A Dynamic Spectrum Sharing Design in the DARPA Spectrum Collaboration Challenge

Tan F. Wong, Tyler Ward, John M. Shea, Marco Menendez, David Greene, and Caleb Bowyer
Department of Electrical and Computer Engineering
University of Florida
Email: {twong,jshea}@ece.ufl.edu

***Abstract*—This paper presents an overview of Team GatorWings' dynamic spectrum sharing design that won the top prize in the DARPA Spectrum Collaboration Challenge (SC2). The overview highlights the design approaches employed by Team GatorWings' radio network to understand the overall communication environments, to provide agile and robust access to the wireless channel while sharing the spectrum with other peer networks, and to optimize the decision engine in order to win the competition. A summary of the observations made and lessons learned throughout SC2 is also provided in hopes of providing directions for further research and development in dynamic spectrum sharing.**

***Keywords*—Dynamic spectrum sharing; dynamic channel access; autonomous spectrum management; software-defined radio.**

I. Introduction

We present an overview of the approach to dynamic spectrum sharing used by our team (Team GatorWings) to win the DARPA Spectrum Collaboration Challenge (SC2), which employs the format of tournament competition to jump start the development of "a new wireless paradigm in which radio networks will autonomously collaborate and reason about how to share the RF spectrum, avoiding interference and jointly exploiting opportunities to achieve the most efficient use of the available spectrum" [1]. This new spectrum management paradigm aims to take humans out of the loop so as to shrink the time scale of its dynamics from years down to seconds.

In SC2 matches, multiple teams of radios must operate in a shared radio frequency (RF) environment that emulates mobile ad hoc networking scenarios. Each team can score points by delivering traffic flows, but teams have a cooperative objective that in each measurement period, their score will be limited to the lowest score of all of the teams unless all teams' scores are above a specified threshold. Teams share information about their channel usage, radio locations, and performance with other teams, and each team tries to use this, along with other information the team collects, to maximize their team's score. Achieving this is complicated by the fact that teams that are scoring above the threshold can just report the scoring threshold, and not their true score. In addition, there is no real-time scoring information available, so all scores used for adapting the spectrum sharing strategy are based on online score estimates generated by each team. To further complicate matters, teams do not have any information about the radio implementations and strategies of the other teams, except for information they can gather during the matches. This results in SC2 matches being mixed collaborative/competitive games with partial information about peers' performances and strategies.

Points are scored in SC2 matches by meeting specific quality-of-service requirements (called *mandates*) for traffic flows assigned to the radios of a team. Teams must build decision engines to determine which traffic flows to attempt to deliver and which channel resources to use to deliver those flows, with the goal of maximizing the team's score. In addition to the flow mandates, the inputs to the decision engine include our and other teams' spectrum usage, estimated scores, and node locations, as well as our estimates of channel throughput. Specific scenarios also include input information on degradations in channel capacity due to jamming or use by incumbents that must be protected from interference. The combination of 10s-100s of flows with varying mandates, up to 40 MHz of bandwidth available, and 50 radios across five teams results in an optimization problem with a huge state space, and not nearly enough training games available to even consider directly applying a machine-learning (ML) approach to the full optimization problem.

Based on these observations, our team's decision engine decomposes the overall optimization problem into a sequence of optimization problems, each of which has a much more limited scope than the overall problem. These optimization problems, the approaches we use to solve them and our observations and recommendations for future improvements are the emphasis of this paper.

II. SC2 Overview

The dynamic spectrum access (DSA) system presented in SC2 consists of multiple teams (networks) of radios communicating in a specified frequency band. Teams engage with each other during *matches*, which have reproducible, time-varying radio channel characteristics (produced using *Colosseum* which is a large-scale channel emulator developed by DARPA) and traffic flows. Each team receives a score based

Our participation in the DARPA SC2 was supported in part by a prize from the DARPA Spectrum Collaboration Challenge, the National Science Foundation under Grant 1738065, and by AFOSR award number FA9550-19-1-0169.

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 *mandate* is provided that details the quality-of-service (QoS) that must be achieved to score points for that flow. Flows either come regularly and have specified throughput and latency requirements, or come as file bursts, for which 90% of the packets have to be delivered before a specified file transfer deadline. Each flow has an associated number of points that can be achieved in each measurement period 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. Each team has a *mandate threshold*, and the match score achieved in any scoring interval is limited to the lowest score among the teams if any team is below its mandate threshold. Thus, the mandate threshold determines the required level of cooperation in the mixed cooperative/competitive game.

Each team has one radio that acts as a gateway (GW), and the GW can communicate a limited set of collaboration information to peer networks over a separate collaboration channel. The information carried over the collaboration channel must adhere to a specified Collaborative Intelligent Radio Network Interaction Language (CIL) and includes:

- locations of the radios, specified as GPS coordinates,
- frequencies used and information on which radios are using which frequency bands, and
- number of achieved mandates and the mandate threshold.

When teams are scoring below the mandate threshold, they must report their actual scores to their peers. However, when teams are scoring at or above the mandate threshold, they can report the mandate threshold instead.

The SC2 championship event (SCE) consisted of two parts. In the first part, a series of five *elimination rounds* were conducted to winnow the field of teams from 10 to 5. In each such round, a single scenario was run many times with different sets of teams in the scenario, and scores are summed up within that round. Then the two lowest-scoring teams play in an elimination match with the three best teams to see which of those two teams will be eliminated from the tournament. In the second part, the *final part*, of the tournament, the final five teams competed in four different scenarios, with points being awarded in each match for the relative finishing position in that match. The final placement was determined by adding the points across the four matches in the final part.

III. Design Overview

Our vision for the SC2 program was to develop a collaborative intelligent radio network (CIRN) that can autonomously sense the overall communication environment, including the RF channel characteristics, traffic conditions, peer networks' strategies and operations in order to create an understanding of the environment and from that decide on and actuate a spectrum sharing strategy that can optimize a set of given communication targets. The structure of the SCE drives the design approach of our CIRN. Specifically, the elimination and final rounds of the SCE have somewhat conflicting objectives. On the one hand, as the cumulative match scores over many matches consisting of different teams determine the ranking of the teams in the elimination rounds, the design objective of our CIRN should be to maximize its cumulative match score. This, as a consequence of the SCE scoring function, requires our CIRN to be more cooperative. On the other hand, the objective in the final round is simply to win each individual match, and hence our CIRN should be designed to maximize the match score for each individual match. This objective requires our CIRN to be more competitive.

The analysis above leads us to the following three-prong CIRN design approach that we have adopted for the SCE:

1) **Understanding of the overall communication environment:** To efficiently utilize the available spectrum and to intelligently share the spectrum with existing systems and other peer CIRNs, we build a CIRN that maps the interference terrain using collaborative information obtained from peer CIRNs through the CIL protocol as well as our own network measurements. We also develop a machine learning (ML) classifier to identify peer teams competing in each match. This allows us to tailor our decision strategy to individual and groups of peer teams. For example, when we identify that we are competing with best performing teams in a match, we increase how aggressively we are in utilizing the radio spectrum. Details are in Section III-A.
2) **Agile and robust channel access capabilities:** We build a highly agile radio, together with an accompanying medium access scheme, to affect the data flow and channel assignment decisions made by the decision engine. In particular, the medium access scheme must allow us to quickly exploit transmission opportunities arising in time, frequency, and space. In addition, our CIRN will inevitably need to work in interference-dominated RF environments; thus, our physical-layer (PHY) and medium-access control (MAC) radio design must be flexible and robust enough to support communication in the presence of interference from peer networks. We achieve these design requirements by employing multiple modulation and coding options, frequency-domain equalization, and adaptive receive beamforming in the PHY, dynamic time-frequency channelization in the MAC, flexible link-layer automatic repeat request (ARQ) schemes as well as a digital, multi-transceiver radio architecture implemented in the FPGA fabric of the X310 radio. This multi-transceiver radio supports simultaneous transmission and reception over multiple channels and fast channel switching. For more detail, see Section III-B.
3) **Optimization of decision engine to win the SCE:** As previously mentioned, we build a decision engine that solves the problems of admission control, channel usage, and resource allocation by breaking the problem down into determining which channels to use, which flows to support in the specified number of channels,

and which channels and which time slots will be used to deliver the chosen flows. Details are in Section III-C. Under our approach, we control how cooperative or how competitive our CIRN is by controlling the choice of the number of channels and how many channels are used to cause intentional interference to other networks. We further discuss our strategies relating to cooperative vs. competitive operation in Section IV.

The overall block diagram of our software-defined radio (SDR) is shown for reference in Fig. 1. The low-layer physical-layer signal processing functions were built using RFNoC [2] and our own custom blocks, and the other functions were custom-built, multi-threaded C++ modules.

A. Environment Understanding Design

Each radio in our network has a spectrum sensor that can measure the power spectral density (PSD) of the whole frequency band. The PSD measurements are used to estimate the occupancy percentage of each channel. Spectrum usage and GPS information of peer networks obtained from the CIL 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 estimates 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. The spectrum sensor is also responsible for estimating channel occupancies and identifying active incumbents (such as radar systems) and jammers.

We employ characteristics of the CIL messages, such as the interarrival times of different types of CIL messages and the precision levels of some of the quantities exchanged in the CIL messages, as features to identify individual competing teams. Recognized feature vectors representing all competing teams are employed to identify individual teams in a match based on a simple minimum distance classifier. Due to the fact that there are no more than 12 teams actively competing in the last stage of the SC2, our simple classifier can typically achieve an almost perfect identification performance within the first 30 seconds of a match. One main use of the team identities obtained is to flag teams that provide inaccurate or nonsensical voxel, performance, and/or GPS location information in the CIL messages. Another main usage is to identify the group of best performing teams. Our decision engine uses a more aggressive strategy when the best performing teams are identified to be in a match.

B. Channel Access Design

Channel access in our radio network follows a time-frequency structure as shown in Fig. 2. The available frequency band is channelized into overlapping channels of 1 MHz in bandwidth. Adjacent channels are separated by 0.5 MHz. A subset of non-overlapping channels is dynamically selected to support the data flows admitted by our radio network.

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. 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 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.

C. Decision Engine Design

The decision engine (DE) is responsible for determining what channels our radio network will use and which flows can be supported using those channels. In what follows, we partition the set of flows into those that are *latency bound*, meaning that they require more than one pocket per frame and those that are *non-latency bound*. The inputs to the DE include:

- the set of specified mandates for our team's flows,
- the estimated number of achieved mandates and the total mandates for our network,
- information on the throughput per pocket that is expected between each source-destination pair,
- the channels used by our network and by the peer networks,
- estimated channel occupancies from our spectrum sensor,
- computed SINRs from our interference map, and
- the estimated achieved and total mandates from competitor networks,
- interference power reports from incumbents, and
- jammer and incumbent channel-occupancy information from the spectrum sensor.

Using these inputs, the DE determines which flows are transmitted and in which pockets they will be transmitted, with the goal of maximizing our team's match score. The output from this process is the *pocket schedule*, which is a list of pockets (time-frequency slots) and the source and destination(s) that will communicate in that slot. Because of the complexity of this optimization problem, the problem is decomposed into the following three steps:

1) Channel selection: We choose the set of channels C to use by first identifying the number of channels to use and then choosing the particular set of channels of the specified size. Here, we give an overview of the basic functionality of these algorithms for the case in which there are no incumbents or jammers and provide some comments on how these algorithms change in the presence of such impairments. Let N_C denote the total number of channels available in the scenario, and let

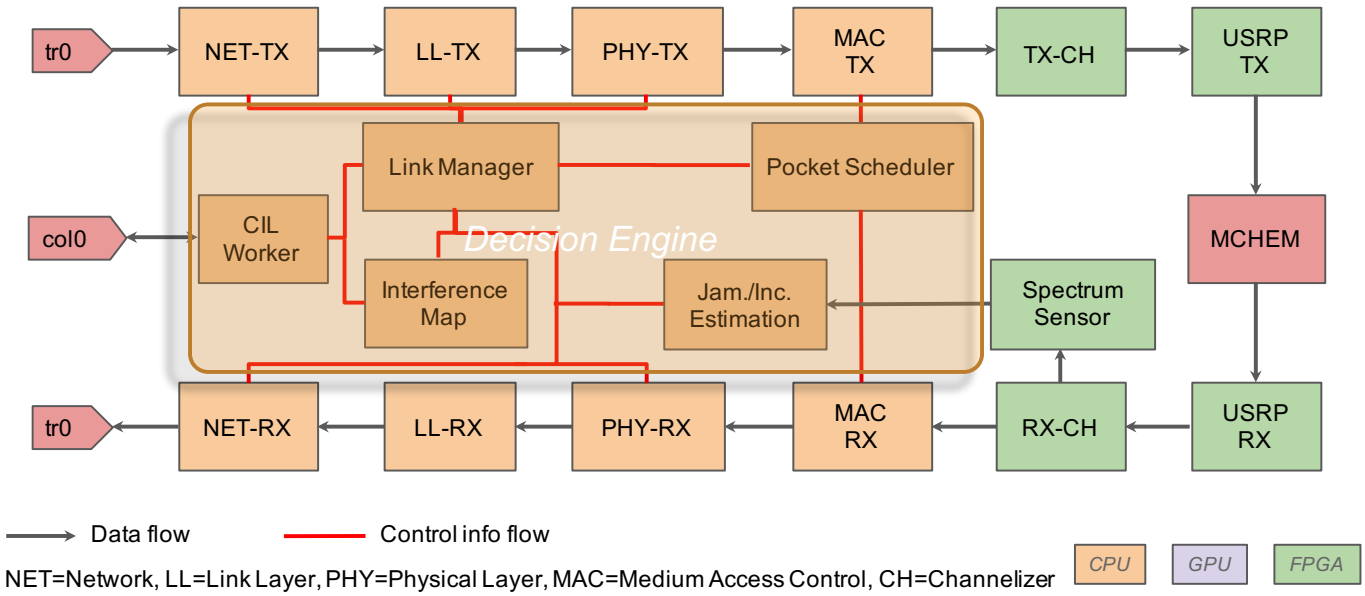
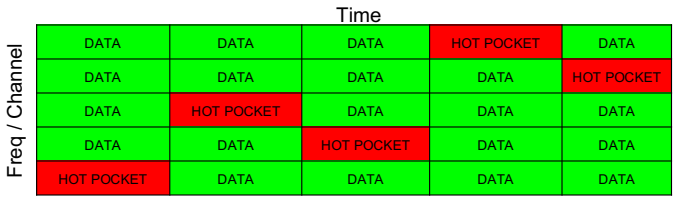
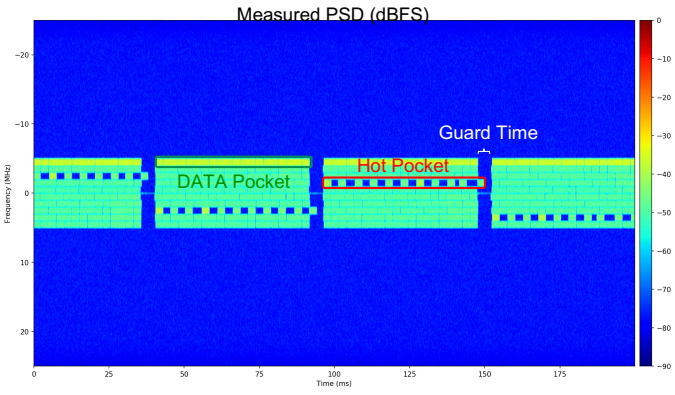


Fig. 1. Overall system block diagram of our software-defined radio.



(a) Illustration of a scheduling frame



(b) Measured power spectral density

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

N_T denote the total number of teams (including our team). We say that our *fair share* of channels is $F = N_C/N_T$. If we identify that the three other best teams (assumed to be MarmotE, Zylinium, and Erebus) are present, then we say we are in *championship mode*.

In our SC2 championship design, the choice of the number of channels to use was essentially a combined controls and expert systems (ES) approach[1]. In this approach, the number of channels is adapted based on observing the effect of the previous choices of numbers of channels on the performance, which is measured in terms of our team's score in comparison to our peers' scores. Our algorithm uses many different cases to deal with different situations for the mix of users and their scores relative to our own score and the scoring threshold(s). Here we present a reduced set of cases that omits some of the details to give a simpler "big picture" overview of our decision engine's operation:

1) At the beginning of stage, all scores are zero and the flows and mandates have just been received, so there is not enough information to choose a good $|C|$. Because of the way that scoring works, we do not want to be outscored in this period, so we are somewhat aggressive in our use of channel resources. If in championship mode, we set $|C| = 3F$; otherwise, $|C| = 2F$.
2) If all peers are scoring below the threshold and doing worse than us, then we reduce $|C|$ in proportion to the difference between our score and the highest of the peer's scores.
3) If some peers are below the threshold and below our team's score, we generally do not change $|C|$ (instead, our channel choices will avoid the channels of lower-scoring networks). However, if we have only have one peer peer below us and either have two higher-scoring peers or have one higher-scoring peer and are in championship mode, we will increase $|C|$ in proportion to the difference between the highest scoring peer's score and our score.

[1]In addition, we have also attempted to solve this problem using a reinforcement learning algorithm. Details are in [3].

4) If some peers are below the threshold, but all are above our score, then we will increase $|C|$ in proportion to the difference between the highest scoring peer's score and our team's score. The constant of proportionality is changed depending on whether we are in championship mode.
5) If all peers are at or above the threshold, we will increase our number of channels used by 2 if in championship mode, and 1 otherwise.

When jammers or incumbents are present, the number of channels may be reduced to avoid interference to an incumbent, or increased due to reduced capacity of a channel for jammed channels or channels with partial-time incumbents.

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 we were using in the previous interval that no other team is trying to use,
2) **Unoccupied channels**: channels that no team is currently using,
3) **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,
4) **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
5) **Channels used only by lower-scoring peers**: these are a last resort.

Within each set, the channels that are least occupied (according to our spectrum sensor) are chosen first. When an incumbent or jammer is present, additional categories of channels are introduced to capture the relative value of channels with different types of impairments (i.e., stationary vs sweeping jammer).

2) 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.

3) 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. We note that the last two steps above can be formulated into typical linear, integer, and mixed integer optimization problems, for which we apply standard optimal and suboptimal solutions.

IV. Lessons Learned

The results in SC2 demonstrate that heterogeneous wireless networks can autonomously and independently manage a shared spectrum in order to achieve their individual communication tasks. In this section, we conclude our development efforts in SC2 by stating a few observations that we have made and lessons that we have learned in our participation in SC2 in hopes of highlighting some further research and development directions that we believe are important for dynamic spectrum sharing:

1) **How algorithms are evaluated will drive the strategy:** A big problem in the SC2 competition is the need to operate in both a cooperative mode and in a competitive mode and there is not enough information provided to know which strategy to employ. For instance, in the round robin matches of the SC2 championship, it is important to be cooperative with other radios so that even if you do not win a particular match, you can maximize your score across matches. On the other hand, in the final round of the SC2 championship, it is important to be as competitive as possible to try to win each match. A key observation that helped us win the SC2 championship is that a very competitive strategy was sufficient. If our team performed poorly during the elimination rounds, it would be sent to an elimination match for which a competitive strategy was optimal. In fact, our team faced elimination twice during the five elimination rounds, but survived both elimination matches. Thus, for any practical system, there must be carefully design "scoring functions" that drive the teams toward the desired behavior.
2) **Practical scoring design:** In addition to developing scoring functions that drive teams to achieve a certain behavior (e.g., cooperation), applications of SC2 technology to commercial or military systems will need a scoring functions that are practical to implement and that measures a more long-term performance of a CIRN. For example, each CIRN may need to declare a score target that describes its targeted QoS performance and a resource target that describes the predicted amount

of spectrum resources it needs. Then the actual performance score and amount of resources used by the CIRN are measured over a longer-term period. Rewards and penalties should be handed out to the CIRN depending on whether the measured values are close to or significantly deviate from the targets, and may also be assigned based on the efficiency of use of the shared radio spectrum.

3) **No machine-learning black box:** The DSA problem presented in SC2 is too massive and complicated to solve by a single ML black-box algorithm. It is clear to us through our design process that very significant domain-specific engineering is needed to break the DSA problem into smaller pieces, some of which may be best solved by ML algorithms and some other may be best solved by classical optimization approaches. It appears most promising to continue on this line of approach by breaking down the DSA problem and determine what subproblems should be best solved by ML algorithms in the near future.
4) **Not enough training and validation data from Colosseum:** We have not been able to obtain enough training and validation data from running enough RF emulation jobs in Colosseum. It appears that a main hindrance of employing ML in SC2 is the inability to do sufficient training of ML algorithms in an emulated or real channel environment, because doing so would require too much of Colosseum's resources. There is an urgent need to develop a simulation-based counterpart of Colosseum to generate sufficient training and validation data for developing ML-based DSA solutions.
5) **Frequent changes in peer radios and strategies:** During the last few weeks before the SCE code submission deadline, we saw very frequent updates in the strategies and even basic PHY and MAC radio design of other teams. The whole "train-and-then-operate" paradigm of typical ML algorithms simply could not catch up with those rapid updates. This motivated us to use our expert-system algorithm to performance channel selection instead of our ML approach. More research is needed to develop approaches that use more exploration, and switching systems could be employed to cope with rapid updates like what we saw near the end of the SC2.
6) **CIL protocol:** The CIL protocol employed in SCE is sufficient, but perhaps not the most efficient, for the DSA problem considered. It is sufficient in the sense that it allows a minimal set of information to be exchanged between peer CIRNs to affect a reasonable level of collaboration between different CIRNs with a large variety of heterogeneous designs. On the other hand, it may not be efficient in that a much higher level of collaboration may be achieved if more detailed information about the CIRNs' operations can be exchanged via the CIL protocol. In addition, the current CIL ignores any privacy and security considerations about the exchanged information. It appears that much more research effort is needed to develop a general CIL protocol with minimal information exchanges that allows efficient spectrum sharing collaboration while addressing privacy and security concerns. We believe a more fruitful practical approach is to consider developing a CIL protocol that is restricted to among CIRNs with similar designs in which the efficiency and privacy issues could be less challenging.

V. Conclusions

We presented an overview of the spectrum-sharing approach used by our team's radio design to win the DARPA SC2. We built a robust and flexible lower-level radio that allowed us to take advantage of openings in the radio spectrum across frequency, time, and space. We then built a decision engine that can fuse information from the CIL and our own observations to try to maximize our score. Our approach decomposes a problem with a huge state space into several sub-problems that can be solved using conventional optimization and control approaches. The SC2 program has shown the potential for autonomous agents to do spectrum sharing, and it has also revealed some of the area that still need additional work, as detailed in the Lessons Learned (Section IV).

References

[1] DARPA, "SC2: Spectrum collaboration challenge," https://www.spectrumcollaborationchallenge.com, retrieved on October 28, 2019.

[2] Ettus Research, "RFNoC," https://kb.ettus.com/RFNoC, retrieved on January 10, 2019.

[3] 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)*, 2019, to appear. Available: http://wireless.ece.ufl.edu/jshea/pubs/dyspan19.pdf.