Deep Multi-User Reinforcement Learning for Distributed Dynamic Spectrum Access

Muhammad A. Jadoon

ITN WindMILL Study Group Meeting

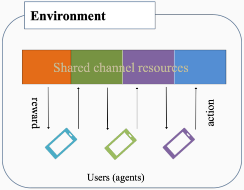
May 13, 2020 (Quarantine day 62)

Agenda

Introduction

Introduction

- ALOHA-based random access
- Each user is an RL agent and learns to access the spectrum independently
- No coordination among users. message passing or carrier sensing. Learning is only based on ACK signal
- Deep Q-network (with LSTM, dueling and doubling)



System Model and Problem Formulation

System Model

- Set $\mathcal{N} = \{1, 2, \ldots, N\}$ of users
- Set $\mathcal{K} = \{1, 2, \ldots, K\}$ of shared orthogonal channels
- At each time slot t, each users transmits its own packet with some probability (ALOHA-like)
- Partially observable environment - each user relies on its own ACK received in the past $t-1$ time slots
- Each user, after transmission, receives ACK as a binary observation:

$$o_n(t) = \begin{cases} 1, & \text{if success} \\ 0, & \text{if collision} \end{cases} \tag{1}$$

Problem Formulation (1/2)

- Distributed setting - no message exchange between users for access
- Partially observable environment
- Let $a_n(t) \in \{0, 1, \ldots, K\}$ be the action of user n at time t. For each user n, actions till time t and the binary observations $o_n(t)$ together define the history:

$$H_n(t) = \{a_n(0), a_n(1), \ldots, a_n(t-1), a_n(t)\} \times \{o_n(0), o_n(1), \ldots, o_n(t-1), o_n(t)\} \tag{2}$$

Problem Formulation (2/2)

Since each user has the goal to find the empty slots independently; the policy π_n for each user n at time t is defined as a mapping from history $H_n(t)$ to a probability mass function over action space $\mathcal{K} = \{0, 1, \ldots, K\}$. We write the policy for user n as

$$\pi_n(t) = \big(p_{n,0}(t), p_{n,1}(t) \ldots, p_{n,K}(t)\big), \tag{3}$$

where

$$p_{n,k}(t) = Pr\big(a_n(t) = k\big) \tag{4}$$

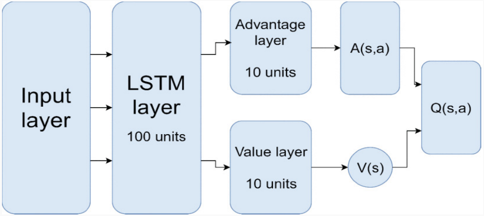
is the probability that user n takes action $a_n(t) = k$ at time t.

Deep Q Network

I am not going in the details of Q learning, rewards, policy and deep Q network assuming you already know about it... if you don't, you should!

Proposed Multiuser RL

Proposed DQN (1/3)



Input $\boldsymbol{x}_n(t)$: a vector of size $2K+2$. First $K+1$ entries are one-hot vectors of action at $t-1$.

If users didn't choose to transmit, the first entry will be 1 and rest K entries equal to zero. $(k+1)$th entry is 1, if user selected channel k. The next K entries are residual channel capacity and the last entry is ACK.

Proposed DQN (2/3)

Input Layer Example:

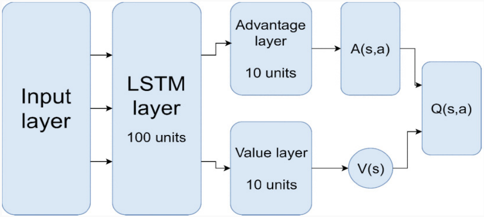
For 2 users and 2 channels case,

if action $a = [2, 2]$, then $x_1(t) = [0, 0, 1, 1, 1, 0]$ and $x_2(t) = [0, 0, 1, 1, 1, 0]$;

and

if action $a = [0, 1]$ then $x_1(t) = [1, 0, 0, 1, 1, 0]$ and $x_2(t) = [0, 1, 0, 0, 1, 1]$

Proposed DQN (3/3)



Value and Advantage layers: Q-value for selecting action $a_n(t)$ at time t is updated by:

$$Q(a_n(t)) \leftarrow V + A(a_n(t)) \tag{5}$$

- Double Q-learning (DQN_1 and DQN_2) is used to decouple the selection of actions from the evaluation of Q-values.

How it works

- At each time slot t, obtain observation $o_n(t)$, and feed the input vector $\boldsymbol{x}_n(t)$ to DQN_1
- Generate Q-values $Q(a)$ for all actions
- Play strategy $\pi_n(t)$ and draw action $a_n(t)$ according to the following distribution:

$$\Pr\big(a_n(t) = a\big) = \frac{(1-\alpha)e^{\beta Q(a)}}{\sum_{a \in \{0,1,\ldots,K\}} e^{\beta Q(a)}} + \frac{\alpha}{K+1} \qquad \forall a \in \{0, 1, \ldots, K\}\,, \tag{6}$$

for small $\alpha > 0$, and temperature β.

Competitive Reward Maximizaton

When each user aims at maximizing its own rate then

$$r_n(t) = \mathbf{1}_n(t-1). \tag{7}$$

Where

$$\mathbf{1}_n(t) = \begin{cases} 1 & \text{if ACK} = 1 \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

for user n at time slot t.

Equilibrium points for this case are efficient when $N \leq K$, and highly inefficient when $N \geq K$.

Cooperative Reward Maximization

When each user aims to maximize a global utility function. Let,

$$r_n(t) = 0, \text{for all} 1 \leq t \leq T-1, \tag{9}$$

and

$$r_n(T) = \sum_{n=1}^{N} f\Big(\sum_{t=1}^{T} \mathbf{1}_n(t-1)\Big) \tag{10}$$

Results

Channel Utilization Results

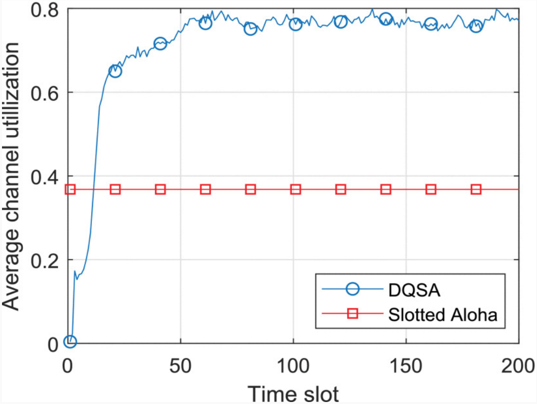


Figure 1: Channel Throughput

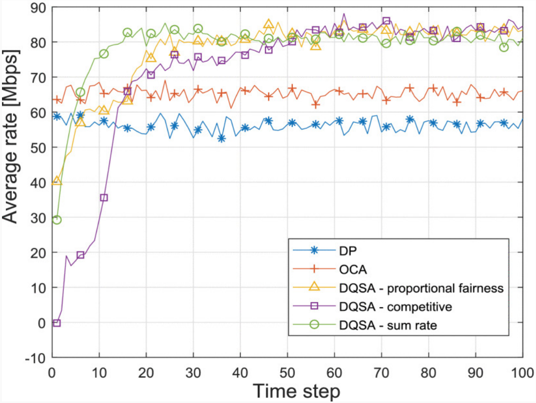


Figure 2: Average user rate for 100 users and 50 shared channels

That's all :)

Thank You!