CooperativeQ: Energy-Efficient Channel Access Based on Cooperative Reinforcement Learning

Mehmet Emre*, Gürkan Gür†, Suzan Bayhan‡ and Fatih Alagöz*
*Dept. of Computer Engineering, Bogazici University, Istanbul, Turkey
Email: {mehmet.emre, fatih.alagoz}@boun.edu.tr
†Provus - A MasterCard Company, Istanbul, Turkey. Email: gurkan_gur@mastercard.com
‡Dept. of Computer Science, University of Helsinki, Helsinki, Finland. Email: bayhan@hiit.fi

***Abstract*—Cognitive Radio (CR) with the capability of discovering the unused spectrum promises higher spectrum efficiency – a pressing requirement for 5G networks. However, CR owes this capability to power-hungry tasks, most particularly to spectrum sensing. Given that advances in battery capacity has a slower pace compared to advances in device capabilities and traffic growth, it is paramount to develop energy-efficient CR protocols. To this end, we focus on spectrum sensing and access from an energy efficiency perspective. Our proposal *CooperativeQ* lets each CR decide with an energy efficiency objective on its actions based on its buffer occupancy, buffer capacity, and its observations about the primary channel states. Different than traditional reinforcement learning, CooperativeQ facilitates CRs to share their local knowledge with others periodically. With this information, CR chooses which action to take for the current time slot: (i) idling, (ii) sensing, and (iii) if channel is decided to be idle adapting transmission power to one of the power levels. We evaluate the performance of our proposal under various PU channel types, idling penalty coefficient, and information sharing period. Our results show that CooperativeQ outperforms greedy throughput-maximizing approach or a random channel selection owing to its adaptation and learning capability as well as cooperative mode of operation.**

I. Introduction

The pressing need for more efficient use of the radio spectrum makes Cognitive Radio (CR) often as the most fundamental part of next generation smart networks. A CR not only improves the spectrum efficiency but also adapts to its internal and external operating environment. To handle cognitive operations like spectrum sensing and learning, CRs require powerful battery equipment which is constrained on mobile devices [1]. Moreover, energy expenditure is a significant fraction (20–30 per cent) of total mobile operator costs and a dominant carbon footprint contributor for wireless communications [2]. As for the end user, higher energy efficiency (EE) leads to longer battery lifetime on mobile devices, which increases user satisfaction. Hence, EE is paramount for both network operators and the users.

EE and related aspects in CR networks (CRN) have attracted significant interest recently [3]–[8]. As primary user (PU) detection accuracy and spectrum discovery are main goals of dynamic spectrum access, an energy-efficient scheme must meet the desired objectives while minimizing energy consumption. For example, EE of cooperative sensing improves if data fusion accounts for the sensing outcomes from CRs with higher sensing accuracy and filters the reports of less reliable CRs. According to [3], letting CRs report their sensing results only after some degree of self-confidence achieves better results than traditional voting. Similarly, [6] shows the gain in terms of energy consumption due to ignoring the sensing results in a fuzzy region. Another work in this line is [4] which discusses two thresholds rather than a sharp single threshold for sensing hypothesis test. Additionally, [4] proposes tuning the length of the sensing period according to the PU activity statistics to improve sensing accuracy which is crucial for higher EE.

Adapting transmission power is another solution for improving EE as transmission energy consumption and achievable throughput are functions of the transmission power. [9] formulates the trade-off between power consumption and throughput considering sensing and transmission parameters including transmission power and sensing time.

In this paper, we consider a set of independent and non-collaborating CRs each of which aims to opportunistically transmit through PU channels in a dynamic environment with no a priori information (Section II). Each CR aims to maximize its EE while preventing buffer overflows. The distributed and stochastic nature of the model lends itself to a learning approach. Reinforcement learning has found applications in the CR literature such as [10]. However, current proposals stick to learning by each CR without letting CRs benefit from the *wisdom of crowds.* We propose a cooperative reinforcement learning scheme, namely *CooperativeQ*, to let CRs learn and adapt to the environment they are in, and share their information among themselves (Section III). Different than [10], we consider the buffer constraints and the cost of switching between PU channel frequencies in our model, which may be dominant factor for devices with small form-factors, i.e, mobile phones [11].

Our experiments show that *CooperativeQ* improves EE significantly by adapting to the dynamic network environment (Section IV). Moreover, its performance is highly dependent on the environmental and operational parameters. Therefore, we investigate the effect of these parameters during performance evaluation of our algorithm.

II. System Model

We assume a CRN which consists of a set of non-cooperative CRs seeking for spectrum opportunities in M

primary user channels. At the beginning of each time slot, each CR generates traffic according to a batch Bernoulli process with rate λ. The objective of each CR is to transmit the packets in its buffer with maximum EE while not causing a buffer overflow. As power adaptation is one of the key techniques of energy-efficient operation, each CR can adapt its power to one of K different transmission power levels $P_{tx}(k) = P_k \in \mathcal{P} = \{P_1, P_2, \cdots, P_K\}$. To mitigate the collisions, each CR waits for a randomly chosen time at the beginning of each time slot of T_{slot} time units. Next, each CR decides on an action for this time slot. It can either stay idle or choose to transmit over a channel after sensing that channel and detecting it as idle. A CR can detect a busy PU channel with probability p_d and may incorrectly decide with probability p_f that a channel is occupied although it is not. The cost (both time and energy) of switching from current channel i to channel f in order to transmit is determined by the spectral distance $|f - i|$ between channel frequencies [12].

A. Mobility Model

We consider a mobility model where some of the agents are moving and others are stationary. The agents are positioned to points uniformly picked from a disk with radius r_{init}. The moving agents are making a random walk. Formally, they update their positions according to the following equations: $x_t = x_{t-1} + v\cos(\Theta)$, $y_t = y_{t-1} + v\sin(\Theta)$ and $\Theta \sim \mathcal{U}(\theta; 0, 2\pi)$ where v is speed of the agent in terms of meters per time slot and t is the current time slot. We model stationary agents as a special case where $v = 0$.

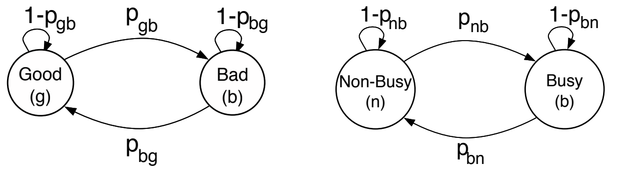
B. Buffer Model

Each CR has a buffer with capacity of storing M_{max} packets to keep the generated traffic until they are transmitted successfully. Let M denote the number of packets in the buffer of a CR. At the beginning of each time slot, $N \sim \mathcal{U}(n; 0, \lambda)$ new packets are generated. Hence, buffer occupancy is updated as: $M \leftarrow M + N$ packets, and $M \leq M_{max}$. Clearly, a CR must decide its actions such that it does not experience buffer overflows which may lead to data loss.

C. Channel Model and PU Traffic

Each transmission channel has two properties: channel quality and channel traffic. For modeling the channel quality, we use 2-state Markov chains similar to Gilbert-Elliott channels where the spectral noise power density changes between *Good* and *Bad* as shown in Fig. 1a. Regarding the PU traffic, we use a Markov-modulated model with two states: *Busy* (high traffic activity) and *Non-Busy* (low traffic activity). At the beginning of each time slot t, for each channel i, probability of having traffic $p_{traffic}(i, t)$ is generated using the hidden Markov chain seen in Fig. 1b.

The quality of each channel i at time t is defined as signal-to-noise-ratio (SNR) denoted by $\gamma_i(t)$ and updated according to the hidden Markov chain at the beginning of each time slot. At the physical layer, we use free-space path loss model for calculating received signal strength [13]. To model heterogeneity between channels, we consider two channel types with different noise power density values and transition probabilities. Type-I channels ($\mathcal{C}_{\text{I}}$) have favorable conditions which result in statistically low packet drop rate whereas Type-II channels ($\mathcal{C}_{\text{II}}$) have lower SNR hence higher packet drop rate compared to Type-I channels. Besides, we assume that there exists a common control channel through which CRs share control messages and data required for cooperation reliably and in a secure manner.



(a) PU channel quality model. (b) PU channel traffic model.

Fig. 1: PU model.

D. CR Actions and Outcomes

Each CR decides to stay idle or senses the spectrum itself and acts based on the outcome of its sensing. Below, we present all possibilities resulting in various throughput and energy consumption.

1) **CR stays idle.** Independent of the PU channel state, the CR decides to stay idle for entire time slot due to internal factors such as low buffer levels or external factors such as bad channel condition or other CR transmissions. The energy consumption in this state can be formulated as follows:

$$E = P_{id}T_{slot}. \tag{1}$$

As no packets are transmitted, throughput $b = 0$ in this case. In the following cases, CR senses the channel and experiences different scenarios due to the channel's quality or the PU traffic state.

2) **Correct detection of spectrum opportunity.** In this case, PU channel is idle and CR discovers this spectrum opportunity correctly. In case this CR does not collide with other CRs, it transmits over the channel f. Assume $N_{success}$ packets been have successfully transmitted out of $N^{'}$ packets that are tried to be sent during this transmission. After transmission, the number of packets in buffer is updated as follows: $M \leftarrow M - N_{success}$. Assume that $P_{tx}(k)$ is transmission power at level k. Furthermore, t_s is the sensing time, $t_{sw}|f - i|$ is the switching time between the selected channel f and channel of CR at beginning time slot i. W is channel bandwidth and $I_f(t)$ is noise of channel f at time t. Given that each packet is L bits in size and channel capacity is $C(f, k, t)$ bps, the resulting throughput in case of transmitting for $t_{tx}(f, k, t)$ time units is:

$$b_{f,k}(t) = t_{tx}(f, k, t)C(f, k, t)\frac{N_{success}}{N^{'}} \text{ bits.} \tag{2}$$

$C(f,k,t)$ is the channel capacity at time slot t:

$$C(f,k,t) = W\log_2\left(1+\frac{P_{tx}(k)}{I_f(t)}\right). \tag{3}$$

Transmission time is defined as:

$$t_{tx}(f,k,t) = \min\left\{\frac{ML}{C(f,k,t)}, T_{slot}-t_s-t_{sw}|f-i|\right\}. \tag{4}$$

Let P_{sw} denote the frequency switching power, and P_{id} is the power consumption of idling. Then, the energy consumption in this case can be formulated as:

$$\begin{aligned} E = \ & |f-i|t_{sw}P_{sw}+t_sP_s+t_{tx}(f,k,t)P_{tx}(k) \\ & +P_{id}(T_{slot}-t_s-t_{sw}|f-i|-t_{tx}(f,k,t)). \end{aligned} \tag{5}$$

3) **All packets are lost in the channel.** This case is a special case of the previous case. All transmitted packets are lost during transmission, i.e. $N_{success} = 0$ and $b_{f,k}(t) = 0$. The energy consumption is same as above and calculated according to (5).
4) **False alarm.** In this case, the sensed PU channel is idle but CR decides to stay idle after falsely detecting an ongoing PU activity. The resulting energy consumption of this state is:

$$E = |f-i|t_{sw}P_{sw}+t_sP_s+P_{id}(T_{slot}-t_s-t_{sw}|f-i|). \tag{6}$$

Moreover, throughput is $b = 0$ in this case since no packets are transmitted.

5) **Correct detection of the PU presence.** In this case, the sensed PU channel is busy and CR detects the ongoing PU transmission. That means, CR keeps idle for the whole time slot which leads to the following energy consumption value:

$$E = |f-i|t_{sw}P_{sw}+t_sP_s+P_{id}(T_{slot}-t_s-t_{sw}|f-i|). \tag{7}$$

Throughput is $b = 0$ in this case since CR keeps idle after sensing.

6) **Misdetection of spectrum opportunity.** This is the worst case in terms of both EE and reliability. In this case, CR cannot detect the ongoing PU transmission and transmits data, then causes PU collision. The energy consumption in this state can be formulated as follows:

$$\begin{aligned} E = \ & |f-i|t_{sw}P_{sw}+t_sP_s+t_{tx}(f,k,t)P_{tx}(k) \\ & + P_{id}(T_{slot}-t_s-t_{sw}|f-i|-t_{tx}(f,k,t)). \end{aligned} \tag{8}$$

All packets drop due to PU collision in this case hence throughput is $b = 0$.

III. Energy-Efficient Channel Access with Reinforcement Learning

A. Problem Formulation

Let $b_{i,k}(t)$ denote the number of bits transmitted by a CR at time t over frequency i with power level k, and $E_{i,k}(t)$ denote the corresponding energy consumption. Let $A_{i,k}(t)$ be a binary random variable denoting the action of a CR at time t. If CR decides to transmit over frequency i with power level k at time t then $A_{i,k}(t) = 1$, and otherwise it is zero. Let t denote the current time slot, we can formulate the EE channel access problem as follows:

$$\max_{A_{i,k}(t)} \frac{\sum_{i=1}^{N_{ch}}\sum_{k=1}^{K} A_{i,k}(t)b_{i,k}(t)}{\sum_{i=1}^{N_{ch}}\sum_{k=1}^{K} A_{i,k}(t)E_{i,k}(t)} \tag{9}$$

$$\text{s.t.} \sum_{i=1}^{N_{ch}}\sum_{k=1}^{K} A_{i,k}(t) \leq 1 \tag{10}$$

$$M(t)+\hat{M}(t+1)-\frac{\sum_{i=1}^{N_{ch}}\sum_{k=1}^{K} A_{i,k}(t)b_{i,k}(t)}{L} \leq M_{max} \tag{11}$$

where N_{ch} is number of PU channels and $\hat{M}(t)$ is expected number of packets generated at time slot t. A CR can choose only one action per time slot, as described by (10). While CR tries to maximize its EE defined as the number of bits transmitted per unit energy consumption in (9), it also needs to maintain its throughput to satisfy the buffer constraint. The buffer constraint described by (11) imposes that transmission at each time t must balance the expected number of packets generated in the next time slot to prevent buffer overflow.

Instead of solving this problem which requires knowledge of some system states, we propose a learning based channel access scheme that is an online algorithm and approximates the above-defined solution.

B. EE Channel Access with Cooperative Q-Learning

In this section, we introduce Cooperative Q-Learning-based algorithm dubbed as *CooperativeQ* which lets a CR - referred to as *agent*– learn while taking actions and making observations in its environment. We first define the states of our system as well as actions and the corresponding rewards.

C. States

We represent the state of a CR as a tuple $s = (l,i)$ where l is CR's buffer occupancy and i is the frequency the CR's antenna is tuned to. For simplicity, we quantize the buffer occupancy to B levels denoted by $\mathcal{B} = \{0,1,2,\cdots,B-1\}$. Given $\mathcal{C} = \{1,2,\cdots,N_{ch}\}$ is the set of channel frequencies, the state space of our system is $\mathcal{B}\times\mathcal{C}$ which consists of BN_{ch} states.

D. Actions

A CR can either stay idle or transmit over a channel f with power P_k. So the set of actions are defined as $\mathcal{A} = \{\text{IDLE}\} \cup (\{\text{TRANSMIT}\} \times \mathcal{C} \times \mathcal{P})$. There are $N_{ch}K+1$ actions a CR can take.

E. Rewards

After observing the outcomes of its actions, each CR gets a reward. The reward function $r_t(s,a) : \mathcal{S}\times\mathcal{A} \rightarrow \mathcal{R}$ defines the desirability of an action a performed on a state s. Reward function takes different values for each possible outcome defined in Section II-D. We calculate $r_t(s,a)$ as follows:

1) **CR stays idle.**

$$r_t(s,a) = -\beta_{idle}\mathcal{R}\frac{T_{slot}}{E} \tag{12}$$

where $\mathcal{R}$ is the bitrate that agent would transmit, T_{slot} is duration of a time slot, β_{idle} is the idling penalty for staying idle, and E is the energy consumption in current time slot as in (1).

2) **Correct detection of spectrum opportunity.** This is the only case that reward function gets a positive value. It is defined as

$$r_t(s,a) = \frac{b}{E} \tag{13}$$

where b is the number of bits transmitted in current time slot as calculated in (2), and E is the energy consumption in current time slot as in (5).

3) **All packets are lost in channel.** This case is actually a special case of Case 2 where $b = 0$. Corresponding reward is:

$$r_t(s,a) = -\beta_{loss}\mathcal{R}\frac{T_{slot}}{E} \tag{14}$$

where β_{loss} is the penalty coefficient in case of packet loss, and E is the energy consumption in current time slot which is defined in (5).

4) **False alarm.**

$$r_t(s,a) = -\mathcal{R}\frac{T_{slot}}{E} \tag{15}$$

where E is the energy consumption in current time slot which is defined in (6).

5) **Correct detection of PU presence.**

$$r_t(s,a) = -\beta_{idle}\mathcal{R}\frac{T_{slot}}{E} \tag{16}$$

where β_{idle} is the penalty for idling, E is the energy consumption in current time slot as in (7).

6) **Misdetection of spectrum opportunity** In this case, the agent gets negative reinforcement to evade further collisions.

$$r_t(s,a) = -\beta_{md}\mathcal{R}\frac{T_{slot}}{E} \tag{17}$$

where β_{md} is the penalty coefficient for misdetection, and E is the energy consumption in current time slot which is defined in (8).

F. Individual Q-Learning Algorithm

Let assume that each CR decides its actions with its local knowledge, i.e., it does not exchange any information with other CRs. We call this scheme as *Individual Q-Learning* in which Q-Learning works by estimating the Q-values $Q(s,a)$. The Q-value $Q(s,a)$ is defined as the expected sum of future rewards obtained by taking action a at state s then *following the optimal policy*. After initializing Q-values to random numbers, it is proven that Q-learning will converge to the optimal policy that maximizes rewards [14]. The main advantage of Q-learning is that, besides being able to converge to optimal policy in case of complete information, it is able to converge to *sufficiently-well* policies under partial knowledge, i.e. modeling system as a Partially Observable Markov Decision Process (POMDP) [15]. This advantage makes Q-learning useful for our problem where CR cannot obtain complete knowledge via observation. *IndividualQ* algorithm makes decisions maximizing EE by using a reward function that resembles EE.

Algorithm 1 *IndividualQ:* Q-learning-based channel access

```
Initialize:
for all $s \in \mathcal{S}, a \in \mathcal{A}$ do
    initialize $Q(s,a)$ with random values
Learning:
loop
    stay idle for a random time between 0 and $2T_{sense}$.
    observe current state $s_t = (l,i) \in \mathcal{S}$ on time slot $t$.
    generate a uniform random number $R \in (0,1)$.
    if $R < \varepsilon$ then
        select action $a_t \in \mathcal{A}$ randomly.
    else
        select action $a_t = \text{argmax}_{a \in \mathcal{A}} Q(s_t, a)$.
    if $a_t = (\text{TRANSMIT}, f, k)$ then
        switch from channel $i$ to channel $f$.
        sense for PU presence.
        if PU is detected then
            stay idle for the rest of the time slot.
        else
            try to transmit $t_{tx}(f,k,t)C(f,k,t)$ bits.
    else                                          ▷ $a_t = \text{IDLE}$
        stay idle for the rest of the time slot.
    get reward $r_t(s_t, a_t)$.
    observe next state $s_{t+1}$.
    $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(t)[r_t(s_t, a_t)$
            $+\gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t)]$.
```

In Algorithm 1, ε is the exploration ratio which determines the probability of exploration of state-space, $\gamma \in [0,1]$ is the discount factor which determines how much the maximum action in the next state affects the Q-value, $r_t(s,a)$ is the reward function defined in Section III-E, and $\alpha(t)$ is the learning rate which controls how much a learning step will impact the Q-value. For this particular algorithm, $\alpha(t)$ is defined as:

$$\alpha(t) = \bar{\alpha} + \frac{1-\bar{\alpha}}{1 + \text{visit}_t(s_t, a_t)} \tag{18}$$

where $\text{visit}_t(s_t, a_t)$ denotes the number of visits made by CR to a state-action pair (s_t, a_t) up to time t, and $\bar{\alpha}$ is the limit value of learning rate.

Regarding complexity, Algorithm 1 runs in O(1) time for each time slot by computing maximums of available Q-values in constant time. Then, it keeps an array of maximum-valued actions for each state. However, Algorithm 1 needs $\text{O}(|B \times \mathcal{C} \times \mathcal{A}|) = \text{O}(BN_{ch}^2|\mathcal{P}|)$ space to store the Q-matrix.

G. Cooperative Q-Learning Algorithm

Generally speaking, cooperation improves the performance of a task although coming with a cost. We propose a cooperative algorithm that uses Expertness Based Cooperative Q-Learning [16]. This algorithm combines Q-values of several agents at fixed time periods called *sharing periods* ($T_{sharing}$) using the following formula

$$Q_i^{new} = \sum_j W_{ij} Q_j^{old} \tag{19}$$

where W_{ij} is measure of agent i's reliance on the knowledge and expertness of agent j. W_{ij} is defined using expertness measures e_i which are sum of some reinforcements. The precise definition of W_{ij} and e_i is problem and algorithm specific. We use a variation of the *learning from experts* (LE) method, and positive expertness measure described by Ahmadabadi *et al.* [16]. In our variation, the experts are chosen among half of the other agents. We W_{ij} as

$$W_{ij} = \begin{cases} 1 - \tau_i, & \text{if } i = j \\ \tau_i \frac{e_j - e_i}{\sum_{k \in \mathcal{E}_i} (e_k - e_i)}, & \text{if } e_j > e_i \wedge j \in \mathcal{E}_i \\ 0, \text{otherwise} \end{cases} \tag{20}$$

$$e_i = \sum_{t=last\_cooperation}^{now} r_i(t) u(r_i(t)) \tag{21}$$

where τ_i is the *impressibility factor* which indicates how much agent i trusts other agents, e_i is expertness measure of agent i, $\mathcal{E}_i$ is experts set of agent i which is a randomly chosen subset of $\{1, 2, \ldots, N_{agent}\} \setminus \{i\}$ with cardinality $\lfloor \frac{N_{agent}}{2} \rfloor$, and $u(x)$ is the unit step function.

Algorithm 2 *CooperativeQ*

```
Initialize:
for all $s \in \mathcal{S}, a \in \mathcal{A}$ do
    initialize $Q(s, a)$ with random values
Learning:
loop
    if in individual learning mode then
        Act according to Algorithm 1
    else                                        ▷ Cooperative Learning
        Choose $\mathcal{E}_i$ among subsets of $\{1, 2, \ldots, N_{agent}\} \setminus \{i\}$
        $Q_i^{new} \leftarrow 0$
        for $j \leftarrow 1, \ldots, n$ do
            $e_j \leftarrow \sum_{t=1}^{now} r_j(t) u(r_j(t))$
        for $j \leftarrow 1, \ldots, n$ do
            $W_{ij} \leftarrow$ ComputeWeights$(e_i, e_j, \mathcal{E}_i)$
            $Q_i^{new} \leftarrow Q_i^{new} + W_{ij} Q_j^{old}$
```

As for complexity, the dominating operation in Algorithm 2 is linear combination of N_{agent} Q-matrices for each agent, which leads to time complexity of $\mathrm{O}(N_{agent}|B \times \mathcal{C} \times \mathcal{A}|) = \mathrm{O}(N_{agent}^2 B N_{ch}^2 |\mathcal{P}|)$.

IV. Performance Evaluation

In this section, we evaluate the performance of our algorithm with a comparison to the following algorithms:

- *OptHighestSNR*: This algorithm chooses a random channel among channels with highest SNR value (lowest noise) and no traffic. Unlike other algorithms, we simulated this algorithm with *perfect spectrum sensing* to act as a benchmark against our algorithm in specific cases. However, like other algorithms, a CR is also susceptible to collision with other CRs.
- *RandomChannel*: This algorithm chooses a random channel and senses traffic with false alarm and detection probabilities identical to our algorithm.

Moreover, we also consider the individual Q-learning algorithm for highlighting the impact of cooperation in the system. Below, we report results of our experiments that are performed in our packet-level simulator. Results are the average of N_{run} = 30 runs. Table I lists the simulation parameters and their values. Although our approach guarantees convergence to a reward-maximizing Q-function in the single agent case [14], that convergence does not hold in a multi-agent environment. More particularly, the problem turns into a stochastic game in which agents must consider others' actions [17].

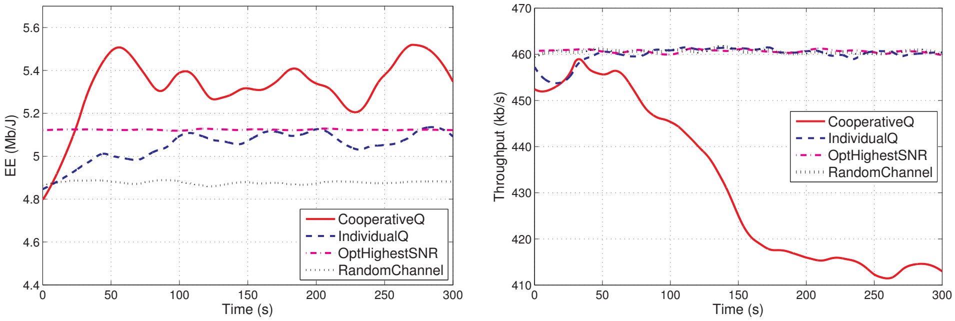
As Fig. 2a shows, in multi-agent environment both Q-learning agents start with the same EE as *RandomChannel*. *IndividualQ* barely reaches the performance of *OptHighestSNR* in this case. However, *CooperativeQ* performs much better than other algorithms despite the cooperation overhead. *CooperativeQ* achieves this by staying idle more than other agents in bad environment conditions hence sacrificing some throughput as seen in Fig. 2b.

A. Effect of Number of Type-I (Better) Channels

$N_{\mathcal{C}_\mathrm{I}}$ value affects EE but that effect diminishes when there are sufficient Type-I channels in the environment. As seen in Fig. 3a, EE performance suffers significantly for $N_{\mathcal{C}_\mathrm{I}} = 1$ while it is similar and favorable for $N_{\mathcal{C}_\mathrm{I}} = 3$ and 5. However, Fig. 3b shows that the change in EE with respect to $N_{\mathcal{C}_\mathrm{I}}$ is caused by the change in throughput. When there are not enough channels with good characteristics, the agents cannot transmit successfully but try the few good channels due to the algorithm's greedy nature. This behavior leads to congestion as shown by the drastic drop in throughput by around 30%.

B. Effect of Idling Penalty Coefficient (β_{idle})

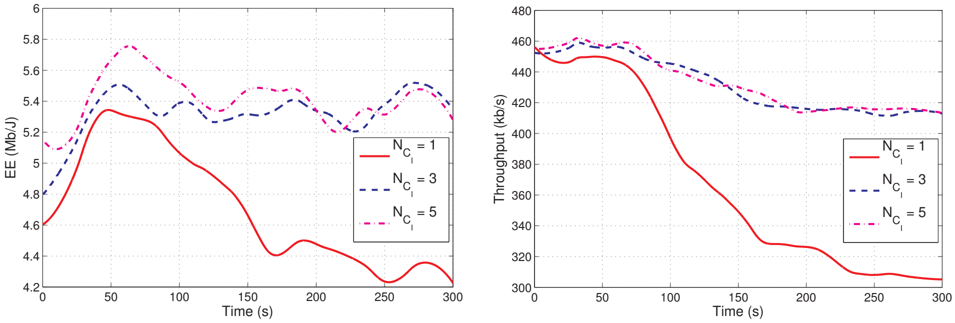
With increasing β_{idle}, transmission becomes more favorable compared to idling. Therefore, for high β_{idle} values CRs would attempt transmission and hence experience channel collision. This aggressive mode then deteriorates the throughput. However, medium values (e.g. $\beta_{idle} = 8$ in the case inspected in Fig. 4) have as good throughput as small values of β_{idle} (Fig. 4b). Moreover, very small β_{idle} values cause long periods of idling followed by transmissions for information chunks. That behavior causes energy wastage due to idling and missing spectrum opportunities.



(a) EE vs. time

(b) Throughput vs. time

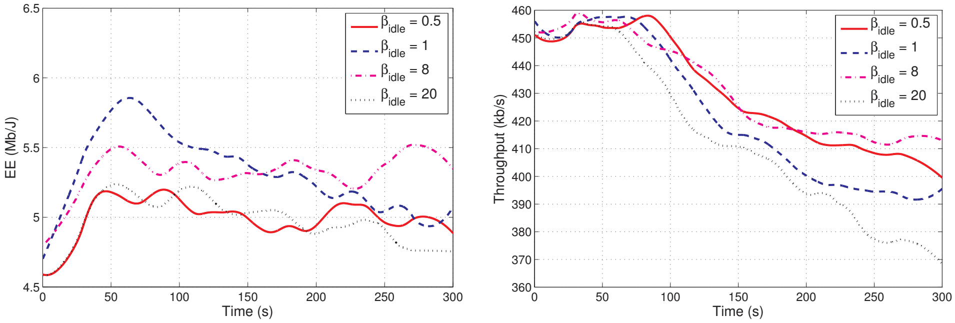
Fig. 2: Comparison of algorithms with respect to EE and throughput (both smoothed using local regression) for multi-agent case. $N_{agent} = 7$, $\beta_{idle} = 8$, $B = 6$, $N_{\mathcal{C}_\mathrm{I}} = 3$, $T_{sharing} = 1000$.



(a) EE vs. time

(b) Throughput vs. time

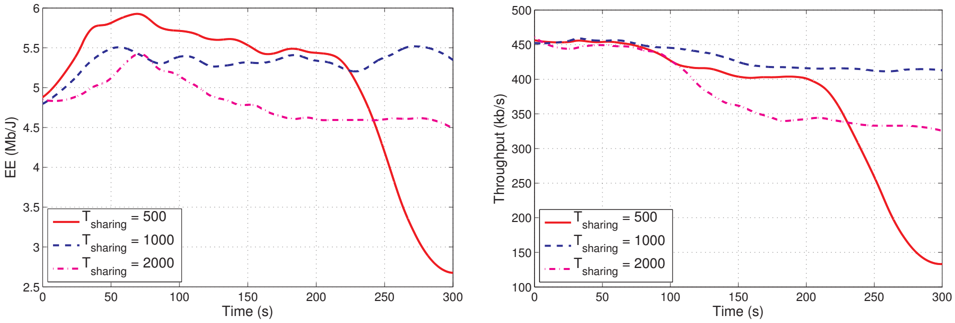
Fig. 3: Effect of $N_{\mathcal{C}_\mathrm{I}}$ on EE and throughput. $N_{agent} = 7$, $\beta_{idle} = 8$, $T_{sharing} = 1000$.



(a) EE vs. time

(b) Throughput vs. time

Fig. 4: Effect of β_{idle} on EE and throughput. $N_{agent} = 7$, $N_{\mathcal{C}_\mathrm{I}} = 3$, $B = 6$, $T_{sharing} = 1000$.



(a) EE vs. time

(b) Throughput vs. time

Fig. 5: Effect of $T_{sharing}$ on EE and throughput. $N_{agent} = 7$, $\beta_{idle} = 8$, $B = 6$, $N_{\mathcal{C}_\mathrm{I}} = 3$.

TABLE I: Simulation Parameters

Parameter	Values
Total simulation time t_{total}	30000 time slots $=$ 300 s
Number of agents N_{agent}	7
Number of stationary agents	$\lfloor \frac{N_{agent}}{2} \rfloor$
Number of channels N_{ch}	5
Number of Type-I channels $N_{\mathcal{C}_{\mathrm{I}}}$	{1, 3, 5}
Radius of simulated area r_{init}	5000 m
Buffer size M_{max}	2560 packets = 320 KiB
Packet size L	1024 bits
Package generation rate per agent	$N \sim \mathcal{U}(n; 0, 8) \frac{\text{packets}}{\text{time slot}}$
Base transmission power P_{tx}	200 mW
Transmission powers $\mathcal{P}$	$\{0.5P_{tx}, P_{tx}, 2P_{tx}, 4P_{tx}\}$
Switching power P_{sw}	$0.5P_{tx}$
Sensing power P_s	$0.5P_{tx}$
Idle power P_s	$0.2P_{tx}$
Time slot duration T_{slot}	10 ms
Sensing time t_s	$0.1T_{slot}$
Switching time between adjacent channels t_{sw}	$0.05T_{slot}$
Frequency of first channel F_0	900 MHz
Channel bandwidth W	1 MHz
Spectral noise densities N_0 of Type-I channels	$N_{0_{good}} = -158.2 \frac{\text{dBmW}}{\text{Hz}}$, $N_{0_{bad}} = -157.2 \frac{\text{dBmW}}{\text{Hz}}$
Spectral noise densities N_0 of Type-II channels	$N_{0_{good}} = -156.7 \frac{\text{dBmW}}{\text{Hz}}$, $N_{0_{bad}} = -148.2 \frac{\text{dBmW}}{\text{Hz}}$
Traffic generation probabilities $p_{traffic}$	Busy channel: 0.7, Non-busy channel: 0.3
State transition probabilities of channels	$p_{gb} = 0.05$, $p_{bg} = 0.4$
Transition probabilities of traffic states	$p_{nb} = 0.3$, $p_{bn} = 0.9$
Bitrate that agents transmit $\mathcal{R}$	3.75 Mbps
Q-Learning Parameters	
Buffer levels B	6
Exploration probability ε	0.03
Discount factor γ	0.2
β_{idle}	{0.5, 1, 8, 20}
β_{md} and β_{loss}	1 and 2
Sharing period $T_{sharing}$	$\{500, 1000, 2000\}$ time slots

C. Effect of Sharing Period

Frequent sharing among CRs, i.e. short sharing period, leads to similarity between Q-matrices of CRs hence CRs start to act similarly. Therefore, this phenomenon leads to very similar channel access patterns for CRs leading to severe congestion and drop in both EE (Fig. 5a) and throughput (Fig. 5b). When $T_{sharing}$ has a relatively large value, the benefits of cooperation become less apparent and adaptation to environment degrades leading to worse performance. As seen in Fig. 5, both EE and throughput have better figures for $T_{sharing} = 1000$ compared to $T_{sharing} = 500$ and $T_{sharing} = 2000$.

V. Conclusion

In this paper, we have modeled the energy-efficient channel access problem using reinforcement learning for a CR. Our proposal *CooperativeQ* aims to maximize energy efficiency while ensuring buffer occupancy is kept below some predetermined level. To this goal, it exploits the knowledge of other CRs via exchange of local information.

Our experiments show that *CooperativeQ* can adapt to the changes in environment such as traffic or channel noise. However, it is highly dependent on the environmental and operational parameters. The algorithm can be further developed by incorporating channel qualities into state space. However, state space grows exponentially with the number of agents which may lead the algorithm become overly-complex. As future work, we are planning to use function approximators to overcome this challenge and extend our model with more realistic settings, e.g., mobility, and interference among CRs. Another research direction is to investigate the implementation details and practical issues of the secure control channel which is used for exchange of Q-values.

References

[1] S. Bayhan and F. Alagöz, “Scheduling in centralized cognitive radio networks for energy efficiency,” *IEEE Trans. Veh. Technol.*, vol. 62, no. 2, pp. 582–595, Feb 2013.

[2] I. Ashraf, F. Boccardi, and L. Ho, “Sleep mode techniques for small cell deployments,” *IEEE Commun. Mag.*, vol. 49, no. 8, pp. 72–79, 2011.

[3] C.-h. Lee and W. Wolf, “Energy efficient techniques for cooperative spectrum sensing in cognitive radios,” in *IEEE Consumer Communications and Networking Conf. (CCNC)*, 2008, pp. 968–972.

[4] H. Su and X. Zhang, “Energy-efficient spectrum sensing for cognitive radio networks,” in *2010 IEEE Int. Conf. on Communications (ICC)*, May 2010, pp. 1–5.

[5] S. Wang, Y. Wang, J. P. Coon, and A. Doufexi, “Energy-efficient spectrum sensing and access for cognitive radio networks,” *IEEE Trans. Veh. Technol.*, vol. 61, no. 2, pp. 906–912, 2012.

[6] S. Maleki, A. Pandharipande, and G. Leus, “Energy-efficient distributed spectrum sensing for cognitive sensor networks,” *IEEE Sensors Journal*, vol. 11, no. 3, pp. 565–573, 2011.

[7] M. C. Oto and O. B. Akan, “Energy-efficient packet size optimization for cognitive radio sensor networks,” *IEEE Trans. Wireless Commun.*, vol. 11, no. 4, pp. 1544–1553, 2012.

[8] G. Gür and F. Alagöz, “Green wireless communications via cognitive dimension: an overview,” *IEEE Network*, vol. 25, no. 2, pp. 50–56, 2011.

[9] D. J. Kadhim, S. Gong, W. Xia, W. Liu, and W. Cheng, “Power efficiency maximization in cognitive radio networks,” in *IEEE Wireless Communications and Networking Conf. (WCNC) 2009*, 2009, pp. 1–6.

[10] Q. Zhao, L. Tong, A. Swami, and Y. Chen, “Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework,” *IEEE J. Sel. Areas Commun.*, vol. 25, no. 3, pp. 589–600, 2007.

[11] A. Nika, Z. Zhang, X. Zhou, B. Y. Zhao, and H. Zheng, “Towards commoditized real-time spectrum monitoring,” in *Proc. 1st ACM workshop on Hot topics in wireless*, 2014, pp. 25–30.

[12] D. Gozupek, S. Buhari, and F. Alagoz, “A spectrum switching delay-aware scheduling algorithm for centralized cognitive radio networks,” *IEEE Trans. Mobile Comput.*, vol. 12, no. 7, pp. 1270–1280, 2013.

[13] T. S. Rappaport, *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall PTR New Jersey, 2002.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[15] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, “Learning policies for partially observable environments: Scaling up,” in *ICML*, vol. 95, 1995, pp. 362–370.

[16] M. Ahmadabadi and M. Asadpour, “Expertness based cooperative Q-learning,” *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 32, no. 1, pp. 66–76, Feb 2002.

[17] M. Wiering and M. van Otterlo, *Reinforcement Learning: State-of-the-Art*, ser. Adaptation, Learning, and Optimization. Springer, 2012.