A Deep Q-Learning Dynamic Spectrum Sharing Experiment

John M. Shea and Tan F. Wong
University of Florida
{jshea, twong}@ece.ufl.edu

***Abstract*—We report results of an experiment in applying deep Q-learning for dynamic spectrum sharing (DSS) in the Alleys of Austin scenario from the DARPA Spectrum Collaboration Challenge. This scenario mimics mobile operations in an urban environment by up to five squads (teams) of soldiers. Each team operates its own wireless network. We consider teamwise–distributed DSS, where there is no central agent to coordinate spectrum usage across teams, but spectrum usage within each team is coordinated by a single member of that team. The spatial distributions of the soldiers creates opportunities for spatial reuse by certain subsets of the teams, and our experiment is set up to evaluate whether the deep Q-learning algorithm can discover and take advantage of these opportunities. The results show that deep Q-learning is able to take advantage of spatial reuse and that doing so results in better performance than a fair-share, disjoint spectrum allocation among the teams.**

I. Introduction

Dynamic spectrum sharing (DSS) offers the potential to improve performance over conventional spectrum allocation schemes because it can adapt the spectrum allocation to time-varying traffic loads and locations of the communicators in a given area. Although there have been many papers that discuss the potential application of machine learning (ML) to cognitive radio systems (see, for example, [1]–[3]), there are few results that report results from experiments using full radio stack implementations with realistic traffic. In this paper, we present results from such an experiment carried out using the Colosseum wireless network emulator. We train a deep-Q network agent to determine the bandwidth a team should use in a five-team wireless networking scenario, where the teams are moving through an emulated mobile environment and experiencing changing traffic loads across three different stages of the scenario.

The experiment in this paper focuses on whether the ML agents that determine the spectrum sharing policy for each team can take advantage of spatial reuse opportunities that exist because of the spatial distribution of squads in the Alleys of Austin (AoA) scenario, which is described more fully in Section II-B. To focus on evaluating the potential for an ML agent to improve performance, we consider a scenario in which each team is using identical physical and link layers. Since each team has the same mix of traffic in AoA, then in the absence of interference, each team needs approximately the same amount of spectrum to deliver their flows. The primary purpose of DSS agents in the experiment is to determine which portions of the spectrum that each team should use, including taking advantage of spatial reuse opportunities in which two teams that are sufficiently separated in space can reuse the same frequency bands.

As reported in our previous work [4], the state space for applying DSS in this scenario is huge compared to that considered in many previous ML works. In [4], we applied domain-specific knowledge to the problem of feature selection and quantization before training a SARSA-based reinforcement learning agent. By contrast, in this paper, we use deep-Q learning to discover effective strategies in the presence of this huge state space. However, we have to limit the deep-Q agent to choosing only the number of channels that a team will use in the next epoch. The particular channels that are used are allocated using an algorithm that tries to avoid other teams' signals, or use spatial reuse if available; more details are in Section III-C. Thus, the deep-Q agent can cause spatial reuse to occur (when the total of the channels allocated by the agents at all the teams exceeds the available spectrum), but it cannot determine which teams or which radios reuse a particular frequency band.

We compare the performance to a system in which each team is allocated a fair share of the total spectrum, equal to the available bandwidth divided by the number of teams. The actual frequencies that are used are still determined using the same distributed process as for the ML agent. We compare the performance for several different configurations of the five teams, where different configurations offer different amounts of interference (and hence spatial reuse opportunities) among the teams. We begin by providing a more detailed discussion of the experimental setup.

II. Experimental Setup Overview

In this section, we give a brief overview of the RF emulation capabilities of Colosseum and the DSS scenario under which the experiment was conducted. We note that the experimental setup described here is a modified version of that we employed in [4]. The current setup provides a more uniform and controlled environment, using which we can better demonstrate and evaluate the efficacy of employing the proposed machine-learning DSS algorithm.

A. Colosseum

Colosseum is a massive RF emulator originally developed by the Defense Advanced Research Projects Agency

This work and our radio design are supported in part by the DARPA SC2, National Science Foundation Grants 1642973 and 1738065, and by AFOSR award number FA9550-19-1-0169. Thanks to the NEU Colosseum Team for access and support.

(DARPA) to support the DARPA Spectrum Collaboration Challenge (SC2). After the completion of the SC2, Colosseum was moved to Northeastern University. It is currently operates under the the National Science Foundation (NSF) Platforms for Advanced Wireless Research (PAWR) program as a national resource for the wireless research community.

The core of Colosseum is a Field Programmable Gate Array (FPGA) matrix, together with supporting RF frontends, that can emulate 256×256 80 MHz RF channel connection over the frequency range from 10 to 6000 MHz. In addition, there are 128 standard radio nodes (SRNs) connected to the RF emulator. Each SRN consists of a USRP X310 software-defined radio (SDR) with 2 transmit and 2 receive chains and a compute server. Most components in the SDR and the compute server are fully user programmable.

In the experiment described in the paper, we employ 50 SRNs, and hence 100×100 RF connections in the channel emulator, over the frequency band from 990 to 1010 MHz.

B. DSS Scenario

The DSS scenario emulated by Colosseum for the experiments in this paper is a member (scenario 10015) of the class of "Alleys of Austin" (AoA) scenarios developed by DARPA for the SC2. This scenario is the only AoA scenario currently supported in Colosseum. Generally, as described by DARPA, the AoA scenarios model a situation in which:
"A platoon from the Texas Army National Guard at Camp Mabry is practicing urban maneuvers and communications in Austin. The platoon is split into five squads consisting of 9 squad members and one UAV (unmanned aerial vehicle). The squads move through the Heritage neighborhood ..."
For the AoA 10015 scenario[1] considered in this paper, the allowed channel bandwidth is 20 MHz[2], and the scenario contains three stages, each of which is 120 s long. The DSS environment in the AoA scenario consists of five teams (networks) of 10 radios each communicating in the allowed 20 MHz frequency band. We note that the AoA scenario can support a maximum of 5 teams of 10 radios each at five different sets of locations. We refer to one of the five sets of team locations as position i, and to the team occupying position i as Team i, for $i = 0, 1, 2, 3, 4$. For example, Fig. 1 shows the trajectories of the five teams over the whole course of the AoA scenario in an experiment run. As previously noted, the inter-team interference experienced by each team varies with the configuration. Each team may determine to use any portion of the allowed 20 MHz band on its own accord with or without regard to spectrum usage decisions of other teams. There is no pre-determined allocation of 20 MHz band among the teams, and all spectrum decisions are made separately by each team: there is no centralized agent to coordinate the sharing of the spectrum.

Each team is to deliver a number of offered traffic flows from prescribed sources to destination radios within the team's network. The offered traffic loads increase at each stage change. In the first stage, the traffic is primarily Voice over IP (VoIP) and Blue Force Tracking (BFT) data, which require low throughput. Stages 2 and 3 add file and video flows, which require much higher throughput. The traffic flows offered to each team in the three stages of the AoA scenario are summarized in Table I.

[1]Hereafter, we will simply refer to this scenario as *the* AoA scenario.
[2]The operating frequency band is from 990 to 1010 MHz.

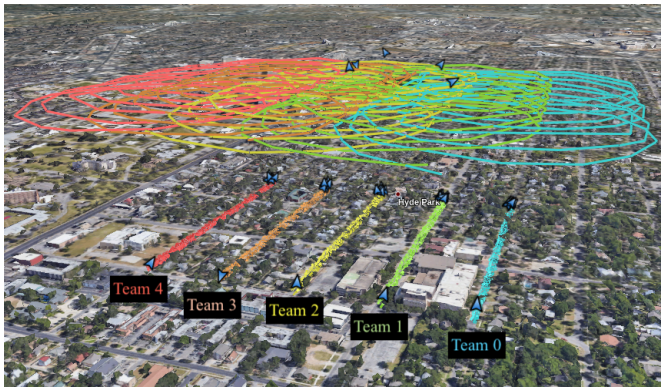

Fig. 1. Trajectories of the five teams of radios in the AoA scenario. The five teams are Teams 0, 1, 2, 3, and 4 occupying Positions 0, 1, 2, 3, and 4, respectively. The nine squad members' radios are close to each other and appear as a cluster of dots along straight lines. The circulating trajectories shows the positions of the UAVs in the teams over time.

Each team receives a score based on the traffic flows that it is able to successfully deliver, as well as the traffic flows delivered by the other teams' networks. For each traffic flow, a quality-of-service (QoS) *mandate* is provided that details the QoS requirements that must be achieved in order to score points for that flow. Two types of flows are offered in the AoA scenario. For a flow with a constant arrival rate, the QoS mandate specifies minimum throughput and maximum latency requirements. For a file burst, the QoS requirement is that 90% of the packets in the file have to be delivered before a specified file transfer deadline. Each flow has an associated number of points that can be achieved in each second in which the QoS is achieved, along with a hold time, which is a number of seconds for which the mandated QoS must be sustained before that flow scores any points. The QoS requirements and the point rewards of different types of flows in the AoA scenario are summarized in Table I.

In addition, each team has a *mandate threshold*, and the actual score that a team achieves in any second is limited to the lowest score among the teams if any team is below its mandate threshold. In our experiment, the mandate threshold is set to be the maximum score that can be achieved at any time. This choice of the mandate threshold ensures a maximum degree of collaboration among the teams in sharing the available 20 MHz spectrum, since each team's actual score is the minimum among the scores achieved by the five teams. Finally, it is worth noting that the traffic load and the latency QoS requirement are that all five teams may simultaneously deliver enough of their respective offered flows to achieve a high score only if proper spectrum sharing decisions are made such that spatial reuse is achieved among the teams. This is particularly true in stages 2 and 3 of the AoA scenario.

TABLE I
SUMMARY OF TRAFFIC FLOWS, WITH THEIR QOS REQUIREMENTS AND POINT REWARDS, OFFERED TO EACH TEAM IN THREE STAGES.

	Flow type	# of flows	Min. throughput (kbps) *[File size (kb)]*	Max. latency (s) *[Transfer deadline (s)]*	Hold time (s)	Point per flow
Stage 1	VoIP	10	36.50	0.37	10	4
	BFT	1	0.26	1.0	10	1
Stage 2	VoIP	10	36.50	0.37	10	4
	BFT	10	0.26	1.0	10	1
	Video	1	918.69	3.0	10	10
	Image	1	*5120*	*10.0*	10	2
	UAV	1	11.23	0.12	10	4
Stage 3	VoIP	10	36.50	0.37	10	4
	BFT	11	0.26	1.0	10	1
	Video	7	401.80	3.0	10	5
	Video	1	918.69	3.0	10	10
	Image	8	*5120*	*10.0*	10	2
	UAV	1	11.23	0.12	10	4

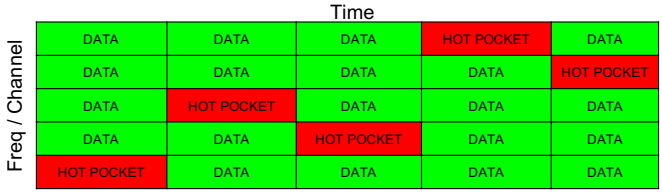

Fig. 2. Time-frequency pocket structure for channel access.

III. RADIO SYSTEM OVERVIEW

Each team in the experiments employed the radio system (physical and link layers) based on our previous design used for the SC2 competition [4]. We provide an overview of the design of the channelization method and spectrum decision process used in our radio system below.

A. Channelization

Channel access by the radios in each team follows a time-frequency structure as shown in Fig. 2. The available 20 MHz spectrum in the AoA scenario is channelized into 20 non-overlapping channels of 1 MHz in bandwidth. A subset of non-overlapping channels is dynamically selected to support the data flows admitted by the team.

The channels are subdivided in time into a repeating schedule of frames, each of which consists of a fixed number of time slots, as illustrated in Fig. 2. In the experiments considered, there are ten 56 ms time slots per frame. A given time-frequency slot is called a *pocket*. Most pockets are used for data transmission from a single source to one or more destinations in the team's network. In addition, a randomized subset of pockets, referred to as *hot pockets*, is used to broadcast network management information and acknowledgments (ACKs). Each hot pocket is divided into minislots, and each radio sends its network management information and ACKs in an assigned minislot.

Transmission in each pocket is packetized into physical-layer (PHY) packets of a fixed duration. The PHY signaling is based on single-carrier frequency-domain (SC/FD) equalization with adaptive modulation and coding that is chosen based on channel conditions and flow QoS requirements. Each radio is capable of simultaneously transmitting and receiving on multiple channels.

B. Network and channel information

Each radio is equipped with a spectrum sensor that can measure the power spectral density (PSD) over the allowed frequency band. The PSD measurements are used to estimate the occupancy percentage of each channel.

Under the AoA scenario, each team does not have any information about the radio implementations and strategies of the other teams, except for information it can gather using its spectrum sensor during the experiment runs. However, the AoA scenario does provide a *collaboration network*[3] over which teams of networks may exchange a limited amount of relevant spectrum sharing information as described below. Each team has one radio that acts as a gateway (GW), and the GW can communicate a limited set of collaboration information to other teams over the collaboration network. The information carried over the collaboration network must adhere to a specified Collaborative Intelligent Radio Network Interaction Language (CIL) and includes:

- locations of the radios, specified as global positioning system (GPS) coordinates,
- frequencies used/are using/plan to use by the team, and
- estimate of a team's score and the mandate threshold.

Spectrum usage and GPS information of other teams' networks obtained from the collaboration network are fused with the PSD measurements to form an interference map at the GW. Through the use of a simple path-loss model, the GW then calculates the interference power seen at each channel of each radio and provides signal-to-interference-and-noise ratio (SINR) estimates for the current and future time. All the information collected by the spectrum sensor and through the collaboration network is fed as input to the DSS decision agent described in the next section.

C. DSS Decision Agent

The DSS decision agent is responsible for determining what channels the team's radio network will use and which flows can be supported using those channels. Based on the

[3]The term *collaboration network* was used by DARPA in the SC2. This spectrum-sharing information exchange network may be implemented in practice using a backbone network, a satellite link, or a cellular network operating in a different frequency band.

channelization method described in Section III-A, that is equivalent to determining which flows are transmitted and in which pockets they will be transmitted. The output produced by the DSS agent is called the *pocket schedule*, which is a list of pockets (time-frequency slots) and the source and destination(s) that will communicate in that slot. Each team's pocket schedule is determined solely based on its own DSS agent, incorporating the scenario information that it measures itself and that it obtains from the other teams via the collaboration network as described in Section III-B.

The goal of the DSS agent is to maximize a team's score over the AoA scenario. However, it is impossible to apply any machine learning algorithm to directly solve this optimization problem in order to obtain the optimal pocket schedule. Take the choices of parameters in the experiment for example. We have 10 time slots per frame and 20 possible channels to use, and hence 200 pockets to assign all the flows in the AoA scenario. From Table I, there are 38 flows in stage 3 of the AoA scenario. Even if we limit the assignment to at most 1 flow per pocket, the action space alone contains 39^{200} possible pocket schedules. That is clearly too large for any practical machine learning implementation. Because of the complexity of this optimization problem, we decompose the problem into the following three steps:

***Channel selection**:* The DSS agent chooses the set of channels C to use by radios in the team by first identifying the number of channels, $|C|$, to use and then choosing the particular set of $|C|$ channels. The DSS agent employs a deep Q-learning algorithm to determine $|C|$ in the first step, taking in the information described in Section III-B as input to the algorithm. The details of the deep Q-learning algorithm will be discussed in Section IV.

After the value of $|C|$ is determined by the agent, the particular set of channels to use is then chosen by iteratively selecting channels from the following sets in order of priority (all channels from the first listed set are added to C before going to the next set, etc.).

1) *Uncontested channels*: channels this team was using in the previous ephoch that no other team is trying to use,
2) *Unoccupied channels*: channels that no team is currently using,
3) *Spatial-reuse channels used in previous epoch*: channels that are in use by one other peer network (at a sufficient separation to support spatial reused[4]) that this team was using in the previous epoch,
4) *Spatial-reuse channels*: channels that are in use by a peer network that are sufficiently separated to allow spatial reuse,
5) *Channels used strictly by higher-scoring peers*: we target the channels of higher-scoring peers to gain additional capacity for our team, while also causing interference to teams that are outperforming us,
6) *Channels used by higher-scoring peers*: these channels may also be used by lower-scoring peers, so there may be some further degradation to their performance, which may hurt us by preventing that team from reaching the scoring threshold, and
7) *Channels used only by lower-scoring peers*: these are a last resort.

Within each set, channels are prioritized based on distance and then occupancy (according to the spectrum sensor).

***Admission control**:* Given the target set of channels C, admission control is performed by estimating the number of pockets or fractions of a pocket needed to support each flow, taking into account the latency and throughput requirements of the flow, as well as the estimated bits/pocket that can be delivered for the source-destination for each flow. The cardinality of C determines the maximum number of pockets available, and an iterative process is used to choose the set of flows that maximizes the number of points that can be scored given $|C|$ and the observed capacities for each source-destination pair.

***Pocket schedule assignment**:* After the set of flows to be supported is determined, a linear program is used to allocate the pockets needed for sources to satisfy the latency requirements of their latency-bound flows to a set of virtual channels, which will be mapped to physical channels in a later step. The linear program determines the number of pockets that each source uses on each channel but does not determine a particular set of pockets that satisfies the specified latency requirements and restrictions on the number of simultaneous transmissions. Thus, an iterative algorithm is used to search for a specific pocket assignment that can be used to satisfy the latency requirements for all sources. The remaining pockets are assigned to satisfy the total throughput requirements from each source, subject to constraints on the number of possible simultaneous transmissions from a radio in a slot. Finally, the virtual channels are mapped to physical channels based on maximizing the worst-case SINR of any of the source-destination pairs assigned to the virtual channel.

[4]For these experiments, a distance of approximately 290 m or more was found to effectively support spatial reuse.

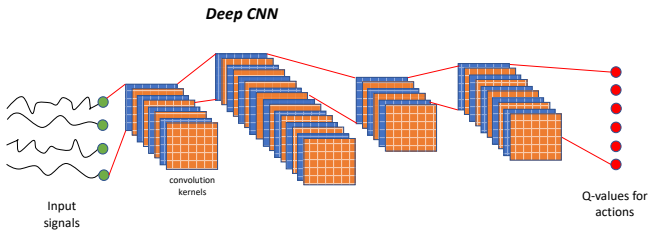

Fig. 3. Convolutional neural network (CNN) with 3 hidden layers used in the deep Q-learning algorithm.

IV. Deep Q-Learning for Channel Size Selection

The DSS agent of each team, operating at the GW, employs a deep Q-learning algorithm to determine the number of channels, $|C|$, that the team's network should use. The deep Q-learning algorithm runs on a convolutional neural network (CNN) with 3 hidden layers (and 4 convolution kernels) as shown in Fig. 3. The action space A consists of the possible number of channels that the team may select, i.e., $A = \{0, 1, \ldots, 20\}$. The input to the CNN is a 28-dimensional

feature vector $\mathbf{s} = (s_0, s_1, \ldots, s_{21}) \in [0,1]^{28}$ that is obtained from the set of network and channel information described in Section III-B as follows:

$$s_0 = (\text{our current score})/r_{\max}$$
$$s_1 = (\text{our current mandate threshold})/r_{\max}$$
$$s_2 = (\text{our current load})/r_{\max}$$
$$s_3 = (\text{our current bandwidth})/b_{\max}$$

where "our" means the team of the DSS agent, $r_{\max}$ = maximum possible score in AoA and $b_{\max} = 20$ MHz is the bandwidth of the allowed spectrum, and for $i = 1, 2, 3, 4$,

$$s_{4+6(i-1)} = (\text{peer } i\text{'s current score})/r_{\max}$$
$$s_{5+6(i-1)} = (\text{peer } i\text{'s current mandate threshold})/r_{\max}$$
$$s_{6+6(i-1)} = (\text{peer } i\text{'s current load})/r_{\max}$$
$$s_{7+6(i-1)} = (\text{peer } i\text{'s current bandwidth})/b_{\max}$$
$$s_{8+6(i-1)} = (\text{overlap bandwidth b/w peer } i \text{ \& us})/b_{\max}$$
$$s_{9+6(i-1)} = (\text{distance b/w centroids of our and peer } i\text{'s network})/d_{\max}$$

where "peer" means another team and $d_{\max} = 600$ m is the maximum possible distance between any two radios from any two teams in the AoA scenario. All bandwidths are measured in MHz and all distances are measured in meters. The maximum possible score that a team can achieve in the current stage is interpreted as the *current load* offered to the team in above. The *current score* of a team is the individual score estimated by the team. Note that the current score may not be the actual score awarded to the team (see Section II-B). The feature vector $\mathbf{s}$ and the action a are updated every second. Also note that the action a determined by the optimal policy, normalized by $b_{\max}$ at the current time is the feature s_3 for the next second. With this model choice, we implicitly assume that the evolution of the DSS system as seen by each team can be approximately modeled by a Markov decision process with $\mathbf{s}$ and a as the input state and action, respectively.

The CNN takes the feature vector $\mathbf{s}$ as input to approximate the "Q-value" $Q(\mathbf{s}, a)$ for each action $a \in A$. The kernel sizes of the first, second, third, and fourth convolution kernels are chosen to be 12, 5, 5, and 5, respectively. The depths of the first, second, and third hidden layers are set to 448, 20, and 320, respectively. The CNN is trained offline with feature vectors and actions collected from experiment runs (jobs) in Colosseum. Let $\mathbf{s}[n]$, $a[n]$, and $r[n]$ be the feature vector, action, and reward at the nth second obtained from a Colosseum job. In order to account for the QoS requirement of 10 s hold time, we define the reward $r[n]$ to be the moving average of the actual score awarded to the team over a 10 s forward window staring from time n. The loss function used in the training process is

$$\sum_{n \in T} \left(r[n+1] + \gamma \max_{a' \in A} Q(\mathbf{s}[n+1], a') - Q(\mathbf{s}[n], a[n]) \right)^2$$

where T is a consecutive block of time (seconds) and γ is the discount factor. We set $|T| = 10$ and $\gamma = 0.95$ in the experiment. Epsilon-greedy policies are employed online in experiment runs to collect training data. However, the optimal policy $\text{argmax}_{a \in A} Q(\mathbf{s}, a)$ resulted from training is employed to obtain the performance results presented in the next section. The optimal policy is generated every second. However, the DSS agency only generates a new pocket schedule every 11 seconds in order to make sure that the pocket schedule does not change too quickly to upset the hold time QoS requirement and to limit the frequency of sending of control information about the updated pocket schedule over the team's network. The policy at the time of the pocket schedule update is taken to select $|C|$.

TABLE II
SUMMARY OF TEAM POSITIONS AND CUMULATIVE SCORES IN ALL EXPERIMENTAL JOBS.

Job #	DSS strategy	Team Positions	Cumulative score
1	Fair sharing	0, 1, 2, 3, 4	11284
2	Fair sharing	0, 1, 2, 3, 4	11428
3	Fair sharing	0, 1, 2, 3, 4	10922
4	Fair sharing	0, 1, 2, 3, 4	11594
5	Fair sharing	0, 1, 2, 3, 4	11335
	Fair sharing	**Average score = 11312.6**	
6	Deep Q-learning	0, 1, 2, 3, 4	16833
7	Deep Q-learning	0, 1, 2, 3 ,4	17046
8	Deep Q-learning	0, 1, 2, 3, 4	17315
9	Deep Q-learning	0, 1, 2, 3, 4	15963
10	Deep Q-learning	0, 1, 2, 3, 4	16599
	Deep Q-learning	**Average score = 16751.2**	
11	Fair sharing	2	26058

V. EXPERIMENTAL RESULTS

In this section, we present some experimental results to evaluate the performance of the DSS solutions produced by the DSS agent employing the deep Q-learning algorithm as described in Sections III-C and IV. .

For comparison, we also consider a fair-sharing solution which assigns $|C| = 4$ channels to each team throughout the whole AoA scenario in the first part of the channel selection step in lieu of using the deep-Q algorithm. The rest of the steps in Section III-C are then implemented by the fair-sharing agent to obtain the pocket schedule. We ran five jobs with 5 teams using their respective fair-sharing DSS agents to share the allowed 20 MHz spectrum in the AoA scenario. The cumulative actual scores achieved[5] over the whole AoA scenario are summarized in Table II (Jobs 1–5). The average score of the five fair-sharing jobs is 11312.6. The spectrum sharing plan distributively determined by the five teams in Job 4 is shown in Fig. 4. We can see from the figure that each team's DSS agent was able to grab 4 channels using the approach described in Section III-C based on the DSS information shared via the collaboration network.

To exclude the possibility that any limited performance in score is caused by inefficiency of our radio design, we also ran a job with a single team in postion 2. In this case, the

[5]Since the mandate threshold is chosen to the maximum possible score, the actual score for each teams is the minimum of individual scores achieved among the five teams at any time.

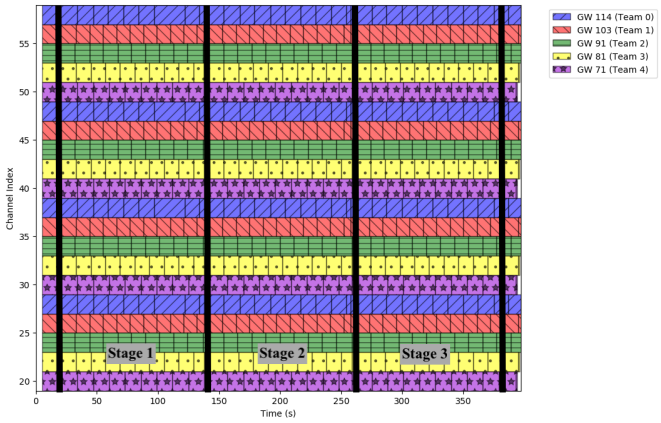

Fig. 4. Fair-share spectrum sharing plan distributively generated by the teams in Job 4.

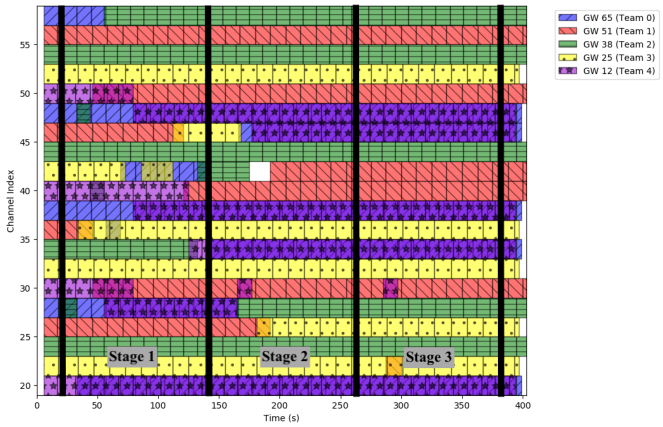

Fig. 5. Deep Q-network spectrum sharing plan distributively generated by the teams in Job 6.

DSS agent fixed $|C| = 20$ channels (i.e., the team may use the whole 20 MHz band) throughout the whole AoA scenario in the first part of the channel selection step in lieu of using the deep-Q algorithm. The result is shown in the last row (Job 11) of Table II. The score achieved is 26058, which is very close to the maximum possible score (26760) that can be achieved in the AoA scenario. The small difference is due to dropping of flows during the transition from one stage to the next. As a result, we can conclude that the decrease in score performance in Jobs 1–5 is due to the reduction in bandwidth that each team can use.

We ran eight jobs employing epsilon-greedy policies to generate data for training the deep-Q CNN described in Section IV. The optimal policy of the CNN model obtained from the training process was then employed to run five additional jobs. The cumulative scores for these five jobs using the deep-Q CNN to select $|C|$ are summarized in Jobs 6-10 in Table II. We see from the table that the DSS agents using the deep-Q channel selection algorithm outperformed the fair-sharing agents. The average cumulative score over Jobs 6–10 using the deep-Q agent is 16751.2, which is 48% higher than the average score obtained using the fair-sharing agent.

The deep-Q agent achieves the scoring advantage by taking advantage of spatial reuse in all stages of the AoA scenario. This is best demonstrated by examining the spectrum sharing plan obtained in Job 6 as shown in Fig. 5. We see from the figure that the deep-Q agent of each team decided to use between 4 to 6 channels throughout the three stages. Since each agent selected to use 5 for the majority of time, the teams must share overlapping channels. The DSS agents were able to make the right spatial reuse decisions in a distributed manner. To see that, consider the channel plan in stage 3 in which the channel assignments of the 5 teams are stable, except in a few transient periods. For Teams 1, 2, and 3, their respective DSS agents decided on using three non-overlapping subsets of 5 channels because the two teams' networks were close in locations that no spatial reuse advantage could be exploited. On the other hand, the DSS agents of Teams 0 and 4 converged on using the same subset of 5 channels because the two teams' networks were sufficiently far apart such that full spatial reuse advantage could be exploited between them.

Finally, we observe that there are transient periods in the channel sharing plan as shown in Fig. 5. These transients in channel assignments caused the traffic flows assigned to use these channels to fail their hold-time QoS requirements, and hence resulted in the lower score. Nevertheless, we should point out that these channel assignment transients are not caused by the deep Q-learning channel selection algorithm described in Section IV, but rather by the distributed and asynchronous operation of the DSS agents of different teams.

VI. Conclusion

We present results from Colosseum experiments that demonstrate the efficacy of a deep-Q network for dynamic spectrum sharing. In particular, the results show that the deep-Q agent is able to identify and take advantage of spatial reuse opportunities in the Alleys of Austin scenario, achieving an approximate 48% improvement in traffic delivery (based on the SC2 QoS-based scoring system). The use of deep reinforcement learning for this problem is still limited by the ability to both generate sufficient training data, as well as the time required for training. Future work will consider the impact of diverse traffic flows to the agents, as well as agents that use heterogeneous strategies and/or physical and link layers.

References

[1] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1136–1159, 2012.

[2] J. Lunden, V. Koivunen, and H. V. Poor, "Spectrum exploration and exploitation for cognitive radio: Recent advances," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 123–140, 2015.

[3] Y. Wang, Z. Ye, P. Wan, and J. Zhao, "A survey of dynamic spectrum allocation based on reinforcement learning algorithms in cognitive radio networks," *Artificial Intelligence Review*, vol. 51, no. 3, pp. 493–506, 2019.

[4] C. Bowyer, D. Greene, T. Ward, M. Menendez, J. Shea, and T. Wong, "Reinforcement learning for mixed cooperative/competitive dynamic spectrum access," in *Proc. IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, Newark, New Jersey, Nov. 2019, pp. 1–6.