Massive Access in Beyond 5G IoT Networks with NOMA: NP-hardness, Competitiveness and Learning

Zoubeir Mlika, and Soumaya Cherkaoui, *Senior Member, IEEE*

***Abstract*—This paper studies the problem of online user grouping, scheduling and power allocation in beyond 5G cellular-based Internet of things networks. Due to the massive number of devices trying to be granted to the network, non-orthogonal multiple access method is adopted in order to accommodate multiple devices in the same radio resource block. Different from most previous works, the objective is to maximize the number of served devices while allocating their transmission powers such that their real-time requirements as well as their limited operating energy are respected. First, we formulate the general problem as a mixed integer non-linear program (MINLP) that can be transformed easily to MILP for some special cases. Second, we study its computational complexity by characterizing the NP-hardness of different special cases. Then, by dividing the problem into multiple NOMA grouping and scheduling subproblems, efficient online competitive algorithms are proposed. Further, we show how to use these online algorithms and combine their solutions in a reinforcement learning setting to obtain the power allocation and hence the global solution to the problem. Our analysis are supplemented by simulation results to illustrate the performance of the proposed algorithms with comparison to optimal and state-of-the-art methods.**

***Index Terms*—Internet of things, machine-to-machine, non-orthogonal multiple access, scheduling, power allocation, NP-hardness, online competitive algorithms, learning algorithms.**

I. Introduction

Tens of billions of objects will be connected to the Internet in the near future [1]. These objects form the well-known Internet of things (IoT) [2], which is one of the promising applications in future wireless networks, including fifth generation (5G) standard and beyond 5G (B5G). To realize IoT, machine-to-machine (M2M) communication is proposed where objects communicate with each others without (or with little) human interactions [3]. The applications of M2M in IoT include smart cities, smart grids, industrial automation, healthcare, intelligent transportation systems, to name only few. Due to the maturity of cellular networks and their ability to provide wide-area coverage, the integration of M2M communication with cellular networks can be viewed as a viable solution to the realization of such applications. For example, narrow-band IoT (NB-IoT) [4] is proposed by the 3rd generation partnership project as a cellular-based M2M communication network using the long term evolution (LTE) standard.

The main requirement of cellular-based M2M communication is massive connectivity, i.e, a massive number of objects (or interchangeably called devices) communicating with each other through cellular networks needs to be supported in the future [5, 6]. For example, in [7], it was expected that more than two billions objects will be directly connected to cellular networks. Besides massive connectivity, M2M traffic is generally different from traditional cellular traffic. It is sporadic and characterized by small-sized packets. Thus, maximizing the sum-rate is not the first priority anymore in cellular-based M2M networks. Further, the cellular-based M2M networks have more stringent requirements including stringent latency and energy-efficiency requirements. Consequently, new resource allocation methods that take into account these requirements are of great importance in such networks.

In this paper, we study an online resource allocation problem in cellular-based M2M networks. We deviate from most previous works that (i) study the sum-rate-related objectives and (ii) use stochastic Lyapunov framework to solve the problem. Particularly, we consider a more important objective that is well suited to the massive access problem in such networks. In other words, we maximize the number of served devices (NSD). To accommodate a large number of devices, non-orthogonal multiple access (NOMA) technique is used. The problem is therefore to maximize the NSD while (i) grouping them into the available resource blocks, (ii) scheduling their transmission to respect their real-time requirements, and (iii) allocating their transmission powers to respect their limited operating energy levels. This problem is solved in the online computation [8] and the learning [9] frameworks in order to provide competitive and learning algorithms. This problem is called Grouping & Power Allocation (GPA).

Remark 1 (Maximizing the NSD is Better)**.** *The following simple example shows that maximizing the sum-rate can be achieved while serving only few devices. Assume that time is divided into two slots and there are two devices. The channel gain of the first device over the two slots is* $[6, 6]^\top$ *whereas the channel gain of the second device over the two slots is* $[2, 1]^\top$*. The maximum transmission powers of the two devices is* $[1, 1/2]^\top$*. Maximizing the sum-rate subject to power constraints over the two slots would serve only the first device on the two slots (with allocated transmission power of* $1/2$ *on each slot)—achieving a sum-rate of* 4 *bps/Hz. However, serving the second device in the first slot with allocated transmission power of* $1/2$ *and the first device in the second slot with allocated transmission power of* 1*, would achieve a sum-rate of* $1 + \lg 7$ *bps/Hz (*< 4 *bps/Hz). Therefore, a larger sum-rate is achieved with fewer served devices.*

A. Related Work

Most previous works considered the problem of resource allocation in M2M networks from the perspective of either maximizing the sum-rate or minimizing the sum-power (or the energy consumption) or other related objectives. Further, few research papers focus on the online competitive analysis and learning frameworks. A detailed survey on radio resource management in M2M networks is given in [3].

In [10], the authors give a short survey for the problem of uplink grant in M2M networks. Comparison between coordinated and uncoordinated access are shown. The problem is formulated as a prediction problem: in order to reduce delay, devices do not have to send random access requests to the BS, instead, the BS allocates its resources to the devices by predicting which device has packets to send. Further, the authors developed a two-stage solution based on machine learning. In [11], a multi-armed bandit approach is proposed to solve the problem of fast uplink grant access in M2M networks. The objective is to maximize a utility function that is a combination of data rate, access delay, and value of data packets. Since the set of possible actions is not known in advance, a *sleeping* multi-armed bandit technique is used. In [12], the authors studied the resource management problem in green IoT multihop networks. IoT devices obtain energy through either grid power or harvesting. The problem is formulated as a stochastic optimization problem where the objective is to maximize a time-average network utility combined with energy purchasing costs. Lyapunov optimization techniques are used to obtain a stable solution in large- and small-time scales. In [13], the authors formulate a dynamic scheduling and power allocation problem in IoT networks using NOMA technique. A stochastic optimization problem is formulated. The objective is to minimize the long average power consumption over time subject to maximum transmission powers, scheduling and long average rate constraints. Well-known techniques are used to derive the Lyapunov function and the upper bound on the drift plus penalty. The problem is transformed into a set of static optimization problems that are solved iteratively. Then, branch and bound technique is used to solve the problem. In [14], the authors study clustering and power allocation in NOMA systems to answer the following question: how to group the users into the resource blocks and how to allocate transmission powers in order to maximize the system throughput while guaranteeing minimum rate requirements of the users and without exceeding maximum transmission powers. The solutions is divided into (1) user clustering by developing algorithm that satisfy the successive interference cancellation (SIC) constraints and (2) power allocation using Lagrangian methods. In [15], a dynamic power control and user pairing problem is solved in delay-constrained multiple access networks with hybrid OMA and NOMA techniques. The objective is to minimize the long-term time-average transmission power while guaranteeing stable queues and minimum time-average rate. By fixing the user pairing, power allocation is derived. Further, for given power allocation, the user pairing problem is solved using matching techniques. [16] considers hybrid OMA and NOMA techniques to solve the problem of maximizing the energy-efficiency while guaranteeing minimum rate requirements and maximum transmission power. Swap matching algorithm is proposed to solve the user pairing problem under fixed power allocation. Once the user pairing is found, the power allocation is solved while maximizing the ratio of rate to total power consumption. The authors of [17] study the problem of maximizing the NSD to solve the uplink access problem in NOMA systems. Specifically, the objective is to maximize the NSD while allocating the channel to the devices and further guaranteeing the minimum rate requirements and the maximum transmission power. The authors first find the power allocation by solving the feasibility problem of minimum rate requirements. The NOMA channel assignment problem is solved by reducing it to a maximum independent set problem. The analysis is only given for device pairing (two devices per NOMA group) and there is no NP-hard proof nor real-time requirements for the devices. In [18], the authors study the problem of minimizing the maximum access delay under minimum rate requirements and prove its NP-hardness. They divide it into user scheduling and power control subproblems. The user scheduling is solved using a graph cutting method while the power control is solved using an iterative algorithm. The authors of [19] study the problem of maximizing the NSD in NOMA systems. They proposed a mathematical programming formulation to solve the problem. No NP-hardness is provided. Further, there are no real-time requirements for the devices.

In the previously cited papers, there are some important research gaps that we fill in this work. First, the majority of the works focus on sum-rate related objectives. Second, even when the objective is the NSD, the previously studied problems and our problem are fundamentally different and there are lacks of studies on (1) the computational complexity of the problem, (2) competitive and learning algorithms, and (3) stringent requirements including real-time, rate and power requirements.

B. Contributions

The main contributions of this work are summarized in the following list.

- We use mathematical programming techniques to model GPA as a mixed integer non-linear program, which can be easily transformed into an integer linear program for some special cases of interest.
- We characterize the computational complexity of GPA by studying its NP-hardness in different cases. We give a complete analysis of each case by either presenting a formal proof of NP-hardness or a polynomial-time algorithm.
- We start by analyzing GPA in some important special cases, where the problem involves NOMA grouping and scheduling, and we derive online competitive algorithms to solve it.
- We propose to combine our proposed online competitive algorithms in a machine learning setting in order to solve GPA in the general case. That is, we propose learning algorithms that find the power allocation solution using the NOMA grouping and scheduling solutions.

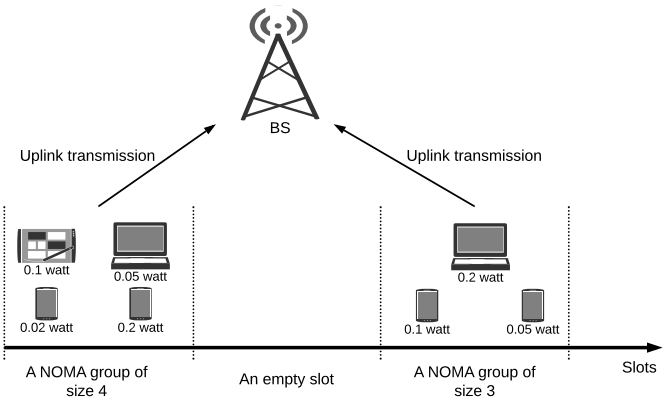


Fig. 1: System Model with a single frame. Power allocation is shown below each device. An empty slot represents the case of constraints violation (e.g., real-time or power constraints).

C. Organization

The paper is organized as follows. Section II presents the system model. Section III formulates GPA as a mathematical program. Section IV studies its NP-hardness in different cases and discusses its offline solutions. Section V presents the proposed online competitive solutions whereas Section VI presents the proposed learning algorithms. Section VII shows some simulation results, and finally, section VIII draws some conclusions.

D. Notations

Lowercase and boldface letters denote vectors whereas uppercase and boldface letters denote matrices and three dimensional arrays. A set of n elements is denoted by $[n] \coloneqq \{1, 2, \ldots, n\}$ and its cardinality is denoted by $|[n]| \coloneqq n$. Interval of integers is denoted by $\{a..b\}$ including a and b. The interval $[0, x]$ is sampled using the power level $\tau \geqslant 1$ to obtain the set $[x]_\tau \coloneqq \{0, x/\tau, 2x/\tau, \ldots, x\}$ with cardinality $|[x]_\tau| = \tau + 1$. The symbol $\mathcal{O}(\cdot)$ denotes the big-O notation.

II. System Model

We consider a cellular-based M2M network composed of one base station (BS) and m devices. Time is divided into k frames where each frame is composed of n time-slots with unit length each. In each frame, device i may have a packet to send. The length (in bits) of device i's packet in frame t is $L_i^t \geqslant 0$. (In general, we may have $L_i^t = 0$ for some t; meaning that i has no packet to send in frame t.) The arrival time and the deadline of device i's packet in frame t are denoted by a_i^t and d_i^t, respectively. The considered traffic pattern is similar to but more general than the well-known frame-synchronized traffic pattern [20, 21]. Device i has $\bar{e}_i$ units of energy stored in its battery. For simplicity, a resource block (RB) is represented simply by a time-slot (but time/frequency RBs can also be used). Every frame has n RBs and we denote a RB by the pair (j, t) for time-slot j of frame t. A RB has a bandwidth of W Hz. An example of the system model is given in Fig. 1.

The wireless channel between device i and the BS using RB (j, t) is given by h_{ij}^t, which may include fast and slow fading. Let x_{ij}^t be a binary variable that is 1 if and only if device i is served using RB (j, t). Also, let p_{ij}^t denotes the transmission power of device i using RB (j, t). Finally, let z_i^t be a binary variable that is 1 if and only if device i is served in frame t. We use $\mathbf{X}$, $\mathbf{P}$, and $\mathbf{Z}$ to denote the multidimensional variables corresponding to x_{ij}^t, p_{ij}^t and z_i^t, respectively.

The signal to interference-plus-noise ratio (SINR) achieved by device i when served by the BS using RB (j, t) is given by:

$$SINR_{ij}^t(\mathbf{X}, \mathbf{P}) = \frac{x_{ij}^t p_{ij}^t g_{ij}^t}{1 + I_{ij}^t(\mathbf{X}, \mathbf{P})}, \tag{1}$$

where $g_{ij}^t = |h_{ij}^t|^2$ is the channel power gain[1] and $I_{ij}^t(\mathbf{X}, \mathbf{P})$ is the power of the interference coming from other devices transmitting using RB (j, t).

To serve a large number of devices, power-domain NOMA technique is used [14], where a group of devices are transmitting to the BS using the same RB. Successive interference cancellation (SIC) is used at the BS for the decoding. Let $\mathbb{A}_j^t$ be the set of devices that are transmitting on RB (j, t). It is well-know to use the highest channel decoding order in uplink NOMA [14, 16]. In other words, the interference received by device i comes from all devices that have lower channel gains. We order the devices in $\mathbb{A}_j^t$ with respect to g_{ij}^t to obtain a new set $\mathbb{B}_{ij}^t \coloneqq \{i' \in \mathbb{A}_j^t : g_{i'j}^t < g_{ij}^t\}$. With that said, the interference received by device i using RB (j, t) can be calculated as follows:

$$I_{ij}^t(\mathbf{X}, \mathbf{P}) = \sum_{i' \in \mathbb{B}_{ij}^t} x_{i'j}^t p_{i'j}^t g_{i'j}^t. \tag{2}$$

The achievable rate between device i and the BS using RB (j, t) is given by:

$$R_{ij}^t(\mathbf{X}, \mathbf{P}) = W \lg(1 + \mathrm{SINR}_{ij}^t(\mathbf{X}, \mathbf{P})). \quad \textit{[in bits/s]} \tag{3}$$

The objective of GPA is to maximize the number of times the devices are served during the time horizon of k frames. This has to be done while grouping the devices in each RB and allocating their transmission powers. GPA guarantee that the served devices respect (i) their data requirements by sending all of their bits in each frame when they are served, (ii) their maximum transmission powers, and (iii) their arrival times and deadlines.

GPA is studied in the *online* scenario under the full information assumption [22]. That is, each device knows only the current and previous information of all other devices (including itself), i.e., at time-slot j of frame t, the devices get to know the channel gains g_{ij}^t, the arrival times a_i^t and deadlines d_i^t and the data requirements L_i^t for all i. The local information assumption [22], where each device knows only its own information in an online manner, is left for our future work.

To be able to solve GPA optimally (using off-the-shelf solvers) in the offline scenario, we formulate it as a mathe-

[1]We normalize the channel power gain to get a noise power of 1.

matical program in the next section.

III. Problem Formulation

GPA is formulated as follows:

$$\underset{\mathbf{X},\mathbf{P},\mathbf{Z}}{\text{maximize}} \quad \sum_{i=1}^{m}\sum_{t=1}^{k} z_i^t \tag{P1a}$$

$$\text{subject to} \quad x_{ij}^t, z_i^t \in \{0,1\}, p_{ij}^t \geqslant 0, \forall i,j,t, \tag{P1b}$$

$$\sum_{j=1}^{n} R_{ij}^t(\mathbf{X},\mathbf{P}) \geqslant L_i^t z_i^t, \forall i,t, \tag{P1c}$$

$$p_{ij}^t \leqslant \bar{e}_i x_{ij}^t, \forall i,j,t, \tag{P1d}$$

$$\sum_{j=1}^{n}\sum_{t=1}^{k} p_{ij}^t \leqslant \bar{e}_i, \forall i, \tag{P1e}$$

$$x_{ij}^t = 0, \forall i, j \notin \{a_i^t..d_i^t - 1\}, t, \tag{P1f}$$

$$x_{ij}^t \leqslant z_i^t, \forall i,j,t, \tag{P1g}$$

$$\sum_{i=1}^{m} x_{ij}^t \leqslant M, \forall j,t, \tag{P1h}$$

$$z_i^t \leqslant L_i^t, \forall i,t. \tag{P1i}$$

The objective function in (P1a) maximizes the number of times the m devices are served during the time horizon of k frames. Constraints (P1b) list the optimization variables. Constraints (P1c) guarantee a minimum of L_i^t bits for device i when served in frame t. Constraints (P1d) relate the variables x_{ij}^t and p_{ij}^t: if $x_{ij}^t = 0$, then so is p_{ij}^t and if $x_{ij}^t = 1$, then device i can use any p_{ij}^t that is less than the maximum transmission power $\bar{e}_i$ (if device i is not served in slot j of frame t, then it is not using any power and it is using at most the maximum otherwise). Constraints (P1e) restrict the limit of transmission powers used by device i in all RBs (j,t). Constraints (P1f) force $x_{ij}^t = p_{ij}^t = z_i^t = 0$ for all i,j,t, whenever device's i packet has not yet arrived or its deadline is already due. Constraints (P1g) relate the variable x_{ij}^t and z_i^t: if $z_i^t = 0$ then so is x_{ij}^t (if device i is served in frame t, then there must exists a slot j when it is served). Constraints (P1h) limit the number of devices served in RB (j,t) to a positive number $M \leqslant m$. Finally, constraints (P1i) mark device i as not-yet-served in frame t if it has no packet to send.

We can see that (P1) is non-linear and non-convex due to the multiplication of $\mathbf{X}$ and $\mathbf{P}$ in constraints (P1c). Note that the variables $\mathbf{X}$ and $\mathbf{P}$ are equivalent and can be related, i.e., $p_{ij}^t > 0$ if and only if $x_{ij}^t = 1$. Thus, we can get rid of $\mathbf{X}$ from constraints (P1c) by writing $R_{ij}^t(\mathbf{X},\mathbf{P}) = R_{ij}^t(\mathbf{1},\mathbf{P})$. Despite this fact, (P1) is still mixed integer non-linear program, which is very hard to solve in general.

In the sequel, transmission powers of each device is assumed to belong to some discrete set. This is a realistic assumption in many real systems [23–25]. Although the continuous power assumption can ease mathematical derivations through mathematical programming, GPA is still NP-hard even under the discrete power assumption (as it will be shown shortly). Under the discrete power assumption, every device i can choose its transmission power p_{ij}^t from the set $[\bar{e}_i]_{\tau_i} := \{0, \bar{e}_i/\tau_i, 2\bar{e}_i/\tau_i, \ldots, \bar{e}_i\}$, where $\tau_i \geqslant 1$ is the power level of device i. An important case, called the binary power (BP) case, is when $\tau_i = 1$ for all i, and thus $[\bar{e}_i]_1 = \{0, \bar{e}_i\}$. We call the general case of $[\bar{e}_i]_{\tau_i}$ the general power (GP) case. The BP case is worth studying because it helps understand the intrinsic difficulty of the problem and helps in characterizing the structure of the solution in more general cases.

Remark 2 (Reduction from Multiple Frames to Single Frame)**.** *Since the devices have to be served during k frames while respecting their limited operating energy levels, we observe that, in any optimal solution to GPA, every frame will be associated an amount of allocated power for each device. Thus, if we could find how much power to allocate to the devices in each frame, we could reduce the problem to k single frame problems and solve each one separately. We start by analyzing the problem in the case of a single frame (the superscript t is dropped from all the notations). Then, we solve the problem in the more general case of multiple frames by applying machine learning techniques.*

In the next section, we study GPA in the offline scenario with a single frame. We give some insights into the offline solutions. Further, we study its computational complexity by characterizing its NP-hardness in different cases.

IV. NP-hardness and the Offline Scenario

A. The Offline Problem in the BP Case

We consider the offline version of GPA for $M = 1$. In this case, GPA is equivalent to the following: maximize the NSD during n slots subject to the constraints of arrival times, deadlines, data requirements, and matching capacity (i.e., no more than one device in the same slot and no more than one slot for each device). We can solve this problem by reducing it to a maximum matching problem in a bipartite graph. First, we create a bipartite graph where the slots represent the left vertexes and the devices represent the right vertexes. An edge exists between slot j and device i if and only if $\bar{e}_i g_{ij} \geqslant (2^{L_i/W} - 1)$ and $j \in \{a_i..d_i - 1\}$. Every edge in the graph has capacity 1. By introducing a source vertex and sink vertex, we can find the optimal solution to this maximum matching problem in polynomial-time by applying some known maximum flow algorithm.

Solving GPA for general $M > 1$ seems to be a hard task. Indeed, we show in the following that GPA is NP-hard for any fixed $M \geqslant 3$. Nonetheless, it remains open whether or not GPA in the BP case is NP-hard for $M = 2$.

Theorem 1. *GPA is NP-hard in the BP case for $M \geqslant 3$.*

Proof: We reduce 3-bounded 3-dimensional matching (3DM3) [26] to GPA. In 3DM3, we are given a set $\mathbb{T} \subseteq \mathbb{W} \times \mathbb{X} \times \mathbb{Y}$, where $\mathbb{W}$, $\mathbb{X}$, and $\mathbb{Y}$ are disjoint sets having the same number ℓ of elements and $|\mathbb{T}| = r$. Also, in 3DM3, no element of $\mathbb{W} \cup \mathbb{X} \cup \mathbb{Y}$ occurs in more than three triples of $\mathbb{T}$. We may assume without loss of generality that $\ell < r < 2\ell$. The goal is to find in $\mathbb{T}$ a matching of maximum size, i.e., a subset $\mathbb{M} \subseteq \mathbb{T}$ where no two elements of $\mathbb{M}$ agree in any coordinate.

Given an instance of 3DM3, an instance of GPA is obtained as follows. Let $M = 3$. We create $n = r$ slots; slot j

corresponds to the 3-element set t_j from $\mathbb{T}$. We create also $m = 2\ell + r$ devices. Specifically, for each $i \in \mathbb{W}$, we have a device w_i, for each $i \in \mathbb{X}$, we have a device x_i, and for each $i \in \mathbb{Y}$, we have a device y_i. Also, there are $r - \ell$ additional devices $\{z_1, z_2, \ldots, z_{r-\ell}\}$. Let $a_i = 1$ and $d_i = n$, that is device's i packet arrived at the beginning of the frame and is due at its end. For all devices i, set $\bar{e}_i = 1$ and let Δ be a large number. For each device w_i and slot j corresponding to the 3-element set t_j, let $b_{w_i} = 1$ and

$$g_{w_i j} \coloneqq \begin{cases} 3, & \text{if } i \in t_j, \\ 2 + w_i/\Delta, & \text{otherwise.} \end{cases}$$

For each device x_i and slot j corresponding to the 3-element set t_j, let $b_{x_i} = 1/2$ and

$$g_{x_i j} \coloneqq \begin{cases} 1, & \text{if } i \in t_j, \\ 2 + x_i/\Delta, & \text{otherwise.} \end{cases}$$

For each device y_i and slot j corresponding to the 3-element set t_j, let $b_{y_i} = 1/2$ and

$$g_{y_i j} \coloneqq \begin{cases} 1/2, & \text{if } i \in t_j, \\ 2 + y_i/\Delta, & \text{otherwise.} \end{cases}$$

And for each additional device z_i, and slot j, let $b_{z_i} = 1$ and $g_{z_i j} = 2 + z_i/\Delta$. This instance is clearly created in polynomial-time. We prove that 3DM3 is solved with a matching of size ℓ if and only if GPA is solved with $2\ell + r$ served devices and each slot serves at most 3 devices.

On the one hand, if 3DM3 is solved, then for each 3-element set m_j of the matching $\mathbb{M}$, we can serve three devices in slot j—one for each element of m_j. This is valid because each element of m_j comes, respectively, from $\mathbb{W}$, $\mathbb{X}$, and $\mathbb{Y}$ and thus the corresponding three devices have channel gains equal, respectively, to 3, 1, or 1/2. This means that, for the device w_i we have $3/(1 + 1 + 1/2) = 6/5 \geqslant 1$, for the device x_i we have $1/(1 + 1/2) = 2/3 \geqslant 1/2$ and for the device y_i we have $1/2 \geqslant 1/2$. We conclude that the three devices meet their data requirements. Since $\mathbb{M}$ is a matching in $\mathbb{T}$ of size ℓ, the slots corresponding to $\mathbb{M}$ serves a total of 3ℓ devices—three devices in each slot. The remaining slots can serve at most $(r - \ell)$—at most one device per slot. To maximize the number of devices served, each remaining slot can serve exactly one device. Thus, there are a total of $3\ell + r - \ell = 2\ell + r$ devices served where each slot serves at most three devices.

On the other hand, assume that GPA is solved where each slot serves at most three devices with a total of $2\ell + r$ served devices. We argue that if a slot serves three devices, then these devices correspond to some triple in $\mathbb{T}$ (they have channel gains 3, 1 or 1/2). Thus, all slots that serve exactly three devices correspond to a matching in $\mathbb{T}$. Note that, in the solution to GPA, we cannot serve two devices in each slot (for a total of $2r$ devices) because $2\ell + r > 2r$. Thus, to maximize the number of devices served, ℓ slots need to serve three devices (and $r - \ell$ slots serves one device each), which corresponds to a matching in $\mathbb{T}$.

The reduction is clearly done in polynomial-time. Finally, since 3DM3 is well-known NP-hard problem [26], the theorem follows. ∎

B. The Offline Problem in the GP Case

In the following, we consider the GP case and we prove that GPA is NP-hard even when $M = 1$, i.e., only OMA technique is used.

Theorem 2. *GPA is NP-hard in the GP case even for $M = 1$.*

Proof: The proof is to show that a special case of GPA is NP-hard. Let $M = 1$. Also, let $a_i = 1$ and $d_i = n$, that is device's i packet arrived at the beginning of the frame and is due at its end. Let us assume that $\bar{e}_i$ is large enough so that device i cannot deplete its energy. Device i is transmitting with fixed power p_{ij} such that $\sum_{j=1}^{n} p_{ij} \leqslant \bar{e}_i$. Denote by $G_{ij} \coloneqq \lg(1 + p_{ij} g_{ij})$.

Under this restriction, we prove that GPA is NP-hard by reduction from maximum independent set (MIS) problem in graph theory. MIS is defined [26] as follows: given a graph and a positive integer ζ. Is there an independent set in the graph of size ζ or more? *An independent set is a set of vertexes that do not share any edge.* On the other hand, the restricted version of GPA can be recasted as: given the coefficients G_{ij} and the size of the packets L_i, is there a scheduling of more than σ devices, denoted by the set $\mathbb{O}$, such that a slot is used by at most one device and $\sum_{j \in \mathbb{O}} G_{ij} \geqslant L_i$?

Given an instance of MIS, we create an instance of GPA in polynomial-time as follows: the vertexes are the devices and the edges are the slots. The edges are numbered as $1, 2, \ldots, n$. There is an edge between two devices if and only if one of them can be served at that slot. Let $\mathbb{O}_i$ be the set of slots that device i can be served at. Let $L_i \coloneqq |\mathbb{O}_i|$ for each device i and set $\zeta = \sigma$. The coefficients G_{ie} for device i and slot e is given by:

$$G_{ie} \coloneqq \begin{cases} 0, & \text{if } e \notin \mathbb{O}_i, \\ 1, & \text{if } e \in \mathbb{O}_i. \end{cases} \tag{4}$$

This reduction is clearly done in polynomial-time. Now it remains to prove that: "there are more than ζ served devices, denoted by the set $\mathbb{O}$, such that a slot is used by at most one device and $\sum_{j \in \mathbb{O}} G_{ij} \geqslant L_i$" if and only if "there is independent set in the graph of size ζ or more".

On the one hand, assume that we have an independent set in the graph of size ζ or more. By setting $x_{ie} = 1$ for all i in the independent set and $e \in \mathbb{O}_i$, we have more than ζ devices served. Further, since we have an independent set, it is true that the served devices are not overlapping with one another.

On the other hand, assume that we have a solution to the restricted version of GPA, then we can see, by construction, that for device i to be satisfied, it must be scheduled in all slots in $\mathbb{O}_i$ (since $L_i = |\mathbb{O}_i|$ and G_{ie} are binary). Since we have more than ζ non-overlapping served devices, these devices form, in the corresponding graph, an independent set of size more than ζ.

Summarizing, we have reduced MIS to the restricted version of GPA in polynomial-time such that MIS is solved if and only if GPA is solved. Since MIS is well-know NP-hard [26], so is GPA. This proves the theorem. ∎

All of our NP-hardness results are presented in table I. We use "Poly" to denote the polynomial-time complexity class and "Open?" to denote that, to the best of our knowledge, the problem is still open.

V. Competitive Algorithms

As previously discussed in remark 2, we start by solving GPA in the single frame case. Then, we solve it in the more general case of multiple frames. Before going into the details, we give the following definition.

Definition 1 (c-competitive algorithm [8])**.** *An online algorithm* ALG *is* ***c-competitive*** *if there is a constant α such that for all finite input sequences,*

$$O \leq cA + \alpha, \tag{5}$$

where A (or O) is the value returned by ALG *(or an optimal algorithm* OPT*) for a given input.* ALG *is* ***strictly c-competitive*** *if it is c-competitive and $\alpha \leq 0$.*

We assume that device i can choose its transmission power from $[\bar{e}_i]_1 = \{0, \bar{e}_i\}$. That is, device i can either transmit in a slot or stay silent once and forever during the frame. This assumption implies that a device can use at most one RB. This is realistic in massive IoT networks where devices normally have short data packets to send [6, 28].

Remark 3 (The Selfish Algorithm at Slot j)**.** *Assume that, without loss of generality, $g_{1j} < g_{2j} < \cdots < g_{mj}$. According to the decoding order of largest channel gains in uplink NOMA, device 1 knows that if $\bar{e}_1 g_{1j} < 2^{L_1/W} - 1$, then $p_{1j} = 0$. Now, if $\bar{e}_1 g_{1j} \geqslant 2^{L_1/W} - 1$, then setting $p_{1j} = \bar{e}_1$ would satisfy device 1 but may interfere with other devices. The following is called the selfish algorithm for device i: whenever $\bar{e}_i g_{ij} \geqslant 2^{L_i/W} - 1$, set $p_{ij} = \bar{e}_i$. We can prove that the selfish algorithm can perform very badly. Say device 1 acts selfishly. Then, we can find an instance in which only device 1 will be served in slot j (due to the severe interference that it generates)—removing device 1 from slot j would satisfy all other devices in that slot. Assume we have $L_i = \lg(1+i)$ for all i and $g_{1j} = 1$ but for all $i \neq 1$, $g_{ij} = \sqrt{2^{(i-2)(i+1)}}\left(2^i - 1\right)$. In this case, if device 1 transmits in slot j, then no other device can transmit in that slot. However, if it does not, then all device $i \neq 1$ can transmit. This shows that the selfish algorithm has a competitive ratio of at least $m - 1$ which is very large.*

The previous remark proves that acting selfishly is not very good in terms of maximizing the NSD. To provide better results, we first study the case of $M = 1$ and then generalize the analysis to larger M.

For $M = 1$, we can reduce GPA to an online matching problem in a bipartite graph as follows. The devices represent the right vertexes of the bipartite graph. The slots represent the left vertexes that appear online one-by-one. An edge exists between slot j and device i if and only if $\bar{e}_i g_{ij} \geqslant (2^{L_i/W} - 1)$ and $j \in \{a_i..d_i - 1\}$. When slot j appears, the channel gain g_{ij} is revealed for all devices i and thus the edges incident to it are also revealed. Once revealed, an online algorithm must make an irrevocable decision of which device to serve at slot j (i.e., match the corresponding edge). This online matching problem can be solved using the well-known *ranking* algorithm that has a competitive ratio of $\frac{e}{e-1} \approx 1.58$ [27]. The ranking algorithm chooses a random permutation ρ of the devices. For each slot j, it finds the set of not-yet-served devices $\mathbb{Y}_j$ that can transmit in this slot, i.e., those devices i that have $\bar{e}_i g_{ij} \geqslant (2^{L_i/W} - 1)$ and $j \in \{a_i..d_i - 1\}$. If $\mathbb{Y}_j$ is not empty, then the ranking algorithm chooses a device i from $\mathbb{Y}_j$ that minimizes $\rho(i)$. The ranking algorithm is equivalent to assigning priorities to the devices and choosing the not-yet-served device that has the highest priority.

To solve the problem for general $M \geqslant 1$, we transform it into a many-to-one matching problem and we adopt a greedy approach to solve it. We create the previous same bipartite graph. Now, contrary to the case of $M = 1$, each slot can be matched to at most M devices from those connected to it by an edge. For each slot $j \in [n]$, let $\mathbb{N}_j$ denotes the set of neighbors of j (i.e., $\mathbb{N}_j \coloneqq \{i \in [m] : \{i, j\} \text{ is an edge}\}$). Once slot j is revealed, the problem is reduced to finding a set of (at most M) devices $\mathbb{D}_j \subseteq \mathbb{N}_j$ of maximum cardinality such that:

$$\bar{e}_i g_{ij} \geqslant \left(2^{L_i/W} - 1\right)\left(1 + \sum_{i' \in \mathbb{D}'_j} \bar{e}_{i'} g_{i'j}\right), \tag{6}$$

is valid for each $i \in \mathbb{D}_j$, where $\mathbb{D}'_j \coloneqq \{i' \in \mathbb{D}_j : g_{ij} > g_{i'j}\}$.

It is known that the complexity of SIC decoding increases as the number of users transmitting on the same RB increases [29]. Thus, in general, M is chosen small in order to keep the complexity of SIC decoding low. In fact, multiple research papers consider the case of user pairing when $M = 2$ [17, 30]. Hence, for small and fixed M, one could generate, in slot j, all combinations of at most M devices and matches the maximum-cardinality set $\mathbb{D}_j$ that respects (6). This leads to a polynomial-time (only for fixed M) worst-case complexity of $\mathcal{O}(m^M)$. However, by analyzing the problem structure based on (6), we provide an optimal way of finding a maximum cardinality set that satisfies (6) in $\mathcal{O}(m)$ worst-case time complexity.

Lemma 1. *Once slot j is revealed, finding a maximum cardinality set $\mathbb{D}_j$ that satisfies* (6) *can be done in $\mathcal{O}(m)$ worst-case time complexity.*

Proof: We prove that the greedy algorithm, given in Algorithm 1 below and called binary-matching-j (BM$_j$), which is applied at slot j, gives a maximum cardinality set that satisfies (6) in $\mathcal{O}(m)$ time in the worst-case; assuming that the channel gain $\mathbf{g}_j$ is sorted, otherwise the complexity would be $\mathcal{O}(m \lg m)$.

The worst-case time complexity of BM$_j$ is clearly $\mathcal{O}(m)$. It remains to show that the algorithm returns a feasible solution of maximum cardinality that satisfies (6).

First, it is clear that $|\mathbb{D}_j| \leqslant M$. Using mathematical induction on each iteration of the algorithm, we prove that the set $\mathbb{D}_j$ represents a feasible solution that satisfies (6). Let $\mathbb{D}_j^p$ be the set of devices returned by BM$_j$ after iteration p. For $p = 1$, device 1 is added to $\mathbb{D}_j^1$ only if $\bar{e}_1 g_{1j} \geqslant b_1$ and thus, $\mathbb{D}_j^1$ is feasible. Assume that $\mathbb{D}_j^p$ is feasible. Is $\mathbb{D}_j^{p+1}$ feasible? At iteration $p+1$, the algorithm adds device $p+1$ to

TABLE I: Complexity Classification

	GPA with GP	GPA with BP		
Group Size	$M = 1$ (no NOMA)	$M = 1$ (no NOMA)	$M = 2$ (NOMA)	$M \geqslant 3$ (NOMA)
Complexity Class	NP-hard	Poly	Open?	NP-hard
Online Algorithm	Open?	1.58-competitive [27]	2-competitive	2-competitive

Algorithm 1 The BM$_j$ algorithm

Input: $M, m, \mathbb{N}_j, \mathbf{g}_j, \mathbf{L}, \bar{\mathbf{e}}$
Output: $\mathbb{D}_j$

$\mathbb{X} \leftarrow \emptyset$
$X \leftarrow 0$
for $i \leftarrow 1$ **to** m **do**
  if i **in** $\mathbb{N}_j$ **then**
    if $\bar{e}_i g_{ij} \geqslant (2^{L_i/W} - 1)(1 + X)$ **then**
      $\mathbb{X} \leftarrow \mathbb{X} \cup \{i\}$
      $X \leftarrow X + \bar{e}_i g_{ij}$
if $|\mathbb{X}| \leqslant M$ **then**
  $\mathbb{D}_j \leftarrow \mathbb{X}$
else
  Let $\mathbb{D}_j \subset \mathbb{X}$ with $|\mathbb{D}_j| = M$
return $\mathbb{D}_j$

$\mathbb{D}_j^{p+1}$ only if $\bar{e}_{p+1} g_{p+1j} \geqslant b_{p+1}(1 + X)$ where $X = \sum_{i \in \mathbb{D}_j^p} g_{ij}$. If this condition is not met, then $\mathbb{D}_j^{p+1} = \mathbb{D}_j^p$ and we are done. Otherwise, $\mathbb{D}_j^{p+1} = \mathbb{D}_j^p \cup \{p+1\}$. Since the channel gains are sorted in increasing order, thus $g_{p+1j} \geqslant g_{ij}$ for all $i \in \mathbb{D}_j^p$. According to the largest channel gain decoding order of SIC, the devices already in $\mathbb{D}_j^p$ will not be affected by the transmission of device $p+1$. Because device $p+1$ is added to $\mathbb{D}_j^p$ only if (6) are respected, we conclude that $\mathbb{D}_j^{p+1}$ is feasible. Combining the base case and the inductive hypothesis, we finally have that the returned set $\mathbb{D}_j = \mathbb{D}_j^m$ is feasible.

To prove the optimality, let $\{i_1, i_2, \ldots, i_{\ell_1}\}$ be the set of served devices in the order they were added to $\mathbb{D}_j$ and let $\{i_1^*, i_2^*, \ldots, i_{\ell_2}^*\}$ be the set of devices in the order they were added to $\mathbb{O}_j$ returned by some optimal algorithm. We assume without loss of generality that $g_{i_1 j} < g_{i_2 j} < \cdots < g_{i_{\ell_1} j}$ and $g_{i_1^* j} < g_{i_2^* j} < \cdots < g_{i_{\ell_2}^* j}$. The optimality is to prove that $\ell_2 = \ell_1$.

First, we prove by induction on $\ell \leqslant \ell_1$ that $g_{i_\ell^* j} \geqslant g_{i_\ell j}$. The base case, for $\ell = 1$, is clearly true: the first device served by BM$_j$ has the smallest channel gain. Assume now that for $\ell > 1$ the statement is true for $1, 2, \ldots, \ell - 1$, i.e., $g_{i_{\ell-1}^* j} \geqslant g_{i_{\ell-1} j}$, is it true for ℓ? If $g_{i_\ell^* j} < g_{i_\ell j}$, then BM$_j$ would have chosen i_ℓ^* instead of i_ℓ because, using the inductive hypothesis, $g_{i_\ell j} > g_{i_\ell^* j} \geqslant b_{i_\ell^*}(1 + \sum_{i=i_1^*}^{i_{\ell-1}^*} g_{ij}) \geqslant b_{i_\ell^*}(1 + \sum_{i=i_1}^{i_{\ell-1}} g_{ij})$. Thus, for all $\ell \leqslant \ell_1$, it is true that $g_{i_\ell^* j} \geqslant g_{i_\ell j}$.

Now, we use the previous fact to prove, by contradiction, that $\ell_1 = \ell_2$. Assume that $\ell_2 > \ell_1$. In other words, there exists a device $i_{\ell_1+1}^* \in \mathbb{O}_j$, or equivalently, the optimal algorithm chooses $i_{\ell_1+1}^*$ in iteration $\ell_1 + 1$. Thus, $g_{i_{\ell_1+1}^* j} \geqslant b_{i_{\ell_1+1}^*}(1 + \sum_{i=i_1^*}^{i_{\ell_1}^*} g_{ij}) \geqslant b_{i_{\ell_1+1}^*}(1 + \sum_{i=i_1}^{i_{\ell_1}} g_{ij})$, where the last inequality follows form the previous fact. Since the optimal algorithm chooses $i_{\ell_1+1}^*$ in iteration $\ell_1 + 1$, then $g_{i_{\ell_1+1}^* j} > g_{i_{\ell_1}^* j} \geqslant g_{i_{\ell_1} j}$. We can see that, in iteration $\ell_1 + 1$, device $i_{\ell_1+1}^*$ has larger channel gain than device i_{ℓ_1} and can be added to $\mathbb{D}_j$. Since BM$_j$ stopped adding devices at iteration ℓ_1, we reach a contradiction and we conclude that $\ell_1 = \ell_2$.

Finally, the set $\mathbb{D}_j$ returned by BM$_j$ is of maximum cardinality and is obtained in $\mathcal{O}(m)$ worst-case time complexity. This proves the lemma. ∎

The proposed algorithm to solve GPA is called the binary-matching-slots (BMS) algorithm and its pseudo-code is given in Algorithm 2. For each arriving slot, BMS calls MB$_j$ and serves the maximum possible number of devices in that slot. Then, it updates the set of not-yet-served devices and continues in this way. We prove that this algorithm is 2-competitive.

Algorithm 2 The BMS algorithm

Input: Bipartite graph, $M, m, n, \mathbf{g}, \mathbf{L}, \bar{\mathbf{e}}$
Output: $\{\mathbb{D}_1, \mathbb{D}_2, \ldots, \mathbb{D}_n\}$

$\mathbb{X} \leftarrow [m]$
for each slot j **do**
  $\mathbb{D}_j \leftarrow \text{BM}_j(M, m, \mathbb{N}_j, \mathbf{g}_j, \mathbf{L}, \bar{\mathbf{e}})$
  $\mathbb{X} \leftarrow \mathbb{X} \backslash \mathbb{D}_j$
return $\{\mathbb{D}_1, \mathbb{D}_2, \ldots, \mathbb{D}_n\}$

Theorem 3. *BMS is 2-competitive.*

Proof: Let $\mathbb{D}_M \coloneqq \{\mathbb{D}_1, \mathbb{D}_2, \ldots, \mathbb{D}_n\}$ be the set of devices served by BMS and let $\mathbb{O}_M = \{\mathbb{O}_1, \mathbb{O}_2, \ldots, \mathbb{O}_n\}$ be the set of devices served by some optimal algorithm OPT, where $\mathbb{D}_j$ (resp. $\mathbb{O}_j$) are the devices served by BMS (resp. by OPT) at slot j.

Based on lemma 1, it is clear that the number of devices in $\mathbb{O}_j \backslash \mathbb{D}_M$ is at most the number of devices in $\mathbb{D}_j$; since otherwise BMS would have chosen $\mathbb{O}_j \backslash \mathbb{D}_M$ instead of $\mathbb{D}_j$. Thus, by summing over j, the number of devices in $\mathbb{O}_M \backslash \mathbb{D}_M$ is at most the number of devices in $\mathbb{D}_M$. Since $\mathbb{O}_M \subseteq (\mathbb{D}_M \cup \mathbb{O}_M \backslash \mathbb{D}_M)$, thus we obtain:

$$\begin{aligned} |\mathbb{O}_M| &\leqslant |\mathbb{D}_M| + |\mathbb{O}_M \backslash \mathbb{D}_M|, \\ &\leqslant |\mathbb{D}_M| + |\mathbb{D}_M| \leqslant 2|\mathbb{D}_M|. \end{aligned}$$

∎

Theorem 4. *There is no deterministic online algorithm with better competitive ratio than BMS.*

Proof: Assume $m = n = 2$. Let $\bar{e}_1 = \bar{e}_2 = 1$ and $L_1 = L_2 = 1$. The channel gains in slot 1 is $\mathbf{g}_1 = [1, 1]$. Now, if an online algorithm decides to serve device 1 (resp. 2) in slot 1, then we can choose the channel gains in the slot 2 as $\mathbf{g}_2 = [1, 0]$ (resp. $\mathbf{g}_2 = [0, 1]$). An optimal offline algorithm can serve device 2 in slot 1 and device 1 in slot 2 if $\mathbf{g}_2 = [1, 0]$ or it can serve

device 1 in slot 1 and device 2 in slot 2 if $\mathbf{g}_2 = [0, 1]$. In any case, the offline-to-online ratio is 2. ■

A. Benchmarks Algorithms

For comparison purposes, in this section, we present an adapted version of a clustering algorithm, called hereinafter ATH (by M. S. Ali, H. Tabassum, and E. Hossain), proposed in [14]. The original algorithm is offline and works with channel gains that are independent of the RBs. We transform it to an online algorithm as follows. First, for simplicity, here, we assume that m is a multiple of M and denote by $\kappa \coloneqq m/M$. For each new slot j, ATH creates κ clusters where each cluster contains exactly M devices by sorting the channel gains in descending order. That is, if $g_{1j} > g_{2j} > \cdots > g_{mj}$, then cluster l will contain the devices $\{l, \kappa+l, 2\kappa+l, \ldots, M\}$. Now, for each slot j, ATH iterates the clusters and checks whether the arrival times, deadlines and the data rate requirements of the devices in cluster l are respected. If not, the devices are removed iteratively from cluster l until the constraints are not violated. Once all clusters are checked, ATH picks the cluster with the maximum NSD. For subsequent slots, ATH proceeds similarly with the exception that an already served device is removed from the clusters.

We present another adapted version of a benchmark algorithm, called ZZ (by D. Zhai and R. Zhang), proposed in [17]. The original algorithm is offline, based on solving independent sets in graphs, and proposed only for $M = 2$ [17]. The modified version, ZZ, works as follows. Since $M = 2$, it generates all pairs of devices in each slot. By checking the constraints of arrival time, deadlines, and data rate requirements, the pairs are updated in each slot—meaning that a pair can be reduced to a single element or to empty if necessary. ZZ constructs an undirected graph G where the set of nodes is the possible set of devices (paired or not) in each slot. So, a node v is given by the tuple (c, j) where c is either a single device or a pair of devices served in slot j. An edge between node (c, j) and node (c', j') exists if and only if $j = j'$ or $c \cap c'$ is not empty. Once the graph is constructed, ZZ creates a new graph H by splitting every node (c, j) in G with $|c| = 2$ into two nodes that has the same neighbors as in G but are not linked by an edge [17]. Device pairing is now obtained by solving the problem of maximum independent set in the new graph H using a greedy approach. ZZ is still offline since it must construct the graph G by knowing all the upcoming slots.

B. Running Time Complexity

Here, we analyze the worst-case time complexities of the different algorithms. We summarize the results in table II. The complexity of BMS is clearly $\mathcal{O}(nf(m))$ where $\mathcal{O}(f(m))$ is the complexity of BM_j which is equal to $\mathcal{O}(m)$ for sorted channel gains $\mathbf{g}_j$ or $\mathcal{O}(m \lg m)$ otherwise. Similar analysis can be done to obtain $\mathcal{O}(nf(m))$ complexity for ATH. As for ZZ, the generation of all pairs is done in $\mathcal{O}(m^2)$. To construct the graph, one has to iterate the slots and this gives a complexity of $\mathcal{O}(nm^2)$. Once the graph is constructed, finding a maximal independent set using the classical greedy approach requires $\mathcal{O}(m^4)$ complexity since the constructed graph has $\mathcal{O}(m^2)$ nodes.

VI. Learning Algorithms

When there are multiple frames, the problem involves power allocation as well as NOMA grouping and scheduling. Note that without further assumption, one cannot hope to obtain good performances in terms of competitiveness. Specifically, say there are two frames and a single device. If an online algorithm decided to allocate some transmission power $p > 0$ in frame one. Then, an adversary can always choose the channel gains such that $p \max_j\{g_j\} < 2^{L/W} - 1$ but $p' g_j \geqslant 2^{L/W} - 1$ for some slot j with $p' > p$. Next, the adversary can also choose the channel gains in the second frame such that $\bar{e} \max_j\{g_j\} < 2^{L/W} - 1$. In this manner, the adversary can serve the device once in frame one with p' but an online algorithm never served the device. For this reason, we are motivated to consider a relative performance measure and thus we adopt the learning framework.

In order to obtain a global solution (for multiple frames) to GPA, we use machine learning techniques. More specifically, we combine our proposed online competitive algorithms and reinforcement learning techniques to obtain the power allocation solution to GPA.

We model GPA with multiple frames as an online deterministic Markov decision process (MDP). This modeling is important to apply reinforcement learning technique and helped us to transform the problem into an online (stochastic) shortest path problem. The MDP is deterministic because the transition probabilities are known. The corresponding transition graph (TG) is constructed as follows. A state (or a node) in the TG is a tuple $(\mathbf{e}^t, t)$, for $t = 2, \ldots, k+1$, where $\mathbf{e}^t = [e_1^t, e_2^t, \ldots, e_m^t]^\top$ represents the remaining battery level of the devices in frame t. When $t = 1$, the node $\mathbf{s} \coloneqq (\mathbf{e}^1, 1)$ is called the starting node, where $e_i^1 = \bar{e}_i$ for all i. There is a terminal node denoted by $\mathbf{t} \coloneqq (\mathbf{e}^{k+2}, k+2)$, where $e_i^{k+2} = 0$ for all i. For $t = 1, 2, \ldots, k+1$, a transition from $(\mathbf{e}_1^t, t)$ to $(\mathbf{e}_2^{t+1}, t+1)$ happens with probability one if and only if $\mathbf{e}_1^t - \mathbf{e}_2^{t+1} \geqslant \mathbf{0}$. No other transition is allowed. For $t = 1, 2, \ldots, k+2$, the action set corresponding to state $(\mathbf{e}^t, t)$ is given by the Cartesian product $[e_1^t]_{\tau_1} \times [e_2^t]_{\tau_2} \times \cdots \times [e_m^t]_{\tau_m}$, that is, an action taken in state $(\mathbf{e}_1^t, t)$ and transitions to state $(\mathbf{e}_2^{t+1}, t+1)$ is a transmission power vector $\mathbf{p}^t = [p_1^t, p_2^t, \ldots, p_m^t]^\top$. In other words, the possible actions in state $(\mathbf{e}_1^t, t)$ are given by the outgoing edges of node $(\mathbf{e}_1^t, t)$. Denote by $m_i \coloneqq |[\bar{e}_i]_{\tau_i}| = \tau_i + 1$ the power level of device i and by $m_x \coloneqq \Pi_{i=1}^m m_i$. The TG contains $2 + km_x$ states and $m_x(m_x(k-1)+k+3)/2$ directed edges. An example of this TG is given in Fig. 2. The reward of choosing action $\mathbf{p}^t = [p_1^t, p_2^t, \ldots, p_m^t]^\top$ in state $(\mathbf{e}^t, t)$ is the NSD in frame t, which can be obtained by applying the previously proposed online competitive algorithm BMS.

Under this modeling, GPA can be seen as an **s**-**t** shortest path problem in the corresponding TG, or equivalently, as finding the **s**-**t** path with the highest reward (by transforming rewards to losses we can move from shortest path to longest path). Nonetheless, finding such an **s**-**t** path is too complex because the number of nodes and the number of edges in the TG is exponentially large, e.g., for three frames, fifteen devices and a power level of two ($m_i = 2$ for all i), the TG contains approximately one hundred thousand nodes and

TABLE II: Worst-Case Time Complexities

Algorithms	Complexity
BMS	$\mathcal{O}(nm \lg m)$
ATH	$\mathcal{O}(nm \lg m)$
ZZ	$\mathcal{O}(nm^2 + m^4)$

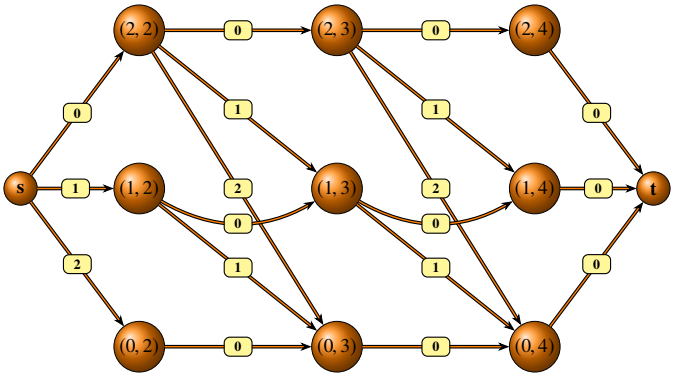


Fig. 2: An instance of the TG of one device with $[2]_2 = \{0, 1, 2\}$ and three frames. The rounded squares in the middle of the edges represent the actions.

one billion edges. Due to the curse of dimensionality, we follow a *distributed* approach to solve the power allocation learning problem. The approach is distributed in the sense that each device learns its own **s**-**t** shortest path by exchanging information among other devices through the BS. (Of course, there is a trade-off here between complexity and information exchange that we left its analysis for future works.) We use a modified version of EXP3 [31]—a popular reinforcement learning algorithm for the adversarial multi-armed bandit problem. For comparison purposes, we also adopt the classical tabular Q-learning algorithm [32].

A. EXP3-based Distributed Learning

EXP3 is based on exponential-weighting for exploration and exploitation and is proposed to solve the non-stochastic (adversarial) multi-armed bandit problem [31]. Each device learns its own **s**-**t** path by applying a modified version of EXP3. Each device has its own TG. As before, a state in each TG i is given by (e_i^t, t) where e_i^t represents the remaining energy level at frame t in device i's battery with $e_i^1 = \bar{e}_i$. For any state (e_i^t, t) of TG i, an action is given by the transmission power $p_i^t \in [e_i^t]_{\tau_i}$. See Fig. 2. Normally, when device i, in state (e_i^t, t), chooses action $p_i^t \in [e_i^t]_{\tau_i}$, its reward is a binary number that represents whether or not it is served. Designing the rewards in this way teaches the devices to act selfishly and thus does not necessarily give good outcome, i.e., the total NSD could be very low because each one will learn to use its transmission power to get served regardless of others (see remark 3). It is thus necessary to redesign the rewards to improve the learning outcome. Instead of the binary rewards, each device receives its reward as the NSD in each frame. This can be acquired by information feedback between the devices and the BS. The main lines of the learning algorithm, called the path-learning (PL) algorithm is given for each round as follows.

- Device i chooses an action p_i^t for each frame t according to some probability, i.e., it chooses an $\mathbf{s}_i$-$\mathbf{t}_i$ path in TG i. We denote this path by the vector $\mathbf{p}_i = [p_i^1, p_i^2, \ldots, p_i^k]^\top$.
- Device i sends its chosen $\mathbf{s}_i$-$\mathbf{t}_i$ path to the BS.
- The BS runs the online per-frame competitive algorithm BMS at frame t with power allocation $\mathbf{p}^t = [p_1^t, p_2^t, \ldots, p_m^t]^\top$ and calculates the NSD.
- The BS broadcasts the rewards to each device (the reward received by device i is the NSD in frame t). That is, device i knows, not only the rewards of its chosen $\mathbf{s}_i$-$\mathbf{t}_i$ path, but also the rewards in each edge of that path.
- Device i updates the probabilities.

PL operates in rounds, where in each round, it is applied at device i that chooses an $\mathbf{s}_i$-$\mathbf{t}_i$ path according to some probability (proportional to the path weight). This probability is chosen to follow a distribution over the set of all $\mathbf{s}_i$-$\mathbf{t}_i$ paths in order to get a mixture between exponential weighting of biased estimates of the rewards and uniform distribution to ensure sufficiently large exploration of each edge of any $\mathbf{s}_i$-$\mathbf{t}_i$ path. After choosing an $\mathbf{s}_i$-$\mathbf{t}_i$ path, device i gets to know the rewards on each edge of that path, i.e, it gets to know the NSD in each frame. Then, PL updates the probability distribution (by updating the paths weights) and continues similarly.

We notice that every TG i has $2 + km_i$ nodes and $m_i(m_i(k-1) + k + 3)/2$ directed edges. Every path in TG i has length $k + 1$. Let $\mathbb{P}_i$ be the set of all $\mathbf{s}_i$-$\mathbf{t}_i$ paths in TG i and let $\sigma_i := |\mathbb{P}_i|$ denotes the number of such paths. We can prove that $\sigma_i = \binom{k+m_i-1}{k}$, which is exponentially large and thus choosing the paths in this way according to their weights is not efficient. However, a simple modification can improve the algorithm enormously [33]. First, instead of assigning weights to paths, they are assigned to edges. Second, we construct a set of edge-covering $\mathbf{s}_i$-$\mathbf{t}_i$ paths $\mathbb{C}_i$, which is defined as the set of paths in TG i such that for any edge e in TG i, there is a path $\mathbf{p}_i$ in $\mathbb{C}_i$ such that $e \in \mathbf{p}_i$. Such an edge-covering paths $\mathbb{C}_i$ can be obtained in $\mathcal{O}(km_i^2 + km_i \lg(km_i))$ time using Dijkstra's algorithm where $|\mathbb{C}_i| = \mathcal{O}(km_i^2)$. Now, instead of each path, each edge e of TG i is assigned a weight $w(e)$ (initialized to one for each edge at the beginning of the rounds) and the weight of an $\mathbf{s}_i$-$\mathbf{t}_i$ path is given by the product of the weights of its edges. For each round, PL, applied at device i, chooses an $\mathbf{s}_i$-$\mathbf{t}_i$ path (1) uniformly from $\mathbb{C}_i$ with probability γ or (2) according to the paths weights with probability $1 - \gamma$. If the latter is to be done, then the $\mathbf{s}_i$-$\mathbf{t}_i$ path can be chosen by adding its vertexes one-by-one according to edges' weights (and not to paths' weights) [33]. Next, PL finds the probability of choosing each edge in the TG i, which can also be done using edges' weights only. (It

can be proven that choosing paths and updating the edges' probabilities can be done efficiently based on paths kernels and dynamic programming [34].) Then, for each frame (or equivalently for each edge), the rewards are obtained using BMS, where the reward at any edge r is normalized by the probability of that edge $q(e)$, i.e., the normalized reward is $(\beta + r\mathbb{1}_{\{e\in\mathbf{p}_i\}})/q(e)$, with $\mathbb{1}_{\mathbb{A}}$ denotes the indicator function and $\beta \in (0, 1]$. Finally, the edges' weights are updated as $w(e) \leftarrow w(e)e^{\eta r}$ where $\eta > 0$.

The per-round complexity of PL is given by $\mathcal{O}(kmm_i^2 + knm \lg m)$, where $\mathcal{O}(knm \lg m)$ is the complexity of applying BMS in all frames and $\mathcal{O}(kmm_i^2)$ is the complexity of choosing the paths according to the edges' weights and updating the probability of each edge.

B. Tabular-based Distributed Learning

We use the tabular Q-learning algorithm [32]. The Q-learning algorithm is called QL and it proceeds in episodes. In each episode, each device i chooses an $\mathbf{s}_i$-$\mathbf{t}_i$ path, receives a reward and updates its Q-table. Precisely, each device i has a Q-table $Q_i(s, a)$ that measures the quality of a state-action combination (s, a), where s represents a node in the TG i and a represents a chosen transmission power in state s. For each episode, each device i starts the learning in the initial state $\mathbf{s}_i$. For each step in that episode, that is for each frame t, device i chooses a possible action (according to its state) using the epsilon-greedy approach and moves to the next state s'. Once all devices choose their actions, the BS runs the online competitive algorithm BMS in frame t and fed back the rewards to each device (device i receives the NSD in frame t). Next, each device move to the next state and updates its Q-table. As soon as the last frame is reached and the Q-table is updated, the devices move to the next episode and the Q-learning algorithm continues. Updating the Q-table is done as follows:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \max_a Q(s', a) - Q(s, a)). \quad (7)$$

The per-episode complexity of QL is given by $\mathcal{O}(knm \lg m)$, where $\mathcal{O}(nm \lg m)$ is the complexity of applying BMS in each frame and updating the Q-table.

VII. Simulation Results

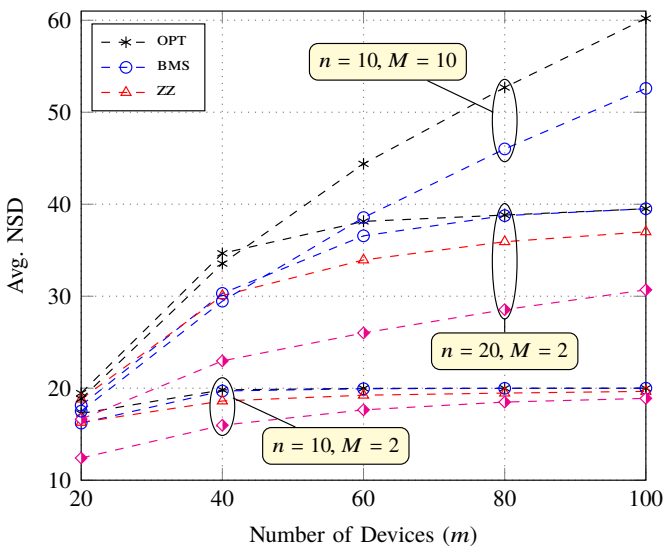
This section illustrates the performance of the proposed algorithms through computer simulations. We consider a geographical zone modeled by a square of side 1000 meters. The BS is located at the center of this zone whereas the devices are randomly and uniformly distributed inside the square. The simulations parameters are based on 3GPP specifications [35, p. 481] as in [6, 19]. The carrier frequency is f_c = 900 MHz and the path-loss (in dB) at f_c is given by $120.9 + 37.6 \log(\text{dist}_i^t) + \alpha_G + \alpha_L$ [35, p. 481], where dist_i^t is the distance (in km) between device i and the BS at frame t, $\alpha_G = -4$ dB represents the antenna gain and $\alpha_L = 10$ dB is the penetration loss. Flat Rayleigh fading is also considered and thus g_{ij}^t includes the previous path-loss model as well as an exponential random variable with unit parameter. The power spectral density of the noise is -174 dBm/Hz and the noise figure is 5 dB. Unless specified otherwise, the next parameters are fixed as follows. Each device i has a maximum transmission power of $\bar{e}_i = 23$ dBm [35, p. 481]. The group size is $M = 2$. The bandwidth is 200 kHz and the bandwidth of a single RB is $200/n$ kHz where n is the total number of RBs. The data requirements of the devices follow an uniform distribution as $L_i^t \sim \text{unif}\{0, L_{\max}\}$ with $L_{\max} = 100$ kbits. The arrival times are given by $a_i^t \sim \text{unif}\{1, n\}$ and the deadlines are given by $d_i^t \sim \text{unif}\{a_i^t+1, n+1\}$. The optimization problem (P1) is modeled in AMPL [36] and solved using the CPLEX. All simulations are performed for independent random realizations and averaged out.

The next figures show the results for the case of a single frame.

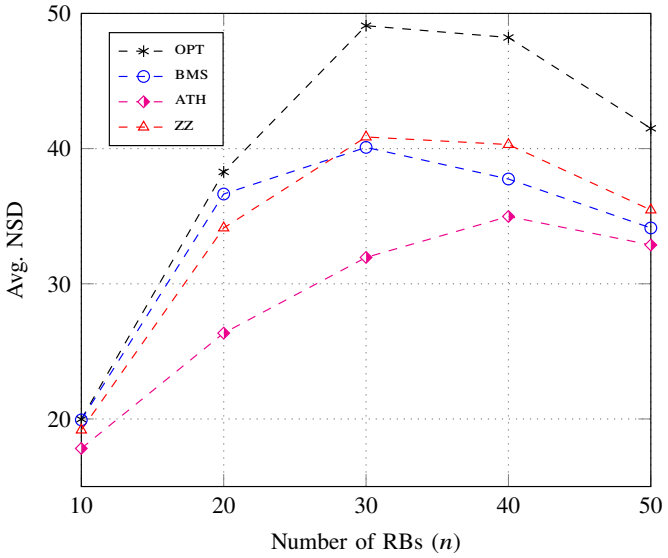
Fig. 3a shows the performance of the proposed online competitive algorithm BMS for different values of M and n against two benchmark algorithms, ATH and ZZ, proposed for NOMA grouping, and also against the optimal offline algorithm OPT obtained by solving (P1) through AMPL modeling and using the CPLEX solver. When the number of RBs n or the group size M increases, the number of served devices increases faster with m. When the number of devices is small and the number of RBs is large, the offline algorithm ZZ achieves slightly better performance compared to our proposed algorithm BMS, despite being online. This is might be due to the heuristic approach used in ZZ to find a maximal independent set in the graph H as discussed in V-A. Another point is that BMS achieves much better performance compared to ATH, because (1) the latter mainly optimizes the sum-rate objective and not the NSD and further (2) the pairs of devices in the latter are fixed a priori (thus, because the stringent constraints in GPA very few pairs can satisfy these constraints according to NOMA). Lately, we see that BMS, despite being online and despite the proven 50% theoretical gap in Theorem 3, its performance is not very far from OPT even for large values of n and M. The worst gap between BMS and OPT is about 87% which is much better than the proven 50% theoretical gap.

Fig. 3b presents the performance of BMS for $m = 60$ against ATH, ZZ and OPT. When the number of devices and the group size are fixed, there is an optimal value of n at which the performance is maximized. When n increases above this optimal value, the NSD starts to decrease since the bandwidth of each RB becomes small. In other words, when n continue to grow which decreases the bandwith of each RB, the interference inside each NOMA group become intolerable and the devices cannot meet their requirements. Lately, the performance of BMS is still the best amongst the non-optimal algorithms except for large n where the offline ZZ becomes better (due to its offline nature and probably to exploring more nodes in the graph H and thus selecting larger maximal independent sets) at the expense of higher running time complexity. Finally, BMS achieves close-to-optimal performance for different n and the worst gap is about 78% which is much better than the proven 50% theoretical gap.

Fig. 4 presents the performance of BMS against ATH, ZZ, and OPT when the battery capacity of the devices $\bar{\mathbf{e}}$ changes. Increasing the battery capacity is able to increase the NSD quickly (the increase rate is faster when the number of RBs is larger). However, since the number of RBs and the group size



(a) Impact of m.



(b) Impact of n.

Fig. 3: Impact of m and n.

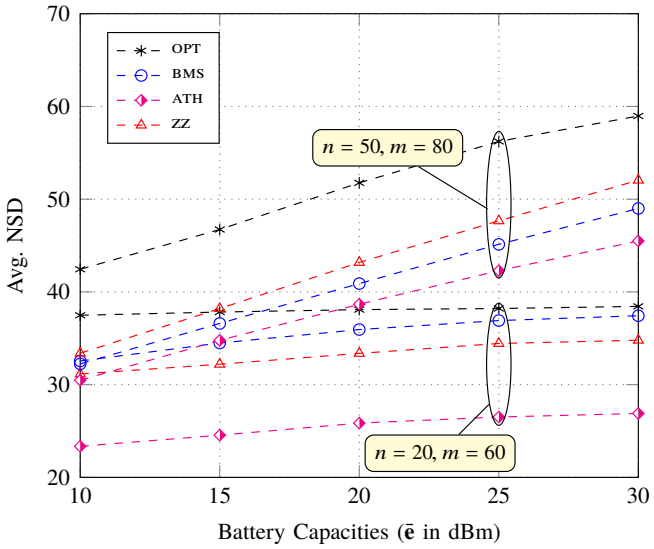


Fig. 4: Impact of $\bar{\mathbf{e}}$

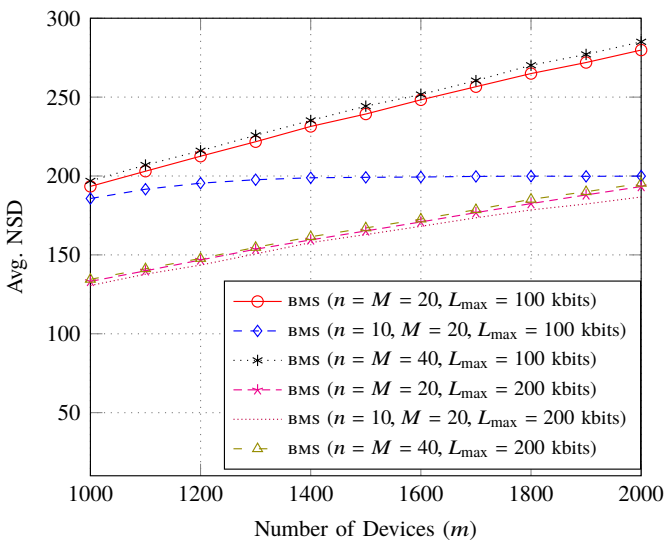
are limited, the increase rate slows down as the battery capacity increases, and thus the curves start to converges. Despite being online and much simpler, BMS is very close to OPT and, as discussed previously, ZZ, due to its offline nature, is better than BMS only for large n but at the expense of higher running time complexity.

Fig. 5a shows the performance of BMS for large m and M. (Comparison with other algorithms is missing due to running time issues and incompatibility with large M.) When the minimum rate requirements $L_{\max}$ is large, increasing M does not help improve the performance even for different n because the interference inside a NOMA group will become large and thus no more devices can be admitted. However, when the $L_{\max}$ is not very large, then increasing n and M can help improve the performance. When the network is really dense, it is not beneficial to increase n or M very largely. Due to increased SIC complexity and to the limited performance improvements, it is better to keep the values of n and M not very large, e.g., when $L_{\max} = 100$ kbits, $n = M = 20$ serves about 13.95% of the devices but $n = M = 40$ serves about 14.25% of the devices. It is thus better to choose $n = M = 20$ rather than $n = M = 40$ (the latter gives a gain of only 0.3%).

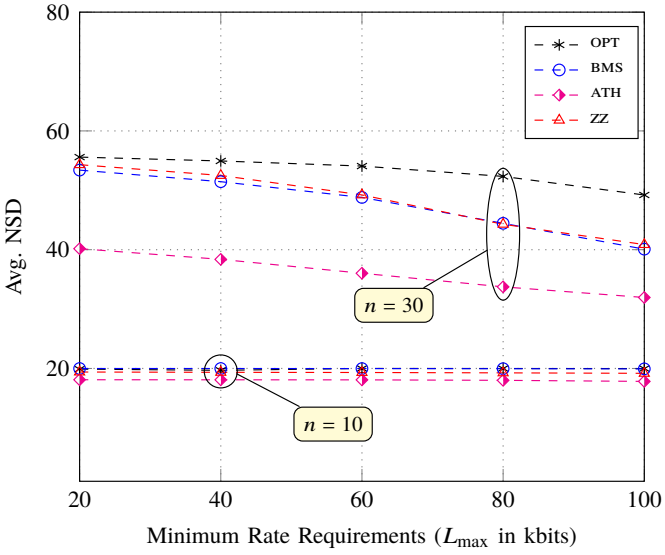
Fig. 5b illustrates the impact of the minimum rate requirements $L_{\max}$ on the performance of the algorithms for $m = 60$. When the number of RBs n is small, the NSD slightly decreases for large values of $L_{\max}$. However, when n is large, the NSD decreases faster with $L_{\max}$. This is because when n is large and $L_{\max}$ is small, an important number of devices can be grouped using NOMA (almost 80% of the devices are served). As soon as $L_{\max}$ increases, some important number of devices will be unsatisfied and thus the performance drops. Nonetheless, the rate of dropping is much better when n is small since a very few number of devices are grouped using NOMA (almost 30% of the devices are served) for small $L_{\max}$. Thus, unless $L_{\max}$ is not large, NOMA can help serve the same *few* number of devices with different rate requirements. On the other hand, NOMA can serve a larger number of devices but is more influenced by their stringent rate requirements. This remark was derived in [19] when comparing NOMA and OMA.

In the next simulations (Fig. 6), we show the performance of the proposed learning algorithms which learn the power allocation across the frames. The number of rounds (or episodes) is denoted by T. Unless stated otherwise, $k = 20$, $n = 10$, $m = 100$, and $T = 100$. The learning rate of QL is $\alpha = 0.5$, the PL's parameters are $\gamma = 0.5$, $\beta = 0.01$, and $\eta = \gamma/(2(k+1)\sigma_i)$.

In Fig. 6a, we compare PL against QL and a very simple algorithm, called random learning (RL), which assigns a random amount of power in each frame (from the amount

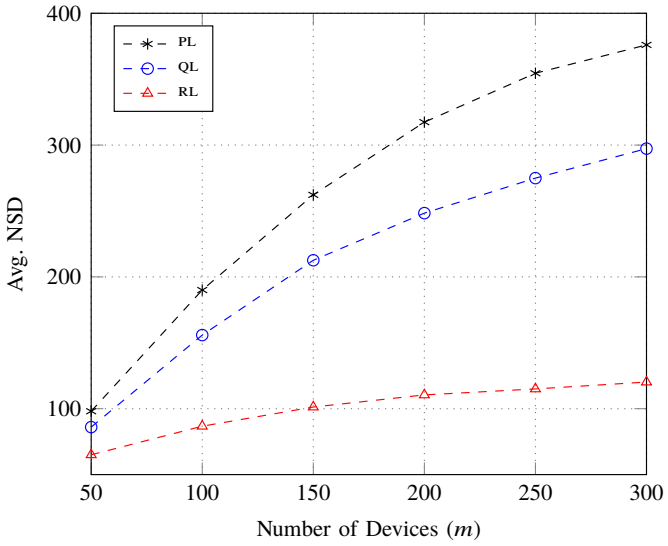


(a) Impact of large m and M.

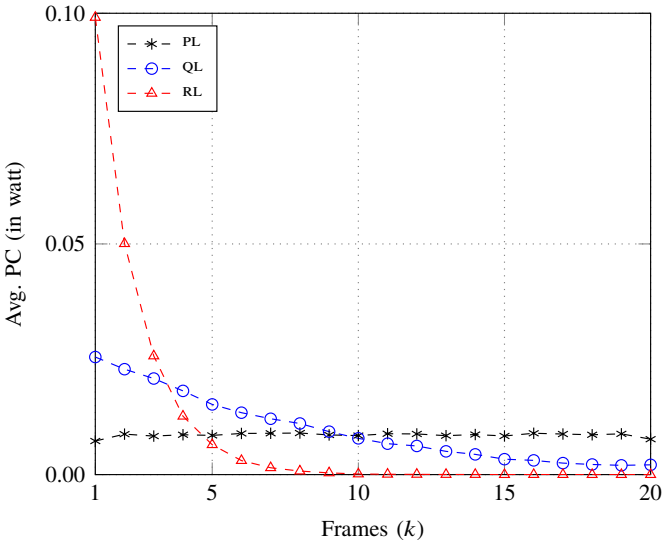


(b) Impact of $L_{\max}$.

Fig. 5: Impact of m, M, and $L_{\max}$.



(a) Impact of m.



(b) Power consumption.

Fig. 6: Impact of m and the power consumption across the frames.

of power left). Learning the transmission powers using PL achieves the best performance whereas the worst performance is achieved, as expected, by RL that makes the energy deplete quickly as the number of frames increases due to its random choices. Comparing PL and QL, the performance of the latter degrades as the number of devices increases. The performance gap between PL and QL is about 1.13 for $m = 50$ whereas it becomes about 1.26 for $m = 300$. This is due to the design principle of PL which exploits the problem structure through exploring the TG edges and exploiting the best edges. It is also proven in [33] that PL achieves a regret proportional to $1/\sqrt{T}$ even for adversarial inputs.

In Fig. 6b, we plot the average power consumption (PC) of all devices across the frames. The PC is averaged over all devices and over all random realizations and it measures the average power allocation of all devices in each frame. We see that RL depletes its transmission powers in the first few frames to end up without energy at later times and thus serves few devices. This is because as the number of frames increases, the random choices available to RL decreases since the sampled power set $[\bar{e}_i]_{\tau_i}$ shrinks. The average PC of QL is much better than RL but still the former allocates more transmission powers to the first frames. However, PL allocates the transmission powers good enough which gives a good learning outcome. Indeed, PL almost have uniform average PC across the frames and thus it saves more energy for future frames. Consequently, the power allocation obtained by PL helps improving the performance by serving the largest number

of devices compared to QL and RL.

VIII. Conclusion

In this paper we studied the online grouping and power allocation problem in beyond 5G cellular-based IoT NOMA networks where we introduced stringent requirements including real-time, rate, and energy constraints. First, we formulated the problem as a mathematical optimization model using integer programming techniques. Then, we studied the complexity of the problem by characterizing its NP-hardness in different and important cases. To solve the problem in a practical way, we divided it into subproblems of NOMA grouping and scheduling. Then, we proposed online competitive algorithms to group the devices using NOMA. To obtain the transmission power allocation solution, we proposed to use machine learning techniques and combined the NOMA grouping and scheduling solutions to obtain a global solution. Specifically, by modeling the power allocation problem as an online (stochastic) shortest path problem in directed graphs, we proposed two reinforcement learning algorithms: (1) based on exponential-weighting for exploration and exploitation and (2) Q-learning. We showed that our proposed solutions can help solving, in an online fashion, the massive access problem efficiently in future networks.

References

[1] A. Bakshi, L. Chen, K. Srinivasan, C. E. Koksal, and A. Eryilmaz, "EMIT: An Efficient MAC Paradigm for the Internet of Things," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1572–1583, Aug. 2019.

[2] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of Things in the 5G Era: Enablers, Architecture, and Business Models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[3] N. Xia, H. Chen, and C. Yang, "Radio Resource Management in Machine-to-Machine Communications–A Survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 791–828, Firstquarter 2018.

[4] Y. . E. Wang, X. Lin, A. Adhikary, A. Grovlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi, "A Primer on 3GPP Narrowband Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 117–123, Mar. 2017.

[5] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, "Toward Massive Machine Type Cellular Communications," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 120–128, Feb. 2017.

[6] M. Shirvanimoghaddam, Y. Li, M. Dohler, B. Vucetic, and S. Feng, "Probabilistic Rateless Multiple Access for Machine-to-Machine Communication," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 6815–6826, Dec. 2015.

[7] H. S. Dhillon, H. C. Huang, H. Viswanathan, and R. A. Valenzuela, "On Resource Allocation for Machine-to-Machine (M2M) Communications in Cellular Networks," in *Proc. IEEE Globecom Workshops*, Dec. 2012, pp. 1638–1643.

[8] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. New York, NY, USA: Cambridge University Press, 1998.

[9] S. Shalev-Shwartz, "Online Learning and Online Convex Optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[10] S. Ali, N. Rajatheva, and W. Saad, "Fast Uplink Grant for Machine Type Communications: Challenges and Opportunities," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 97–103, Mar. 2019.

[11] S. Ali, A. Ferdowsi, W. Saad, and N. Rajatheva, "Sleeping Multi-Armed Bandits for Fast Uplink Grant Allocation in Machine Type Communications," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–6.

[12] D. Zhang, Y. Qiao, L. She, R. Shen, J. Ren, and Y. Zhang, "Two Time-Scale Resource Management for Green Internet of Things Networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 545–556, Feb. 2019.

[13] D. Zhai, R. Zhang, L. Cai, B. Li, and Y. Jiang, "Energy-Efficient User Scheduling and Power Allocation for NOMA-Based Wireless Networks With Massive IoT Devices," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1857–1868, Jun. 2018.

[14] M. S. Ali, H. Tabassum, and E. Hossain, "Dynamic User Clustering and Power Allocation for Uplink and Downlink Non-Orthogonal Multiple Access (NOMA) Systems," *IEEE Access*, vol. 4, pp. 6325–6343, 2016.

[15] M. Choi, J. Kim, and J. Moon, "Dynamic Power Allocation and User Scheduling for Power-Efficient and Delay-Constrained Multiple Access Networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4846–4858, Oct. 2019.

[16] M. Zeng, A. Yadav, O. A. Dobre, and H. V. Poor, "Energy-Efficient Joint User-RB Association and Power Allocation for Uplink Hybrid NOMA-OMA," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5119–5131, Jun. 2019.

[17] D. Zhai and R. Zhang, "Joint Admission Control and Resource Allocation for Multi-Carrier Uplink NOMA Networks," *IEEE Wireless Commun. Lett.*, vol. 7, no. 6, pp. 922–925, Dec. 2018.

[18] D. Zhai, R. Zhang, L. Cai, and F. R. Yu, "Delay Minimization for Massive Internet of Things With Non-Orthogonal Multiple Access," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 3, pp. 553–566, Jun. 2019.

[19] A. E. Mostafa, Y. Zhou, and V. W. S. Wong, "Connection Density Maximization of Narrowband IoT Systems With NOMA," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4708–4722, Oct. 2019.

[20] L. Deng, W. S. Wong, P. Chen, Y. S. Han, and H. Hou, "Delay-Constrained Input-Queued Switch," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 11, pp. 2464–2474, Nov. 2018.

[21] L. Deng, C. Wang, M. Chen, and S. Zhao, "Timely Wireless Flows With General Traffic Patterns: Capacity Region and Scheduling Algorithms," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, pp. 3473–3486, Dec. 2017.

[22] N. Buchbinder, L. Lewin-Eytan, I. Menache, J. Naor, and A. Orda, "Dynamic Power Allocation Under Arbitrary Varying Channels - The Multi-User Case," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[23] Y. Liu, X. Fang, and M. Xiao, "Discrete Power Control and Transmission Duration Allocation for Self-Backhauling Dense mmWave Cellular Networks," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 432–447, Jan. 2018.

[24] F. Shan, J. Luo, W. Wu, M. Li, and X. Shen, "Discrete Rate Scheduling for Packets With Individual Deadlines in Energy Harvesting Systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 438–451, Mar. 2015.

[25] E. Altman, K. Avrachenkov, G. Miller, and B. Prabhu, "Discrete Power Control: Cooperative and Non-Cooperative Optimization," in *Proc. IEEE INFOCOM*, May 2007, pp. 37–45.

[26] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1979.

[27] B. Birnbaum and C. Mathieu, "On-line Bipartite Matching Made Simple," *ACM SIGACT News*, vol. 39, no. 1, pp. 80–87, Mar. 2008.

[28] D. Niyato, P. Wang, and D. I. Kim, "Performance Modeling and Analysis of Heterogeneous Machine Type Communications," *IEEE Trans. Wireless Commun.*, vol. 13, no. 5, pp. 2836–2849, May 2014.

[29] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. New York, NY, USA: Cambridge University Press, 2005.

[30] M. A. Sedaghat and R. R. Müller, "On User Pairing in Uplink NOMA," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, pp. 3474–3486, May 2018.

[31] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The Non-stochastic Multiarmed Bandit Problem," *SIAM J. Comput.*, vol. 32, no. 1, p. 48–77, Jan. 2003.

[32] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[33] A. Gyorgy, T. Linder, and G. Lugosi, "The Shortest Path Problem in the Bandit Setting," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Mar. 2006, pp. 87–91.

[34] E. Takimoto and M. K. Warmuth, "Path Kernels and Multiplicative Updates," in *Computational Learning Theory*, J. Kivinen and R. H. Sloan, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 74–89.

[35] 3GPP, "Cellular System Support for Ultra-Low Complexity and Low Throughput Internet of Things (CIoT)," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 45.820, Nov. 2015, version 13.1.0.

[36] R. Fourer, D. M. Gay, and B. W. Kernighan, "A Modeling Language for Mathematical Programming," *Manage. Sci.*, vol. 36, no. 5, p. 519–554, May 1990.