Reinforcement Learning Approach for RF-Powered Cognitive Radio Network with Ambient Backscatter

Nguyen Van Huynh[1], Dinh Thai Hoang[1], Diep N. Nguyen[1], Eryk Dutkiewicz[1], Dusit Niyato[2], and Ping Wang[2]
[1] School of Electrical and Data Engineering, University of Technology Sydney, Australia
[2] School of Computer Science and Engineering, Nanyang Technological University, Singapore

***Abstract*—For an RF-powered cognitive radio network with ambient backscattering capability, while the primary channel is busy, the RF-powered secondary user (RSU) can either backscatter the primary signal to transmit its own data or harvest energy from the primary signal (and store in its battery). The harvested energy then can be used to transmit data when the primary channel becomes idle. To maximize the throughput for the secondary system, it is critical for the RSU to decide when to backscatter and when to harvest energy. This optimal decision has to account for the dynamics of the primary channel, energy storage capability, and data to be sent. To tackle that problem, we propose a Markov decision process (MDP)-based framework to optimize RSU's decisions based on its current states, e.g., energy, data as well as the primary channel state. As the state information may not be readily available at the RSU, we then design a low-complexity online reinforcement learning algorithm that guides the RSU to find the optimal solution without requiring prior- and complete-information from the environment. The extensive simulation results then clearly show that the proposed solution achieves higher throughputs, i.e., up to 50%, than that of conventional methods.**

Keywords- Ambient backscatter, RF energy harvesting, cognitive radios, MDP, reinforcement learning.

I. INTRODUCTION

Radio frequency (RF) powered cognitive radio networks (CRNs) have been seen as an emerging solution to address both the radio spectrum shortage and the energy limitation for low-power secondary systems (e.g., in industrial IoT applications). In an RF-powered CRN, while the primary transmitter, e.g., the base station, broadcasts signals to its receivers, the secondary transmitter (ST) can harvest energy from such signals through RF energy harvesting techniques. The harvested energy is then stored in the battery of the ST and used to transmit its own data to the secondary receiver (SR) when the primary channel becomes idle, i.e., the base station ceases broadcasting. In this way, the secondary system can operate with minimal human intervention and without causing any interference to the primary system. As a result, there are paramount applications of RF-powered CRNs in practice such as low-energy sensor and IoT networks [1]. However, in an RF-powered CRN, the performance of secondary system heavily depends on the activities of the primary channel that controls both energy and radio frequency of STs. In particular, when the primary channel is usually busy, i.e., the base station broadcasts signals most of the time, the ST has very limited opportunities to transmit data, resulting in a low throughput. This problem can be tackled by recent advances in ambient backscattering.

Ambient backscatter communication (ABC) allows wireless devices to communicate by modulating and reflecting the surrounding ambient RF signals [2]. The ABC technology bears close resemblance with radio frequency identification (RFID), but while RFID requires transmissions from a dedicated carrier emitter, ABC can modulate surrounding ambient signals transmitted by existing wireless systems. Hence, ABC systems can share spectrum with exiting systems and achieve better spectral efficiency than that of RFID systems. Furthermore, ABC devices are relatively simple and consume much less power than active transmitters, and thus ABC allows ultra-low-power operation with low cost implementation [3]. As a result, ABC technology has been receiving significant attention recently, and it was listed as one of the 10 breakthrough technologies in 2016 by MIT Technology Review [4]. For RF-powered CRNs that employ ABC, while the primary channel is mostly busy, instead of spending whole time to harvest energy, the ST can use a fraction of time to transmit data by modulating and backscattering the received signals through ABC technique. Thus, ABC enables secondary systems to simultaneously optimize the spectrum usage and energy harvesting to maximize their performance.

There were some research works in the literature studying solutions to integrate ABC into RF-powered CRNs. In [2], the authors introduced a circuit diagram together with a prototype for an ambient backscattering device with RF energy harvesting capability, i.e., ST. This device includes three main components, i.e., an antenna, an energy harvesting circuit, and a controller. The prototype device can achieve information rates of 1 $Kbps$ over the distances of 2.5 feet. The authors in [5] then extended [2] by introducing a novel coding scheme to improve the backscatter transmission rate as well as the communication range. In this technique, each data bit is represented by one symbol, and each symbol in turn is represented by a predefined chip sequence. Through experiments, the authors showed that the backscatter transmission rate and the communication range can be extended up to 1 Mbps and 20 meters, respectively.

Some other solutions were also proposed to improve the performance for secondary systems. In [6], a hybrid backscatter communications for RF-powered CRNs was introduced in order to improve transmission range and rate for the secondary system. Under this model, the ST can flexibly select between an ambient RF source or a dedicated RF source to support its transmissions based on its location, i.e., indoor-zone or outdoor-zone. Then, an energy trade-off problem is formulated to maximize the throughput for the hybrid backscatter communications. In [7], the time trade-off between the harvest-then-transmit and backscatter processes for an RF-powered backscatter CRN was studied. The numerical results demonstrate that the integration of ambient backscatter technique into RF-powered CRNs always achieves the higher transmission rate than that of using either the ambient backscatter commu-

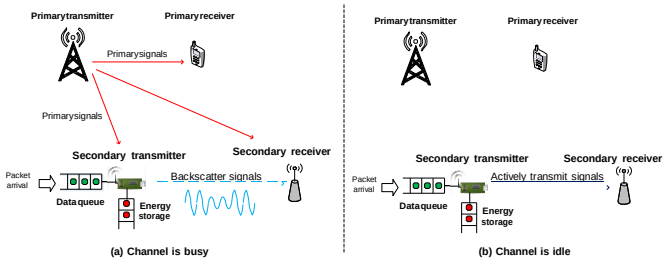


Fig. 1: System model.

nication or the harvest-then-transmit scheme alone.

For RF-powered CRN with ABC, the optimal decision of ST on when to backscatter, when to harvest, when to transmit has to account for the dynamics of primary channel state, energy/battery status, data to be transmitted. Unfortunately, these dynamics are either not readily available at a ST or difficult to be predicted. In this paper, we develop a low-complexity online reinforcement learning algorithm to deal with these dynamics of the environment and aim to maximize the ST's throughput. Specifically, we first formulate the optimal decision problem for the ST as a Markov decision process. We then develop an online learning algorithm which enables the ST to find the optimal policy through "learning" from its interactions with the environment. Through simulation results, we demonstrate that our proposed learning algorithm achieves the best performance compared to existing methods and close to that of the optimal solution achieved when all environment information is known in advance.

II. System Model

Consider a primary system and a secondary system coexisting in an area as shown in Fig. 1. The secondary system consists of a secondary transmitter (ST) which wants to transmit data to its secondary receiver (SR). The ST is equipped with RF energy harvesting and ambient backscatter circuits. While the primary channel is busy, the ABC allows the ST to either harvest energy from the primary signals (to store in its energy storage) or backscatter the signals to transmit data as shown in Fig. 1(a). In contrast, while the channel is idle, i.e., Fig. 1(b), the ST can actively transmit data to its SR by using the energy in its energy storage. Let E and D be the maximum energy storage capacity and maximum data queue size of the ST, respectively. In each time slot, a packet arriving at the data queue with probability a. The probability of the primary channel being idle is denoted by η. When the channel is busy and the ST performs backscattering, i.e., backscatter policy, the ST can transmit d_b data units successfully with probability β. However, if the ST chooses to harvest energy in the busy period, it can harvest e_h units of energy successfully with probability γ. When the channel becomes idle, the ST can use e_t units of energy to successfully transmit d_t data units to its receiver with probability σ. This process is also known as harvest-then-transmit (HTT) mode [9]. Note that our proposed system model can be straightforwardly extended to multiple STs that operate on different primary channels to avoid collision. In the proposed system, two successive working periods of the PT, i.e., idle and busy, are taken into account. As mentioned, the ST can choose to harvest energy or backscatter data in busy periods, and actively transmit data in idle periods. This leads to a trade-off problem among data backscattering, energy harvesting, and data transmitting time to achieve the optimal network throughput. Intuitively, based on its current state, i.e., the data queue state, the energy storage state, and the primary channel state, the ST needs to make a decision to transmit data, harvest energy, backscatter data, or stay idle. However, in practice, the environment parameters, e.g., channel idle probability and successful data transmission probability, may not be available in advance. Therefore, in the following, we introduce an online learning algorithm that can help the ST make the optimal decisions without requiring the complete environment parameters.

III. Problem Formulation

A. MDP Description

We define the state space of the ST as follows:

$$\mathcal{S} = \{(\mathcal{C}, \mathcal{D}, \mathcal{E}); \mathcal{C} \in \{0, 1\}, \mathcal{D} \in \{0, \ldots, d, \ldots, D\}, \mathcal{E} \in \{0, \ldots, e, \ldots, E\}\}, \quad (1)$$

where $c \in \mathcal{C}$ represents the state of the primary channel, i.e., $c = 1$ when the primary channel is busy and $c = 0$ otherwise, $d \in \mathcal{D}$ and $e \in \mathcal{E}$ represent the number of data units in the data queue and the energy units in the energy storage of the ST, respectively. Then, we define the state of the ST as a 3-tuple $s = (c, d, e) \in \mathcal{S}$, where c, d and e are the channel state, the data state, and the energy state, respectively. As mentioned, the ST can choose one of four actions, i.e., harvest energy, transmit data, backscatter data, or stay idle, to perform. Therefore, we define the action space of the ST as follows:

$$\mathcal{A} = \{a : a \in \{1, \ldots, 4\}\}, \quad (2)$$

where

$$a = \begin{cases} 1, & \text{when the ST stays idle,} \\ 2, & \text{when the ST transmits data,} \\ 3, & \text{when the ST harvests energy,} \\ 4, & \text{when the ST backscatters data.} \end{cases} \quad (3)$$

Moreover, when the ST is in state s, its action space is denoted by $\mathcal{A}_s$. Note that $\mathcal{A}_s$ consists of *feasible* actions that do not lead a transition to an unreachable state. Therefore, $\mathcal{A}_s$ can be defined as follows:

$$\mathcal{A}_s = \begin{cases} \{1\}, & \text{if } c = 0 \text{ and } d < d_t \\ & \text{OR } c = 0 \text{ and } e < e_t \\ & \text{OR } c = 1,\ e = E \text{ and } d < d_b, \\ \{1, 2\}, & \text{if } c = 0,\ d \geq d_t \text{ and } e \geq e_t, \\ \{3\}, & \text{if } c = 1,\ d < d_b \text{ and } e < E, \\ \{4\}, & \text{if } c = 1,\ d \geq d_b \text{ and } e = E, \\ \{3, 4\}, & \text{if } c = 1,\ d \geq d_b \text{ and } e < E. \end{cases} \quad (4)$$

The first condition refers to the case when the primary channel is idle and there is not enough data, e.g., no data, or insufficient energy for active transmission. This condition also applies to a special case when the energy storage is full, the primary channel is busy, and the number of data units in the no data for backscattering. Thus, the ST can only select to stay idle, i.e., $a = 1$. The second condition corresponds to the case in which the primary channel is idle and there are data and sufficient energy to perform active transmission. When the primary channel is busy, if there is not enough data, e.g., no

data, for backscattering, and the energy storage is not full, the ST will choose to harvest energy, i.e., the third condition. Otherwise, if there is data to backscatter, the ST can choose to backscatter data or harvest energy if the energy storage is not full, i.e., the fourth and fifth conditions.

When the ST successfully transmits or backscatters data to its receiver, it will receive an immediate reward, i.e., throughput T, denoted as follows:

$$\mathsf{T}(s,a) = \begin{cases} \sigma d_t, & (a=2), \\ \beta d_b, & (a=4), \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

When all environment parameters, e.g., channel idle probability and successful data transmission, are known, we can derive the transition probability matrix for the MDP and use conventional algorithms [8], e.g., value iteration algorithm, to obtain the optimal policy for the ST. However, in practice, some environment parameters may not be available in advice. As a result, we are unable to derive the transition probability matrix for the MDP. In the following, we propose the reinforcement online learning algorithm to resolve this issue. The optimal policy obtained by the MDP using value iteration algorithm will be used as a benchmark to evaluate the performance of the proposed solution.

B. Parameterization for the MDP

We consider a randomized parameterized policy [10] with *softmax* action selection rules [11] to find decisions for the ST. With the randomized parameterized policy, the ST will choose action a at state s with the normalized probability as follows:

$$x_\Theta(s,a) = \frac{\exp(\theta_{s,a})}{\sum_{a' \in \mathsf{A}} \exp(\theta_{s,a'})}, \tag{6}$$

where $\Theta = [\ \cdots\ \theta_{s,a}\ \cdots\ ]^\top$ is the parameter vector of the learning algorithm. By interacting with the environment, the algorithm will update this parameter vector iteratively. Furthermore, $x_\Theta(s,a)$ must not be negative and meets the following constraint:

$$\sum_{a \in \mathsf{A}} x_\Theta(s,a) = 1. \tag{7}$$

The parameterized immediate throughput function of the ST is then as follows:

$$\mathsf{T}_\Theta(s) = \sum_{a \in \mathsf{A}} x_\Theta(s,a)\mathsf{T}(s,a), \tag{8}$$

where $\mathsf{T}(s,a)$ denotes the immediate throughput. Similarly, the parameterized transition probability function can also be derived as follows:

$$P_\Theta(s,s') = \sum_{a \in \mathsf{A}} x_\Theta(s,a) P_{s,s'}(a), \quad \forall s,s' \in \mathsf{S}, \tag{9}$$

where $P_{s,s'}(a)$ is the transition probability from state s to state s' when action a is taken. After that, the average throughput of the ST can be parameterized as follows:

$$\xi(\Theta) = \lim_{t \to \infty} \frac{1}{t} \mathsf{E}_\Theta \left[\sum_{k=0}^{t} \mathsf{T}_\Theta(s_k) \right], \tag{10}$$

where s_k is the state of the ST at time step k. $\mathsf{E}_\Theta[\cdot]$ is the expectation of the throughput. Then, we make following assumptions:

Assumption 1. *There exists a recurrent state s^* which is visited by the online learning algorithm for each of the Markov chain, and this Markov chain needs to be aperiodic.*

Assumption 1 ensures that the considered system has a Markov property. Additionally, we have the following balance equations:

$$\sum_{s \in \mathsf{S}} \pi_\Theta(s) = 1 \text{ and } \sum_{s \in \mathsf{S}} \pi_\Theta(s) P_\Theta(s,s') = \pi_\Theta(s'), \forall s' \in \mathsf{S}, \tag{11}$$

where $\pi_\Theta(s)$ is the steady-state probability of state s under the parameter vector Θ. With (10) and (11), we can express the parameterized average throughput as follows:

$$\xi(\Theta) = \sum_{s \in \mathsf{S}} \pi_\Theta(s)\mathsf{T}_\Theta(s). \tag{12}$$

We aim to maximize $\xi(\Theta)$ given the parameter vector Θ.

C. Policy Gradient Method

We define the differential throughput $d(s,\Theta)$ at state s as follows:

$$d(s,\Theta) = \mathsf{E}_\Theta \left[\sum_{k=0}^{T-1} (\mathsf{T}_\Theta(s_k) - \xi(\Theta)) \,|\, s_0 = s \right], \tag{13}$$

where $T = \min\{k > 0 | s_k = s^*\}$ is the first future time that the online learning algorithm visits the recurrent state s^*. Then, with the differential throughput $d(s,\Theta)$, the gradient of the average throughput $\xi(\Theta)$ can be easily derived as stated in Proposition 1.

Proposition 1. *Under Assumption 1 and Assumption 2, we have*

$$\nabla \xi(\Theta) = \sum_{s \in \mathsf{S}} \pi_\Theta(s) \left(\nabla \mathsf{T}_\Theta(s) + \sum_{s' \in \mathsf{S}} \nabla P_\Theta(s,s') d(s',\Theta) \right). \tag{14}$$

The proof of Proposition 1 can be found in [10]. In addition, we make an assumption as follows:

Assumption 2. *For every state $s, s' \in \mathsf{S}$, the immediate throughput function $\mathsf{T}_\Theta(s)$ and the transition probability function $P_\Theta(s,s')$ satisfy the following conditions: (1) twice differentiable and (2) the first and second derivatives are bounded.*

Assumption 2 ensures that the average throughput is well defined for every Θ and does not depend on the initial state.

D. Idealized Gradient Algorithm

As stated in [12], the idealized gradient algorithm is formulated through Proposition 1 as follows:

$$\Theta_{k+1} = \Theta_k + \rho_k \nabla \xi(\Theta_k), \tag{15}$$

where ρ_k is a step size satisfied Assumption 3.

Assumption 3. *The step size ρ_k is nonnegative, deterministic, and satisfies*

$$\sum_{k=1}^{\infty} \rho_k = \infty, \text{ and } \sum_{k=1}^{\infty} (\rho_k)^2 < \infty. \tag{16}$$

Specifically, the step size has to approach to zero when the time step approaches to infinity. With the policy gradient method, the algorithm will begin with an initial parameter vector $\Theta_0 \in \mathrm{R}^{|S|}$, and the parameter vector Θ will be adjusted at each time step by using (15). With Assumption 2 and Assumption 3, as stated in [12], it is proved that $\lim_{k\to\infty} \nabla \xi(\Theta_k) = 0$, and thus $\xi(\Theta_k)$ converges.

E. Learning Algorithm

By calculating the gradient of the function $\xi(\Theta_k)$ with respect to Θ at each time step k, the average throughput $\xi(\Theta_k)$ can be maximized based on the idealized gradient algorithm. Nevertheless, the gradient of the average throughput $\xi(\Theta_k)$ may not be exactly calculated if the size of the state space S is very large. Therefore, the proposed online learning algorithm adopts an approach that can estimate the gradient $\xi(\Theta_k)$ and update the parameter vector Θ at each time step as follows.

Under the constraint (7), with $\sum_{a\in\mathsf{A}} x_\Theta(s, a) = 1$, we have $\sum_{a\in\mathsf{A}} \nabla x_\Theta(s, a) = 0$. Hence, from (8), $\nabla \mathsf{T}_\Theta(s)$ can be expressed as:

$$\begin{aligned} \nabla \mathsf{T}_\Theta(s) &= \sum_{a\in\mathsf{A}} \nabla x_\Theta(s,a) \mathsf{T}(s,a) \\ &= \sum_{a\in\mathsf{A}} \nabla x_\Theta(s,a) (\mathsf{T}(s,a) - \xi(\Theta)). \end{aligned} \tag{17}$$

In addition, for all $s \in \mathsf{S}$, we have:

$$\begin{aligned} &\sum_{s'\in\mathsf{S}} \nabla P_\Theta(s,s') d(a', \Theta) = \\ &\sum_{s'\in\mathsf{S}} \sum_{a\in\mathsf{A}} \nabla x_\Theta(s,a) P_a(s,s') d(s', \Theta). \end{aligned} \tag{18}$$

Then, under Proposition 1, the gradient of $\xi(\Theta)$ can be expressed as follows:

$$\begin{aligned} \nabla \xi(\Theta) &= \sum_{s\in\mathsf{S}} \pi_\Theta(s) \Big(\nabla \mathsf{T}_\Theta(s) + \sum_{s'\in\mathsf{S}} \nabla P_\Theta(s,s') d(s', \Theta) \Big) \\ &= \sum_{s\in\mathsf{S}} \pi_\Theta(s) \Big(\sum_{s'\in\mathsf{S}} \nabla x_\Theta(s,a) \big(\mathsf{T}(s,a) - \xi(\Theta)\big) \\ &\quad + \sum_{s'\in\mathsf{S}} \sum_{a\in\mathsf{A}} \nabla x_\Theta(s,a) P_a(s,s') d(s', \Theta) \Big) \\ &= \sum_{s\in\mathsf{S}} \sum_{a\in\mathsf{A}} \pi_\Theta(s) \nabla x_\Theta(s,a) q_\Theta(s,a), \end{aligned} \tag{19}$$

where

$$\begin{aligned} q_\Theta(s,a) &= \big(\mathsf{T}(s,a) - \xi(\Theta)\big) + \sum_{s'\in\mathsf{S}} P_a(s,s') d(s', \Theta) \\ &= \mathsf{E}_\Theta \Big[\sum_{k=0}^{T-1} \big(\mathsf{T}(s_k, a_k) - \xi(\Theta)\big) \,|\, s_0 = s, a_0 = a \Big]. \end{aligned} \tag{20}$$

Here $T = \min\{k > 0 | s_k = s^*\}$ is the first future time that the learning algorithm visits the recurrent state s^*. In addition, $q_\Theta(s, a)$ can be expressed as the differential throughput if the ST chooses action a at state s based on policy x_Θ. Then, we introduce Algorithm 1 that updates the parameter vector Θ at each time it visits the recurrent state s^*. In Algorithm 1, the step size ρ_m satisfies Assumption 3 and υ is a positive constant. The gradient of the randomized parameterized policy function in (6) is derived as $\nabla x_{\Theta_m}(s_{k'}, a_{k'})$. Additionally, $F_m(\Theta_m, \hat{\xi}_m)$ is the estimated gradient of the average throughput calculated by the cumulative sum of the total estimated gradient of the average throughput between the m-th and $(m+1)$-th visits of the algorithm to the recurrent state s^*. Through Algorithm 1, the parameter vector Θ and the estimated average throughput $\hat{\xi}$ are adjusted iteratively. Then, the convergence result of Algorithm 1 is derived as in Proposition 2.

Algorithm 1 Algorithm to update parameter vector Θ at each time it visits the recurrent state s^*

1: **Inputs:** υ, ρ_m, and Θ_0.
2: **Initialize:** initiate parameter vector Θ_0 and randomly select a policy for the ST.
3: **for** *k=1 to T* **do**
4: Update current state s
5: **if** $s_k \equiv s^*$ **then**

$$\Theta_{m+1} = \Theta_m + \rho_m F_m(\Theta_m, \hat{\xi}_m), \tag{21}$$

$$\hat{\xi}_{m+1} = \hat{\xi}_m + \upsilon\rho_m \sum_{k'=k_m}^{k_{m+1}-1} \big(\mathsf{T}(s_{k'}, a_{k'}) - \hat{\xi}_m\big), \tag{22}$$

where

$$F_m(\Theta_m, \hat{\xi}_m) = \sum_{k'=k_m}^{k_{m+1}-1} q_{\Theta_m}(s_{k'}, a_{k'}) \frac{\nabla x_{\Theta_m}(s_{k'}, a_{k'})}{x_{\Theta_m}(s_{k'}, a_{k'})}, \tag{23}$$

$$q_{\Theta_m}(s_{k'}, a_{k'}) = \sum_{k=k'}^{k_{m+1}-1} \big(\mathsf{T}(s_k, a_k) - \hat{\xi}_m\big). \tag{24}$$

6: $m = m + 1$
7: **end if**
8: Update ρ_m
9: **end for**
10: **Outputs:** The optimal value of Θ

Proposition 2. *Under Assumption 1-3, let $(\Theta_0, \Theta_1, \ldots, \Theta_\infty)$ be a sequence of the parameter vectors generated by Algorithm 1. Then, $\xi(\Theta_m)$ converges and*

$$\lim_{m\to\infty} \nabla \xi(\Theta_m) = 0, \tag{25}$$

with probability one.

The proof of Proposition 2 can be found in [10] and [12]. Specifically, based on the stochastic approximation method [13], it is proved that $\xi(\Theta)$ and $\hat{\xi}(\Theta)$ converge to a common limit. Then, the process of updating the parameter vector Θ can be expressed as a gradient method with diminishing errors, thereby we can prove that $\nabla \xi(\Theta_m)$ converges to 0, i.e., $\nabla_\Theta \xi(\Theta_\infty) = 0$.

With Algorithm 1, we need to store all values of $\frac{\nabla x_{\Theta_m}(s_k, a_k)}{x_{\Theta_m}(s_k, a_k)}$ and $q_{\Theta_m}(s^k, a^k)$ between the m-th and $(m+1)$-th visits in order to update the values of the parameter vector

Θ. This may lead to a slow processing especially when the size of the state space $\mathcal{S}$ is large. To deal with this shortcoming, the Algorithm 1 is modified to be able to update parameter vectors iteratively with simple calculations. First, we reformulate $F_m(\Theta_m, \xi_m)$ as follows:

$$F_m(\Theta_m, \xi_m) = \sum_{k'=k_m}^{k_{m+1}-1} \sum_{(s_{k'}, a_{k'})} q_{\Theta_m} \frac{\nabla \chi_{\Theta_m}(s_{k'}, a_{k'})}{\chi_{\Theta_m}(s_{k'}, a_{k'})}$$

$$= \sum_{k'=k_m}^{k_{m+1}-1} \frac{\nabla \chi_{\Theta_m}(s_{k'}, a_{k'})}{\chi_{\Theta_m}(s_{k'}, a_{k'})} \sum_{k=k'}^{k_{m+1}-1} \left(\mathcal{T}(s_k, a_k) - \xi_m\right) \quad (26)$$

$$= \sum_{k'=k_m}^{k_{m+1}-1} \left(\mathcal{T}(s_k, a_k) - \xi_m\right) z_{k+1},$$

where

$$z_{k+1} = \begin{cases} \frac{\nabla \chi_{\Theta_m}(s_k, a_k)}{\chi_{\Theta_m}(s_k, a_k)}, & \text{if } k = k_m, \\ z_k + \frac{\nabla \chi_{\Theta_m}(s_k, a_k)}{\chi_{\Theta_m}(s_k, a_k)}, & k = k_m + 1, \ldots, k_{m+1} - 1. \end{cases} \quad (27)$$

Then, the algorithm now can be expressed as in Algorithm 2, where υ is a positive constant and ρ_k is the step size of the algorithm. Instead of calculating the value of $\frac{\nabla \chi_{\Theta_k}(s_k, a_k)}{\chi_{\Theta_k}(s_k, a_k)}$ directly, we can use some mathematical manipulation to transform it into an equivalent form by $1 - x_\Theta(s, a)$. Thus, at each computing step, the ST just needs to perform basic calculations without any complex functions, thereby the online learning algorithm can be efficiently implemented on power-constrained devices.

Algorithm 2 Algorithm to update Θ at every time step

1: **Inputs:** υ, ρ_k, and Θ_0.
2: **Initialize:** initiate parameter vector Θ_0 and randomly select a initial policy for the ST.
3: **for** *k=1 to T* **do**
4: Update current state s_k
5:

$$z_{k+1} = \begin{cases} \frac{\nabla \chi_{\Theta_k}(s_k, a_k)}{\chi_{\Theta_k}(s_k, a_k)}, & \text{if } s_k = s^*, \\ z_k + \frac{\nabla \chi_{\Theta_k}(s_k, a_k)}{\chi_{\Theta_k}(s_k, a_k)}, & \text{otherwise,} \end{cases} \quad (28)$$

$$\Theta_{k+1} = \Theta_k + \rho_k(\mathcal{T}(s_k, a_k) - \xi_k) z_{k+1}, \quad (29)$$

$$\xi_{k+1} = \xi_k + \upsilon \rho_k (\mathcal{T}(s_k, a_k) - \xi_k). \quad (30)$$

6: Update ρ_k
7: **end for**
8: **Outputs:** The optimal value of Θ

IV. Performance Evaluation

A. Experiment Setup

We perform the simulations using MATLAB to evaluate the performance of the proposed solution under different parameter settings. In particular, when the primary channel is busy, we assume that if the secondary transmitter (ST) harvests energy, it can successfully harvest one unit of energy with probability 0.9. Otherwise, if the ST performs backscattering to transmit data, it can successfully transmit one unit of data with probability 0.9. When the channel is idle and if the ST wants to transmit data actively, the ST requires one unit of energy to transmit two units of data. The successful data transmission probability when the channel is idle is also assumed to be 0.9. The maximum data size and the energy storage capacity are set to be 10 units. Unless otherwise stated, the idle channel probability and the packet arrival probability are 0.5. For the learning algorithm, i.e., Algorithm 2, we use the following parameters for the performance evaluation. At the beginning, the ST will start with a randomized policy, i.e., stay idle or transmit data if the primary channel is idle, and harvest energy or backscatter data otherwise. We set the initial value of $\rho = 0.00001$ and it will be updated after every 18,000 iterations as follows: $\rho_{k+1} = 0.9\rho_k$. We also set $\upsilon = 0.01$. To evaluate the proposed solution, we compare its performance with three other schemes, i.e., optimal policy [8], HTT policy [9], and backscatter policy [2]. The optimal policy is obtained through using the value iteration algorithm when all environment information is available in advance. The optimal policy will be used as a benchmark to evaluate the performance of the proposed learning algorithm when the environment information is not available in advance.

B. Numerical Results

1) Convergence of the learning algorithm: We first show the learning process and the convergence of the proposed algorithm. As shown in Fig. 2, the performance of the ST is fluctuated in the first 4,000 iterations as the ST is still learning to adjust the parameter Θ. After that, the learning process begins to stabilize, and then the average throughput co

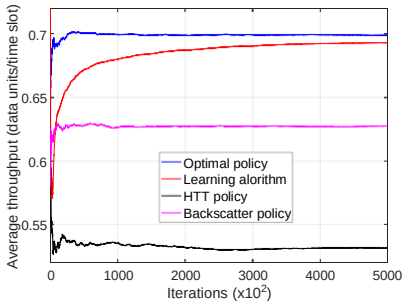
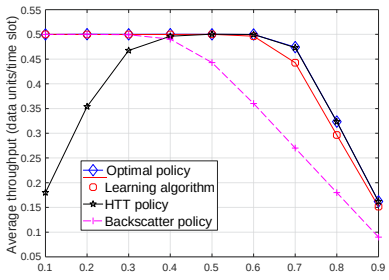
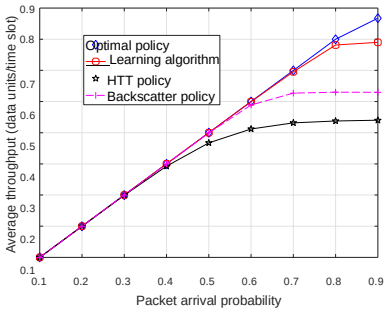


Fig. 2: The convergence of the learning algorithm.

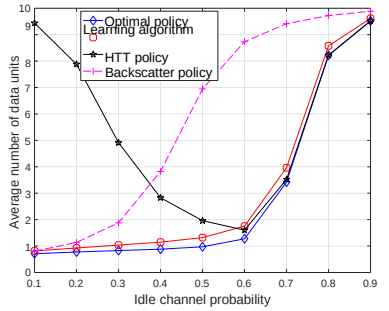
2) Network performance: Next, we perform simulations to evaluate the performance of the proposed solution, i.e., Algorithm 2, and compare with the three other policies, i.e., the optimal, HTT, and backscatter policies, in terms of the average throughput, delay, and blocking probability. In Figs. 3(a) and 3(b), we show the average throughput of the ST obtained by different policies when the idle channel and packet arrival probabilities are varied. Obviously, when the channel idle probability increases, the average throughput of the ST decreases accordingly. However, the learning algorithm always achieves the throughput close to that of the optimal policy. Note that, when the idle channel probability is low, i.e., less than 0.5, the average throughput obtained by HTT policy increases. This is from the fact that the ST has higher opportunities to transmit data as the primary channel is likely to be idle. Nonetheless, when the idle channel probability is

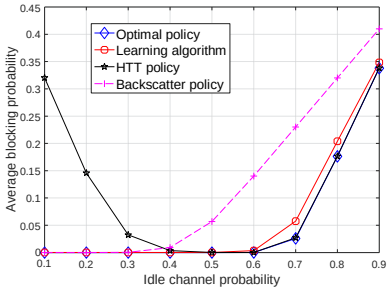




(b) Packet arrival probability is varied

Fig. 3: The average throughput of the ST.





(b)

Fig. 4: (a) The average number of data units in the data queue and (b) blocking probability.

high, i.e., higher than 0.6, the throughput obtained by HTT policy decreases as the ST has little time to harvest energy for data transmission process. Similarly, in Fig. 3(b), the throughputs of all the policies increase when the packet arrival probability increases. When the packet arrival probability is higher than 0.4, the optimal policy achieves the highest throughput followed by the learning algorithm.

We then investigate the blocking probability and delay of all policies as shown in Fig. 4. Clearly, when the idle channel probability increases, the average number of data units in the data queue and the blocking probability also increase. This is due to the fact that the ST has less opportunities to backscatter data and does not have sufficient energy to transmit data to its receiver as the primary channel is likely to be idle. However, the proposed learning algorithm always achieves the performance close to that of the optimal policy.

V. Summary

In this paper, we have considered the RF-powered backscatter cognitive radio network in which the secondary transmitter is equipped with wireless energy harvesting and backscattering capabilities. In this network, the secondary transmitter can harvest energy or backscatter data to its receiver when the channel is busy. To maximize the network throughput, we propose an online learning algorithm that enables the secondary transmitter to adjust its decision to obtain the optimal policy by interacting with the environment. Through numerical results, we have demonstrated that the proposed solution can achieve much higher throughout than the conventional methods and close to that of the optimal policy without requiring the complete information from the environment in advance.

Acknowledgment

This work was supported in part by WASP/NTU M4082187 (4080), Singapore MOE Tier 1 under Grant 2017-T1-002-007 RG122/17, MOE Tier 2 under Grant MOE2014-T2-2-015 ARC4/15, NRF2015-NRF-ISF001-2277, and EMA Energy Resilience under Grant NRF2017EWT-EP003-041.

References

[1] D. Niyato, E. Hossain, D. Kim, V. Bhargava, L. Shafai, *Wireless Powered Communication Networks: Architectures, Protocols, and Applications*, Cambridge University Press, 2016.

[2] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: Wireless communication out of thin air," in *ACM SIGCOMM*, pp. 39-50, Hong Kong, Aug. 2013.

[3] N. V. Huynh, D. T. Hoang, X. Lu, D. Niyato, P. Wang, and D. I. Kim, "Ambient Backscatter Communications: A Contemporary Survey," *IEEE Communications Surveys & Tutorials*, 2018.

[4] 10 Breakthrough Technologies 2016. Available Online: https://www.technologyreview.com/lists/technologies/2016/.

[5] A. N. Parks, A. Liu, S. Gollakota, and J. S. Smith, "Turbocharging ambient backscatter communication," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 619-630. Oct. 2014.

[6] S. H. Kim, and D. I. Kim, "Hybrid Backscatter Communication for Wireless-Powered Heterogeneous Networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 10, Oct. 2017, pp. 6557-6570.

[7] D. T. Hoang, D. Niyato, P. Wang, D. I. Kim, and Z. Han, "Ambient backscatter: A new approach to improve network performance for RF-powered cognitive radio networks," *IEEE Transactions on Communications*, vol. 65, no. 9, pp. 3659-3674, Jun. 2017.

[8] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[9] S. Park, H. Kim, and D. Hong, "Cognitive radio networks with energy harvesting," *IEEE Transactions on Wireless Communications*, vol. 12, no. 3, Mar. 2013, pp. 1386-1397.

[10] P. Marbach, and J. N. Tsitsiklis, "Simulation-based optimization of Markov reward processes," in *IEEE Transactions on Automatic Control*, vol. 46, pp. 191-209, Feb. 2001.

[11] R. S. Sutton, and A. G. Barto, "Reinforcement learning: An introduction," *MIT press*, 1998.

[12] D. P. Bertsekas, "Nonlinear Programming," Athena Scientific, Belmont, MA, 1995.

[13] V. S. Borkar, "Stochastic Approximation: A Dynamic Systems Viewpoint,"*Cambridge University Press*, 2008.