Maximizing Packets Collection in Wireless Powered IoT Networks With Charge-or-Data Time Slots

Xiaoyu Song and Kwan-Wu Chin

***Abstract*—This paper studies data collection in a wireless powered Internet of Things (IoT) network with a hybrid access point (HAP). A fundamental problem at the HAP is to determine the number of time slots over a given planning horizon that is used to charge and collect data from devices. To this end, we outline a mixed integer linear program (MILP) to determine (i) the mode (charge or data) of each slot, (ii) the HAP's transmit power allocation, and (iii) transmitting devices in data slots. Further, we propose a receding horizon approach whereby the HAP solves the said MILP over a time window using channel estimates from a Gaussian mixture model (GMM). We also outline a data-driven approach. In its *offline* stage, an IoT network operator first solves the said MILP over an exhaustive collection of channel power gains. The MILP solution for each channel power gain realization is then stored in neural networks. In the *online* stage, the HAP accesses the trained neural network of each time slot to retrieve its mode. The results show that the rolling horizon and data-driven approach allow the HAP to receive up to 106% and 90% more packets as compared to competing approaches.**

***Index Terms*—Model predictive control, neural network, uplinks transmissions, wireless or RF charging.**

I. Introduction

INTERNET of Things (IoT) networks have applications in various industries such as healthcare [1], and they form the foundation of smart cities [2]. In these applications, devices are used to gather data such as velocity, temperature or humidity to name a few [3]. This information is then transmitted to a sink or base station, which may change the environment through one or more actuators; e.g., a device could switch an air-conditioner on/off as per the temperature of a room [4].

The data collection process of an IoT network is affected by the amount of energy at devices, especially, those powered by non-rechargeable batteries. To this end, applying radio frequency (RF) energy harvesting technology [5] to replenish the energy of sensing devices is now of interest. Specifically, devices can be powered by RF signals emitted by television towers or access points [6]. For example, the Powercast P2110B receiver is able to harvest 2.75 mW of power from RF signals transmitted on the 850 to 950 MHz band [7].

Manuscript received 19 October 2022; revised 13 January 2023 and 24 February 2023; accepted 19 April 2023. Date of publication 26 April 2023; date of current version 14 August 2023. The associate editor coordinating the review of this article and approving it for publication was Y. Gao. *(Corresponding author: Xiaoyu Song.)*

The authors are with the School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, Wollongong, NSW 2522, Australia (e-mail: xs579@uowmail.edu.au; kwanwu@uow.edu.au).

Digital Object Identifier 10.1109/TCCN.2023.3269508

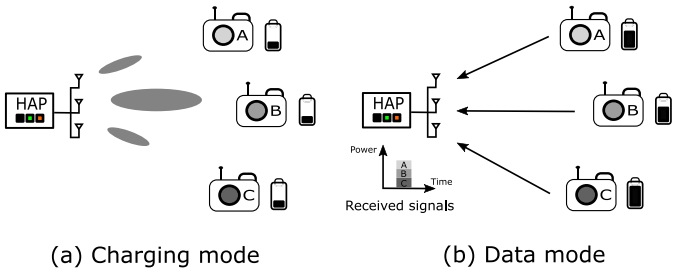

Fig. 1. A SIC-enabled multi-antenna HAP, and N RF harvesting devices. (a) In *charging* time slots, the HAP uses switched beamforming to charge devices. (b) in *data* time slots, the HAP selects a subset of devices to transmit their packet.

Another example is [8], where a prototype comprising of a hybrid access point (HAP) with multiple-input multiple-output (MIMO) powers devices located 50 meters away.

Another approach to improve data collection in IoT networks is to employ multi-user detection [9]. Specifically, interference or collision is a fundamental channel access issue, which limits the amount of data uploaded by devices. As a result, recent works such as [10] consider nodes with a successive interference cancellation (SIC) radio. This radio allows a receiver to decode multiple concurrent transmissions if their respective signal to interference plus noise ratio (SINR) meets a given threshold.

In this work, we study data collection in a wireless powered IoT network that uses a HAP with switched beamforming to deliver energy and SIC for packet reception. Fig. 1 illustrates our system. Specifically, a beamforming HAP charges devices using RF signals. The HAP has SIC capability, and thus it is able to decode multiple uplink transmissions simultaneously. Our *aim* is to maximize the amount of packets transmitted by devices over multiple time slots or a planning time horizon. Unlike past works, see Section II, we *do not* assume the conventional harvest-then-transmit frame structure [11]. Instead, a HAP determines the operation mode of each time slot; namely, a time slot is either in *charging* or *data* mode. In particular, if the channel gains from the HAP to devices are favourable for energy delivery, then it uses a time slot for charging. Conversely, if the channel gains lead to a high number of SIC decoding successes, then the HAP uses a time slot to receive a packet from devices. In contrast, for the harvest-then-transmit frame structure [11], the HAP and devices are required to deliver energy and transmit data regardless of channel condition.

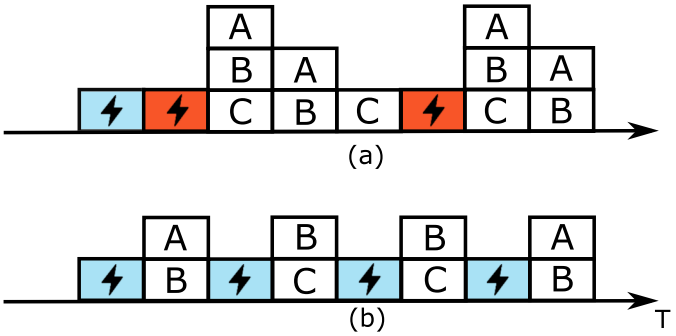

Fig. 2. Two examples of charging and data collection schedule: (a) Mode selection structure, and (b) Conventional harvest-then-transmit frame structure. The light blue and red colored boxes respectively indicate charging over poor and good channel condition. The letter A, B and C correspond to the devices in Figure 1.

Fig. 2 shows an example of the mode time structure considered in this work, and the conventional harvest-then-transmit structure [11]. Specifically, light blue boxes/slots indicate charging over poor channel condition, resulting in poor energy delivery. In contrast, red colored boxes/slots denote good channel condition. We see that the channel condition in the first slot leads to devices having a low amount of harvested energy. Consequently, only two devices are able to transmit their packet using the harvest-then-transmit schedule. Now consider the case whereby the HAP considers whether a slot is better suited for charging or data transmission. In this example, the channel condition is good in the second slot, meaning if it is used for charging, devices are able to harvest more energy. As a result, there are three more transmissions in Fig. 2(a). Observe also that there is a coupling between time slots, meaning the energy level at devices is a function of past charging and data transmission slots.

The number of packets collected by a HAP is affected by the following factors: (i) the number of *charging* and *data* slots over a given planning horizon, where there is a fundamental trade-off between the number of *charging* and *data* slots, and (ii) devices that are selected in each data slot. For factor (i), the main issue is that devices must be charged before they are able to transmit. However, having more *charging* slots means they are fewer opportunities for data transmissions. A main challenge in this respect is that the HAP has non-causal channel information. Specifically, the HAP is not aware of the amount of energy harvested by devices or the number of uplink transmissions in future slots. As for factor (ii), as the HAP is equipped with a SIC radio, it is able to decode multiple uplink transmissions in each *data* slot. Here, a key problem is to select transmitting devices in each *data* slot, where uplink transmissions must satisfy a given SIC condition relating to the SINR of each uplink transmission at the HAP.

To this end, we make the following contributions:

- We address a novel data collection problem in a wireless powered IoT network comprising of an HAP with beamforming and SIC capabilities. We outline the first mixed integer linear program (MILP) to determine (i) the mode of time slots over a planning horizon T, (ii) the weight of each antenna in charging slots, and (iii) transmitting devices in data slots. The objective of the MILP is to maximize the total number of uplink transmissions over multiple time slots. In particular, given T time slots, we optimize how many time slots are used for energy and data transmissions. Note that the MILP produces the optimal upper bound on the maximum number of data transmissions assuming causal channel gain information.
- We outline a rolling horizon approach, namely *RollH*, that enables the HAP to determine the mode of future time slots. Briefly, we use a Gaussian mixture model (GMM) [12] to estimate future channel power gains to devices over planning horizon T. The estimated channel power gains are then used to solve the said MILP to yield the mode of each slot. In addition, we propose a data-driven approach, namely *n-RollH*, that utilizes neural networks to retrieve the mode of each slot. Advantageously, unlike *RollH*, this approach allows a HAP to retrieve a solution that is computed *offline*. Specifically, *n-RollH* shifts the computational expensive task of solving the MILP to the offline stage. This thus allows it to take advantage of the massive computational power at data centers and historical channel power gains information.
- We outline the first study of the said solutions. The results show that *RollH* achieves up to respectively 106% and 90% higher system throughput than competing solutions. Lastly, *RollH* and *n-RollH* outperform the approach in [13] by 87% and 84.7%, respectively.

In Section II discusses relevant works and highlight gaps in the literature. In Section III, we present our network model and in Section IV, we formalize our problem using an MILP. Our proposed approaches are detailed in Section V. Their evaluation is presented in Section VI, followed by our conclusion in Section VII.

II. Related Works

Our work overlaps with research into MIMO RF energy harvesting SIC-enabled wireless powered communications networks (WPCNs). To the best of our knowledge, none of them, except [13], consider selecting a mode for each time slot. Specifically, past works assume a frame that consists of a charging and data upload phase; aka harvest-then-transmit frame structure [11], i.e., they do not jointly optimize the mode of multiple time slots. We further elaborate on this fact in the discussion to follow.

Many works have studied data collection in WPCNs. For example, the works surveyed in [14] assume devices use time division multiple access (TDMA), where devices transmit in a pre-assigned time slot. In this respect, channel access is not a concern. Moreover, they assume only a single transmission for each time slot. A number of works consider random access protocols. For example, works such as [15], [16] consider carrier sense multiple access (CSMA), where devices first listen to the channel before transmitting their data. Specifically, the authors of [16] design a protocol to adjust the charging period of a HAP and the contention window of devices. The work in [17] and [18] considers slotted Aloha, where in [17], idle

slots are used to charge devices. In [18], the authors design a learning approach to adjust the frame size for a given number of devices. A major limitation of these works is that they do not consider non orthogonal multiple access (NOMA), meaning they have low spectrum efficiency.

In terms of NOMA works, their aims are to maximize the performance of a WPCN in terms of its sum-rate, minimum throughput and/or energy efficiency. None of these works, however, consider a mode-based time structure. Moreover, they only consider energy and data transmission over *one* frame. In general, these works consider optimizing the charging power of a HAP, the transmit power of devices, and charging and/or data transmission duration of a harvest-then-transmit frame. The work in [19] considers a two-user WPCN and assume a linear energy harvesting model. They aim to maximize the sum-throughput of devices by jointly allocating a HAP's charging duration and device transmit power. Reference [20] studies throughput maximization for both SIC and conventional single-user decoding. In [21], the authors propose a centralized and a distributed learning strategy to determine the transmit power of devices. The authors in [22] study the impact of charging duration and SIC decoding order on system throughput. In [23], the authors select devices to transmit and vary the charging duration of a HAP to maximize system throughput. The work in [24], [25], [26] considers a HAP with multiple antennas. They aim to maximize sum-rate by optimizing the HAP's energy beamforming, charging duration and device transmit power. In particular, the authors in [26] consider a MIMO WPCN and aim to improve both uplink and downlink sum-rate. The authors of [27] equip a HAP and all devices with beamforming capabilities and aim to maximize sum-rate by jointly optimizing downlink and uplink beamforming. In [28], the authors consider a WPCN where devices are grouped into clusters. They design an algorithm to adapt the beamforming weight for each cluster. The work in [29] and [30] focuses on maximizing the minimum throughput of devices. Specifically, the authors in [29] jointly optimize energy transfer duration of a HAP and device transmit power. On the other hand, [30] optimizes the charging and data transmission duration. In addition, there also are works that aim to minimize the energy transmitted by the HAP in NOMA-based WPCNs. For example, the authors of [31] aim to optimize the time used for charging and data transmission as well as the HAP's transmit power. These works, however, do not consider a mode time structure; i.e., they do not consider energy and data transmissions over multiple slots.

There is also research into user scheduling in NOMA WPCNs, e.g., [32], [33], [34], [35], [36], [37]. Their aim includes throughput maximization, energy efficiency and overcoming the double near-far effect problem [11]. For example, the work in [32] uses a column generation approach to schedule devices for data transmission. In [33], a HAP learns the best transmission schedule with a high throughput using only imperfect channel state information. The work in [34], [35], [36], [37] addresses the double near-far effect problem. In [34], the authors optimize the SIC decoding order of devices. Reference [35] proposes two decoding order strategies to improve the data rate of devices. In [36], the authors aim to schedule concurrent energy and data transfer. These works, however, only schedule devices over *one* time slot/frame, where they do not take advantage of favorable downlinks/uplinks channel condition in future time slots. In addition, their system does not have MIMO energy transfer capability.

TABLE I
A COMPARISON OF RELATED WORKS

References	RF Charging	SIC	MIMO	Mode Selection	Learning
[14]–[17]	✓	✗	✗	✗	✗
[18]	✓	✗	✗	✗	✓
[19], [20], [23], [35], [29]–[31], [37]–[41]	✓	✓	✗	✗	✗
[32]–[34], [36]	✓	✓	✗	✗	✗
[21]	✓	✓	✗	✗	✓
[24], [25], [42]	✓	✓	✓	✗	✗
[28]	✓	✓	✓	✗	✗
[43], [44]	✗	✓	✗	✗	✗
[13]	✓	✗	✗	✓	✗
Our work	✓	✓	✓	✓	✓

Table I compares our work against prior research. The main novelties of our work include channel access over a mode-based frame structure. Specifically, it is the first to consider such a frame structure in a WPCN with a beamforming SIC-capable HAP. Further, it outlines a data-driven method that uses neural networks to store past solutions that can be retrieved by the HAP in an online setting. The closest work to ours is [13], where the authors aim to determine the operation mode of an HAP over multiple time slots. However, as shown in Table I, [13] considers a different system setup. Moreover, it does not consider storing past solutions using a neural network.

III. System Model

Time is divided into T slots, each with length τ (in second), and indexed by t, where $t \in \mathcal{T} = \{1, 2, \ldots, T\}$. We consider a wireless powered network that contains a HAP and a set $\mathcal{N} = \{1, 2, \ldots, N\}$ of devices, indexed by n. Each device has an energy storage with size Φ. The distance between device n and the HAP is d_n. The HAP has a maximum transmit power of P_{max}. It uses switched beamforming with a set of antennas denoted as $\mathcal{K} = \{1, 2, \ldots, K\}$. We use the model in [45]. Specifically, the HAP is capable of generating K single-lobe beams, and it forms multi-lobe beam patterns by weighting single-lobe beams. Let the weight (transmit power) of the k-th beam in slot t be a_k^t, where

$$\sum_{k \in \mathcal{K}} a_k^t \leq P_{max}. \tag{1}$$

In each time slot t, the HAP either operates in *charge* or *data* mode. Let $I_c^t \in \{0, 1\}$ and $I_d^t \in \{0, 1\}$ be binary variables, where $I_c^t = 1$ means slot t is a charging slot and $I_d^t = 1$ indicates slot t is a data slot. In a charging slot, the channel power gain between device n and the k-th antenna of the HAP is modeled as

$$h_{k,n}^t = \underline{h}_{k,n}^t \, d_n^{-\beta}, \tag{2}$$

where $\underline{h}_{k,n}^t$ is an Exponentially distributed random variable with unit mean and β denotes the path loss exponent. We assume block fading, which means the channel power gain is fixed in each time slot but varies across time slots. Define $P_{n,k}^{t*} = P_{max} h_{k,n}^t$ as the received power at device *n* when the HAP transmits using P_{max} over beam *k* only. As per [45], the received power at device *n* is

$$P_n^t = \sum_{k=1}^{K} a_k^t P_{n,k}^{t*}. \tag{3}$$

Each device *n* is equipped with an RF-energy harvester. We consider a non-linear energy harvesting model. Specifically, device *n*'s harvested power given its received power P_n^t is defined as

$$\psi\left(P_n^t\right) = \frac{\frac{M}{1+\exp(-aP_n^t+b)} - M/(1+\exp(ab))}{1 - 1/(1+\exp(ab))}. \tag{4}$$

Specifically, *M* is the maximum received power when the circuit is saturated; the constants *a* and *b* relate to a circuit's specification such as its resistance and capacitance, and their values are obtained via curve fitting [46]. As a result, the energy harvested by device *n* is

$$H_n^t = \psi\left(P_n^t\right)\tau. \tag{5}$$

In data slot *t*, the energy cost of a packet is ϵ Joule, which may also include circuit and retransmissions cost [47]. Apart from that, the HAP receives packets using an omni-directional beam pattern. The channel power gain is denoted as

$$g_n^t = \underline{h}_n^t d_n^{-\beta}, \tag{6}$$

where $\underline{h}_n^t$ is drawn from an Exponential distribution with unit mean, and β is the path loss exponent.

The HAP has SIC capability, meaning it is able to decode multiple uplink transmissions simultaneously [10]. To do so, the HAP first decodes the signal with the highest received power. This decoded signal is then removed/subtracted from the received signal. The HAP then decodes the next signal with the second highest received power. The decoding process ends when all signals are decoded or a signal is decoded unsuccessfully. As devices use the same transmit power for each packet transmission, their decoding order is pre-determined according to their channel power gain. In particular, the order is denoted by a function $o^t : \mathcal{N} \to \{1, 2, \ldots, N\}$, where $o^t(n)$ is the position of device *n* in the decoding order in slot *t*. We have $o^t(n) \geq o^t(m)$ if $g_n^t \leq g_m^t$, which means the device with a higher uplink channel power gain will be decoded earlier.

In data slot *t*, device *n* transmits a signal to the HAP using transmit power p_n^t. The SINR of this signal is

$$\gamma_n^t = \frac{p_n^t g_n^t}{\sum_{m \neq n, o^t(n) \leq o^t(m)} p_m^t g_m^t I_m^t + N_o}, \tag{7}$$

where N_o is the noise power. A signal is successfully decoded if $\gamma_n^t \geq \Gamma$. A binary variable $I_n^t\{0,1\}$ is used to indicate if device *n* transmits a packet in slot *t*, where $I_n^t = 1$ denotes that the packet sent by device *n* in slot *t* is successfully decoded.

An example of the SIC decoding process is given as follows. Assume there are three devices: n_1, n_2, n_3, where their transmit power is p_1, p_2, p_3, respectively. The received power at the HAP is $p_1 g_1, p_2 g_2, p_3 g_3$. Assume $p_1 g_1 > p_2 g_2 > p_3 g_3$, which means $o^t(1) < o^t(2) < o^t(3)$. The SINR of n_1 is

$$\gamma_1 = \frac{p_1 g_1}{p_2 g_2 + p_3 g_3 + N_o}. \tag{8}$$

The signal is successfully decoded and removed from the composite/received signal if $\gamma_1 \geq \Gamma$. Otherwise, the transmission fails and the decoding process ends. After that, the HAP decodes the signal from n_2, which has the SINR

$$\gamma_2 = \frac{p_2 g_2}{p_3 g_3 + N_o}. \tag{9}$$

If we have $\gamma_2 \geq \Gamma$, the packet sent by device 2 is successfully decoded and it is removed from the composite signal. Finally, the HAP decodes the signal from n_3. The signal is successfully decoded if $\gamma_3 \geq \Gamma$, with the resulting SNR being

$$\gamma_3 = \frac{p_3 g_3}{N_o}. \tag{10}$$

In slot $t+1$, the energy at device *n* is calculated as $E_n^{t+1} = E_n^t + H_n^t I_c^t - \epsilon I_n^t$. After *T* slots, the number of packets collected from devices is calculated as

$$R = \sum_{t=1}^{T} \sum_{n=1}^{N} I_n^t. \tag{11}$$

Note that in this paper we only aim to maximize the number of packets collected from devices; we do not consider sum-rate because in practice devices transmit packets. Further, we emphasize that considering specific application requirements is outside the scope of this paper. We note, however, that Eq. (11) can be revised to weigh the transmissions of devices differently depending on the needs of applications. We assume the HAP uses an omni-directional beam pattern in data time slots. It is possible to optimize the beam pattern used for data receptions at the expense of higher signaling cost. To elaborate, given a set of transmitting devices, the AP will have to measure their channel gain for each beam pattern. It then selects the beam pattern that yields the highest SINR for each device. Table II summarizes our notations.

TABLE II
TABLE OF NOTATIONS

Notation	Description
$\mathcal{N}$	Set of devices
d_n	Distance from the HAP to device n
Φ	Battery size
τ	Slot duration
P_{max}	Maximum transmit power of the HAP
β	Path-loss exponent
M	Maximum received power
a, b	Constants in energy harvesting circuit model
H_n^t	Harvested energy
ϵ	Energy cost for transmitting a packet
Γ	SIC decoding SINR threshold (in dB)
γ_N^t	SINR of a packet transmitted by device n in slot t
N_o	Noise power
I_c^t	A binary variable indicating if slot t is in *charge* mode
I_d^t	A binary variable indicating if slot t is in *data* mode
I_n^t	A binary variable indicating if device n transmits in slot t
a_k^t	The weight of the k-th beam in slot t

IV. Problem Formulation

Our aim is to maximize the number of packets R collected by the HAP. To do so, we have to (i) determine the mode of time slots, i.e., I_d^t, I_c^t, (ii) optimize the transmit power allocation or antenna weight a_k^t at the HAP for charging slots, and (iii) schedule the set of transmitting devices in data time slots. Formally, in a data time slot t, the set of transmitting devices is defined as $\{n|I_n^t = 1 \wedge I_d^t = 1 \wedge n \in \mathcal{N}\}$. Define a vector $\mathbf{v} = [a_k^t, I_d^t, I_c^t, I_n^t]$. The problem of maximizing the number of data transmissions can be modeled as the following mixed integer non-linear program (MINLP):

$$\underset{\mathbf{v}}{\text{maximize}} \quad R \tag{12a}$$

$$\text{subject to} \quad \sum_{k=1}^{K} a_k^t \leq P_{max} I_c^t, \quad \forall t \in \mathcal{T} \tag{12b}$$

$$I_n^t \leq I_d^t, \quad \forall t \in \mathcal{T}, \forall n \in \mathcal{N} \tag{12c}$$

$$I_c^t + I_d^t = 1, \quad \forall t \in \mathcal{T} \tag{12d}$$

$$E_n^t \geq 0, \quad \forall t \in \mathcal{T}, \forall n \in \mathcal{N} \tag{12e}$$

$$\gamma_n^{t\prime} \geq \Gamma, \quad \forall t \in \mathcal{T}, \forall n, m \in \mathcal{N}. \tag{12f}$$

In the above formulation, we define $\gamma_n^{t\prime}$ as

$$\gamma_n^{t\prime} = \frac{p_n^t g_n^t + \Phi\big(1 - I_n^t\big)}{\sum_{m \neq n, o^t(n) \leq o^t(m)} p_m^t g_m^t I_m^t + N_o}. \tag{13}$$

In the previous equation, Φ is a large constant that is used to disable constraint (12f).[1] Specifically, when we have $I_n^t = 0$, the left hand side of inequality (12f) is always higher than the right hand side. Thus, constraint (12f) is disabled. On the other hand, when $I_n^t = 1$, the second term in the numerator is removed, meaning Constraint (12f) must be satisfied/enabled.

Constraint (12b) sums the weight of antennas to be less than the maximum transmit power of the HAP in a charging slot. In a data slot, the weight of each antenna is limited to zero. Constraint (12c) shows that devices are able to transmit packets only when slot t is in *data* mode. Constraint (12d) indicates that the HAP can only select one mode in each slot. Constraint (12e) means that the available energy of each device must be non-negative. Constraint (12f) indicates that if device n successfully transmits a packet, its SINR value must be higher than a threshold.

A. Linearization

As mentioned earlier, we have an MINLP problem. This is because of the non-linear RF-energy conversion process, i.e., Eq. (4). To this end, we linearize Eq. (4) by approximating it with linear segments. Doing so results in a MILP, which can then be solved to optimality using the many computationally efficient algorithms adopted by solvers such as Gurobi [48].

Define a set $\mathcal{S} = \{1, 2, \ldots, S\}$ of intervals, where P_s^L and P_s^U denote the lower and upper bound of interval s. If the received power P_n^t falls within the interval s, we use conversion efficiency η_s. In particular, η_s is calculated as

$$\eta_s = \frac{\psi\big(P_s^L\big)}{2P_s^L} + \frac{\psi\big(P_s^U\big)}{2P_s^U}. \tag{14}$$

[1]This is also known as the big-M method.

In Eq. (14), ψ is the non-linear energy harvesting model; see Eq. (4). Let $M_{s,n}^t$ denote the output power of device n in slot t for interval s, where $M_{s,n}^t = \eta_s P_n^t$ if the received power P_n^t falls within the interval s.

Define a binary variable $J_{s,n}^t$ for each slot t. It is set to one if the received power of device n falls in the s-th interval. For all devices in slot t, the variable $J_{s,n}^t$ is set as follows

$$\big(J_{s,n}^t - 1\big)\Phi + P_s^L \leq P_n^t \leq P_s^U + \big(1 - J_{s,n}^t\big)\Phi. \tag{15}$$

In other words, when the received power of device n falls in the s-th interval in slot t, where $P_s^L \leq P_n^t \leq P_s^U$, we have $J_{s,n}^t = 1$. Otherwise, $J_{s,n}^t$ is set to zero so that inequality (15) holds. Here, Φ can set to the maximum battery capacity.

In each charging slot, only one $J_{s,n}^t$ is set to one. In each data slot, each $J_{s,n}^t$ is set to zero. As a result, we have

$$\sum_{s=1}^{S} J_{s,n}^t = I_c^t. \tag{16}$$

Inequality (17) and inequality (18) are used to determine the converted power $M_{s,k}^t$. In particular, when $J_{s,n}^t$ is set to one, the output power $M_{s,n}^t$ is set to $\eta_s P_n^t$. On the other hand, if $J_{s,n}^t$ is zero, the output power $M_{s,n}^t$ is set to zero.

$$\eta_s P_n^t - \big(1 - J_{s,n}^t\big)\Phi \leq M_{s,n}^t \leq \eta_s P_n^t. \tag{17}$$

$$0 \leq M_{s,n}^t \leq J_{s,n}^t \Phi. \tag{18}$$

We are now ready to present out MILP. Rewrite the vector $\mathbf{v}$ as $\mathbf{v} = [a_k^t, I_d^t, I_c^t, I_n^t, J_{s,n}^t, M_{s,n}^t]$. As a result, we have the following MILP:

$$\underset{\mathbf{v}}{\text{maximize}} \quad R$$
$$\text{subject to} \quad (12b)\text{–}(12f), \quad (15)\text{–}(18). \tag{19}$$

We note the value of S, i.e., number of intervals, has an impact on the results between MILP and MINLP (12a). Specifically, to reduce the gap between the results of MILP and MINLP, we can increase the value of S, and thus gaining a better approximation to the energy conversion function $\Psi(.)$. However, this increases computation complexity of the resulting MILP.

We conclude this section by characterizing the computational complexity of our problem, and MILP (19).

Proposition 1: Given T slots and N devices, the solution space has size $\sum_{i=0}^{T} \binom{T}{i} (\sum_{j=1}^{N} \binom{N}{j})^i$.

Proof: For T slots, there are $\binom{T}{i}$ choices of data slots, where i denotes the number of data slots over T slots. In each data slot, it is possible to have zero or at most N transmitting devices. If the HAP selects j devices to transmit in each data slot, there are $\binom{N}{j}$ choices of transmitting devices. These facts imply the size of solution space is $\sum_{i=0}^{T} \binom{T}{i} (\sum_{j=1}^{N} \binom{N}{j})^i$. ■

Proposition 2: Given T slots, N devices and an HAP with K antennas, MILP (19) has $(2 + K)T + (1 + 2S)TN$ decision variables and $2T + 4TN + 6TNS$ constraints.

Proof: Decision variable I_c^t and I_c^t exist for each slot $t \in \mathcal{T}$, which results in $2T$ variables. Each HAP antenna has a decision variable a_k^t in each slot, meaning there are KT such decision variables. For each device, variable I_n^t exists in each

slot, so there are TN variables. In constraints related to linearization, for each interval s, decision variable $J_{s,n}^t$ and $M_{s,n}^t$ exist for each device in each slot. Hence, there are $2STN$ decision variables. The above facts imply the number of decision variables of MILP (19) is $(2 + K)T + (1 + 2S)TN$. We now consider the constraints of MILP (19). There is a transmit power constraint (12b) and a mode constraint (12d) for each time slot, which equates to $2T$ constraints. For each device in each time slot, constraint (12c) ensures a device only transmit packets in a data slot,and constraint (12f) ensures a packet is successfully decoded if its SINR is higher than the threshold. We also use constraint (12e) to ensure the residual energy of a device is non-negative. In addition, we also have constraint (16) to ensure the linearization occurs only in charging slots. The above facts means that we have another $4TN$ constraints. In addition, constraint (15), (17) and (18) exists for each interval of each device in each time slot. As a result, MILP (19) has $2T + 4TN + 6TNS$ constraints in total. This completes the proof. ■

V. Solutions

Recall that solving MILP (19) requires the HAP to have channel gain information to devices in all time slots. However, in practice, the HAP only has access to current and historical information. Hence, the HAP does not have the benefit of foresight when deciding whether the current slot should be used for charging or data transmission. Further, the HAP's current decision will have an impact on its decision in the future. For example, assume the current time slot has good channel condition. The HAP may decide to use it for charging if it doing so leads to more packet transmissions in future time slots. On the other hand, the HAP may use the time slot for data transmission(s) if many prior time slots have been used for charging devices.

To this end, we propose two approaches. Briefly, the first approach, namely Rolling Horizon (*RollH*), requires the HAP to estimate channel power gains to devices and solve an MILP in each time slot. Although this approach yields the optimal solution, it is computationally expensive. The second approach, namely neural network Rolling Horizon (*n-RollH*), has an *offline* and *online* stage. In the *offline* stage, an operator makes use of previously collected channel power gains information. For each channel gain realization, it solves for the optimal mode of each time slot using MILP (19). The mapping between a channel power gain realization and the optimal solution computed by MILP (19) is then stored in neural networks. The operator then stores these trained neural networks at the HAP. In the *online* stage, the HAP is only required to retrieve the mode of a time slot from the said neural networks based on channel gain information. Advantageously, the HAP does not have to solve a computationally expensive MILP in every time slot.

A. RollH – A Rolling Horizon Approach

We start by describing the rolling or receding horizon framework [49] that is used by both *Roll-H* and *n-RollH*. Rolling horizon is a feedback control technique that can be implemented in real-time systems. At each step, a controller uses an estimate of future information to solve an optimization problem in order to obtain the actions over a planning horizon. The controller then applies the first action computed for the planning horizon. It then shifts the planning horizon one time step forward and repeats the previous steps.

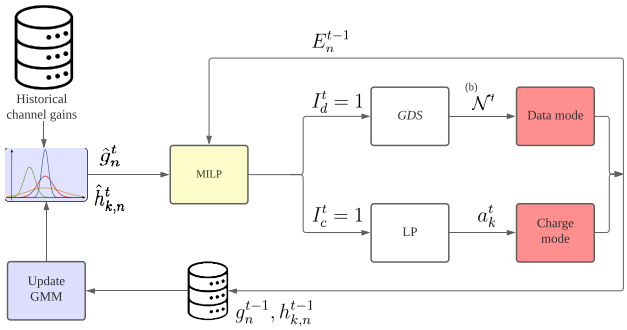

Fig. 3. A block diagram of *RollH*. The HAP first trains the GMM of each device. In each slot, the HAP uses a GMM to generate channel power gain estimates and then solves MILP (19) to obtain the mode of a slot. If $I_c^t = 1$, the system enters *charge* mode. The HAP then determines a_k^t using LP (22a). If $I_d^t = 1$, time slot t is a data slot. The HAP then determines $\mathcal{N}^t$ using *GDS*. After that, the HAP collects g_n^{t-1} and $h_{k,n}^{t-1}$ and uses them to update its GMMs.

Fig. 3 shows the flowchart of *RollH*. The HAP uses a GMM [12] to obtain a channel power gain estimate of each device. This estimate is then used to solve MILP (19). The MILP yields the mode of each slot over a planning horizon with T time slots. The HAP then applies the mode for the current slot. If it is a charging slot, the HAP solves an LP to determine the beam weight of each antenna. If it is a data slot, the HAP decides transmitting devices using the Greedy Device Selection (*GDS*) strategy presented in Section V-A3. At the end of each slot, the HAP collects the channel power gain information of devices and updates the GMM of each device. The same process repeats until the end of the planning horizon.

Next, we introduce GMM. After that, we present the said LP and *GDS* before outlining the main body of *RollH*.

1) Gaussian Mixture Model (GMM): We use GMM [12] to model the distribution of channel power gains between the HAP and a device. Advantageously, it can be trained using historical channel power gains and updated over time when new channel power gains become available. Moreover, the HAP is able to use it to estimate future channel power gains to each device. A GMM is formulated as

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}. \tag{20}$$

It contains a sum of finite Gaussian distributions or components. Each component i is parameterized by its mean μ_i, variance σ_i^2 and weight ω_i [50]. The PDF of a GMM is then represented as

$$p(x|\mu) = \sum_{i=1}^{C} \omega_i g(x|\mu_i, \omega_i). \tag{21}$$

There is a GMM for each antenna of the HAP to each device. Specifically, in the downlink, for device n and antenna k, the PDF of the GMM is represented as $p_{k,n}$. Similarly, in the uplink, for device n, the PDF of the GMM is represented as $p_{0,n}$. In the training process, we use the Expectation-Maximization (EM) algorithm [12] to determine the parameter values of each GMM. We assume the channel power gains between devices and time slots to be independent. This is reasonable because devices are static. Consequently, the channel gains are not a function of time. Instead, they are a function of a device's position and environmental factors, where we model the channel power gains as having a Rayleigh distribution; see Eq. (2). Lastly, we note that our approaches continue to work if GMM is replaced with a recurrent neural network or long short term memory (LSTM). Lastly, we note that training a GMM using EM with C components and G channel scenarios results in a run-time complexity of $\mathcal{O}(CG)$ [51].

2) Transmit Power Control: In a charge slot, the HAP solves an LP to determine the beam weight of each antenna that yields the highest amount of energy delivered to devices. In particular, the HAP solves the following LP:

$$\underset{a_k^t}{\text{maximize}} \quad \sum_{n=1}^{N}\sum_{k=1}^{K} a_k^t h_{k,n}^t \tag{22a}$$

$$\text{subject to} \quad \sum_{k=1}^{K} a_k^t \leq P_{max}. \tag{22b}$$

Constraint (22b) bounds the total transmit power allocated over all antenna elements to be no more than P_{max}.

Proposition 3: Given N devices and an HAP with K antennas, LP (22a) has K decision variables and one constraint.

Proof: Each HAP antenna has a decision variable a_k^t, meaning there are K such decision variables. There is only one transmit power constraint (22b) to ensure that the total transmit power of the HAP is no more than P_{max}. As a result, LP (19) has K decision variables and one constraint. ∎

3) Greedy Device Selection (GDS) Strategy: In each data slot, the HAP uses *GDS* to determine transmitting devices. We define the set of transmitting devices in slot t as $\mathcal{N}^t = \{1, 2, \ldots, |\mathcal{N}^t|\}$. The HAP first checks the SINR of the device that has the lowest channel gain. If the device's SINR γ_n^t is higher than Γ, the HAP will include device n in $\mathcal{N}^t$. The HAP then checks the SINR of the device with the second lowest channel gain. If its SINR γ_m^t is higher than Γ, the HAP will include device m in $\mathcal{N}^t$. The process repeats until all devices are checked. The binary variable I_n^t is then set as

$$I_n^t = \begin{cases} 1, & n \in \mathcal{N}^t, \\ 0, & \text{Otherwise}. \end{cases} \tag{23}$$

Lastly, we note that *GDS* has a run-time complexity of $\mathcal{O}(N \log N)$. This is because it needs to sort the SINR of devices. After that, it iteratively checks each device included in $\mathcal{N}^t$. The maximum size of $\mathcal{N}^t$ is N.

4) Main Body: Algorithm 1 shows the main body of *RollH*. The HAP first initializes all GMMs, see Section V-A1. In slot t, the HAP uses the said GMMs to estimate the channel gain $\hat{g}_n^t$ and $\hat{h}_{k,n}^t$, see line 2. The HAP solves MILP (19) using the estimated channel gain and obtains the mode over $T - t$ time slots. Then it calls *CollectCSI()* to load channel gains for the current slot. After that, the HAP applies the mode for slot t, see line 5-10. The system enters *charge* mode if $I_c^t = 1$. The HAP calls *RFCharge()*, which solves LP (22a), to determine the weight a_k^t of antenna k. Then each antenna broadcasts RF signals with a_k^t. If $I_d^t = 1$, the HAP obtains the transmitting device set $\mathcal{N}^t$ according to *GDS*. Then the HAP calls *ScheduleDevices()* to schedule each device n in $\mathcal{N}^t$ to transmit. After that, as shown in line 11, the HAP updates each GMM using the collected channel gain information.

Algorithm 1 Pseudocode for *RollH*

Input: $\mathcal{T}, \mathcal{N}, \mathcal{K}$

1: **for** each slot $t \in \mathcal{T}$ **do**
2: Estimate $\hat{\mathbf{g}}_n^t = GMM(\mathcal{N})$, $\hat{\mathbf{h}}_{k,n}^t = GMM(\mathcal{N}, \mathcal{K})$
3: $[I_c^t, I_d^t] = \text{MILP}(\hat{\mathbf{g}}_n^t, \hat{\mathbf{h}}_{k,n}^t)$
4: $[\mathbf{g}_n^t, \mathbf{h}_{k,n}^t] = CollectCSI(\mathcal{N})$
5: **if** $I_c^t = 1$ **then**
6: $[a_1^t, a_2^t, \ldots, a_K^t] = LP(\mathbf{h}_{k,n}^t)$
7: $RFCharge([a_1^t, a_2^t, \ldots, a_K^t])$
8: **else**
9: $\mathcal{N}^t = GDS(\mathbf{g}_n^t)$
10: $ScheduleDevices(\mathcal{N}^t)$
11: $UpdateGMM(\mathbf{g}_n^t, \mathbf{h}_{k,n}^t)$
12: **return**

A limitation of *RollH* is that the HAP needs to solve a MILP in each time slot, which is time-consuming. To this end, we propose a data-driven approach in the following subsection.

B. A Data-Driven Rolling Horizon Approach

Our approach, called *n-RollH*, is based on the voice of optimization framework [52]. Briefly, this framework exploits massive data and computing power to *pre-solve* optimization models. The optimal solutions, or its approximations, are then stored in a neural network. When a similar experience or scenario is encountered in an online setting, the corresponding optimal solution is then retrieved from the neural network.

The framework has an *offline* and *online* phase. Fig. 4 shows both phases of *n-RollH*. In the *offline* phase, an IoT network operator first collects as many channel gain values of devices as possible. For example, the operator may conduct a measuring campaign where it requests an HAP to collect channel gain information to devices for an extended period of time. Further, the HAP can use historical channel gains information. Define the set of channel gains to devices for time slot t to T as a *channel scenario*. Let the r-th channel scenario of time slot t be denoted as $\mathbf{C}_r^t$. For each channel scenario, we use it to solve MILP (19). The corresponding solution contains the mode of all time slots from t to T; let this be called a *mode pattern* and denoted as $\mathbf{mt}^t$. We then check whether mode pattern $\mathbf{m}^t$ is new. If so, we assign it a new index and record it. Further, we label the said channel scenario with its corresponding mode pattern. Given a collection of channel scenarios and

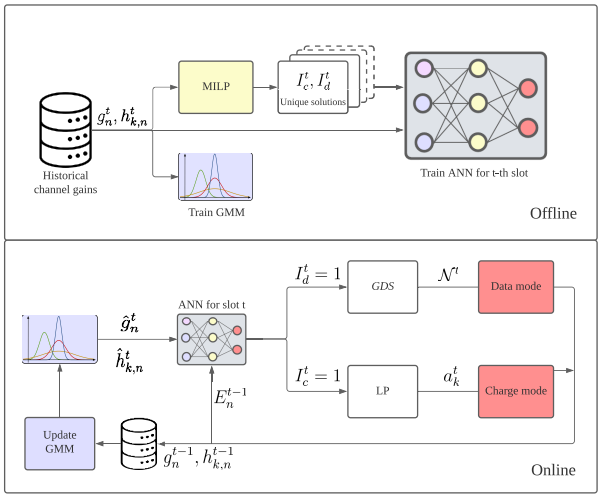

Fig. 4. A block diagram of *n-RollH*. In the *offline* phase, it first generates g_n^t and $h_{k,n}^t$ and uses them to train a GMM for each device. It then solves MILP (19) for I_c^t and I_d^t and trains an ANN for each slot with $g_n^t, h_{k,n}^t, I_c^t$ and I_d^t. In the *online* phase, given the current time slot t, the HAP obtains the actual channel gain to devices, and also uses GMM to estimate the channel gains of future time slots. The HAP then uses these channel gains to retrieve the mode of the current and future time slots from the ANN. If $I_c^t = 1$, then time slot t is used to charge devices. Hence, the HAP determines a_k^t using LP (22a). Otherwise, if $I_d^t = 1$ (data time slot), the HAP determines $\mathcal{N}^t$ using *GDS*. The HAP then collects g_n^{t-1} and $h_{k,n}^{t-1}$ and updates GMMs.

their corresponding index or mode pattern, we then train artificial neural networks (ANNs) to store the mapping between each channel scenario and its mode pattern; see Section V-B1 for more details. Note that we do not store the weight of antennas in ANNs. Instead, in each time slot t, the HAP uses the actual channel gain to devices in order to set the said weight.

After that, in the *online* phase, the HAP obtains the actual channel gain to devices. Further, it uses GMM to generate a channel gain estimate of each device for future time slots. Given these channel gains, the HAP then retrieves the corresponding mode from an ANN. This obviates the need to solve MILP (19), and thereby allowing the HAP to determine the mode of each time slot quickly. If the mode is *charging*, the HAP determines a_k^t using LP (22a). If the mode is *data*, the HAP uses *GDS* to determine transmitting devices. After that, the HAP collects observed channel gains and uses them to update the GMM of each device. We emphasize that there is no ANN training during the *online* phase. Moreover, ANNs are *not* trained on devices.

In the following sections, we provide details of ANNs. After that, we present the main body of *n-RollH*.

1) ANN: In *n-RollH*, there is an ANN in each time slot to store the mode of remaining slots. We denote the ANN of time slot t as $f_{\theta^t}()$, which is parameterized by θ^t. Each ANN has three layers: input layer, hidden layer and output layer. For the input layer, we have one neuron for each g_n^t and each $h_{k,n}^t$ in the planning horizon $\mathcal{T}$. The hidden layer has the same number of neurons as the input layer. As for the output layer, there is one neuron for each unique solution that appears in the training process. Neurons in the input layer are fully connected to neurons in the hidden layer, which are then fully connected to neurons in the output layer.

Algorithm 2 *n-RollH Offline*

Input: $\mathcal{T}$
Output: $\boldsymbol{\theta}$

$\boldsymbol{\theta} = \{\}$
for each slot $t \in \mathcal{T}$ **do**
  $\mathcal{M}^t = \{\}, \mathcal{C}^t = \{\}$
  $\mathcal{Q} = \textit{LoadScenarios}()$
  for each channel scenario $\mathbf{c}_r^t \in \mathcal{Q}$ **do**
    $\mathcal{C}^t = \mathcal{C}^t \cup \mathbf{c}_r^t$
    $\mathbf{m}^t = \textit{MILP}(\mathbf{c}_r^t)$
    if $\mathbf{m}^t \notin \mathcal{M}^t$ **then**
      $\mathcal{M}^t = \mathcal{M}^t \cup \mathbf{m}^t$
  $f_{\theta^t} = \textit{BuildANN}()$
  $\textit{TrainANN}(\theta^t, \mathbf{C}^t, \mathcal{M}^t)$
  $\boldsymbol{\theta} = \boldsymbol{\theta} \cup \theta^t$
return $\boldsymbol{\theta}$

Algorithm 3 *n-RollH Online*

Input: $\mathcal{T}, \mathcal{N}, \mathcal{K}, \boldsymbol{\theta}$

for each slot $t \in \mathcal{T}$ **do**
  $\hat{\mathbf{g}}_n^t = \textit{GMM}(\mathcal{N}), \hat{\mathbf{h}}_{k,n}^t = \textit{GMM}(\mathcal{N}, \mathcal{K})$
  $[I_c^t, I_d^t] = f_{\theta^t}(\hat{\mathbf{g}}_n^t, \hat{\mathbf{h}}_{k,n}^t)$
  $[\mathbf{g}_n^t, \mathbf{h}_{k,n}^t] = \textit{CollectCSI}(\mathcal{N})$
  if $I_c^t = 1$ **then**
    $[a_1^t, a_2^t, \ldots, a_K^t] = \textit{LP}(\mathbf{h}_{k,n}^t)$
    $\textit{RFCharge}([a_1^t, a_2^t, \ldots, a_K^t])$
  else
    $\mathcal{N}^t = \textit{GDS}(\mathbf{g}_n^t)$
    $\textit{ScheduleDevices}(\mathcal{N}^t)$
  $\textit{UpdateGMM}(\mathbf{g}_n^t, \mathbf{h}_{k,n}^t)$
return

We train the ANN for each time slot as follows. The weight θ^t of ANN $f_\theta^t()$ is first initialized randomly. Then we input channel scenarios of time slot t into ANN $f_\theta^t()$, which then produces a mode pattern for time slot t to T. To measure the loss between the mode pattern computed by ANN $f_\theta^t()$ and the actual mode pattern for each scenario, we use the categorical cross-entropy as the loss function. The computed loss is then used to update θ^t. The previous steps are repeated until convergence.

2) Main Body: The main body of *n-RollH* has two phases: *offline* and *online*; see Algorithm 2 and Algorithm 3, respectively. In the *offline* phase, for each time slot t in the planning horizon $\mathcal{T}$, *n-RollH* creates a set $\mathcal{C}$ to store all channel scenarios, and a set $\mathcal{M}$ to store all mode patterns that appear in the training process. It then calls *LoadScenarios()* to obtain the set $\mathcal{Q}$, which contains channel scenarios; each defined as $\mathbf{c}_r = [g_1^t, h_{1,1}^t, \ldots, g_n^T, h_{k,n}^T]$. For each $\mathbf{c}_r$, *n-RollH* first adds $\mathbf{c}_r$ to $\mathcal{C}$. The HAP then solves MILP (19) to obtain a mode pattern that is then stored in the vector $\mathbf{m} = [I_c^t, I_d^t, \ldots, I_c^T, I_d^T]$. If $\mathbf{m}$ is not in $\mathcal{M}$, it is added to $\mathcal{M}$. After that, *n-RollH* calls *BuildANN()* to build an ANN. This ANN is trained by

TABLE III
A SUMMARY OF PARAMETER VALUES

Parameter	Value
Maximum HAP transmit power (P_{max})	0.6 W
Distance of device n to the HAP (d_n)	15 m
Energy to transmit a packet (ϵ)	4.75 μJ
SINR threshold Γ	2 (dB)
Maximum received power (M)	0.024 W
Parameter of Non-Linear Model (a)	150
Parameter of Non-Linear Model (b)	0.014
Noise power	10^{-13} W/Hz
Number of runs	100
Planning horizon (T)	10 slots

calling *TrainANN()* with $\mathcal{C}$ and $\mathcal{M}$. The *online* phase is similar to *RollH*. However, in each time slot, instead of solving MILP (19), the HAP uses an ANN to obtain the mode of each slot.

We note that it is not possible to analyze the run-time complexity of Algorithm 2. This is because it involves training an ANN or minimizing a loss function, which does not have a size. Note that line 3 of Algorithm 3 requires the HAP to retrieve the mode of time slot t from ANN $f_{\theta^t}()$. This involves $\mathcal{O}(((T-t)(1+k)N)^2 + (T-t)(1+k)N|\mathcal{M}|)$ operations. Recall that the ANN has three layers, namely input, hidden and output. Further, ANN $f_{\theta^t}()$ has one neuron for each channel gain g_n^t and $h_{k,n}^t$, meaning it has $(T-t)(1+k)N$ neurons in total in both input and hidden layers. In addition, there are $|\mathcal{M}|$ neurons in its output layer. The most computational expensive step of Algorithm 3 is solving LP (22a); its number of variables and constraints are shown in Proposition-3. Lastly, we have analyzed the run-time complexity of *GDS* in Section V-A3, and *GMM* in Section V-A1.

VI. EVALUATION

Our simulator, written in Python, and experiments are run on an Intel i5 CPU @3.40 GHz with 8 GB of memory. We solve our MILP using Gurobi [48]. Devices are placed at different locations with a distance of 15 m from the HAP [45]. At this distance, they are able to harvest RF-energy using the harvester in [46]. The HAP is equipped with eight antennas [45] and the maximum transmission power P_{max} is 0.6 W. The energy cost of a packet is 475 μJ [53]. The SINR threshold Γ is 2 dB. As for the RF-energy conversion efficiency ψ, its maximum received power M is 0.024 W and its parameter a and b are set to 150 and 0.014 respectively as per [46]. We assume that the circuit noise power is 10^{-13} W/Hz [54]. Each result is an average of 100 runs and each run contains $T = 10$ slots. We train each ANN using 10000 channel scenarios. Except Section VI-D, we assume there are five devices. The aforementioned parameter values are summarized in Table III.

We compare *RollH* and *n-RollH* against four other solutions:

- *Optimal Solution or MILP:* The HAP has non-causal channel gain information, which it uses to solve MILP (19). Hence, it has the optimal mode for each slot, transmit power of each antenna in each charging slot and transmitting devices in each data slot.

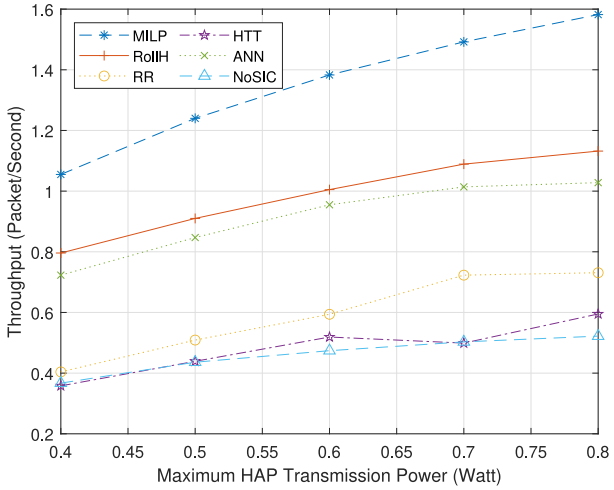

Fig. 5. Throughput versus the maximum HAP transmit power P_{max}.

- *Round Robin (RR):* The HAP selects the mode for each device in a round robin fashion, where the HAP alternates between charging and *data* mode. In data slots, the HAP selects transmitting devices using *GDS*.
- *Alpha Search (AS)* with an adaptive α value [13]*:* The HAP selects the *charge* mode in the first α slots and *data* mode in remaining slots. To determine the value of α, the HAP applies binary search to determine a value that yields the highest throughput. In data slots, it selects one device that has the highest residual energy. Note that AS is different to the conventional harvest-and-transmit method [11], where each slot is used for *both* charging and uplink transmissions. In our case, each slot is either used for charging or data transmission only.
- *RollH* without SIC (*NoSIC*) [13]*:* The HAP uses RollH to determine the mode of each slot. In data slots, it selects one device that has the highest residual energy.

We now introduce the terms *potential transmitters*, *successful transmitters* and *missed transmission opportunities*. Specifically, potential transmitters denote devices with sufficient energy, i.e., $E_n^t \geq \epsilon$; let the set $\mathcal{A} = \{1, 2, \ldots, |\mathcal{A}|\}$ denote potential transmitters. A potential transmitter n is a successful transmitter if $\gamma_n \geq \Gamma$; let $\mathcal{B} = \{1, 2, \ldots, |\mathcal{B}|\}$ denote the set of successful transmitters. In addition, a missed transmission opportunity corresponds to a potential transmitter with $\gamma_n < \Gamma$. The number of missed transmission opportunities in data slot t is denoted as M^t.

In each experiment, we collect the following average results: (i) the system throughput R, (ii) the energy harvested by all devices $\sum_{n=1}^{N} H_n^t$, (iii) the number of data slots, (iv) the number of potential transmitters $|\mathcal{A}|$, (v) the number of successful transmitters $|\mathcal{B}|$, and (vi) the number of missed transmission opportunities M^t.

A. HAP Transmit Power

We vary the HAP's maximum transmit power from 0.4 to 0.8 W. As shown in Fig. 5, the throughput of *MILP* increases by 45% when HAP's maximum transmit power increases from 0.4 to 0.8 W. A higher transmit power allows devices to harvest

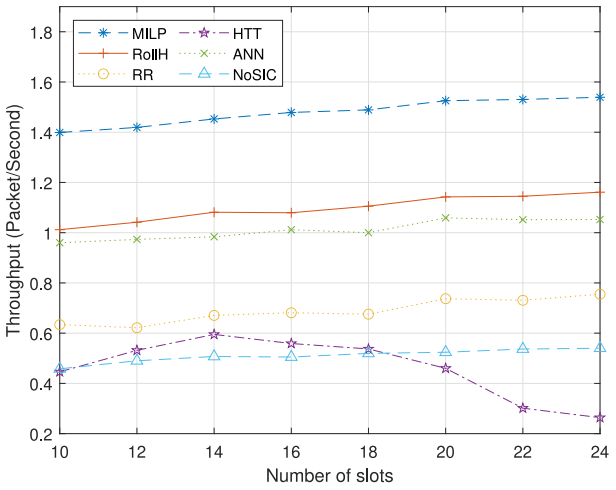

Fig. 6. Throughput versus the number of slots T.

more energy in charging slots. This means there will be more data slots as devices have sufficient energy to transmit their data. In Fig. 5, the throughput of *RollH* and *n-RollH* increased by approximately 3.2 and 2.9 packets per second, respectively, when the HAP's maximum transmit power increased from 0.4 to 0.8 W. This is because devices receive more energy in charging slots. As a result, the number of data slots increases. In addition, the number of potential transmitters in each data slot increases for *RollH* and *n-RollH*. The same reason applies to *AS*. Referring to Fig. 5, the throughput of *AS* increases from 3.5 to 5.9 packet per second because the devices have more energy and the number of potential transmitters in a data slot increases.

Referring to Fig. 5, the throughput of *RollH* is approximately 35.8% lower than *MILP*. This is because *RollH* has only causal channel gain information, where it uses the future channel state information from GMM to solve MILP (19). This information, however, suffers from estimation errors. It is possible for *RollH* to select *data* mode when the channel is suitable for charging mode and vice-versa. In contrast, *MILP* has non-causal channel gain information. As a result, the number of missed transmission opportunities for *MILP* is lower than *RollH*. On the other hand, the throughput of *RollH* is approximately 5.6% higher than *n-RollH*. This is because *RollH* harvests more energy than *n-RollH* given that it has more charging slots. As a result, *RollH* selects more slots to be in *data* mode as compared to *n-RollH*.

In Fig. 5, the throughput of *RollH* is approximately 106.3% higher than *AS*, and around 87% higher than *NoSIC*. This is because for *AS* and *NoSIC*, only a maximum of one device is able to transmit in data slots. In contrast, as for *RollH*, there are on average 1.69 devices in each data slot.

B. Planning Horizon Length

Referring to Fig. 6, the throughput of *MILP* when there are ten time slots increased by around 11.3% when the planning horizon increases from 10 to 24 slots. This is because devices have a higher energy harvesting rate, which allows the HAP to select more slots to be in *data* mode. As per Fig. 6, the throughput of *RollH* rised by 13.1% when the planning horizon length increased from 10 to 24 slots. This is because the longer planning horizon allows *RollH* to select charging mode in slots with a better downlink channel and *data* mode in slots with a better uplink channel. As a result, devices harvest more energy, and thereby allowing them to become potential transmitters. The same reason applies to *n-RollH*; referring to Fig. 6, the throughput of *n-RollH* increased by 9.5% when the planning horizon length increased from 10 to 24 slots. In Fig. 6, the throughput of *AS* first increased by 33.4% when the number of slots increased from 10 to 14, and then decreased by 125.5% when the number of slots increased to 24. This is because the energy harvested by devices is higher when the number of slots increased from 10 to 14. However, when the number of slots exceeds 14, the HAP selects more slots to be in charging mode, which means devices have less chance for data transmissions. As a result, the number of data slots decreases when the number of slots increases from 14 to 24.

From Fig. 6, the throughput of *RollH* is 33.5% lower than *MILP*. This is because *MILP* has the benefit of non-causal channel gain information to decide the mode of each slot where *RollH* only has the channel gain information in current and previous slots. To solve MILP (19), *RollH* obtains channel estimations for future slots, which results in estimation errors. This means it is possible for *RollH* to select *data* mode when the channel is suitable for charging mode, or vice-versa. Thus, the number of potential transmitters for *MILP* is higher than *RollH*. Referring to Fig. 6, the throughput of *RollH* is approximately 8% higher than *n-RollH*. This is because devices in *RollH* harvest more energy than *n-RollH* in each charging slot. As a result, devices in *RollH* accumulate energy faster than *n-RollH*, which allows the HAP to nominate more *data* slots for *RollH*. From Fig. 6, the throughput of *RollH* is around 95.2% higher than *NoSIC*. This is because there is at most one device transmitting in a data slot for *NoSIC*, which is less than *RollH*. Specifically, there are on average 1.66 devices that transmit in each data slot when the number of slots is 14. As shown in Fig. 6, the throughput of *RollH* is 138% higher than *RR*. This is because the number of potential transmitters for *RollH* is higher than *RR*.

C. HAP-Devices Distance

Here, we vary the distance between the HAP and each device from 10 to 14 m. As shown in Fig. 7, the throughput of *MILP* decreases from 20.6 to 15.2 packets per second. This is because the channel gain of devices decreases by around 79.9% if the distance increases from 10 to 14 m according to Eq. (2), which leads to less energy being harvested by devices. From Fig. 7, the throughput of *RollH* and *n-RollH* decreases by 32.1% and 31.6% when the HAP-device distance increased from 10 to 24 m, respectively. This is reasonable due to less energy being harvested by devices. This in turn reduces the number of potential transmitters.

The throughput of *MILP*, as shown in Fig. 7, is around 58.7% higher than *RollH*. This is because *MILP* has non-causal channel gain information, where *RollH* only has channel gain information in current and previous slots and channel

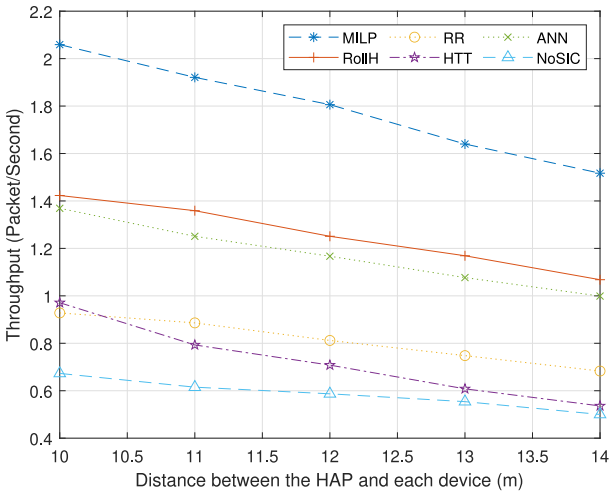

Fig. 7. Throughput versus distance between the HAP and each device.

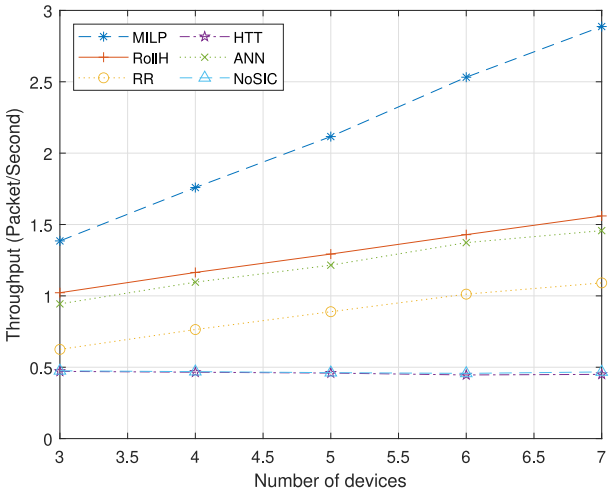

Fig. 8. Throughput versus number of devices.

estimates of future slots. Due to estimation errors, it is possible for *RollH* to select *data* mode when the channel is suitable for charging mode and vice-versa. Accordingly, the number of missed transmission opportunities for *MILP* is lower than *MILP*. In contrast, the throughput of *RollH* is 7.7% higher than *n-RollH*. The reason is because *RollH* selects more slots to be in *data* mode than *n-RollH*. In addition, the number of potential transmitters for *RollH* is higher than *n-RollH*. As shown in Fig. 7, the throughput of *RollH* is 94% higher than *NoSIC*. This is because there is at most one device transmitting in a data slot for *NoSIC*. On the other hand, there are on average 1.75 devices transmitting for *RollH* in each data slot.

D. Number of Devices

Fig. 8 shows the throughput of *MILP* increases from 1.37 to 2.85 packets per second when we increase the number of devices from three to seven. This is because more energy will be harvested in each charging slot if there are more devices. From Fig. 8, when the number of devices increases from three to seven, the throughput of *RollH* and *n-RollH* increases by 55.9% and 53.4%, respectively, when the number of devices

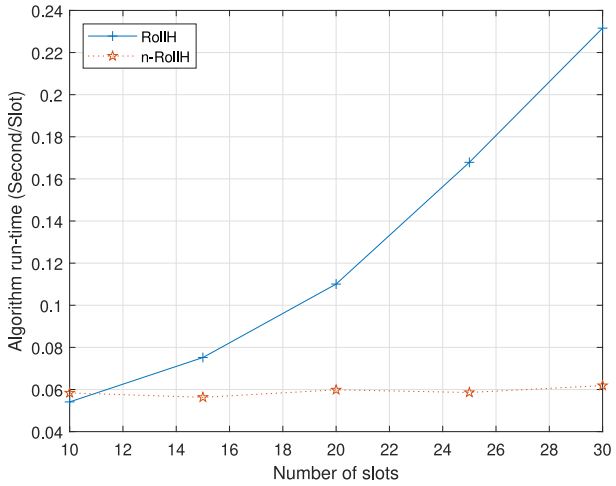

Fig. 9. Number of data slots versus algorithm run-time.

increases from three to seven. The reason is because the energy harvested by devices increases when the number of devices increases from three to seven. As a result, devices harvest sufficient energy more quickly so that the average number of potential transmitters increases for *RollH* and *n-RollH*. From Fig. 8 the throughput of *NoSIC* and *AS* remains stable when the number of devices increases from 3 to 7. This is because for *NoSIC* and *AS*, there is at most one device transmitting in a data slot, which is not affected by the number of devices.

As per Fig. 8, the throughput of *MILP* is 62.1% higher than *RollH*. This is because *RollH* experiences more missed transmission opportunities than *MILP*, which results in fewer transmitting devices in each data slot. In contrast, the throughput of *RollH* is 7.6% higher than *n-RollH*. The reason is because the number of missed transmission opportunities for *n-RollH* is higher than *RollH*. As a result, more devices for *RollH* transmit packets in each data slot. In addition, when the HAP uses *RollH*, it assigns more slots to be *data* mode than *n-RollH*. As shown in Fig. 8, the throughput of *RollH* is 116.9% higher than *AS* when there are three devices. This is because for *AS*, there is at most on device transmitting in a data slot, where as for *RollH*, there are on average 1.74 devices transmitting in each data slot.

E. Run-Time

We vary the number of slots T from 10 to 30 in order to compare the average run-time in each slot of *RollH* and *n-RollH*. Specifically, we record the start and end time of each experiment run using the time module of Python [55], and then calculate the average run-time for each solution. As per Fig. 9, the run-time of *RollH* is around 0.05 seconds per slot, which is approximately the same as *n-RollH* when there are ten time slots. When the number of slots increases to 30, the run-time of *n-RollH* remains around 0.06 seconds per slot, where the run-time of *RollH* increases to 0.23 seconds per slot, which is around 287% higher than *n-RollH*. This is because for *RollH*, the computational complexity to solve MILP Eq. (19) is low if the number of slots is small. When the number of slots increases, the complexity and computational time of the MILP

increases exponentially. As for *n-RollH*, the HAP obtains its decision from a trained neural network in each slot, and it is independent of the number of slots.

VII. Conclusion

This paper considers a WPCN that uses a mode based time structure, where the key problem is to determine whether a time slot is used for charging or data collection. To this end, it outlines a MILP to decide the mode in each slot, the beamforming weight of antennas in charging slots and transmitting devices in data slots. It then presents two rolling horizon approaches, namely *Roll-H* and *n-RollH*, that do not require causal channel gain information. In this respect a key innovation is the use of neural networks to store offline results, which can then be retrieved quickly by a HAP when it encounters similar channel conditions. The results show that the amount of data collected by the HAP is affected by the HAP's charging power, number of slots, distance between the HAP and each device and number of devices. *Roll-H* and *n-Roll-H* respectively achieve up to 106% and 90% higher system throughput as compared to benchmark approaches. A possible future work is to consider distributed solutions. Such solutions will have to consider the RF-energy arrival or charging process in conjunction with the energy and channel state of other devices. Another future work is to extend our formulation to jointly consider different beam patterns and their corresponding collection of transmitting devices in data slots. Specifically, for each beam pattern, there is a corresponding channel gain for each device for each data slot. This means for each beam pattern there will be a collection containing subsets of devices that can transmit simultaneously. The resulting very challenging problem is thus to determine the mode of time slots, and for data slots, select the beam pattern and transmitting devices. Lastly, instead of using a switched beam system at the HAP, an interesting problem is to consider an adaptive antenna array, which involves optimizing antenna weights.

References

[1] M. C. Domingo, "An overview of the Internet of Things for people with disabilities," *J. Netw. Comput. Appl.*, vol. 35, no. 2, pp. 584–596, Mar. 2012.

[2] P. Datta and B. Sharma, "A survey on IoT architectures, protocols, security and smart city based applications," in *Proc. 8th ICCCNT*, New Delhi, India, Dec. 2017, pp. 1–5.

[3] L. M. Borges, F. J. Velez, and A. S. Lebres, "Survey on the characterization and classification of wireless sensor network applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1860–1890, 4th Quart., 2014.

[4] H. Salarian, K.-W. Chin, and F. Naghdy, "Coordination in wireless sensor–actuator networks: A survey," *J. Parallel Distrib. Comput.*, vol. 72, no. 7, pp. 856–867, Jul. 2012.

[5] X. Lu, P. Wang, D. Niyato, and Z. Han, "Resource allocation in wireless networks with RF energy harvesting and transfer," *IEEE Netw.*, vol. 29, no. 6, pp. 68–75, Nov./Dec. 2015.

[6] K. Huang and V. K. N. Lau, "Enabling wireless power transfer in cellular networks: Architecture, modeling and deployment," *IEEE Trans. Wireless Commun.*, vol. 13, no. 2, pp. 902–912, Feb. 2014.

[7] C. Powercast. "Wireless Power Products—Powercastco.com." 2021. [Online]. Available: https://www.powercastco.com

[8] K. W. Choi et al., "Toward realization of long-range wireless-powered sensor networks," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 184–192, Aug. 2019.

[9] L. Dai, B. Wang, Z. Ding, Z. Wang, S. Chen, and L. Hanzo, "A survey of non-orthogonal multiple access for 5G," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2294–2323, 3rd Quart., 2018.

[10] J. G. Andrews, "Interference cancellation for cellular systems: A contemporary overview," *IEEE Wireless Commun.*, vol. 12, no. 2, pp. 19–29, Apr. 2005.

[11] H. Ju and R. Zhang, "Throughput maximization in wireless powered communication networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 1, pp. 418–428, Jan. 2014.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[13] X. Song and K.-W. Chin, "A novel hybrid access point channel access method for wireless-powered IoT networks," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12329–12338, Aug. 2021.

[14] X. Lu, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Wireless networks with RF energy harvesting: A contemporary survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 757–789, 2nd Quart., 2015.

[15] T. Ha, J. Kim, and J.-M. Chung, "HE-MAC: Harvest-then-transmit based modified EDCF MAC protocol for wireless powered sensor networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 3–16, Jan. 2018.

[16] T. Kim, J. Park, J. Kim, J. Noh, and S. Cho, "REACH: An efficient MAC protocol for RF energy harvesting in wireless sensor network," *Wireless Commun. Mobile Comput.*, vol. 2017, no. 11, pp. 1–8, Sep. 2017.

[17] H.-H. Choi, W. Shin, M. Levorato, and H. V. Poor, "Harvest-or-access: Slotted ALOHA for wireless powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11394–11398, Nov. 2019.

[18] A. Iqbal, Y. Kim, and T.-J. Lee, "Learning AP in wireless powered communication networks," *Int. J. Commun. Syst.*, vol. 32, no. 14, Jul. 2019, Art. no. e4027.

[19] G. Gui and B. Lyu, *Non-Orthogonal Multiple Access in Wireless Powered Communication Networks* (SpringerBriefs in Electrical and Computer Engineering). Cham, Switzerland: Springer, 2019, pp. 15–31.

[20] M. A. Abd-Elmagid, A. Biason, T. ElBatt, K. G. Seddik, and M. Zorzi, "Non-orthogonal multiple access schemes in wireless powered communication networks," in *Proc. IEEE ICC*, Paris, France, Jul. 2017, pp. 1–6.

[21] Y. Li and K.-W. Chin, "Random channel access protocols for SIC enabled energy harvesting IoTs networks," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2269–2280, Jun. 2021.

[22] F. R. M. Lima, A. J. d. O. Bastos, and J. d. O. Melo, "Optimal SIC decoding order and WPT time length for WPCN with imperfect SIC," *IEEE Wireless Commun. Lett.*, vol. 10, no. 10, pp. 2170–2174, Oct. 2021.

[23] Z. Ni, Z. Chen, Q. Zhang, and C. Zhou, "Analysis of RF energy harvesting in uplink-NOMA IoT-based network," in *Proc. IEEE VTC*, Honolulu, HI, USA, Nov. 2019, pp. 1–5.

[24] H. Pang, G. Zhang, Q. Wu, and M. Cui, "Throughput maximization for wireless powered non-orthogonal multiple access networks with multiple antennas," in *Proc. 23rd APCC*, Perth, WA, Australia, Mar. 2017, pp. 1–5.

[25] Y. Yuan and Z. Ding, "The application of non-orthogonal multiple access in wireless powered communication networks," in *Proc. IEEE 17th SPAWC*, Edinburgh, U.K., Aug. 2016, pp. 1–5.

[26] C. Qin, W. Ni, H. Tian, and R. P. Liu, "Joint rate maximization of downlink and uplink in multiuser MIMO SWIPT systems," *IEEE Access*, vol. 5, pp. 3750–3762, 2017.

[27] H. Lee, K.-J. Lee, H.-B. Kong, and I. Lee, "Sum-rate maximization for multiuser MIMO wireless powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 11, pp. 9420–9424, Nov. 2016.

[28] D. Song, W. Shin, J. Lee, and H. V. Poor, "Sum-throughput maximization in NOMA-based WPCN: A cluster-specific beamforming approach," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10543–10556, Jul. 2021.

[29] Z. Chen, K. Chi, K. Zheng, Y. Li, and X. Liu, "Common throughput maximization in wireless powered communication networks with non-orthogonal multiple access," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7692–7706, Jul. 2020.

[30] P. D. Diamantoulakis, K. N. Pappi, Z. Ding, and G. K. Karagiannidis, "Optimal design of non-orthogonal multiple access with wireless power transfer," in *Proc. IEEE ICC*, Kuala Lumpur, Malaysia, Jul. 2016, pp. 1–6.

[31] K. Chi, Z. Chen, K. Zheng, Y.-H. Zhu, and J. Liu, "Energy provision minimization in wireless powered communication networks with network throughput demand: TDMA or NOMA?" *IEEE Trans. Commun.*, vol. 67, no. 9, pp. 6401–6414, Sep. 2019.

[32] M. Lei, X. Zhang, T. Zhang, L. Lei, Q. He, and D. Yuan, "Successive interference cancellation for throughput maximization in wireless powered communication networks," in *Proc. IEEE 84th VTC-Fall*, Montreal, QC, Canada, Mar. 2016, pp. 1–6.
[33] Y. Liu, K.-W. Chin, and C. Yang, "Uplinks schedulers for RF-energy harvesting networks with imperfect CSI," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4233–4245, Apr. 2020.
[34] H. Chingoska, Z. Hadzi-Velkov, I. Nikoloska, and N. Zlatanov, "Resource allocation in wireless powered communication networks with non-orthogonal multiple access," *IEEE Wireless Commun. Lett.*, vol. 5, no. 6, pp. 684–687, Dec. 2016.
[35] P. D. Diamantoulakis, K. N. Pappi, Z. Ding, and G. K. Karagiannidis, "Wireless-powered communications with non-orthogonal multiple access," *IEEE Trans. Wireless Commun.*, vol. 15, no. 12, pp. 8422–8436, Dec. 2016.
[36] Y. Liu, X. Chen, L. X. Cai, Q. Chen, and R. Zhang, "Nonorthogonal multiple access for wireless-powered IoT networks," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 112–128, Jan. 2021.
[37] P. D. Diamantoulakis, K. N. Pappi, and G. K. Karagiannidis, "Jointly optimal downlink/uplink design for wireless powered networks," in *Proc. 24th ICT*, Limassol, Cyprus, Aug. 2017, pp. 1–6.
[38] T. A. Zewde and M. C. Gursoy, "NOMA-based energy-efficient wireless powered communications," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 3, pp. 679–692, Sep. 2018.
[39] Z. Na, Y. Liu, J. Wang, M. Guan, and Z. Gao, "Clustered-NOMA based resource allocation in wireless powered communication networks," *Mobile Netw Appl.*, vol. 25, pp. 2412–2420, Jun. 2020.
[40] S. A. Tegos, P. D. Diamantoulakis, A. S. Lioumpas, P. G. Sarigiannidis, and G. K. Karagiannidis, "Slotted ALOHA with NOMA for the next generation IoT," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6289–6301, Oct. 2020.
[41] L. Pei et al., "Energy-efficient D2D communications underlaying NOMA-based networks with energy harvesting," *IEEE Commun. Lett.*, vol. 22, no. 5, pp. 914–917, May 2018.
[42] Q. Zhou, J. Zhang, and Y. Sun, "Energy-efficient transmission for uplink NOMA in wireless-powered D2D communications," in *Proc. IEEE GC Wkshps*, Abu Dhabi, UAE, Feb. 2018, pp. 1–7.
[43] V. Ozduran, "Advanced successive interference cancellation for non-orthogonal multiple access," in *Proc. TELFOR*, Belgrade, Serbia, Jan. 2018, pp. 1–4.
[44] Z. Chen, Y. Liu, S. Khairy, L. X. Cai, Y. Cheng, and R. Zhang, "Optimizing non-orthogonal multiple access in random access networks," in *Proc. IEEE VTC*, Antwerp, Belgium, Jun. 2020, pp. 1–5.
[45] H.-S. Lee and J.-W. Lee, "Contextual learning-based wireless power transfer beam scheduling for IoT devices," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9606–9620, Dec. 2019.
[46] E. Boshkovska, D. W. K. Ng, N. Zlatanov, and R. Schober, "Practical non-linear energy harvesting model and resource allocation for SWIPT systems," *IEEE Commun. Lett.*, vol. 19, no. 12, pp. 2082–2085, Dec. 2015.
[47] J. Vazifehdan, R. V. Prasad, M. Jacobson, and I. Niemegeers, "An analytical energy consumption model for packet transfer over wireless links," *IEEE Commun. Lett.*, vol. 16, no. 1, pp. 30–33, Jan. 2012.
[48] Gurobi Optimization, LLC. "Gurobi Optimizer Reference Manual." 2021. [Online]. Available: https://www.gurobi.com
[49] J. Mattingley, Y. Wang, and S. Boyd, "Receding horizon control," *IEEE Control Syst. Mag.*, vol. 31, no. 3, pp. 52–65, May 2011.
[50] D. A. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometrics*, vol. 741. Boston, MA, USA: Springer, 2009, pp. 659–663.
[51] R. C. Pinto and P. M. Engel, "A fast incremental Gaussian mixture model," *PLoS ONE*, vol. 10, no. 10, pp. 1–12, Oct. 2015.
[52] D. Bertsimas and B. Stellato, "The voice of optimization," *Mach. Learn.*, vol. 110, pp. 249–277, Jul. 2020.
[53] Y. Gao et al., "Low-power ultrawideband wireless telemetry transceiver for medical sensor applications," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 3, pp. 768–772, Mar. 2011.
[54] E. Karipidis, D. Yuan, Q. He, and E. G. Larsson, "Max–min power control in wireless networks with successive interference cancellation," *IEEE Trans. Wireless Commun.*, vol. 14, no. 11, pp. 6269–6282, Nov. 2015.
[55] P. S. Foundation. "Time–Time Access and Conversions." 2022. [Online]. Available: https://docs.python.org/3/library/time.html

Xiaoyu Song received the Bachelor of Engineering degree (Hons.) in telecommunications from Zhengzhou University, China, and University of Wollongong, Australia, in 2019, where he is currently pursuing the Ph.D. degree. His research interests include channel access in WPCNs, NOMA, and 5G networks.

Kwan-Wu Chin received the Bachelor of Science degree (with First Class Hons.) and the Ph.D. degree with commendation from Curtin University, Australia, in 1997 and 2000, respectively. He was a Senior Research Engineer with Motorola from 2000 to 2003. In 2004, he joined the University of Wollongong as a Senior Lecturer. He was promoted to an Associate Professor in 2011. To date, he holds four U.S. patents and has published more than 190 conference and journal articles. His current research areas include medium access control protocols for wireless networks, and resource allocation algorithms/policies for communications networks.