

A Dynamic Spectrum Footprint Adaptation Framework for Collaborative Spectrum Sharing

Ruben Mennes, Jakob Struye, Carlos Donato, and Steven Latré
IDLab, University of Antwerp and IMEC, Belgium
Email: [firstname.lastname]@uantwerpen.be

Abstract—Spectrum sharing is a reality closer than one might think. Dynamic and intelligent spectrum allocation, where different networks collaborate to optimize spectrum usage jointly, is required to overcome spectrum scarcity. Artificial Intelligence (AI) can play a major role to solve this complex problem in dynamic environments with continuously changing data requirements. A part of the problem can be solved by using smart AI-enabled flow control. Smart flow control can have a major impact on the spectrum footprint of mobile networks where it can optimize the Quality of Service for the own and neighboring networks. This paper presents the architecture and the basic principles of the dynamic spectrum footprint control based on flow prioritization of the SCATTER radio system, a wireless end-to-end communication system that participated in the DARPA Spectrum Collaboration Challenge. The flow control mechanism is a policy-based framework.

Index Terms—wireless flow control, collaboration, system architecture, differentiated quality of service

I. INTRODUCTION

SPECTRUM scarcity is a phenomenon to be tackled in the current decade. Most users in the wireless spectrum are limited to using the unlicensed, relatively narrow industrial, scientific, and medical (ISM) radio bands. As the demand for wireless spectrum is only expected to increase in the upcoming years [1], these bands will only become more congested. Other parts of the spectrum remain underutilized as they are licensed to only a small number of users, as decided by governmental agencies. Fundamental changes in the approach of sharing this scarce resource with many users are needed to scale with the foreseen demand for new capacity [2].

Ongoing efforts to tackle this issue are generally focused on one technology, such as Wi-Fi, Bluetooth, and ZigBee. These efforts usually rely on the technology's inherent characteristics, making use of sense-and-avoid techniques or duty cycle changes [3]. One common example is Cognitive Radios (CRs), where Secondary Users (SUs) can use a band only if its Primary User (PU), owning the band, is inactive [4]. Collaboration could be seen as smart and fair spectrum usage across different networks without static priority between networks based on flow-level Quality of Service (QoS) requirements.

In this paper we present the architecture of a flow control policy-based framework, that could be implemented on wireless networks. The policy framework enables optimizing

flow control, and as a result the spectrum footprint, based on collaboration and inter-network QoS by using Artificial Intelligence (AI)-driven policies.

This framework is a crucial part of the SCATTER radio system, a wireless end-to-end communication system that participated in the Defense Advanced Research Projects Agency (DARPA) Spectrum Collaboration Challenge (SC2). During the SC2 competition, the presented architecture was employed for optimizing the scoring and the spectrum footprint. This three-year competition was focused on collaboration between wireless networks in the same collision domain [5]. Each participating team built a wireless radio based on technologies of their choosing. These competed and collaborated in a wide array of scenarios, simulating collision domain-wide collaboration with multiple independent networks. As team SCATTER, we made it through the finals and ultimately reached the 6th position.

II. RELATED WORK

Wireless network slicing is a proven technique to optimize spectrum usage, allowing different devices to adapt their spectrum footprint based on their needs and requirements. On the licensed part of the spectrum, a lot of different techniques are proposed and currently in use [6]. Tehrani et al. classified the cellular spectrum sharing scenarios into a few categories. Slicing techniques that focus on inter-operator spectrum sharing, and techniques available for sharing spectrum between different operators. Unfortunately, due to the business concerns and lack of strong evidence in favor of spectrum sharing, Mobile Network Operators (MNOs) are not showing any interest in these techniques. They have implemented almost none of these existing techniques. Most of these techniques have proposed a primary-secondary hierarchy, where own users are primary.

Different implementations are proposed in [7], where the authors consider sharing their radio network controller, and two MNOs could operate in Wideband Code Division Multiple Access. While in [8], the authors proposed to roam other users of adjacent MNOs on their network in a distributed manner without a centralized controller. Additional work to support incumbents in the spectrum is described in the literature as well [9]. Note that all of these works are related to CR technologies, where typically PUs and SUs are defined, and SUs can only access the spectrum if the PU is not using the

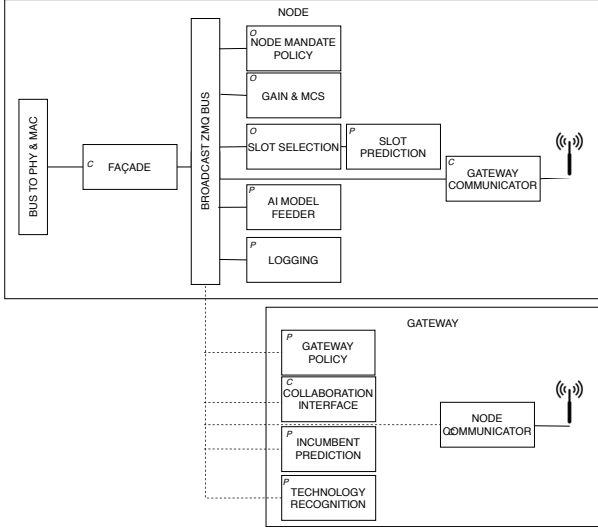


Fig. 1: Overview of the SCATTER decision making submodules

spectrum [4]. This functionality is implemented in physical and Medium Access Control (MAC) layers. In this paper, we focus on flow control and broadcasting state information to other optimization modules that can optimize the physical layer and MAC parameters to provide CR-like approaches if necessary. While during the SC2 competition the goal was to slice on a per-flow basis rather than a per-user basis, the framework proposed in this paper is more generic than most implementations of CR Networks (CRNs). It could be used for primary-secondary user slicing as well.

III. BRIEF OVERVIEW OF THE SCATTER SYSTEM

In previous work, we already described the overall architecture of the SCATTER system [10]. In this work, we focus on one specific part of the overall system: the architecture of the policy framework.

The overall design of the SCATTER radio system is based on five layers: the network interface, MAC-layer, PHY-layer, RF-monitor and the decision engine. Each layer is completely independent and can interact with others via a shared message bus using clearly defined messages. The policy framework, discussed in this work, is a part of the *decision engine*. The framework consists of two different modules: 1) the gateway policy module, and 2) the node flow policy module. These modules interact mainly with the *network interface* that controls the flows in the system and provides the framework with information about the flows. One node of the network acts as the gateway. The gateway node has a wired backbone and can communicate with other teams. Additional modules of the *decision engine* are started on the gateway as well.

One gateway module acts as the collaboration interface. This module exchanges messages with other networks concerning performance and traffic load. In the context of the DARPA SC2 competition, this communication was strictly defined by the participating teams, and their score was exchanged

in real-time during each match [11]. The way of using this module is out of the scope of this work, but it is essential that our framework can access the performance of the neighboring wireless networks.

IV. FRAMEWORK ARCHITECTURE

In this section, we discuss the architecture of the framework. The goal of the framework is to optimize and prioritize flows in the system and inform other modules in the system about the current operational state of working. In this way, it is possible to control some link-level parameters, like the Modulation and Coding Scheme (MCS) or gain. The framework is designed to support more advance learning techniques, such as Reinforcement Learning, online learning, semi-supervised learning, or other Machine Learning (ML) techniques, and support legacy rule based algorithms. An example of this is the slot prediction module that uses online supervised learning [12], or incumbent recognition that uses semi-supervised learning as described in Section V.

To build our framework, we assume that 1) there is an interaction with other modules via a control mechanism, 2) one node is a gateway node as discussed in Section III, 3) communication between gateway and all other nodes is possible, but expensive, 4) every node can observe (parts of) the behavior of the own system, 5) the framework can override other modules' control over, for example, link behavior, and 6) flows may have specific requirements, known across all nodes. We assume that every flow has some priority-based score to be optimized; these scores should be known.

The framework consists of four basic structures: 1) An abstract representation of a flow, 2) Observers that observe one part of the system and provide information to all other classes, 3) Policies responsible for controlling the flows and the other parts of the system, 4) Handlers that control the policies.

A. Abstract Flow

The basic building block of the framework is a class that abstracts all complexity about a flow. Through this class it is possible to read the flow's requirements (e.g., minimal throughput or maximal latency), access its current performance, and (un)block the flow.

Note that the implementation of the flow abstraction depends on the type of flow and where it is used. Different types of flows can have a different implementations. Two subclasses implement the specific functionality of a flow on the node and gateway, respectively. On the gateway, it is not possible to immediately block or unblock flows, and wireless communication is necessary to update flow statistics. The actual *Flow* class is implemented as a subclass of the *AbstractNodeFlow* and *AbstractGatewayFlow*. Multiple child classes of the *AbstractNodeFlow* and *AbstractGatewayFlow* class can exist if different implementations are necessary for different flow types, such as Voice over IP (VoIP) and Hypertext Transfer Protocol (HTTP) flows.

B. Policies

Policies are the main control objects in our framework. Each policy has a set of constraints that describes when a policy can be active. Once a policy is activated, it has (along with all other active policies) control over the flows, if necessary, by using the correct handler in the system and can communicate with other modules in the system to change their behavior. The behavior of the policy could be rule-based or ML could be used to learn specific behavior. The observers could be used to describe a state or to trigger events. Each policy needs to have a gateway and a node implementation, as the network state is described centralized on the gateway, where policies are activated or deactivated. This information is forwarded to the nodes, and they activate the same policies. On each node, policies have immediate control over the flows originating there, and can communicate directly with other modules.

We propose to have two classes of policies: performance-driven policies and environment-driven policies.

The performance-driven policies are policies that try to optimize the active flows (and as a side effect, the spectrum footprint). Precisely one of the performance-driven policies is active at any time. Within the SCATTER system, these policies are based on the performance of the teams. If the minimal performance is lower than a specific threshold, we have a specific policy that limits the number of active flows, freeing spectrum for weaker teams.

Environment-driven policies, on the other hand, are policies that consider additional information about the environment. Zero or more of these policies could be active at the same time. The environment-driven policies can interact with the active performance-driven policy and can override unblocking flows or block more flows. These are unable to unblock flows, as we assume that an environment-driven policy describes a limitation or constraint of the environment, indicating that less spectrum than expected is available. An example of such an environment-driven policy is an incumbent policy. From the moment the incumbent is detected in the environment, the incumbent policy becomes active. This policy prohibits all unblocking operations proposed by the active performance policy; blocks more flows if there is a violation, and limits maximum allowed TX gain.

C. Observers

The observers are responsible for tracking one specific part of the system, collecting and aggregating data. This data is available for all other classes on the same node or gateway. Note that on every node, only one instance of a specific observer is necessary, and these observers could be implemented as a singleton. To provide information at the gateway, it might be necessary to forward data from the nodes to the gateway. This is possible but must be limited as it is expensive and has to scale with the number of nodes in the system. An observer that should be available on each node and the gateway is the *FlowObserver (FO)*. The *FO* observes the status of all the flows on a node, or all the flows of the system on the gateway. All the policies, described

above, are able to get information about the flow via the *FO* in a representation of an *AbstractFlow* class. In this way, the active policies can manipulate these flows. Within the SCATTER system, we implemented three major observers. 1) The *FO* on node and gateway side as discussed before, 2) The *NodeObserver* that observes the state of every node, and 3) the *NeighborNetworkObserver (NNO)* that observes the state and performance (score) of the neighboring networks. In the context of the SC2 competition, the *NNO* uses the score announced via the collaboration interface. In general, this could be done by any metric, as long it describes the performance of the network.

D. Handlers

The handlers are the glue of the design. They are responsible for distributing information from the gateway to nodes and making sure that decisions about the flows are taken correctly. We proposed to use at least these two handlers 1) the *Policy Handler* and 2) the *Argument Handler*.

The *Policy Handler* is responsible for the activation and deactivation of the policies and controls the evaluation of blocking or unblocking different flows. Note that the policies first become active on the gateway side. We make a distinction between the performance-driven policies and the environment-driven policies. We propose to use an additional *Performance Policy Handler*, as a member of the *Policy Handler*, that takes care of the performance-driven policies. This handler decides which performance-driven policy is active at any time. In contrast, the environment-driven policies, of which any combination could be active, can initiate their own (de)activation. On the gateway, the handler polls the performance-driven policy for a proposal of flow block and unblock operations, at a fixed interval. Then, each active environment-driven policy evaluates this proposal, removing unblock operations or inserting block operations as deemed necessary. The final proposal is sent to the nodes along with the set of active policies. There, similar handlers process the changes. These handlers also evaluate the proposal using node-side versions of the policies, meaning changes can also be initiated by the node.

The *Argument Handler*, on the other hand, is an additional way to represent state. Each policy could have arguments. An example of this is the center frequency of an incumbent for the incumbent policy. This could help the policy to make changes to the flows or communicate this to other modules, like slot selection. The argument handler is responsible for distributing these arguments across all nodes while minimizing the additional spectrum usage. One way of implementing this is by using an acknowledgment, where each node piggybacks the ID of the currently used argument on a packet sent to the gateway. From the moment each node is using the correct arguments, no additional messages need to be exchanged.

V. VALIDATION

Within the context of the DARPA SC2 competition, flow control to reduce spectrum footprint was key. Performance of teams was measured by a scoring formula and collaboration

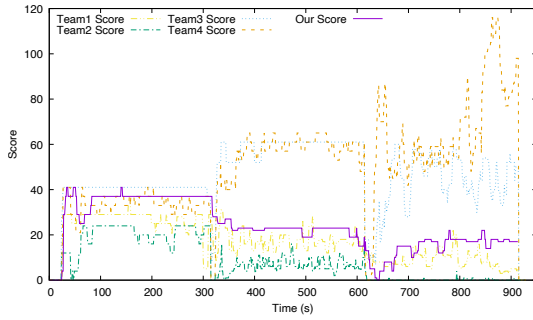


Fig. 2: Example of spectrum footprint control. The purple line represents the score of the SCATTER radio. The other lines represent the score of other teams in the game.

was enforced by using the following scoring rules: each team received the score of the worst-performing team. The score is based on the stable flows of each team. Because, for each scenario, many matches were played, collaboration was required, mainly due to the available spectrum being limited, and because incumbents or jammers could be active in the same environment [5]. By playing many matches with four other randomly selected teams, pushing (weaker) teams to have a better overall performance leads to a higher average score. If a team cannot reach the minimal score, the spectrum footprint of the other teams may be too high. Flow control is an implicit way of controlling the spectrum footprint and influences the score of the own team and the other teams.

The following examples are games executed on the SC2 testbed: Colosseum. This testbed consists of 128 Software Defined Radios (SDRs), combined with an RF-simulator. Each game is played against 2 or 4 other randomly selected teams that participated in the finals of the SC2 competition.

A main *Performance-driven policy* was implemented to enable flows up to a score a bit above the weakest team. This results in minimizing the spectrum footprint. Weaker teams could improve their performance. On the gateway side, a simple AI model was implemented searching an optimal the scoring range $[s + \epsilon_1, s + \epsilon_2]$, where s is the score of the weakest neighboring team, and $\epsilon_1 < \epsilon_2$. The model uses the information of the *NNO* and the currently known active flows to generate a range. The policy enables more flows if the own scoring is lower than $s + \epsilon_1$. In this case we are underperforming in compression with the neighboring networks. If, on the other hand, the own score is greater than $s + \epsilon_2$ than the policy blocks flows to reduce the spectrum footprint because weaker teams are struggling with setting up their flows. This information is forwarded to the nodes. The policies on the nodes apply the proposed changes. An example of the behavior of this policy is shown in Figure 2.

VI. CONCLUSION

We presented the architecture of the policy framework, responsible for flow control and state representation in the SCATTER radio system built for the DARPA SC2 competi-

tion. The framework represents state by enabling or disabling AI-driven policies. The active policies are able to control the flows and inform the other modules of the system about the state of the system. In this way, it is possible to do flow control in a distributed, centralized, or hybrid way depending on the state of the network. We defined two types of policies: *performance-driven policies*, of which only one can be active at any moment in time, and that has the initiative to enable or disable flows, and *environment policies*, of which any number could be active at the same time. The *environment policies* can only reduce the (proposed) active flows.

This framework was implemented in the SCATTER radio system and optimizes collaborative flow control, which was a key feature during the SC2 competition.

ACKNOWLEDGMENT

This work has been funded by Research Foundation Flanders (FWO) and the Flemish Government (1S52917N).

REFERENCES

- [1] I. Cisco Systems, "Cisco visual networking index: Global mobile data traffic forecast update, 2016-2021 (white paper)," *Cisco Public Information*, pp. 1-35, 2017.
- [2] L. Zhang, M. Xiao, G. Wu, M. Alam, Y.-C. Liang, and S. Li, "A survey of advanced techniques for spectrum sharing in 5g networks," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 44-51, 2017.
- [3] A. Garcia-Saavedra, P. Patras, V. Valls, X. Costa-Perez, and D. Leith, "Orla/olaa: Orthogonal coexistence of llaa and wifi in unlicensed spectrum," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2665-2678, 12 2018.
- [4] I. F. Akylidiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "A survey on spectrum management in cognitive radio networks," *IEEE Communications magazine*, vol. 46, no. 4, 2008.
- [5] P. Tilghman, "Ai will rule the airwaves: A darpa grand challenge seeks autonomous radios to manage the wireless spectrum," *IEEE Spectrum*, vol. 56, no. 6, pp. 28-33, 2019.
- [6] R. H. Tehrani, S. Vahid, D. Triantafyllou, H. Lee, and K. Moessner, "Licensed spectrum sharing schemes for mobile operators: A survey and outlook," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2591-2623, 2016.
- [7] G. Salami, A. U. Qudus, D. Thilakawardana, and R. Tafazolli, "Non-pool based spectrum sharing for two umts operators in the umts extension band," in *2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2008, pp. 1-5.
- [8] A. Yarmohammad and R. Tafazolli, "Decentralized inter-radio access network dynamic spectrum selection scheme," in *2009 International Conference on Wireless Communications & Signal Processing*. IEEE, 2009, pp. 1-5.
- [9] M. Matinmikko, H. Okkonen, M. Palola, S. Yrjola, P. Ahokangas, and M. Mustonen, "Spectrum sharing using licensed shared access: the concept and its workflow for lte-advanced networks," *IEEE Wireless Communications*, vol. 21, no. 2, pp. 72-79, 2014.
- [10] S. D. Giannoulis, C. Donato, R. Mennes, F. A. Pereira de Figueiredo, I. Jabandzic, Y. De Bock, M. Camelo, J. Struye, P. Maddala, M. Mehari, A. Shahid, D. Stojadinovic, M. Claeys, F. Mahfoudhi, W. Liu, I. Seskar, S. Latré, and I. Moerman, "Dynamic and collaborative spectrum sharing: The SCATTER approach," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN) (IEEE DySPAN 2019)*, Newark, USA, Nov. 2019.
- [11] D. Stojadinovic, F. A. P. d. Figueiredo, P. Maddala, and I. Seskar, "Sc2 cil: Evaluating the spectrum voxel announcement benefits," in *Proceedings of the IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, Newark, NJ, USA, Nov 2019.
- [12] R. Mennes, M. Claeys, F. A. P. De Figueiredo, I. Jabandzic, I. Moerman, and S. Latré, "Deep learning-based spectrum prediction collision avoidance for hybrid wireless environments," jan 2019, submitted to IEEE Access.