data rate with the resources available to perform the transmissions. In addition, we had to respect the physical limitations imposed by the OFDMA and EH systems. Now, we formulate a new problem that adds the difficulty to satisfy QoS requirements for each user. Since the amount of harvested energy is random by nature, we cannot hope to satisfy all users only with renewable energy sources [9]. Hence, we include a fixed energy supply from a non-renewable source in order to fulfill the demand for power. Therefore, similarly to [13], the BS is powered by a hybrid power system composed of a renewable and a non-renewable source. This complicates our problem, since we have to observe the QoS constraints as we maximize the total rate, and because we have to respect the power limitations of the non-renewable source.

A. Problem Formulation and Optimal Solution

Before we present the new problem formulation, we define $p^w_{i,n}$ as the power allocated from source w to subcarrier n in the i-th TTI. We use $w = 1$ for the renewable source, and $w = 2$ to the non-renewable source. This means that the power allocated to a subcarrier comes in part from source 1, in part from source 2. Thus, we define a continuous decision variable $y_{i,n,w,j,m} \in [0, 1]$, that assumes a value bigger than zero when subcarrier n transmits to user j during the i-th TTI with energy from source w that provides $100 \cdot y_{i,n,w,j,m}$ percent of the power needed to achieve MCS m.

The binary decision variable $x_{i,n,j,m}$ previously defined is also present in the new problem, since we calculate the total data rate as expressed in equation (2). Moreover, the problem remains constrained by the subcarrier and MCS assignment restrictions in equation (3). Finally, in order to determine a feasible configuration for $y_{i,n,w,j,m}$, we define the following restrictions:

$$y_{i,n,1,j,m} + y_{i,n,2,j,m} = x_{i,n,j,m},\ \forall i,\ \forall n,\ \forall j,\ \forall m. \quad (17)$$

The restrictions in (17) ensure that the powers $p^1_{i,n}$ and $p^2_{i,n}$ will always complement each other, as $p^1_{i,n} + p^2_{i,n}$ sums up to 100% of the power required to load subcarrier n at TTI i. In other words, the percentages in $y_{i,n,1,j,m}$ and $y_{i,n,2,j,m}$ must sum up to 1 or 0, as indicated by $x_{i,n,j,m}$.

Furthermore, we define $p^1_{i,n}$, the power spent by the renewable source in subcarrier n at TTI i, in terms of $y_{i,n,w,j,m}$:

$$p^1_{i,n} = \sum_{j=1}^{J} \sum_{m=1}^{M} p^{\text{r}}_{i,n,j,m} y_{i,n,1,j,m},\ \forall i,\ \forall n. \quad (18)$$

Consequently, we rewrite the energy harvesting constraints in terms of $p^1_{i,n}$:

$$\sum_{i=1}^{t} \sum_{n=1}^{N} p^1_{i,n} \leq \frac{1}{\tau} \sum_{i=1}^{t} H_i,\ t = 1, \ldots, T, \quad (19a)$$

$$\sum_{i=1}^{t+1} H_i - \tau \sum_{i=1}^{t} \sum_{n=1}^{N} p^1_{i,n} \leq B_{\max},\ t = 1, \ldots, T-1. \quad (19b)$$

Then, we define $p^2_{i,n}$, the power spent by the non-renewable source in subcarrier n at TTI i, in terms of $y_{i,n,w,j,m}$:

$$p^2_{i,n} = \sum_{j=1}^{J} \sum_{m=1}^{M} p^{\text{r}}_{i,n,j,m} y_{i,n,2,j,m},\ \forall i,\ \forall n. \quad (20)$$

The next restriction regards the power provided by the non-renewable source that provides a fixed amount of power, p_{fix}, per TTI. Hence, $p^2_{i,n}$ is restricted to:

$$\sum_{n=1}^{N} p^2_{i,n} \leq p_{\text{fix}},\ \forall i. \quad (21)$$

The final restriction regards the QoS constraints that ensure a minimum data rate, Q_j, to be experienced by each user j throughout the T TTIs. Therefore, the QoS requirements are:

$$\sum_{i=1}^{T} \sum_{n=1}^{N} \sum_{m=1}^{M} r_m x_{i,n,j,m} \geq Q_j,\ \forall j. \quad (22)$$

Now, we need the values of $x_{i,n,j,m}$ and $y_{i,n,w,j,m}$ that maximize the total data rate for the optimization problem:

$$\max_{\boldsymbol{x},\, \boldsymbol{y}} \sum_{i=1}^{T} \sum_{n=1}^{N} \sum_{j=1}^{J} \sum_{m=1}^{M} r_m x_{i,n,j,m}, \quad (23)$$

subject to

$$\text{Equations (3), (17), (19a), (19b), (21) and (22),} \quad (24a)$$

$$x_{i,n,j,m} \in \{0,1\},\ \forall i,\ \forall n,\ \forall j,\ \forall m, \quad (24b)$$

$$y_{i,n,w,j,m} \in [0,1],\ \forall i,\ \forall n,\ \forall w,\ \forall j,\ \forall m. \quad (24c)$$

This problem is a MILP (Mixed Integer Linear Programming) mathematical optimization, since we have continuous and discrete values for the decision variables. In order to obtain the optimal solution, we run the BB algorithm implemented by the CPLEX solver. The computational complexity has grown considerably in comparison with the problem presented in section IV, because of the many additional restrictions in (17), and specially because of the QoS constraints in (22). These QoS requirements completely change the reasoning behind the heuristic solution in the offline case. Our new heuristic solution is shown in Algorithm 6.

B. Heuristic Solution

We start Algorithm 6 by running the first four lines of Algorithm 1, where we compute the values of $p^{\mathrm{r}}_{i,n,j,m}$, $q^{\mathrm{a}}_{i,n}$, $u_{i,n}$ and $p^{\mathrm{s}}_{i,n,m}$. Differently from Algorithm 1, we do not decrease the power requirements in q_i by running procedure *AdjustReq*, because each value of q_i is decremented by p_{fix} in line 11 of Algorithm 6. This decrease is enough to remove the extra power in overloaded subcarriers. Thus, running *AdjustReq* would easily set q_i to zero, what worsens our results since $q_i = 0$ means no need to spend power from the battery at TTI i. However, this is only true when p_{fix} is sufficient to achieve MCS M in all subcarriers at TTI i. Hence, the best action is simply to subtract p_{fix} from q_i. Then, we continue Algorithm 6 by running lines 17 to 22 of Algorithm 1, where we set the variables Δr_m, $\Delta p^{\mathrm{s}}_{i,n,m}$, $c_{i,n,m}$, $p^{\mathrm{a}}_{i,n}$, $\lambda_{i,n}$ and b_1.

After we determine the values for all these variables, we run the *for* loop from lines 2 to 10 in Algorithm 6, that calculates the power allocated from the non-renewable source, $p^2_{i,n}$, for each TTI. This means that we raise the first MCS levels with power from p_{fix}, and then, we add power from the battery in order to achieve higher MCS levels. Also, we compute an initial power allocation without checking the QoS requirements, and next, we satisfy the QoS constraints by running the procedure *SatisfyQoS*. In line 3, we initialize s_{pow}, p and n on the goal to start the *while* loop from lines 4 to 6. This loop runs the HH-based method described in lines 34 to 40 of Algorithm 1, with the exception of loading $p^2_{i,n}$ rather than $p^{\mathrm{a}}_{i,n}$. Moreover, we limit the spent power to p_{fix} instead of limiting to l_i, and after the loop ends we obtain the values of $p^2_{i,n}$ for TTI i. In line 7, we calculate β_i that is the remaining power from the fixed power supply at TTI i. Then, in line 8, we define ζ_i as the difference between the last power increment p and β_i. Thus, ζ_i is the power taken from the battery that complements β_i in order to apply the power increase p. Lastly, in line 9, η_i stores the subcarrier n that corresponds to p at TTI i.

In line 11, we compute q_i by subtracting p_{fix} from the sum of $q^{\mathrm{a}}_{i,n}$ for each n, and in case $q_i < 0$, we set it to zero in line 12. Next, we initialize all values of d_i to zero in line 13, and we begin the *for* loop from lines 14 to 35, that calculates

Algorithm 6 Heuristic Solution for Hybrid Power Systems

1: Repeat Algorithm 1 from lines 1 to 4 and 17 to 22
2: **for** $i \leftarrow 1$ **to** T **do**
3: $s_{\mathrm{pow}} = 0 \quad p = 0 \quad n = 1$
4: **while** $s_{\mathrm{pow}} < p_{\mathrm{fix}}$ **do**
5: Repeat Algorithm 1 from lines 34 to 40 by replacing $p^{\mathrm{a}}_{i,n}$ for $p^2_{i,n}$
6: **end while**
7: $\beta_i = p_{\mathrm{fix}} - \sum_{n=1}^{N} p^2_{i,n}$
8: $\zeta_i = p - \beta_i$
9: $\eta_i = n$
10: **end for**
11: $q_i = -p_{\mathrm{fix}} + \sum_{n=1}^{N} q^{\mathrm{a}}_{i,n},\ \forall i$
12: $q_i = 0,\ \forall i$ where $q_i < 0$
13: $d_i = 0,\ \forall i$
14: **for** $i \leftarrow 1$ **to** T **do**
15: **if** $i < T$ **and** $\sum_{t=i}^{T} q_t > 0$ **then**
16: Repeat Algorithm 1 from lines 26 to 30
17: **end if**
18: $l_i = b_i - d_i \quad s_{\mathrm{pow}} = \zeta_i \quad p = \zeta_i \quad n = \eta_i$
19: **if** $s_{\mathrm{pow}} < l_i$ **then**
20: $p^2_{i,n} = p^2_{i,n} + \beta_i$
21: $\beta_i = 0$
22: **end if**
23: Repeat Algorithm 1 from lines 33 to 41 by replacing $p^{\mathrm{a}}_{i,n}$ for $p^1_{i,n}$
24: **if** $i < T$ **then**
25: $f = (h_{i+1} + b_i - \sum_{k=1}^{N} p^1_{i,k}) - b_{\max}$
26: **if** $f > 0$ **and** $s_{\mathrm{pow}} < b_i$ **then**
27: $p^2_{i,n} = p^2_{i,n} + \beta_i$
28: $p^1_{i,n} = p^1_{i,n} + p$
29: $\lambda_{i,n} = \lambda_{i,n} + 1$
30: **else if** $f > 0$ **then**
31: *AdjustPA*$(\boldsymbol{\lambda}, \boldsymbol{u}, \boldsymbol{p^1}, \boldsymbol{p^{\mathrm{r}}}, i, f)$
32: **end if**
33: $b_{i+1} = h_{i+1} + b_i - \sum_{n=1}^{N} p^1_{i,n}$
34: **end if**
35: **end for**
36: $\xi_j = 0,\ \forall j$
37: **for** $i \leftarrow 1$ **to** T **do**
38: **for** $n \leftarrow 1$ **to** N **do**
39: $m = \lambda_{i,n}$
40: $j = u_{i,n}$
41: $\xi_j = \xi_j + r_m$
42: **end for**
43: **end for**
44: *SatisfyQoS*$(\boldsymbol{\lambda}, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{\beta}, \boldsymbol{p^1}, \boldsymbol{p^2}, \boldsymbol{p^{\mathrm{r}}})$
45: $p^{\mathrm{a}}_{i,n} = p^1_{i,n} + p^2_{i,n},\ \forall i,\ \forall n$
46: $r_{\mathrm{total}} = \sum_{j=1}^{J} \xi_j$

the power allocated from the battery, $p^1_{i,n}$, for each TTI. If we pass the conditions in line 15, then we determine the value of d_i as described in lines 26 to 30 of Algorithm 1. In this case, we add an extra condition regarding the sum of q_t in order to prevent division per zero in the computation of d_i. Hence, we continue in line 18, where we set l_i, s_{pow}, p and n. The last three variables are set according to ζ_i and η_i, since we will firstly raise the MCS level of the subcarrier pointed by η_i. If l_i is greater than ζ_i, then we add the remaining power β_i to $p^2_{i,n}$ in line 20. Thereafter, β_i is set to zero in line 21, since we spent all power from p_{fix} at TTI i.

If we satisfy the condition in line 19, then we add ζ_i to $p^1_{i,n}$ as we start the loop from lines 33 to 41 of Algorithm 1. This loop will gradually load $p^1_{i,n}$ until we exceed the power limit l_i. After the loop ends, we calculate the battery overflow f in line 25, and if $f > 0$ and $s_{\mathrm{pow}} < b_i$, then we apply the last power increment p. In line 27, we add β_i to $p^2_{i,n}$ since we may not run line 20 neither line 23, and we would still apply the power increase $p = \beta_i + \zeta_i$. Finally, if the overflow f remains greater than zero, then we run the procedure *AdjustPA* in line

31 by inputting $p^1_{i,n}$ rather than $p^{\mathrm{a}}_{i,n}$. Now, running procedure *AdjustPA* has some advantages, because as we take subcarrier n from user k in order to assign n to user j, we may favor a user that still needs to satisfy its QoS requirements. This is what happens most often, because k is more favored than j when the subcarrier assignment changes. In line 33, we update the battery level for the next TTI, and as the loop ends we have completed the initial power allocation for $p^1_{i,n}$ and $p^2_{i,n}$. However, we need to check the QoS requirements, and then we compute the data rate experienced by each user j along all TTIs, given by ξ_j. The calculation of ξ_j is shown in lines 36 to 43. Consequently, we run the procedure *SatisfyQoS* in line 44, and after it runs, we determine $p^{\mathrm{a}}_{i,n}$ and the total data rate as described in lines 45 and 46, respectively.

Algorithm 7 shows the procedure *SatisfyQoS*, that iterates over each user j until all users are satisfied. We start in lines 3 and 4 by sorting the channel gains for user j in descending order, where $\theta_{i,n}$ stores the sorted gains and $\mu_{i,n}$ stores the subcarriers corresponding to the sorted values. In line 5, we calculate ΔQ, that is the difference between the desired QoS and the current data rate for user j. Then, in line 6 we set the variable f, that determines the fraction of ΔQ that must be added to ξ_j when we assign subcarrier n to user j. Initially, we set f to one, what means that we wish to add the whole value of ΔQ to ξ_j in the first subcarrier assignment. If this is not possible, then we increment f until we are able to add at least $\Delta Q/f$ to ξ_j. This is done to minimize the number of changes in $u_{i,n}$, what maximizes r_{total} since the initial assignment stores the best matches between n and j in the sense of rate maximization.

The *while* loop from lines 7 to 36 runs until we satisfy the QoS constraints for user j. And after it starts, we load the variables $\theta_{i,n}$ and $\mu_{i,n}$ to $g_{i,n}$ and $\psi_{i,n}$, respectively. This is done to refresh the values of $g_{i,n}$ and $\psi_{i,n}$ after we leave the loop beginning at line 10. This loop finishes if we satisfy the QoS restrictions or if we use up all the values of $g_{i,n}$. In line 11, we determine i, that is the TTI corresponding to the greatest gain for user j found in $g_{t,1}, \forall t \in \mathcal{T}$. Next, in line 12, we set n to the subcarrier that has the greatest gain as pointed by $\psi_{i,1}$. Hence, in lines 13 and 14, m stores the achieved MCS in subcarrier n at TTI i, and k stores the user who was assigned to subcarrier n at TTI i. For TTI i, the idea is to take the power for allocating n with MCS m to user k and use it to allocate n with MCS c to user j.

Therefore, in line 15, we define $p = \beta_i + p^2_{i,n} + p^1_{i,n}$ as the power available to assign subcarrier n with MCS c to user j at TTI i. Now, we have to determine the MCS c that can be achieved with the available power p. This is done by running the *for* loop from lines 16 to 21, that computes the biggest value of c for $p > p^{\mathrm{r}}_{i,n,j,c}$. If we do not pass the condition in line 17, this means that $p > p^{\mathrm{r}}_{i,n,j,M}$, and $c = M$ since this is the last value of c in the *for* loop. Then, in line 22, we check the conditions needed to apply the new subcarrier assignment. Firstly, r_c has to add at least $\Delta Q/f$ to ξ_j. Secondly, we need to maintain user k satisfied, as we subtract r_m from ξ_k still respects the QoS constraints for user k. If the conditions are favorable, we take subcarrier n from user k and assign n to user j. Finally, this results in subtracting r_m from user k and in adding r_c to user j, as shown in lines 23 and 24.

Algorithm 7 Satisfy QoS Constraints

```
1: procedure SatisfyQoS(λ, u, ξ, β, p^1, p^2, p^r)
2:     for j ← 1 to J do
3:         θ_i = sortdesc(α_{i,n,j} : ∀n ∈ N), ∀i, where θ_i = {θ_{i,n} : ∀n}
4:         μ_i = argsortdesc(α_{i,n,j} : ∀n ∈ N), ∀i,    μ_i = {μ_{i,n} : ∀n}
5:         ΔQ = Q_j − ξ_j
6:         f = 1
7:         while ξ_j < Q_j do
8:             g_{i,n} = θ_{i,n}, ∀i, ∀n
9:             ψ_{i,n} = μ_{i,n}, ∀i, ∀n
10:            while ξ_j < Q_j and Σ_{t=1}^{T} g_{t,1} > 0 do
11:                i = argmax(g_{t,1} : ∀t ∈ T)
12:                n = ψ_{i,1}
13:                m = λ_{i,n}
14:                k = u_{i,n}
15:                p = β_i + p^2_{i,n} + p^1_{i,n}
16:                for c ← 1 to M do
17:                    if p < p^r_{i,n,j,c} then
18:                        c = c − 1
19:                        break
20:                    end if
21:                end for
22:                if r_c ≥ ΔQ/f and ξ_k − r_m ≥ Q_k then
23:                    ξ_k = ξ_k − r_m
24:                    ξ_j = ξ_j + r_c
25:                    λ_{i,n} = c
26:                    u_{i,n} = j
27:                    p = p^2_{i,n}
28:                    p^2_{i,n} = p^r_{i,n,j,c} − p^1_{i,n}
29:                    β_i = β_i + p − p^2_{i,n}
30:                end if
31:                ψ_{i,n} = ψ_{i,n+1}, for n = 1, ..., N − 1
32:                g_{i,n} = g_{i,n+1}, for n = 1, ..., N − 1
33:                g_{i,N} = 0
34:            end while
35:            f = f + 1
36:        end while
37:    end for
38: end procedure
```

Moreover, we update the MCS and the subcarrier assignment stored in $\lambda_{i,n}$ and $u_{i,n}$, as shown in lines 25 and 26. We also need to adjust the power spent in this new assignment, thus, we update $p^2_{i,n}$ as we keep the power allocated from the battery unchanged. This is done in line 28, where we set $p^2_{i,n}$ to the difference between $p^{\mathrm{r}}_{i,n,j,c}$ and $p^1_{i,n}$. Lastly, in line 29, we update β_i by adding to it the difference $p - p^2_{i,n}$, where p is the previous value of $p^2_{i,n}$ stored in line 27. In this case, β_i tends to increase because $c \leq m$. Nevertheless, it may decrease if β_i, in addition to $p^2_{i,n} + p^1_{i,n}$, has enough power to raise an extra MCS level, what may turn c equal to m. On the other hand, the total data rate tends to decrease since $c \leq m$, and in the best case, it simply remains unchanged. Hence, it becomes clear how the QoS constraints decrease r_{total}. Continuing Algorithm 7, in lines 31 and 32 we update $\psi_{i,n}$ and $g_{i,n}$, and in line 33 we set $g_{i,N}$to zero to indicate that one gain was used up. After the loop ends in line 34, we increment f by one in line 35, and after we satisfy all users the procedure *SatisfyQoS* terminates.

C. Performance Evaluation

In order to assess the effectiveness of Algorithm 6, we ran simulations for three different values of QoS requirements. Initially, we define $\mu = \frac{1}{M}\sum_{m=1}^{M} r_m$ as the average over all possible values of r_m. Next, we define $Q_j = T \cdot \mu$, $\forall j \in \mathcal{J}$, as the middle QoS requirement for each user j. This means

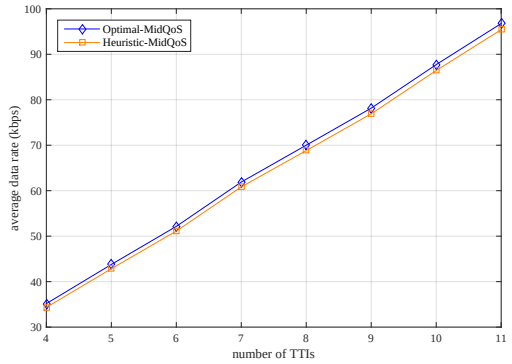

Fig. 6. Average data rate versus number of TTIs, for $M = 16$, $N = 12$, $J = 6$, $T = 4$ to 11, $h_{\max} = 112$ mW, $b_{\max} = 240$ mW, $p_{\text{fix}} = 96$ mW, $S = 7$, and Q_j for middle QoS.

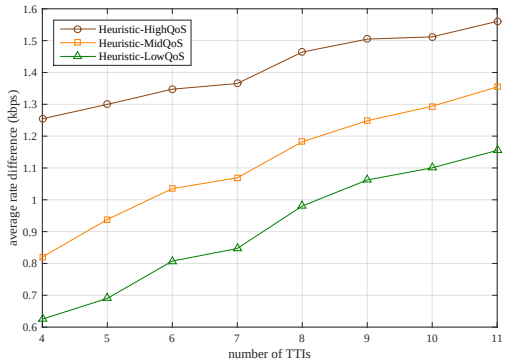

Fig. 7. Average rate difference versus number of TTIs, for $M = 16$, $N = 12$, $J = 6$, $T = 4$ to 11, $h_{\max} = 112$ mW, $b_{\max} = 240$ mW, $p_{\text{fix}} = 96$ mW, $S = 7$, and Q_j for high, middle and low QoS according to each curve.

that, at the end of T TTIs, all users have to achieve at least the data rate that is equivalent to experiment μ bps at each TTI. Consequently, we define the low and high QoS requirements for each user j as $Q_j = (T-1)\cdot\mu$ and $Q_j = (T+1)\cdot\mu$, respectively. We also define $p_{\text{fix}} = 96$ mW, and we decrease the values of N and J to $N = 12$ and $J = 6$. This reduction aims to minimize the effects of high memory consumption and high computational time to obtain the optimal solution. Additionally, we increase the battery capacity to $b_{\max} = 240$ mW in order to reduce the occurrences of battery overflow, since we added p_{fix} to the power supply. The Algorithm 6 generated the results for the Heuristic-HighQoS, Heuristic-MidQoS and Heuristic-LowQoS curves. The CPLEX solver generated the results for the Optimal-MidQoS and Optimal-LowQoS curves. Figure 6 shows the average data rate over 1,000 instances of the problem for $T = 4$ to 11.

Also, this figure shows the results for optimal and heuristic solutions when Q_j assumes a middle QoS requirement. We realize that the difference between both solutions increases as T grows but, as we shall see, not in the same way for each value of QoS. Figure 7 shows how these differences grow as we vary the required QoS. We observe that, for low and middle QoS scenarios, the differences from the optimal solution grow faster than in high QoS scenarios. On the other hand, the heuristic solution presents the worst performance with high QoS, because it performs extra changes in the subcarrier assignment on the goal to satisfy the QoS constraints.

Curiously, even though the differences are greater, they slowly increased in the high QoS problem, specially in the interval between $T = 9$ and $T = 10$. Like this, the curves in Figure 7 tend to become closer since we have more time to fulfill the QoS requirements as T grows. For example, in $T = 4$ the HighQoS and MidQoS curves differ by 434 bps, and in $T = 11$ they differ only by 206 bps. Thus, higher QoS demands have minor effects in the heuristic solution for bigger values of T. Despite this fact, the difference between optimal and heuristic will always grow because the problem is more complex when T increases. Moreover, we calculated the NMSE for each value of QoS in order to measure the impact of higher QoS in the heuristic solution. The NMSE for LowQoS, MiddleQoS and HighQoS, according to the samples in Figure 7, are: $\text{NMSE}_{\text{low}} = 0.201 \times 10^{-3}$, $\text{NMSE}_{\text{mid}} = 0.301 \times 10^{-3}$ and $\text{NMSE}_{\text{high}} = 0.477 \times 10^{-3}$. As expected, the error grows as we raise the required QoS because higher demands complicate the resource management, what results in greater deviation from the optimal solution.

After we compared two different solutions for the same QoS scenarios, now we compare the same solution for different QoS scenarios. This is shown in Figure 8 that compares the progress of the heuristic solution for low and high QoS requirements, where the performance improves when Q_j decreases. This figure displays the opposite behavior of that shown in Figure 6, since the difference between the heuristic solutions decreases as T grows.

This behavior is explained by the fact that, as we add extra TTIs, we have more available power and more combinations for setting up $p^{\text{a}}_{i,n}$ and $u_{i,n}$. Therefore, the loss in r_{total} caused by higher QoS requirements is reduced as T increases. This is clearly visible in Figure 9, that shows how the differences from the high QoS scenario decrease for the optimal and heuristic solutions. This figure aims to demonstrate how lower QoS demands result in higher values of r_{total}. This is done by comparing Heuristic-HighQoS to Heuristic-MidQoS and Heuristic-LowQoS. Similarly, we compare Optimal-HighQoS to Optimal-MidQoS and Optimal-LowQoS. In Figure 9, it is noticeable that the optimal solution has superior performance over the heuristic algorithm, because the optimal curves are well below the heuristic curves. This shows that r_{total} presents greater variation in the heuristic solution, meaning that the heuristic is more affected by the QoS requirements.

Furthermore, the curves for middle QoS are below its respective curves for low QoS, what means that the differences decrease when Q_j grows. In addition, we see that the distance between the heuristic curves is much greater in comparison to the optimal curves. This is clear in $T = 11$, since the heuristic curves differ by 243 bps while the optimal curves differ by 44 bps. Other observations include the faster decay displayed by the heuristic solution, and the considerable reduction in the rate difference when compared to Figure 7. For example, in Figure 9 the greatest difference is 0.89 kbps when $T = 4$ in the Heuristic-LowQoS curve, whereas Figure 7 shows 1.56 kbps of difference when $T = 11$ in the Heuristic-HighQoS curve. Thus, the differences are much bigger between different solutions with same QoS than between same solutions with

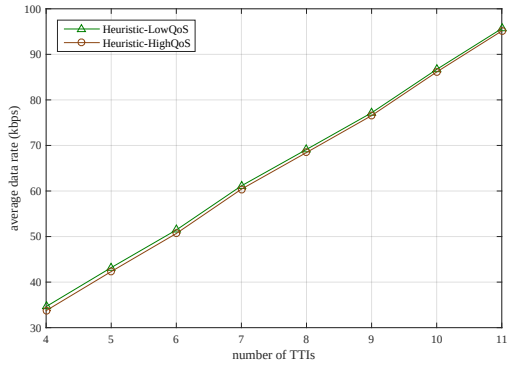

Fig. 8. Average data rate versus number of TTIs, for $M = 16$, $N = 12$, $J = 6$, $T = 4$ to 11, $h_{\max} = 112$ mW, $b_{\max} = 240$ mW, $p_{\text{fix}} = 96$ mW, $S = 7$, and Q_j for low and high QoS according to each curve.

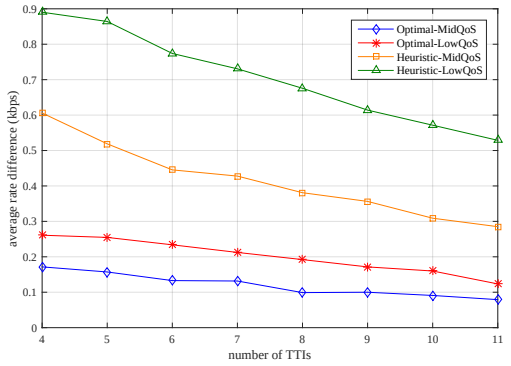

Fig. 9. Average rate difference versus number of TTIs, for $M = 16$, $N = 12$, $J = 6$, $T = 4$ to 11, $h_{\max} = 112$ mW, $b_{\max} = 240$ mW, $p_{\text{fix}} = 96$ mW, $S = 7$, and Q_j for low and middle QoS according to each curve.

different QoS. Lastly, we analyze the computational time for optimal and heuristic solutions as Q_j increases.

We define the pair (t_1, t_2), where t_1 and t_2 are the average execution times in ms for optimal and heuristic solutions, respectively. Then, for $T = 6$, we obtained $E_6 = \{(t_1, t_2) : (5466, 45.5), (6135, 48.3), (6683, 51.4)\}$, where the three pairs correspond to low, middle and high QoS, respectively. We also define the triple $G_T = (g_1, g_2, g_3)$, where g_i is the ratio t_1/t_2 with $i = 1, 2, 3$ corresponding to low, middle and high QoS, respectively. Hence, for $T = 6$, we have $G_6 = (120, 127, 130)$ with g_i truncated to an integer. The results in E_6 show that the execution time grows considerably for the optimal solution as Q_j increases, while the time for the heuristic solution grows at reduced rates. Additionally, G_6 shows a good compromise between performance and complexity, since for low QoS, the heuristic runs 120 times faster than the optimal solution at the cost of a small loss in data rate. The results in G_6 improve as Q_j grows, because the gain in computational time increases to 127 and 130 for middle and high QoS. For higher values of T, the trade-off between performance and complexity becomes more evident. In order to illustrate this, for $T = 8$ we have $E_8 = \{(t_1, t_2) : (11523, 61.8), (12108, 63.3), (12681, 65.5)\}$ and $G_8 = (186, 191, 193)$. In this case, the times for optimal solution almost doubled in comparison to $T = 6$, whereas the times for heuristic solution increased by small amounts. Consequently, the gains in G_8 achieved a maximum of 193 at the cost of losing 1.47 kbps in the high QoS scenario.

VII. Conclusions

In this paper we studied the problem of resource allocation for rate maximization in OFDMA systems with an EH Base Station transmitting to several users. Firstly, we formulated the problem in an offline scenario in the form of an ILP. Secondly, we proposed a heuristic solution that, according to simulation results, achieves near-optimal performance at the cost of low computational complexity in the simulated scenario. We also implemented the discrete relation between SNR and data rate through MCSs, that is a more realistic assumption than the continuous mapping proposed in several studies referenced by this work. Finally, our EH model follows a Markov chain that, differently from other models found in literature, has states represented by continuous intervals, giving the advantage of simulating the actual nature of the energy arrivals. To the best of our knowledge, the problem formulation proposed in section IV has not been presented in literature yet, and the heuristic in Algorithm 1 introduces a novel solution for problems that deal with EH technology in wireless communications.

In addition, we performed an analysis of the online case as we developed two different solutions only with causal information. The results in section V show the advantage of saving energy for future TTIs, and how performance improves according to better predictions of the harvested energy. Next, in section VI, we presented a new problem formulation that considers QoS constraints to be satisfied for all users. This problem describes a new scenario where the BS is powered by a hybrid power system, and the subcarriers are loaded with power from two types of sources: renewable and non-renewable. Afterwards, we proposed a heuristic solution that solves the problem for offline settings, and simulation results demonstrated the considerable compromise between performance and complexity. As far we know, the algorithms shown in section V, the problem formulation described in section VI, and the solution presented in Algorithm 6 are new contributions still not available in the literature.

References

[1] Jair A. de Carvalho, Juno V. Saraiva, F. Rafael M. Lima, Tarcisio F. Maciel, and F. Rodrigo P. Cavalcanti, "Resource Allocation for OFDMA Systems and Energy Harvesting Communications in Multi-User Offline Scenarios," in *XXXV Brazilian Telecommun. Symp. (SBrT'17)*, Sep. 2017.

[2] A. Minasian, S. Shahbazpanahi, and R. S. Adve, "Energy Harvesting Cooperative Communication Systems," *IEEE Trans. Wireless Commun.*, vol. 13, no. 11, pp. 6118-6131, Nov. 2014. doi: 10.1109/TWC.2014.2320977

[3] C. K. Ho and R. Zhang, "Optimal Energy Allocation for Wireless Communications with Energy Harvesting Constraints," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4808-4818, Sep. 2012. doi: 10.1109/TSP.2012.2199984

[4] M. R. Zenaidi, Z. Rezki, and M. Alouini, "Performance Limits of Online Energy Harvesting Communications With Noisy Channel State Information at the Transmitter," *IEEE Access*, vol. 5, pp. 1239-1249, Mar. 2017. doi: 10.1109/ACCESS.2017.2654454

[5] O. Ozel, K. Tutuncuoglu, J. Yang, S. Ulukus, and A. Yener, "Transmission with Energy Harvesting Nodes in Fading Wireless Channels: Optimal Policies," *IEEE Journal Sel. Areas Commun.*, vol. 29, no. 8, pp. 1732-1743, Sep. 2011. doi: 10.1109/JSAC.2011.110921

[6] Y. Li and J. Niu, "Forecast of Power Generation for Grid-Connected Photovoltaic System Based on Markov Chain," IEEE Asia-Pacific Power and Energy Engineering Conference, vol 1, pp. 652-655, Oct. 2009.

[7] C. K. Ho, D. K. Pham, and C. M. Pang, "Markovian Models for Harvested Energy in Wireless Communications," presented at the IEEE Int'l Conf. Commun. Syst. (ICCS), Singapore, Nov. 17-20, 2010.
[8] P. Poggi, G. Notton, M. Muselli, and A. Louche, "Stochastic Study of Hourly Total Solar Radiation in Corsica Using a Markov Model," *Int'l J. Climatology*, vol. 20, no. 14, pp. 1843-1860, Nov. 2000. doi: 10.1002/1097-0088(20001130)20:14<1843::AID-JOC561>3.0.CO;2-O
[9] R. Loodaricheh, S. Mallick and V. Bhargava, "Admission Control and Power Allocation for Energy Harvesting Systems with QoS Provisioning," 3rd Int'l Conf. on Signal Process. and Integrated Networks, Feb. 2016.
[10] M. Qin, Q. Yang, J. Yang, D. Park, and K. S. Kwak, "Energy-Aware Resource Allocation for OFDMA Wireless Networks with Hybrid Energy Supplies," *IET Commun. J.*, vol. 11, iss. 11, pp. 1671-1678, Jul. 2017. doi: 10.1049/iet-com.2016.1328
[11] I. Ahmed, A. Ikhlef, R. Schober, and R. Mallik, "Joint Power Allocation and Relay Selection in Energy Harvesting AF Relay Systems," *IEEE Wireless Commun. Lett.*, vol. 2, no. 2, pp. 239-242, Apr. 2013. doi: 10.1109/WCL.2013.012513.130007
[12] M. Zhang, J. Zhang, Y. Wang, Y. Dong, S. Wang, "Power Allocation for Uplink Multi-user Energy Harvesting Relay Systems with Sleep Mode," 10th Int'l Conf. on Commun. and Networking in China, Aug. 2015.
[13] D. Ng, E. Lo, and R. Schober, "Energy-Efficient Resource Allocation in OFDMA Systems with Hybrid Energy Harvesting Base Station," *IEEE Trans. Wireless Commun.*, vol. 12, no. 7, pp. 3412-3426, Jul. 2013. doi: 10.1109/TWC.2013.052813.121589
[14] M. Maso, S. Lakshminarayana, T. Quek, H. Poor, "Energy Harvesting for Self-sustainable OFDMA Communications," IEEE Global Communications Conference (Globecom), Dec. 8-12, 2014.
[15] R. Yadav, K. Singh, A. Gupta, and A. Kumar, "Optimal Energy-Efficient Resource Allocation in Energy Harvesting Cognitive Radio Networks with Spectrum Sensing," 84th IEEE Veh. Tech. Conference, Sep. 2016.
[16] T. Hoan and I. Koo, "Multi-Slot Spectrum Sensing Schedule and Transmitted Energy Allocation in Harvested Energy Powered Cognitive Radio Networks Under Secrecy Constraints," *IEEE Sensors Journal*, vol. 17, no. 7, pp. 2231-2240, Apr. 2017. doi: 10.1109/JSEN.2017.2658608
[17] Y. Zhao, V. Leung, C. Zhu, H. Gao, H. Ji, "Energy-Efficient Sub-Carrier and Power Allocation in Cloud-Based Cellular Network With Ambient RF Energy Harvesting," *IEEE Access*, vol. 5, pp. 1340-1352, Feb. 2017. doi: 10.1109/ACCESS.2017.2667678
[18] IBM, "IBM ILOG CPLEX Optimizer." [Online]. Available: http://www-01.ibm.com/software /integration/optimization/cplex-optimizer
[19] D. Hughes-Hartogs, "Ensemble Modem Structure for Imperfect Transmission Media", Patent US 4 833 706, May, 1989.
[20] C. Mehlfhrer, M. Wrulich, J. C. Ikuno, D. Bosanska, and M. Rupp, "Simulating the long term evolution physical layer," in *Signal Processing Conference, 2009 17th European*, pp. 1471-1478, IEEE, 2009.
[21] 3GPP, "Evolved universal terrestrial radio access (E-UTRA); physical layer procedures," 2009. Third Generation Partnership Project, Tech. Rep. TR 36.213 V8.6.0, Mar. 2009.
[22] A. Poli and M. Cirillo, "On the Use of the Normalized Mean Square Error in Evaluating Dispersion Model Performance," *Atmospheric Environment*, vol. 27A, no. 15, pp. 2427-2434, Apr. 1993. doi: 10.1016/0960-1686(93)90410-Z

Jair Alves de Carvalho is currently a B.Sc. student in Computer Engineering from the Federal University of Ceará (UFC), Sobral, Brazil. From 2012 to 2014, as an undergraduate researcher, he engaged in a scientific initiation project funded by FUNCAP for the study of NFC technology in mobile payments. From 2014 to 2015, Carvalho participated in an exchange program at SUNY Oswego, USA, funded by CAPES as a scholarship student from the Science Without Borders program. Lastly, from 2016 to 2018, as an undergraduate researcher, he engaged in another scientific initiation project funded by FUNCAP for the study of radio resource allocation in energy harvesting communications. In 2017, Carvalho has won the best paper award at the Brazilian Telecommunications Symposium 2017 (SBrT' 17), São Pedro, Brazil. His research interests include radio-frequency technologies, radio resource allocation and energy harvesting communications.

Francisco Rafael Marques Lima received the B.Sc. degree with honors in Electrical Engineering in 2005, and M.Sc. and D.Sc. degrees in Telecommunications Engineering from the Federal University of Ceará, Fortaleza, Brazil, in 2008 and 2012, respectively. In 2008, he has been in an internship at Ericsson Research in Luleå, Sweden, where he studied scheduling algorithms for LTE system. Since 2010, he has been a Professor of Computer Engineering Department of Federal University of Ceará, Sobral, Brazil. Prof. Lima is also a researcher at the Wireless Telecom Research Group (GTEL), Fortaleza, Brazil, where he works in projects in cooperation with Ericsson Research. In 2016, prof. Lima has been awarded a research productivity scholarship by FUNCAP. He has published several conference and journal articles as well as patents in the wireless telecommunications field. His research interests include radio resource allocation algorithms for QoS guarantees in scenarios with multiple services, resources, antennas, and users.

Tarcisio Ferreira Maciel received his B.Sc. and M.Sc.degrees in Electrical Engineering from the Federal University of Ceará (UFC) in 2002 and 2004, respectively, and his Dr.-Ing. degree from the Technische Universität Darmstadt (TUD), Germany, in 2008, also in Electrical Engineering. Since 2001, he has actively participated in several projects in a technical and scientific cooperation between Wireless Telecom Research Group (GTEL), UFC, and Ericsson Research. From 2005 to 2008, he was a research assistant with the Communications Engineering Laboratory, TUD. Since 2008, he has been a member of the Post-Graduation Program in Teleinformatics Engineering, UFC. In 2009, he was a Professor of computer engineering with UFC-Sobral and since 2010, he has been a Professor with the Center of Technology, UFC. His research interests include radio resource management, numerical optimization, and multiuser/multiantenna communications.

Francisco Rodrigo Porto Cavalcanti received the D.Sc. degree in Electrical Engineering from the State University of Campinas, São Paulo, Brazil, in 1999. Upon graduation, he joined the Federal University of Ceará (UFC), where he is currently an Associate Professor and holds the Wireless Communications Chair at the Department of Teleinformatics Engineering. In 2000, he founded, and since then has directed, the Wireless Telecom Research Group (GTEL), which is a research laboratory based on Fortaleza, Brazil, which focuses on the advancement of wireless telecommunications technologies. At GTEL, he manages a 19-year long program of research projects in wireless communications sponsored by Ericsson Research. In 2017 he was a visiting researcher to Ericsson's main site at Stockholm, Sweden. Prof. Cavalcanti has produced a varied body of work including books, papers, patents and software dealing with wireless access networks and technologies. Prof. Cavalcanti is a distinguished researcher of the Brazilian Scientific and Technological Development Council for his technology development and innovation record. He also holds a Leadership and Management professional certificate from the Massachusetts Institute of Technology, Cambridge, USA.