INTELLIGENT OPTIMIZATION AND CONTROL FOR ADAPTIVE CELLULAR NETWORKS

A Thesis

by

CHING WEN CHENG

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Srinivas Shakkottai
Committee Members,	I-Hong Hou
	Anxiao Jiang
	Dileep Manisseri Kalathil
Head of Department,	Miroslav M. Begovic

December 2021

Major Subject: Electrical Engineering

ABSTRACT

When streaming media in a cellular network environment with less than ideal resources, simple algorithms do not perform well by not prioritizing specific clients well enough. Therefore, intelligent network sharing must occur to ensure maximum average quality of experience (QoE). We propose using the general idea of RAN intelligent control (RIC) at the level of scheduling at the radio access network (RAN) at dense and sparse timescale levels to enable such sharing. We formulate this problem of this problem of designing intelligent policies as a Constrained Markov decision process. We observe that the evolution of the state at a client is independent of the others given the scheduling decision on what resources to allocate to it. Hence, we may consider the problem as that of single clients that jointly have a resource constraint, but are otherwise unrelated. We develop reinforcement learning-based polices that are able to determine the resource allocation to clients in two settings of sparse-reward and sparse-control (SRSC) and dense-reward and dense-control (DRDC) and show that significant performance improvements are possible in both settings over vanilla and state-of-the-art polices.

DEDICATION

To my family and friends and whomever supported on pursuing my goal.

ACKNOWLEDGMENTS

I would like to first thank my advisor, Dr. Srinivas Shakkottai for his help and support for pursuing my degree. He has always been supportive and insightful and making sure that I'm on the right way. His passion and inspiration is crucial to the thesis. I'd also like to thank my committee members, Dr. Dileep Kalathil, Dr. I-Hong Hou and Dr. Anxiao Jiang for their support.

I would also like to give thanks to my lab mates. Their insights and feedback from them have helped me to solve problems that I have faced.

Special thanks to my friends and family who have supported me go through the time pursuing the degree.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Srinivas Shakkottai, Dr. I-Hong Hou and Dr. Dileep Kalathil of the Department of Electrical and Computer Engineering and Dr. Anxiao Jiang of the Department of Computer Science.

Funding Sources

Graduate study was supported by a student worker position from Dr. Srinivas Shakkottai.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGMENTS iv

CONTRIBUTORS AND FUNDING SOURCES v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

1. INTRODUCTION 1

2. BACKGROUND 4

    2.1 O-RAN System Architecture 4
    2.2 Reinforcement Learning 5
        2.2.1 Policy Gradient Method 7
    2.3 Proximal Policy Optimization Algorithm 8

3. LITERATURE REVIEW 9

4. SYSTEM DESIGN AND APPROACH 10

    4.1 srsRAN: An Open-Source Platform for 4G and 5G Evolution and Experimentation .. 11
    4.2 srsRAN with RIC for Reinforcement Learning 12
    4.3 srsRAN System Architecture for RIC integration 12

5. SIMULATION-BASED TRAINING 14

    5.1 Simulation System 14
    5.2 Algorithms 14

6. REAL-WORLD EVALUATION 17

    6.1 Sparse-Reward and Sparse-Control (SRSC) 17
    6.2 Dense-Reward and Dense-Control (DRDC) 19

7. CONCLUSION 21

REFERENCES 22

LIST OF FIGURES

FIGURE Page

2.1 O-RAN provides a disaggregated stack and APIs for control hitherto unavailable. We will instantiate our learning algorithms at the level of near-RT RIC to enable highly configurable cellular networks. 4

2.2 Reinforcement Learning Framework 6

4.1 Feedback loop between state of YouTube sessions, controller actions (constrained) and end-user QoE (reward) for reinforcement learning in a media streaming application. 11

4.2 srsRAN with integration of intelligent control for PHY-MAC layers. Information from the PHY-MAC layers is ported to OpenAI along with application-level information that is directly sent (in-band using cellular uplink) from the UE to an SQL database. This state information is sent to an algorithm representing the scheduling policy, which provides weights for each connected UE, which is translated into a count of how many sub-frames to allocate to each one of them. The impact at both the application and at the backlog queues at the srsENB is then used to determine reward, hence completing the feedback loop. 13

5.1 Simulation Training Curve for PPO 15

5.2 Simulation Training Curve for PO PPO 15

6.1 Comparison of average QoE 18

6.2 Comparison of QoE CDF 18

6.3 Comparison of average QoE CDF 19

6.4 Comparison of average QoE 20

6.5 Comparison of QoE CDF 20

6.6 Comparison of average QoE CDF 20

1. INTRODUCTION

The next generation of wireless communication networks will support a wide variety of applications ranging from streaming media, cyber-physical systems control, to IoT-based healthcare monitoring. Not only will the networks need to be adaptable to serve the demands of such time-varying and heterogeneous applications, the network components at a hardware and software level need to be disaggregated so as to be hosted on a variety of mobile, edge and cloud computing resources that can change dynamically. Further, given the essential nature of these networks, they must be highly tolerant to disruption events, such as those caused by increasingly volatile weather patterns. Thus, the defining need of the next generation of wireless networks is adaptability attained through a high ability to support spatially and temporally varying application tasks and and integrate heterogeneous, disaggregated system components via rapid autonomous reconfiguration.

With the rapid growth of the complexity of communication systems, wireless modalities, applications, data volume, high bandwidth, and user mobility, while still sharing a common channel and limited mobile energy source, the problems of ensuring adaptability of wireless networks without the aid of well-defined models are emerging where a data-driven machine learning approach to optimal resource utilization has value. How can we adaptively maximize quality of overall user experience, which requires that each individual application be optimized with respect to its own "Quality of Experience" (QoE) subject to the resources consumed by it? With the proliferation of new applications, designing wireless resource scheduling rules for each application becomes increasingly unmanageable by an apriori class division and must be driven by adaptation via a dynamic learning approach.

With the advent of Open Radio Access Networks (O-RAN) for 5G cellular networks, wireless networking is at the cusp of a revolution, engendered by softwarization and disaggregation at all layers of the cellular stack. Managing the problem of reconfiguring multitudes of parameters, spanning across traditional silos of core and RAN in the interest of adaptability and resilience is impossible using rule-based specifications and human intervention, and hence will need built-in

intelligence and autonomy via machine learning. Several challenges need to confronted to realize it that go beyond offline supervised or unsupervised machine learning, and are explicitly designed for autonomous management of communication networks viewed as hierarchical feedback control systems, i.e., robust approaches to online reinforcement learning are critical. Addressing these challenges will need both fundamental contributions to control algorithm design as well as disaggregated implementations and experiments to inform and validate algorithmic frameworks.

In this thesis, we consider the problem of enabling intelligent control at the radio access network (RAN) level, with the specific use case of supporting high quality video streaming. Our goal is to design a system for endowing a cellular base station with the ability to support intelligent control, and to instantiate reinforcement learning based algorithms that dynamically determine the resources to be allocated to each connected user device.

Our main contributions are as follows:

1. We propose a system design that is aligned with the O-RAN architecture in terms of obtaining state information at the RAN level (backlog buffers and channel qualities of connected devices) as well as application level state in the form of buffered video packets and current quality. We use a pub-sub approach to provide this information to an intelligent controller capable of providing fine grain scheduling information in the form of assigning cellular subframes to connected devices.

2. We instantiate our design on the open source srsRAN platform, which provides complete source code. Specifically, we allow a intelligent controller to communicate its policy decisions by information exchanging with the sub-frame level scheduler in srsRAN. Thus, scheduling decisions can be taken at the transmission time interval (TTI) of 1 ms.

3. We develop a simulation environment that is a good approximation of the real-world system and is suitable for training learning-based algorithms. We consider two classes of algorithms, namely (i) sparse-reward and sparse-control (SRSC) under which the control algorithm only obtains application state information and can only suggest scheduling weights at a long time

scale of 10 seconds, and (ii) dense-reward and dense-control (DRDC) under which the control algorithm obtains fine grain backlog and wireless channel quality information in addition to application performance, and schedules at the TTI-level of 1 ms.

4. We utilize the trained algorithms in a real-world setting of user devices supporting YouTube sessions connected to a cellular base station running srsRAN, with the radio unit being a software defined radio (USRP B210). Likewise, the user devices obtain cellular access via srsUE running over the same software defined radios. Our experimental results show that both the SRSC and the DRDC scenarios yield considerable improvement in the average quality experience by the user over the baseline approaches of round robin (sparse setting) or max-weight scheduling (dense setting). The results suggest that RAN intelligent control can indeed have a significant benefit in terms of tailoring resource allocation to suit the current set of applications and that dynamic policies using RL can be the medium of attaining such intelligent control.

2. BACKGROUND

In this chapter, we provide the background on the O-RAN system architecture, including RAN intelligent control (RIC) and the reinforcement learning (RL) algorithms that can be used to drive such intelligent control.

2.1 O-RAN System Architecture

A highly adaptable cellular network needs an architecture that permits configurability at multiple levels and time scales, as well as the capability to utilize these features autonomously. Simultaneously, it must be must leverage the significant technology developments being made in the commercial 5G space that are allowing for a range of disaggregation options hitherto unavailable.

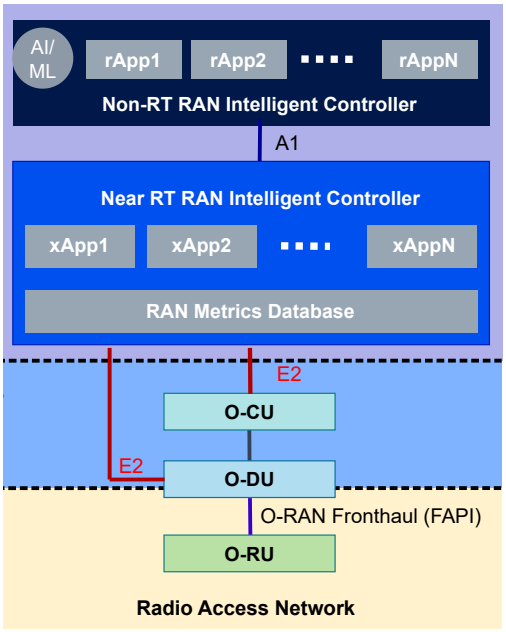

Figure 2.1: O-RAN provides a disaggregated stack and APIs for control hitherto unavailable. We will instantiate our learning algorithms at the level of near-RT RIC to enable highly configurable cellular networks.

Our control systems view of the network prompts us to explore the the use of Reinforcement

Learning (RL), a branch of ML that is explicitly tailored towards learning feedback-control policies towards the creation of our intelligent stack. At a high-level, RL provides a method of learning how to solve Markov Decision Problems (MDPs), which arise commonly in a variety of feedback control systems. Thus, while RL algorithms use DNNs for specific function approximations, the core idea is to "learn by doing" and to tailor the control policy in an online manner based on the information obtained thus far. However, RL in its native form is not conducive to application to communication networks, since several key issues exist such as (i) constrained action space: networks typically have limited power/energy and spectrum availability, (ii) limited observably: the full system state and reward functions are typically not available, and (iii) deployability: a platform for efficient RL coupled with one for feedback control of network systems is required.

The issue of implementing and experimenting with proposed resilient and intelligent stack requires cellular platforms with considerable flexibility, and our approach is to leverage the upcoming disaggregated O-RAN architecture illustrated in Figure 2.1. A popular network software stack option is the 7-2 split, with the open radio unit (O-RU) running the Low-PHY, open distributed unit (O-DU) running High-PHY to radio link control, and open centralized unit (O-CU) supporting higher layers of the stack. Disaggregation also means that the core can be cloud-based, with the ability to provide application-specific slices across core and RAN. The entire stack is configurable via E2 and A1 interfaces that provide the ability for data collection and control using near-realtime and non-realtime RAN intelligent controllers, which are the seat of algorithmic control for learning and adaptation. On the hardware side, we propose to tap into the configurablity provided by software defined radios in the sub-six GHz band.

2.2 Reinforcement Learning

Alongside with supervised learning and unsupervised learning in Machine Learning (ML), Reinforcement Learning (RL) is an area that focusing on how the agents take actions in an environment to maximize the cumulative reward. Reinforcement Learning consists of a set of environment and agent states, S, a set of actions, A, of the agent. The framework of reinforcement learning is as shown in Figure 2.2: an agent takes an action in an environment and the environment gives reward

based on the taken action and current observed state of the agent and feeds next observed state back to the agent based on the transition probability matrix $P(s'|s,a)$. The goal of a reinforcement learning agent is to learn a policy $\pi(s)$ that maximize the cumulative reward.

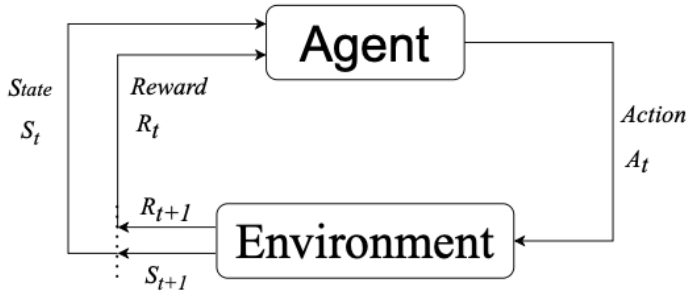

Figure 2.2: Reinforcement Learning Framework

Besides the components of RL, how to evaluate the performance of Reinforcement Learning algorithms is essential. Value function is the expected return in the state or state-action pair. There are several ways to represent the value function. The first one is on-policy value function that represents the value engendered by applying a policy π:

$$V^{\pi}(s) = E_{\tau\sim\pi}[R(\tau)|s_0 = s].$$

The second one is on-policy action-value function, which represents the value of each current action with the application of π thereafter:

$$Q^{\pi}(s,a) = E_{\tau\sim\pi}[R(\tau)|s_0 = s, a_0 = a].$$

The third one is optimal value function, which represents the value engendered by the policy that

maximizes value:

$$V^*(s) = max_\pi E_{\tau\sim\pi}[R(\tau)|s_0 = s].$$

The fourth one is optimal action-value function, that considers the return due to applying particular action currently, with application of the optimal policy thereafter:

$$Q^*(s,a) = max_\pi E_{\tau\sim\pi}[R(\tau)|s_0 = s, a_0 = a]$$

The optimal policy maximizes the cumulative reward and produces maximum return:

$$V^*(s) = max_\pi Q_\pi(s,a).$$

Thus, the optimal policy can interpreted as

$$\pi^*(s) = arg\ max_\pi Q_\pi(s,a).$$

2.2.1 Policy Gradient Method

Policy gradient theorem [1] consider how a learning agent interacts with a Markov decision process. The environment consists of state, action and reward, and its dynamics are characterized by state transition probability and expected rewards. A policy characterizes the agent's decision making procedure. In the paper, the agent's objective are formulated under two ways with function approximation. Both of the function approximations give its own definition of long-term expected reward and value of a state-action pair given a policy.

In the theorem of policy gradient, the gradient shows a relationship with start-state function derived from state-value function for any MDP. Furthermore, policy gradient with function approximation shows that the error of the approximation of value function is orthogonal to the gradient the policy parameterization to further subtract it from the policy gradient theorem. The result indicates

that policy gradient is valuable for value function approximation. The paper also shows that the policy iteration with function approximation can converge.

2.3 Proximal Policy Optimization Algorithm

Proximal Policy Optimization Algorithm [2] is a policy gradient method and aims to solve reinforcement learning problems by optimizing the policy. In the paper, clipped surrogate objective and probability ratio of policies $\gamma_t(\theta) = \frac{\pi_\theta(a_t)|s_t}{\pi_{\theta_{old}}(a_t)|s_t}$ are proposed. The loss function of the algorithm combine the policy surrogate, a value function error term and entropy bonus to ensure exploration and the algorithm contains advantage estimator required by the style of policy gradient. The Actor-Critic style of PPO has two layers of cycling. One cycle is collecting T timesteps of data and computing the advantage estimates in each actor, the other one optimizes the surrogate loss with regard to θ with epochs and minibatch size in each iterations to update parameters θ.

Experiments of Proximal Policy Optimization Algorithms were conducted on seven varied physical robotic MuJoCo environments. The work introduce six different kinds of algorithms to produce rewards on same environments. To test Proximal Policy Optimization Algorithms, A2C, A2C plus trust region, CEM, Adaptive Vanilla Policy Gradient and TRPO are introduced due to their similarity with Proximal Policy Optimization Algorithms. The result of Proximal Policy Optimization Algorithms shows the stability, reliability of trust region methods, simpler on implementation comparing with vanilla policy gradient and also better overall performance.

3. LITERATURE REVIEW

Our problem space can be considered under the framework of constrained MDPs. There has been significant work in this problem area [3, 4]. There has been much recent work on structured policies for media streaming [5, 6, 7] that often take a threshold form. The model presented in [5, 6] is a finite system for which QoE of the users is to be maximized. While [5] provides tradeoffs between startup latency and the probability of stalling, [6] maximizes QoE and discovers a threshold form of the optimal policy. Furthermore, the results suggest that the policy may be decentralized. Other work, such as [7] focuses on the heavy traffic limit and provides optimal online QoE over fading channels.

Reinforcement learning can be divided into model-based methods and model-free methods to determine optimal policies for MDP/CMDP problems. In the context of model-based learning, much recent work has focused on identifying the regret [8, 9] and the sample complexity [10] of learning algorithms. Much recent work on model-free learning has been on high-performance policy gradient approaches [11], such as Proximal policy gradient [12], Trust region methods [13] under the MDP setting, and constrained policy optimization [14] in the CMDP regime.

Reinforcement learning has also been applied to a variety of video streaming applications [15, 16], which show significant improvements over existing approaches. On the one hand, [15] describes a system entitled Penseive that trains an adaptive bit rate (ABR) algorithm to optimize client QoE. On the other hand, [16] considers the complementary question of priority access to network resources in a WiFi access point and shows that model-based and model-free RL approaches can significantly improve QoE. In contrast to these works, our setting is of a cellular base station, and our goal is to understand how sparse and dense state indformation and control may be employed to maximize client QoE performance.

4. SYSTEM DESIGN AND APPROACH

The design space for cross-layer control is large, provided by the vast operational freedom opened up by software-defined, informed infrastructures. Potentially it can provide significant performance improvements both at the user level as well as system-wide level. To realize these gains, systems must be able to navigate such large parameter spaces efficiently and adapt rapidly to changes. This is especially true in contemporary wireless environments as active users, interfering signals, and channel profiles are unique to every deployment location. Altogether, these factors point to the need for online reinforcement learning (RL) algorithms adapted to distributed systems with multiple agents.

Markov Decision Process (MDP) is the standard mathematical framework used for modeling a stochastic dynamical system. An MDP has a state space $\mathcal{X}$, action space $\mathcal{U}$, and reward function $r : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$. The stochastic model of the system is represented by the transition probability P^o, where $P^o(y|x, u)$ represents the probability of transitioning to state y when action u is taken at state x. A policy $\pi : \mathcal{X} \to \mathcal{U}$ prescribes the action to take in any given state of the system. At time step $h = 1, \ldots, H$, given the state x_h, the policy selects an action $u_h \sim \pi_h(x_h, \cdot)$, and the next state is realized according to the model, $x_{h+1} \sim P^o(\cdot|x_h, u_h)$. The value of a policy π is defined as $V_\pi(x) = \mathbb{E}[\sum_{h=1}^{H} r(x_h, u_h)|x_1 = x]$. Depending on the specific application, the value function can be formulated in the infinite horizon average form or infinite horizon discounted form. The optimal control policy π^* is the one which maximizes the value, i.e. $\pi^* = \arg\max_\pi V_\pi$. In most real-world application, the system model P^o is unknown. The goal of a reinforcement learning algorithm is to learn π^* without knowing the true model a priori.

An architecture for wireless networks for supporting the RL approach was developed in the context of WiFi in [16], which we have adapted to the cellular architecture. As shown in Figure 4.1, a learning agent based in at an intelligent controller receives video state information (buffered seconds, stalls) from multiple YouTube sessions (running on mobile devices), assigns flows corresponding to these competing sessions to service classes, and uses the a user QoE model

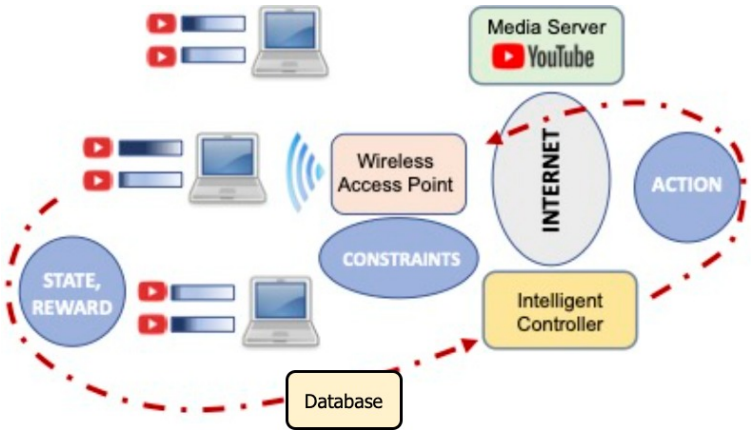

Figure 4.1: Feedback loop between state of YouTube sessions, controller actions (constrained) and end-user QoE (reward) for reinforcement learning in a media streaming application.

[17, 18, 19] to assign rewards to the resulting events. The scheduler at the base station can enhance the state through observing the backlog at the base stations as well as the channel state information, hence providing it with both an application view as well as a network view.

4.1 srsRAN: An Open-Source Platform for 4G and 5G Evolution and Experimentation

Conducting experiments over a 4G or 5G cellular network is not as straightforward as testing over a WiFi system. Several open source platforms that can be used to approximate the O-RAN-plus-RIC architecture shown in Figure 2.1 exist. Specifically, our focus is on an implementation called srsRAN [20], which can emulate a 4G or 5G system (the 5G NR version is under development), eNodeB and UE on a software defined radio (SDR) platform . srsRAN is a open source 4G and 5G (currently non-stand alone) software radio system consisting of three software elements, namely (i) srsUE, (ii) srsENB and (iii) srsEPC.

In the structure of srsRAN, srsUE is implemented in software as a 4G LTE and 5G NR NSA UE modem and is responsible for transmitting and receiving radio signals from the user equipment in the system. srsENB is implemented in software as an LTE eNodeB base station and 5G NSA gNodeB and used to connect to a core network to create a local cell. Both of srsUE and srsENB are run as applications on standard Linux-based operating system. srsEPC is a light weight implementation of a core network and provides the key core components of Home Subscriber Service,

Mobility Management Entity, Service Gateway and Packet Data Network Gateway.

In our approach, the system setup consists of numerous srsUEs, one srsENB and one srsEPC to simulate the 4G/5G environment, along with our own implementation of a RAN intelligent controller (RIC). Both of srsUE and srsENB operate over the USRP B210 software defined radios in the system. The USRP B210 provides a integrated, single-board, Universal Software Radio Peripheral platform with continuous frequency coverage from 79 MHz to 6 GHz.

4.2 srsRAN with RIC for Reinforcement Learning

The foundation of our experiments is to build a reinforcement learning compatible srsRAN system, i.e., to endow the system with the ability for RAN intelligent control (RIC) at some level. RL algorithms require states and reward as feedback of the system and make decisions as actions to control the system. Thus, our first step is to implement a controller that can receive state information at the eNB and UE and take decisions based on that. We desire this to be compatible with standardized RL implementations in the manner of openAI Gym so as to allow off-the-shelf and customized algorithm usage. Our focus is in intelligent downlink scheduling (Layer 2 MAC) and we design our controller to permit scheduling decisions at different timescales. Thus, srsUEs run YouTube sessions and generate application state and rewards. srsENB obtains state information on backlog (queues) and channel information. Based on generated states and rewards, RL algorithms at the RIC make short and long timescale decisions and srsENB executes decisions as scheduling actions.

4.3 srsRAN System Architecture for RIC integration

To evaluate the performance of srsRAN system, we plan to compare reinforcement learning approaches with other well-known and intuitive algorithms. The system design is illustrated in Figure 4.2, which shows the main components of the system. We have an srsENB and srsEPC running on a computer that hosts the USRP radio unit that represents the base station. We use a simple means of in-memory information exchange to port state information from the PHY-MAC layers to the intelligent control element, represented by OpenAI and a learning algorithm. The

algorithm also obtains application information from the application in an in-band manner (using uplink data) that is stored in an SQL database. The decisions of the algorithm are sent back to the srsENB via the same information exchange, resulting in a new schedule for the connected UEs. Such scheduling could be accomplished at the level of TTIs (1 ms timescale) or a much larger timescale of several seconds, representing near-realtime and non-realtime control. Reward information follws the same path as state information, which completes the feedback loop needed for learning and control.

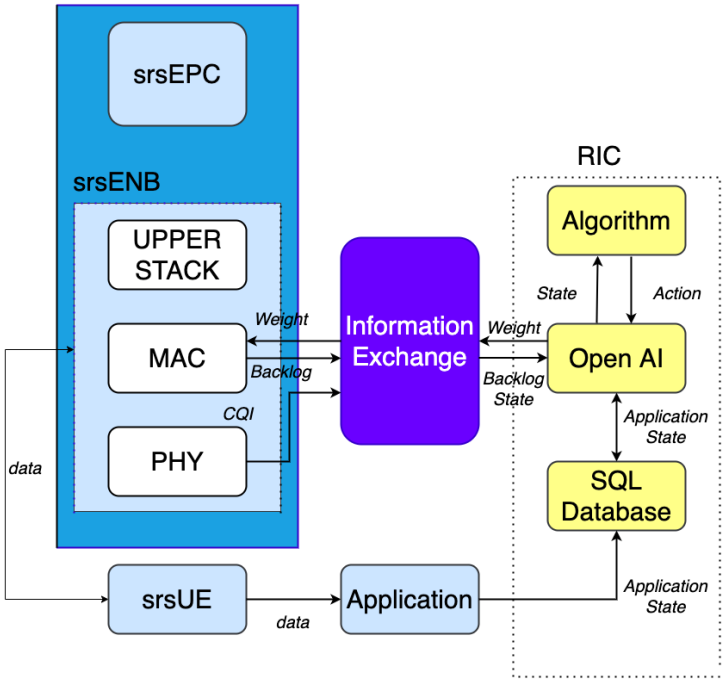

Figure 4.2: srsRAN with integration of intelligent control for PHY-MAC layers. Information from the PHY-MAC layers is ported to OpenAI along with application-level information that is directly sent (in-band using cellular uplink) from the UE to an SQL database. This state information is sent to an algorithm representing the scheduling policy, which provides weights for each connected UE, which is translated into a count of how many sub-frames to allocate to each one of them. The impact at both the application and at the backlog queues at the srsENB is then used to determine reward, hence completing the feedback loop.

5. SIMULATION-BASED TRAINING

5.1 Simulation System

We first train our system by developing a simulator that simulates the dynamics of the real system. The simulator consists of N cellular clients (UEs), whose bandwidth depends on CQI and allocated downlink sub-frames, and which require service. We typically simulate three clients as in the real environment. The controller in the simulator (modeling the scheduler in the srsENB base station) distributes different numbers of time slots in a frame with fixed total number of 10 sub-frames per frame. The throughput for each client is the number of the time slots times its own bandwidth (depending on its CQI). In the simulator, each client has a state consisting of the video buffer length, the number of stalls, QoE, which is combined with the CQI and backlog buffer of that client measured at the base station. The QoE of each client is calculated using the DQS model [17]. Note that the QoE is used as reward in the system instead of cost used in typical model by most of RL implementation.

The state of the whole system is the union states of all three clients, which is infinite since the both of video buffer length and the number of stalls can be unbounded. The action is the number of downlink time slots assigned to each client and the total number is fixed in the simulator which produces finite action space. Intuitively, training on the simulation system should be faster since the execution time for a timestep in the simulation system is faster than a timestep in real system.

5.2 Algorithms

We have five candidate algorithms that operate at different timescales and state/reward information granularity. We present all the algorithms below, as well as the training results for reinforcement learning based methods.

Round Robin (RR): The policy assignment clients receive downlink packets in each time slot in turn. Although it is computationally inexpensive and fairness is guaranteed at some level, the base station cannot distribute the time slots to client that have the best chance of increasing their

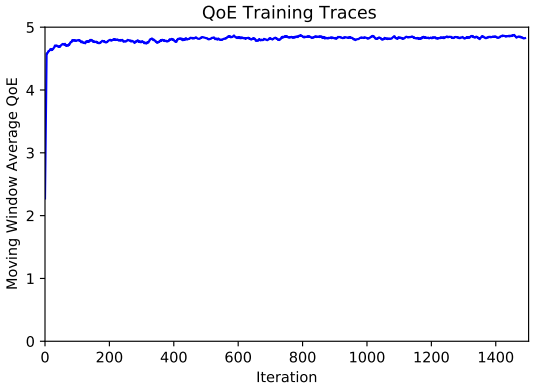

Figure 5.1: Simulation Training Curve for PPO

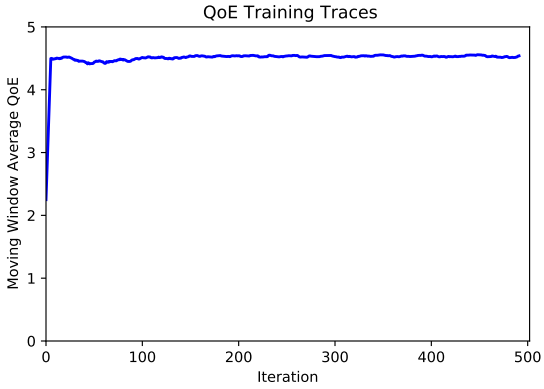

Figure 5.2: Simulation Training Curve for PO PPO

QoE. This can be performed at both short and large timescales, and we implement at the large timescale of 10 seconds.

Greedy Buffer (GB): Smooth playout of a video depends on the size of the playback buffer, which suggests that clients that have low buffer values should be prioritized. The approach assigns more downlink time slots in a subframe to the clients has lowest playback buffer. However, the approach may assign resources to wrong the clients because all clients have low buffers at the end of the video and it does not account for QoE. Weights are chosen at the large timescale of 10 seconds.

Max Weight (MW): The approach always assigns each downlink time slot in a subframe to the client that has the largest queue length in backlog buffer at the base station. This is performed on a per TTI basis.

Proximal Policy Optimization Algorithm (PPO): This algorithm requires actions and states to update its policy and using the policy based on current state to predict next state to maximize the reward. In the experiment, actions are the downlink time slots assigned to each clients in a subframe. State information can be application state, including buffer state, stall and QoE (quality of experience) and backlog states consisting of CQI (channel quality indicator) and backlog queue length. We show the training curve of this algorithm in Figure 5.1.

Partial Observable Proximal Policy Optimization Algorithm (PO PPO): The algorithm is

the previous PPO with partial observation of states. The partial observation states are only CQI and queue length, which could happen in a real world situation where application state is not available. We show the training curve of this algorithm in Figure 5.2. It performs worse than the previous case, as is to be expected.

6. REAL-WORLD EVALUATION

Our experiments are setup in a somewhat similar manner to [16] in which we have four stations. Each station consists of a computer connected to a USRP B210, with three of them acting as srsUEs, while the fourth runs the srsEPC and srsENB. Thus, we have a cellular system with thee client devices. We also host an SQL database at the station hosting the srsEPC to collect data from the applications on state and reward periodically every 10 seconds. We also collect channel quality index (CQI) and buffer backlog (BB) information on a per UE basis from the srsENB every ms. Thus, we can potentially make decisions using fine-grain base station information or coarse grain application information.

The clients all run YouTube sessions and have a list of popular 1080p YouTube videos. They play these videos in a random manner, and flush their video buffers after the completion of each video so that repat plays are seen as totally new. Our typical experiments consisted of running the sytem for three hours at a time. The collected data does not show much statistical variation over longer collection periods. Experiments are performed in a laboratory setting with stationary UEs. However, we note that the CQI of the stations are slightly different due to their location in the laboratory, i.e., the clients are not completely homogeneous.

6.1 Sparse-Reward and Sparse-Control (SRSC)

Our first set of experiments are under the Sparse-Reward and Sparse-Control (SRSC) setting, where the controller only obtains state and reward information from the applications every 10 seconds, and no information on CQI or backlog at the base station. Further, the actions are to select weights for each connected UE, also at the timescale of 10 seconds, with the weights determining how any sub-frames will be allocated to particular UE in each frame. The comparison is with the greedy buffer and round robin policies with a PPO-based policy. As expected, the PPO-based policy significantly outperforms both round robin and greedy buffer as seen in Figure 6.1. This is even more apparent in the graphs representing the CDF of QOE samples ans CDF of QoE on a per

UE basis, shown in Figures 6.2 and 6.3, where over 90% of the samples have a perfect QoE score.

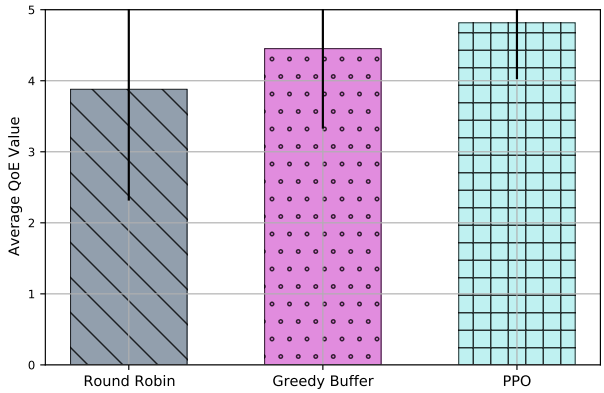

Figure 6.1: Comparison of average QoE

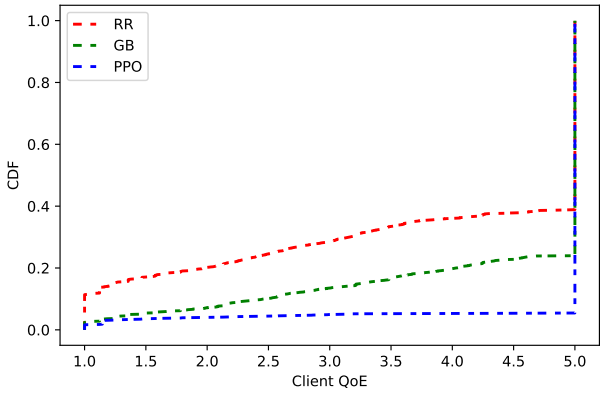

Figure 6.2: Comparison of QoE CDF

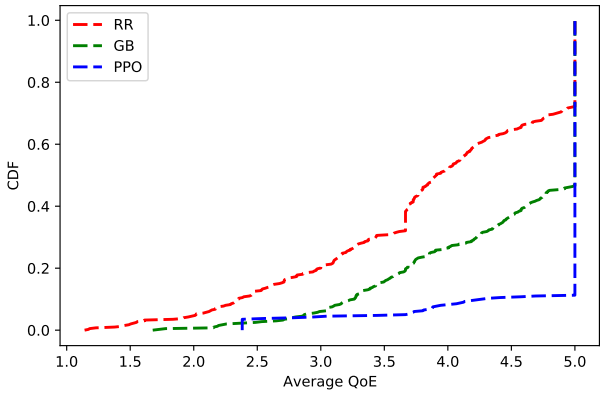

Figure 6.3: Comparison of average QoE CDF

6.2 Dense-Reward and Dense-Control (DRDC)

Our next set of experiments are under the Dense-Reward and Dense-Control (DRDC) setting, where the controller obtains information regarding the queue lengths for each UE at the base station, along with their CQI information at every 1 ms. Thus, very fine grain scheduling on a per TTI basis is possible here. We implement the well-regarded max-weight algorithm that prioritizes the UEs with the largest backlogs, as well as PPO algorithm that uses a reward that depends on maintaining small queue lengths. As is seen in Figure 6.4, both of these algorithms have very similar performance indicating that there is not much to be gained from a per backlog based scheduler. However, we also see that PPO with compete state information on the application outperforms both of these, suggesting that there is indeed much to be gained from tailoring the scheduler based on a application performance. This gain is made clearer in the graphs representing the CDF of QOE samples ans CDF of QoE on a per UE basis, shown in Figures 6.5 and 6.6.

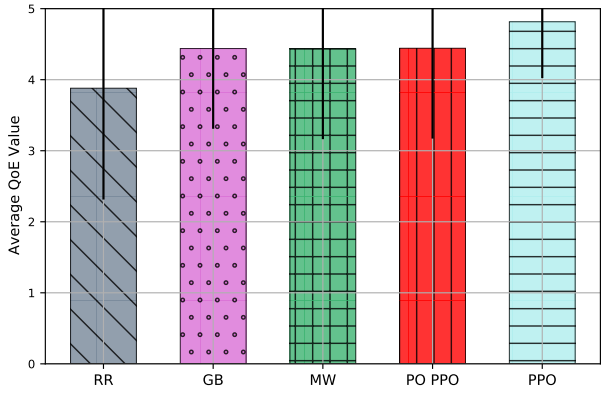

Figure 6.4: Comparison of average QoE

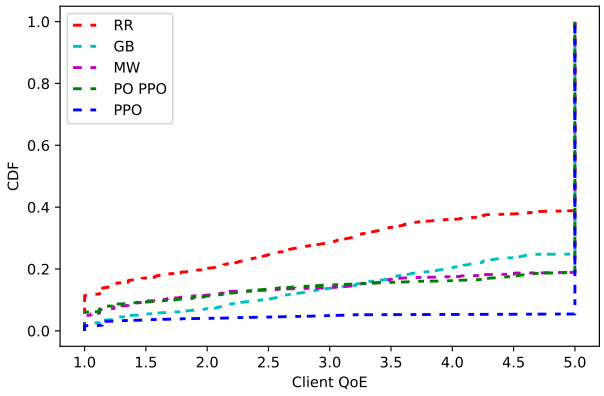

Figure 6.5: Comparison of QoE CDF

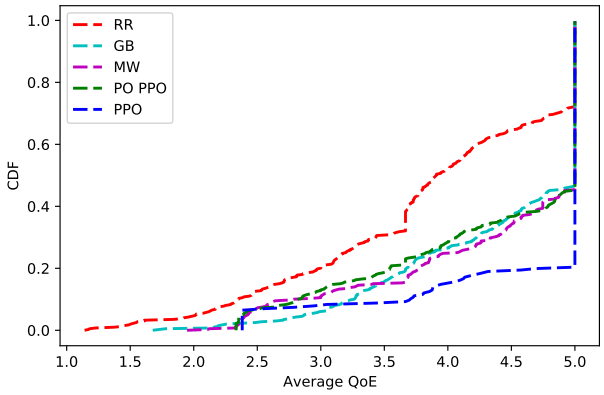

Figure 6.6: Comparison of average QoE CDF

7. CONCLUSION

In this work we explored the value of RAN intelligent control (RIC) for (downlink) medium access in a cellular setting. Our context is of high-quality media streaming and the goal was to understand if such intelligent control can provide significant benefits at the application level. We observed that the evolution of each client application is independent of each other, given the scheduling decision, which makes for a simpler model for simulation and training. We instantiated several intuitive and well-known approaches as well as those based on reinforcement learning in two settings of dense reward and control versus sparse reward and control. Interestingly, we observed that the sparse reward setting does extremely well as it is tailored to the specific application performance, whereas the dense setting with fine grain information at the cellular base station does not do as well as it is application agnostic. This strongly suggests that application-level feedback and control is likely to provide the use-case and performance benefits for RAN intelligent control.

REFERENCES

[1] S. S. Y. M. Richard S. Sutton, David McAllester, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems 12*, 1999.

[2] P. D. A. R. O. K. ohn Schulman, Filip Wolski, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[3] E. Altman, *Constrained Markov decision processes*, vol. 7. CRC Press, 1999.

[4] E. Altman, "Applications of Markov decision processes in communication networks," in *Handbook of Markov decision processes*, pp. 489–536, Springer, 2002.

[5] A. ParandehGheibi, M. Médard, A. Ozdaglar, and S. Shakkottai, "Avoiding interruptions—a QoE reliability function for streaming media applications," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 1064–1074, 2011.

[6] R. Singh and P. Kumar, "Optimal decentralized dynamic policies for video streaming over wireless channels," *arXiv preprint arXiv:1902.07418*, 2019.

[7] P.-C. Hsieh and I.-H. Hou, "Heavy-traffic analysis of QoE optimality for on-demand video streams over fading channels," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1768–1781, 2018.

[8] Y. Efroni, S. Mannor, and M. Pirotta, "Exploration-exploitation in constrained mdps," *arXiv preprint arXiv:2003.02189*, 2020.

[9] D. Ding, X. Wei, Z. Yang, Z. Wang, and M. Jovanovic, "Provably efficient safe exploration via primal-dual policy optimization," in *International Conference on Artificial Intelligence and Statistics*, pp. 3304–3312, PMLR, 2021.

[10] A. HasanzadeZonuzy, A. Bura, D. Kalathil, and S. Shakkottai, "Learning with safety constraints: Sample complexity of reinforcement learning for constrained mdps," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 7667–7674, 2021.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. 2018.

[12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[13] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[14] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International Conference on Machine Learning*, pp. 22–31, PMLR, 2017.

[15] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 197–210, 2017.

[16] R. Bhattacharyya, A. Bura, D. Rengarajan, M. Rumuly, S. Shakkottai, D. Kalathil, R. K. Mok, and A. Dhamdhere, "QFlow: A reinforcement learning approach to high qoe video streaming over wireless networks," in *Proceedings of the twentieth ACM international symposium on mobile ad hoc networking and computing*, pp. 251–260, 2019.

[17] H. Yeganeh, R. Kordasiewicz, M. Gallant, D. Ghadiyaram, and A. C. Bovik, "Delivery quality score model for Internet video," in *Proceedings of IEEE ICIP*, 2014.

[18] N. Eswara, K. Manasa, A. Kommineni, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya, "A continuous QoE evaluation framework for video streaming over HTTP," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. In press, 2017.

[19] D. Ghadiyaram, J. Pan, and A. C. Bovik, "Learning a continuous-time streaming video QoE model," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2257–2271, 2018.

[20] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith, "srsLTE: An open-source platform for lte evolution and experimentation," in *Proceedings of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization*, pp. 25–32, 2016.