Optimal Joint Scheduling of eMBB and URLLC Traffic on the 5G NR

David Black - Robotics and Control Lab, University of British Columbia, Vancouver, Canada

***Abstract*—With the advent of real time AR/VR applications, mission-critical autonomous vehicle communications, and tele-medicine, the 5G and future 6G networks have to handle a combination of high throughput requirements from enhanced mobile broadband users and extreme reliability and latency requirements from other users simultaneously. To achieve this, a number of optimal scheduling regimes for downlink transmission have been proposed, but most rely upon strong and unrealistic assumptions about the system in order to be able to approach the problem using analytical methods such as convex optimization. These assumptions render the resulting "optimal" policies substantially sub-optimal. In this paper we first implement a flexible and realistic simulation of the physical and medium access control layers of the 5G NR on which different scheduling regimes can be implemented and evaluated. We then compare the schedulers from several past papers and propose new methods that outperform the rest, with higher sum data rate on average, and lower block error rates.**

***Index Terms*—URLLC, eMBB, 5G, Scheduling, Optimization, Deep Reinforcement Learning**

I. Introduction

One of the primary future directions of communications research is towards the combination of ultra-reliable low latency communication (URLLC) and enhanced mobile broadband (eMBB) technologies to support applications which require both guaranteed low latency and high data rates [1]. One such example is real time AR/VR, especially with haptic feedback, which has promise for example in tele-medicine [2]. The visual aspect requires high quality, 3D video with minimal latency, while haptics often relies on end-to-end (E2E) delays of about 1ms to maintain stability and teleoperation transparency. However, both URLLC and eMBB alone are already challenging research problems. URLLC is difficult to achieve and requires cross-layer design since the E2E reliability and delay of a network is heavily influenced by interactions between network layers [3]. This is, however, computationally difficult, and traditional optimization techniques for this generally non-convex problem are intractable within the 0.125-1ms transmission time interval (TTI) in the 5G NR.

Instead, deep learning methods have been proposed to tackle the problem. However, the additional requirement for eMBB complicates the issue; URLLC traffic is often scheduled on networks such that it overlaps in time and/or frequency with eMBB traffic, thus sharing the same resources and leading to potential data-rate-loss in the eMBB traffic [4]. This is a problem for the aforementioned example application, where the large-volume data also requires high data rates. For such applications, it is necessary to consider optimal, multilayer joint scheduling of URLLC and eMBB traffic on the 5G NR and in future on the 6G network.

Manuscript received November 26, 2021; Corresponding author: D. Black (email: dgblack@ece.ubc.ca).

In the following subsections of the Introduction, some background will be provided before introducing the problem in more detail and outlining this paper's contribution. The subsequent sections will describe the various scheduling frameworks explored, the methods used to test and compare them, and finally the results.

A. Background

In the 5G NR, the time-frequency space is organized as a resource grid, shown in Fig. 1. Time is divided into time "slots" 1ms in length known as eMBB TTIs. These are subdivided into a number of "mini-slots" or URLLC TTIs, which are further divided into OFDM symbols. In this paper we assume $m_{ms} = 12$ mini-slots and $m_{ofdm} = 12$ OFDM symbols per slot for simplicity, so the two are equivalent. Similarly, a given network uses a portion of frequency space whose width is called the bandwidth, which varies for example between 3G, 4G, and 5G. The bandwidth is divided into many subcarriers, and contiguous groups of m_{sc} subcarriers over one time slot form a resource block (RB). This is illustrated as one row of the resource grid in Fig. 1. For this paper we assume $m_{sc} = 27$, which gives the required numbers of data bits per RB for the error correcting codes described in Section II-A2. In reality, 14 symbols per slot is standard, and the number of mini-slots and subcarrier spacing are both defined by the 5G Numerology, which allows for flexibility in meeting user quality of service (QoS) requirements [5].

One of the main requirements of 5G networks is to support gigabit per second data rates for eMBB at latencies of a few milliseconds, as well as ultra-reliable low latency communication with sub-millisecond latencies and 99.999% reliability [6]. To achieve both, a so-called superposition/puncturing framework is utilized. The RBs in a given slot are shared among the eMBB users, and the allocation remains fixed for the whole eMBB TTI. On the other hand, if URLLC data is received during a slot, it cannot wait until the next slot so it is allocated the next mini-slot, on top of the current eMBB traffic. By assigning a fraction of the power to both the eMBB and URLLC traffic or all the power to just the URLLC traffic for that mini-slot, the base station (gNB) performs superpositioning or puncturing respectively. In this manner, it guarantees the latency and reliability of the URLLC packet at the cost of the eMBB users' rates. After the eMBB TTI has elapsed, the gNB can signal to the user the mini-slot locations where superpositioning or puncturing occurred so the user can decode the signal. This pre-emption of subcarriers has been shown to be more efficient than statically separating eMBB and URLLC traffic because of the sporadic, stochastic nature of URLLC demands [7].

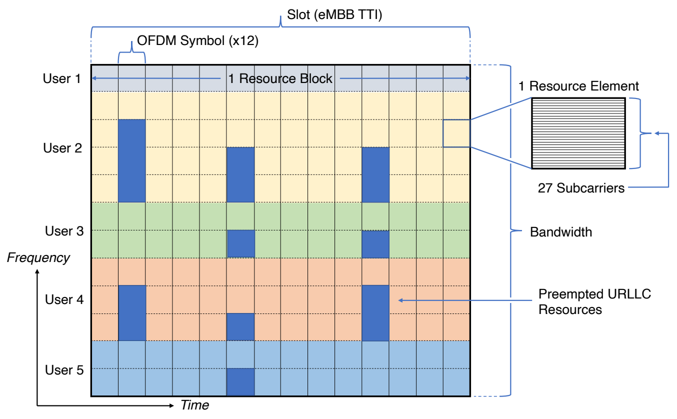

Fig. 1. Time-frequency resource grid of 1 time slot, showing resource block allocation to 5 users. The simulation parameters for the construction of the resource grid are outlined in Table I

However, the more subcarriers that are taken from an eMBB user to fulfill URLLC requirements, the more likely the decoding will fail, causing eMBB data loss. Thus, it is important to spread the "punctures" or pre-empted resources over multiple orthogonal eMBB users to avoid affecting the quality of a single user's interaction. This is a difficult problem because it gives rise to a multi-timescale optimization problem to satisfy eMBB users on slots and stochastic URLLC demands on mini-slots simultaneously. In addition, the rates depend heavily on the modulation and coding schemes, which are chosen dynamically to keep error rates below a certain level. It is therefore impossible to express the eMBB failure probability as a function of URLLC scheduling in an analytically closed form [8]. While [4] approximates the objective function using three models - a linear function, a convex function, and a threshold function - [8] argues that this is unsuitable and proposes instead a model free deep deterministic policy gradient (DDPG)-based multiplexer dubbed DEMUX. This substantially outperforms the analytical approximations.

Yin et al. [9] formulate a dual objective function to maximize proportional fairness in eMBB scheduling while supporting the URLLC preemptions. They solve a mixed integer linear programming problem using convex relaxation and a greedy algorithm, achieving two different solutions with good performance on simulations. However, their simulations do not include the factors such as modulation and low density parity check (LDPC) coding described below, and thus tend to agree with the simplified analytical models which may not be realistic. On the other hand, Karimi et al. [10] take into account a number of 5G physical and MAC layer subsystems, including the physical downlink control channel (PDCCH), the physical downlink shared channel (PDSCH), and hybrid automatic repeat request (HARQ) retransmission. However, they still do not model the coding scheme.

Further deep reinforcement learning-based approaches are presented in Alsenwi et al. [11], and Saggese et al. [12]. While Saggese uses a proximal policy optimization (PPO) algorithm and shows that URLLC latency requirements are never violated, Alsenwi argues that DRL alone cannot guarantee QoS satisfaction, and thus proposes a two-staged, mixed optimization and DRL method. This approach not only considers eMBB rate maximization with URLLC constraints, but also takes into account the variance in past eMBB data rates to quantify the reliability and risk in the eMBB transmissions due to the URLLC interference. Saggese's approach does not address the eMBB scheduling problem at all, but focuses entirely on the URLLC preemption.

One predominant reason why analytical approximations perform poorly is that it is not possible to model precisely the LDPC encoding used for forward error correction in the 5G NR [13], making it difficult to model the likelihood of errors in the decoding. While it is possible to obtain curves for the expected block error rates resulting from the LDPC experimentally, it is infeasible to have a table of these for every possible subcarrier distribution among eMBB users, volume of URLLC traffic, modulation and coding scheme (MCS), channel condition, etc.. Hence, DRL performs well due to its ability to generalize beyond the training. However, the authors of [8] experienced convergence issues with the algorithm and had to use domain knowledge and approximations based on the scheduling method of eMBB. While these may be a reasonable approximation for one network state or configuration, real networks are not stationary and may differ substantially, leading to a model mismatch [1]. Furthermore, training is very time consuming. The largest problem, however, is that DRL has no ability to build in QoS requirements, so there is no guarantee of satisfying URLLC constraints. While Huang et al. attempt to rectify this by using two post-processing functions for the output of their DRL agent, the issue remains problematic [8]. Thus, DRL is also not a perfect solution.

B. Contribution

This paper performs an unbiased and realistic comparison of the various scheduling methods and proposes two new ones by providing the following contributions:

- A simulation environment built by the author on MATLAB's communications and 5G toolboxes which allows fast and realistic simulation of the 5G physical (PHY) and medium access control (MAC) layers and allows easy implementation of custom schedulers.
- An analytical comparison of the accuracy, effectiveness, and viability of the various scheduling algorithms proposed in literature, which are outlined in the previous section.
- A comparison, using the simulation environment, of a subset of the proposed scheduling algorithms.
- Two alternative scheduling schemes that achieve improved results in the simulation.

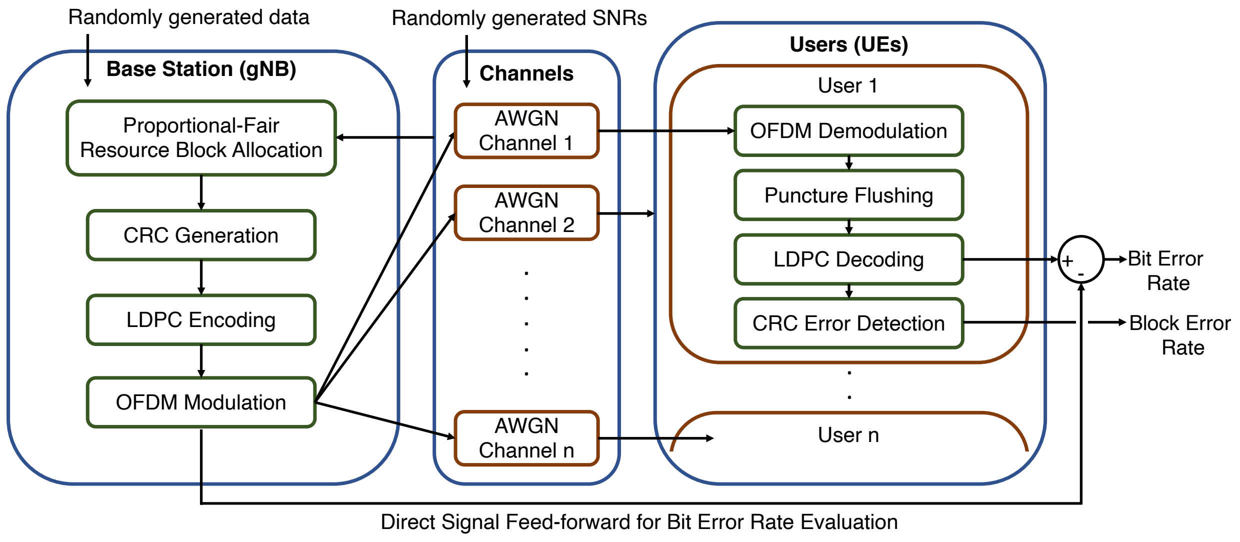

Fig. 2. Block Diagram of the implemented simulation

II. Methods

A. The Simulation

In order to evaluate different scheduling policies effectively, an accurate yet simple and fast simulation was needed to overcome the impossibility of analytical solutions [8]. To this effect, a simulation was designed on MATLAB (Mathworks Inc., Massachusetts, USA), using the communication and 5G toolboxes. The overall structure of the simulation is shown in Fig. 2, and some key parameters are listed in Table I

TABLE I
Key Simulation Parameters

Parameter	Symbol	Value
Number of Users	n	8
Bandwidth	BW	30 MHz
Carrier Frequency	f_c	4 GHz
Number of RBs	m_{RB}	50
Subcarriers/RB	m_{sc}	27
OFDM Symbols/Slot	m_{ofdm}	12
Minislots/Slot	m_{ms}	12
User distances from gNB	d_i	Unif$[50, 100]$

At a high level, n users are generated at uniformly distributed distances, $d_i \sim$ Unif$[50, 1000]$ metres from the base station (gNB). Each user is assigned a communication channel from the gNB with a certain noise power spectral density N_0^i (See Section II-A4). Based on the channel qualities, the gNB performs optimal fair allocation of the available bandwidth to the users (Seciton II-A1). Employing the infinite backlog assumption, it is assumed that each user wants to receive as much data as the gNB can possibly send. Once resource blocks have been assigned to the users, the channel qualities are again used to determine the modulation and coding scheme (MCS) used on each resource block, which dictates how much data can be sent to each user in that slot. This quantity of data is generated uniformly randomly and processed (Section II-A2) before being sent over the respective additive white Gaussian noise (AWGN) channel to the user (section II-A4). Before transmission, however, some RBs in some mini-slots are preempted by URLLC traffic, as determined by the Poisson-random arrival of URLLC packets and the URLLC scheduler. After transmission, each user performs its own signal processing to demodulate and decode the signal, and finally the error rates and data rates are determined (Section II-A6).

1) eMBB Scheduling: Though the eMBB scheduling is not the primary goal of this work, it is important to have a realistic and representative setup. Thus, the simulation performs a proportional fair scheduling based on the calculated MCS levels of each user.

Rather than trying to satisfy a certain data requirement for each user, we assume that each user has an infinite backlog of data waiting to be received, so the sum data rate to all users can be maximized without considering inefficiencies caused by over-allocation of resources to a certain user who does not need them. Thus, to efficiently and equitably allocate the resources, we instead look at the channel qualities to ensure users with poor channel quality are given more bandwidth so they too can receive their data. To this end, we employ a proportional fair scheduling algorithm modified from [14].

We make a further simplification in describing the transmission rate as the concave, increasing function given by the Shannon channel capacity, equation 1 in [14]:

$$r_i(w_i) = w_i \log_2 \left(1 + \beta \frac{p h_i}{N_0^i w_i}\right) \tag{1}$$

where p is the transmit power, which is assumed to be equal for all users, h_i is the channel gain, and w_i is the assigned bandwidth. The parameter $\beta \in [0, 1]$ addresses the discrepancy between the theoretically achievable Shannon capacity and the practically possible rate governed by the MCS [15]. Because LDPC coding approaches the Shannon capacity, however, we choose a value relatively close to 1: $\beta = 0.75$. Now we maximize the sum of the logs of the rates using fmincon in MATLAB, to obtain the proportional fair allocation:

$$\begin{aligned}\underset{w_i}{\text{argmax}} & \sum_{i=1}^{n} \log\left(w_i \log_2\left(1+\beta\frac{ph_i}{N_0^i w_i}\right)\right)\\ \text{subject to } & w_i = \bar{w}_i W_{RB}\\ & \sum_{i=1}^{n} \bar{w}_i = m_{RB}\\ & \bar{w}_i \in \mathbb{Z}_+\end{aligned}$$

The w_i values are continuous bandwidths, but in reality the frequency domain is subdivided into resource blocks of finite bandwidth. Thus, the allocated bandwidths are discretized into numbers of RBs, $\bar{w}_i$, such that $w_i = \bar{w}_i W_{RB}$ where W_{RB} is the bandwidth of a single RB, given by the product of subcarrier spacing and the number of subcarriers per RB. The sum of the number of resource blocks must equal the total available number set for the simulation. In fact, allocation resolution is usually limited to “resource block groups” (RBGs) of several RBs, but for this simulation we assume the number of RBs per RBG is 1. To speed up the solution, we relax the problem to exact linear programming and afterwards discretize the bandwidths by rounding them to the nearest number of RBs while maintaining the sum constraint. This is acceptable since the exact eMBB scheduling optimality is not the goal of this work.

$$\text{SNR} = \frac{ph}{N_0 w} \tag{2}$$

With the bandwidths assigned to each user, we can then find the actual SNRs of each channel using equation 2, and use these to determine the assigned MCS such that the block error rate (BLER) is less than 0.1, which is standard in literature [8]. This mapping was determined by experimentation with the simulation. The SNR was increased for each given MCS until the BLER exceeded 0.1, and this value was chosen as the cutoff. The resultant MCS strategy is shown in Table II. As an explanatory example, a channel with SNR= 24 would be assigned the MCS associated with the next lowest SNR value found in the table, so 21, which gives a QAM16 modulation with code rate 3/4.

TABLE II
SNR CUTOFFS FOR THE 12 POSSIBLE COMBINATIONS OF MODULATION TYPE (QPSK, QAM16/64) AND CODE RATE (1/2, 2/3, 3/4, 5/6).

	1/2	2/3	3/4	5/6
QPSK	4	6	10	16
QAM16	18.5	20	21	25
QAM64	26	26.5	28	30

2) Transmitter Signal Processing: Given the number of RBs and their corresponding MCS allocated to each user, we know the number of data bits that can be sent over each RB, as explained in this section. This number of bits is generated by sampling from Unif$\{0,1\}$ and appending a 16-bit cyclic redundancy check (CRC-16) sequence. This is used for error checking in the receiving user's signal processing.

The data stream is then encoded using Low-Density Parity-Check (LDPC) coding, a linear error-correcting code that allows for error-free transmission over noisy channels. LDPC involves computing another sequence of bits that are again appended to the message. These bits are chosen such that the modified message falls in the nullspace of a certain sparse parity-check matrix. The matrices used in this simulation are taken from cyclic permutation matrices in the IEEE 802.11n standard [16] for three specific block lengths. Depending on the length of the appended codeword, the encoding can be more or less robust to noise. This is characterized by the code rate, R, which describes the ratio of data bits to total bits in the message. In the standard, R can take values 1/2, 2/3, 3/4, or 5/6. For example, a message with block length (i.e. total capacity of the RB) 1296 and coding rate 2/3 sends $1296 \times 2/3 - 16$ data bits, 16 CRC bits, and $1296 \times 1/3$ LDPC parity bits. Thus, the data rate depends on the code rate, which in turn depends on the channel SNR as given in Table II.

After CRC calculation and LDPC coding, the data undergoes OFDM modulation. Similar to the code rate, the modulation scheme also determines how many bits can be sent per RB. For QPSK modulation, pairs of bits are encoded as complex numbers with magnitude and phase forming a grid, or constellation, of four points on the complex plane. Each OFDM symbol carries one such number, meaning that each symbol carries two bits. For QAM16 modulation, the grid of complex numbers is 4x4, so each symbol carries 4 bits. Similarly, for QAM64, each symbol carries 6 bits. An example constellation is found in Fig. 3. By modulating the bits in this way, noise in the signal does not immediately flip bits, but rather just causes a slight spread around the original point, as seen in the figure. By choosing higher modulation schemes, however, the points become closer together and thus more sensitive to noise. Therefore, the gNB chooses the modulation scheme based on the SNR to keep the BLER below 0.1, as shown in Table II. Modulation is the final step of signal processing before the data is sent.

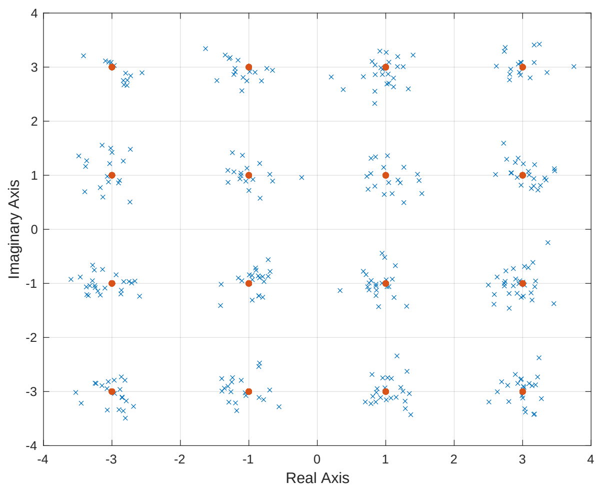

Fig. 3. QAM16 modulation constellation showing ideal amplitude and phase locations (orange) overlaid with noisy data (blue)

3) URLLC Puncturing Scheduler: Before the prepared data can be sent over the channel, URLLC packets may arrive and preempt some of the resources. In particular, URLLC packets are taken to be 432 bytes in size, which, with QPSK modulation and a 1/3 code rate, takes up 432 subcarriers or 16 RBs. In testing, this number is varied from 4 RBs to 20 to simulate different levels of URLLC traffic. The packets arrive as a Poisson-random process with $\lambda = 0.5$ in each mini-slot. A scheduler, described in Section III, chooses which RBs to preempt, replaces their data with the URLLC data, and returns a puncture map indicating the punctured blocks to the user for decoding. The scheduler attempts to minimize the loss in rate caused by the puncturing by employing several different strategies which are tested in Section III-E.

4) Channel: The encoded, modulated, and potentially punctured data is then sent over channels from the gNB to the users. Each user's channel is modeled as an additive white Gaussian noise (AWGN) channel and is assigned a noise power spectral density N_0^i at the start of the simulation. The value N_0^i of the i^{th} channel is sampled randomly from a normal distribution with $\mu_i = 6\log(1050 - d_i) - 18$ and $\sigma_i = 2$, where d_i is the distance of user i from the gNB. Thus, users further from the gNB are more likely to have poorer channel conditions, and the resultant SNR values range roughly from 2.5 to 30, which were found experimentally to cover the full range of MCS (See Table II).

This method was employed because it provided a convenient way to quantify the channel quality by a single value, the SNR, which could be used to determine the MCS. It is also more general and does not require a specific description of the geography, building density, weather, etc. of the network area, which is unlike Rayleigh or Rician multipath fading models [17]. However, it does have limitations. There is no consideration for Doppler shift, multipath interference, or other effects that could induce a phase shift. As seen in Fig. 3, the AWGN model provides some point spread, but no rotation of the constellation, nor magnitude scaling on average.

5) Receiver Signal Processing and Error Checking: The receiver signal processing chain is effectively the same as the transmitter signal processing but in reverse, and with the addition of a preemption flushing step. The received signal is first demodulated using the known MCS. Preemption flushing comes next, which ensures that the preempted bits do not propagate their error throughout the LDPC decoding process. These bits, which are flagged by the preemption map, are nullified before the data is LDPC decoded and finally checked for errors using the CRC checksum. If the CRC fails, the block is taken to be corrupted, so all the contained data is lost, and the data rate for that user decreases for that slot.

6) Time Variation: Practical communication networks are not perfectly constant. Instead, they experience some variation with time. This is modeled in the simulation by adding a δN_0^i to the noise power spectral density at each iteration (each slot), to introduce some variation. The δN_0^i is given by $N \times (0.25 \times 10^{-8})$dBm/Hz where $N \sim \text{Unif}[-1, 1]$. In addition, the moving average data rate, $\bar{R}_i(t)$ for each user is kept track of to inform some URLLC scheduling methods. The calculation ascribes more weight to recently achieved data rates by setting

$$\bar{R}_i(t+1) = \left(\frac{1}{T}\right)\bar{R}_i(t) + \left(1 - \frac{1}{T}\right) r_i(t) \tag{3}$$

Where T is the averaging period in number of mini-slots.

III. Existing Scheduling Methods

In the following sections, I will outline several optimization algorithms proposed in the literature, which were introduced in Section I-A. I will then implement them on the described simulation, and compare their performance. Unfortunately, due to time constraints and the substantial work involved, I will not implement the DRL-based approaches.

A. Anand et al. [4]

In this paper, the authors propose two separate URLLC traffic allocation policies, the Resource Proportional (RP) and Threshold Proportional (TP) placements. The RP scheduler assigns URLLC traffic to the RBs of user m proportional to the amount of resources allocated to user m by the eMBB scheduler. i.e. let γ_m^t be the fraction of URLLC traffic in slot t that is assigned to an RB belonging to user m, and let ϕ_m^t be the fraction of the total bandwidth that is allocated to user m. In the RP regime we have:

$$\gamma_m^t = \phi_m^t \tag{4}$$

For example, if user 1 is allocated half of all RBs, then half of the URLLC traffic is assigned to user 1's RBs. Interestingly, it is shown in the paper that over the long term, this is equivalent to uniformly randomly distributing the URLLC preemptions over the resource grid.

The threshold model approximates a threshold URLLC preemption fraction beyond which the entire RB of data is lost. In particular, let Γ_m^t be the fraction of user m's RBs that are preempted by URLLC traffic in slot t. The authors define a threshold function $\theta(\Gamma, \alpha)$ such that:

$$\theta(\Gamma, \alpha) = \begin{cases} 1 & \Gamma < \alpha \\ 0 & \Gamma \geq \alpha \end{cases} \tag{5}$$

Where the cutoff for user m, $\alpha_m^t \in [0, 1]$, depends on the channel quality of the user during slot t. The TP scheduler then assigns URLLC RBs according to this threshold such that

$$\gamma_m^t = \frac{\alpha_m^t}{\sum_{m'=1}^{n} \alpha_{m'}^t} \tag{6}$$

In the paper, the authors simply assign $\alpha_m = 0.5$ to half the users and $\alpha_m = 0.7$ to the other half. Both the TP and RP methods as described here are tested in the simulation.

B. Yin et al. [9]

The authors solve a mixed integer linear programming problem using convex relaxation and a greedy algorithm to maximize a dual objective function which addresses proportional fairness in eMBB scheduling while satisfying the URLLC requirements. Beyond the scenario described in this paper so far, the authors consider an additional scenario where the URLLC traffic actually has less stringent latency requirements, for example of 10ms, and thus does not need to be scheduled immediately in the next mini-slot. This makes it easier to maximize eMBB data rates while achieving the URLLC requirements. However, in this paper we focus on the scenario where this is not possible, so we ignore this part of the Yin paper.

Yin approximates the rate, $r_m(t)$ of eMBB user m in mini-slot t in terms of the rate that user would have achieved ($\tilde{r}_m(t)$) scaled by the fraction of resource elements that are preempted from the user by URLLC traffic in that mini-slot. In syntax convenient for our simulation, this is expressed in equation 7.

$$r_m(t) = \left(1 - \frac{\sum_{k=1}^{m_{RB}} y^k(t) x_m^k(t)}{\sum_{k=1}^{m_{RB}} x_m^k(t)}\right) \tilde{r}_m(t) \tag{7}$$

Where $y^k(t) \in \{0,1\}$ is 1 if RB k is preempted by URLLC traffic in mini-slot t and 0 otherwise, and $x_m^k(t) \in \{0,1\}$ is 1 if RB k is allocated to eMBB user m, and 0 otherwise. In our simulation, this is given and constant for the duration of one time slot. The value of $\tilde{r}_m(t)$ is also given by the eMBB allocation and MCS. To maximize the long term proportional fair allocation, the average rate over time should be used rather than this instantaneous rate for a single mini-slot. Hence we consider $\bar{R}_m(t)$, the average rate of user m over T mini-slots, as given in equation 3. Thus we can state the integer programming problem at each mini-slot:

$$\begin{aligned} \underset{y^k}{\text{maximize}} \quad & \sum_{m=1}^{n} U(\bar{R}_m) \\ \text{Subject to} \quad & \sum_{k=1}^{m_{RB}} y^k(t) = m_{llc} \\ & y^k(t) \in \{0,1\} \end{aligned}$$

Where $U(\phi_m) = \log \bar{R}_m$ for proportional fairness. Note that additional implicit constraints regarding the number of RBs that can be allocated to are covered by the binary nature of x and y and the lengths of the vectors. Additionally, the constraint that each RB must be allocated to exactly one user is automatically satisfied since x is given. The only explicit constraint, therefore, is that the total number of URLLC packets must equal the prescribed number, m_{llc}, given by the Poisson-random process in the simulation.

To solve this, Yin proposes to use a convex relaxation using the concave convex procedure (CCCP). The the binary constraint can be reformulated as

$$y^k(t)(1 - y^k(t)) \leq 0, \; y^k \in [0,1] \tag{8}$$

For all $k \in \{1, \dots, m_{RB}\}$. It is possible to then formulate the problem as follows, which is shown to be equivalent:

$$\begin{aligned} \underset{y^k}{\text{minimize}} \quad & \sum_{m=1}^{n} -U(\bar{R}(m)) + \zeta \sum_{k=1}^{m_{RB}} y^k(t) \left(1 - y^k(t)\right) \\ \text{Subject to} \quad & \sum_{k=1}^{m_{RB}} y^k(t) = m_{llc} \\ & y^k \in [0,1] \end{aligned}$$

The method for solving this iteratively is given in Algorithm 1 of the paper and is denoted the CCCP Method. Due to concerns about speed, the authors also proposed a greedy algorithm which reaches a slightly different solution. This simply iterates through the RBs needed by incoming URLLC traffic and assigns them one-by-one to the remaining already-allocated RB which has the least effect on the aggregate proportional fair rate. Let x_m be the number of RBs allocated to user m and Δx_m be the number of RBs of user m that are preempted. The proportional fair utility is given by modifying equation 7:

$$P = \sum_{m=1}^{n} \log\left(\left(1 - \frac{\Delta x_m}{x_m}\right) \tilde{r}_m\right) \tag{9}$$

To find the effect of preempting an RB from user m, we must determine

$$\frac{\partial P}{\partial(\Delta x_m)} = \frac{1}{\Delta x_m - x_m} \tag{10}$$

Thus, for every RB to be preempted in turn we choose the user from which to take the RB by minimizing the absolute value of equation 10, whose denominator can never be greater than 0. We denote this as the Greedy Method. Both Yin methods were implemented in our simulation.

C. Karimi et al. [10]

This paper aims to maximize proportional fairness for the eMBB users and URLLC rate by dynamically allocating both at once. This is fundamentally different from the puncturing methods described otherwise in this work, as it does not fill the bandwidth with eMBB traffic and then schedule URLLC separately. It also relies on HARQ retransmission of failed packets, which could be problematic for achieving the required URLLC latencies. The problem formulation is shown below:

$$\begin{aligned} \max_{b^j_{u/k}} \quad & \sum_{u=1}^{n_{llc}} \alpha_u R_u^{llc} + \sum_{m=1}^{n_{mbb}} \log \bar{R}_m^{mbb}, \\ \text{Subject to:} \quad & \sum_{u=1}^{n_{llc}} b_i^j + \sum_{m=1}^{m_{urllc}} b_m^j \leq 1, \forall j \in \{1, \cdots, m_{RB}\}, \\ & \sum_{j=1}^{m_{RB}} b_{u/m}^j \geq \min\left(R_{u/m}^{llc/mbb}, 1\right) \cdot b_{u/m}^{\min}, \; \forall u, m, \\ & R_i^{llc} \leq Q_i^{llc} \quad \forall i, \\ & b_{u/m}^j \in \{0,1\} \quad \forall u, m, j, \end{aligned}$$

Where the binary variable b_i^j is 1 if the jth RB is allocated to the ith user, and 0 otherwise. The users are split into n_{llc} users with URLLC traffic and rates R_u^{llc}, and n_{mbb} users with eMBB requirements and average throughput $\bar{R}_m^{mbb}$. The objective is a weighted sum (with weights $\alpha_u \in [0,1]$) of the URLLC sum rate and the proportional fair eMBB allocation objective function. The first constraint ensures that each RB is allocated to only one user. The second ensures that each user is allocated enough RBs to meet the minimum control channel overhead, $b_{u/m}^{min}$. Lastly, the third constraint keeps the assigned rate for a URLLC user below the queued amount of data, Q_u^{llc}, because the authors do not employ the infinite backlog assumption for the URLLC users since the amount of data is usually relatively small compared to eMBB. The algorithm for URLLC scheduling is outlined in Algorithm 1 of the paper.

This method is problematic because it assumes eMBB and URLLC traffic can be scheduled at the same time scale, which is exactly what we cannot assume, and what makes this problem so difficult. Scheduling eMBB traffic at every mini-slot is not possible because the overhead is too high, whereas scheduling URLLC traffic only once in a slot is also not possible as this would not satisfy the latency requirements. Thus, this method is incompatible with the implemented simulation and will not be tested quantitatively, as the results would not be comparable to other methods.

D. Contributions

In this work I implement two modest improvements of methods described above. The first is a modified, MCS-aware threshold proportional scheduler where the threshold, α_m^t, is taken to be a function of the channel quality of user m instead of being arbitrarily assigned. In particular, recall from Table II that a user with a certain SNR, s_m, is assigned an MCS corresponding to the next lowest SNR, s_- in the lookup table. Suppose the next larger SNR in the table is s_+. This means that the closer s is to s_+, the more gap there is between the SNR of the channel and the cutoff SNR for a BLER less than 0.1 at the given MCS. Hence, the closer s is to s_+, the lower the probability of transmission failure. The largest gap between SNR values in the table is 6, so to have a maximum α value of about 0.7 as it is in [4], we set

$$\alpha_m^t = \frac{s_m - s_-}{9} \tag{11}$$

This method will be referred to as MCS-aware threshold proportional, or MTP. A second algorithm is based on the packetized proportional fair method in [18], and is similar to the CCCP approach. As in Yin's CCCP method, let $\tilde{r}_m(t)$ be the rate user m would achieve in mini-slot t without URLLC preemption. This is unique to every user as it is the number of RBs allocated to the user times the bits per RB for the user, which is determined by the MCS, divided by the number of mini-slots per slot. Now rather than considering the actual RB locations as in equation 7, we follow the syntax from equation 9 and consider instead the number of RBs and preemptions per user. The instantaneous throughput of user m is given by:

$$r_m(t) = \tilde{r}_m \left(1 - \frac{\Delta x_m}{x_m}\right) \tag{12}$$

The moving average rate, $\bar{R}(t)$ over the last T mini-slots is given by equation 3. Now from [18], we can approximate proportional fair allocation in this packetized model by maximizing

$$\sum_{m=1}^{n} \frac{r_m(t)}{\bar{R}_m(t)} \tag{13}$$

However, observe that maximizing equation 12 is the same as minimizing $\frac{\tilde{r}_m \Delta x_m}{x_m}$. Thus we reformulate the problem as follows:

$$\begin{aligned} \underset{\Delta x_m}{\text{minimize}} \; & \sum_{m=1}^{n} \frac{\tilde{r}_m}{x_m \bar{R}_m} \Delta x_m \\ \text{Subject to} \; & \sum_{m=1}^{n} \Delta x_m = m_{llc} \\ & \Delta x_m \in 0, 1, \ldots, x_m \;\forall m \in \{1, \ldots, n\} \end{aligned}$$

This is a mixed integer linear program where $\frac{\tilde{r}_m}{x_m \bar{R}_m}$ is a constant that can be calculated for each user in each mini-slot. The problem is simple enough to be calculated efficiently using CVX in MATLAB. This method is referred to in testing as the Packetized MCS-aware Proportional Fair (PMPF) method.

E. Testing

As an initial sanity check to ensure the functioning of the eMBB scheduler, the MCS selection process, the channel model, and the reliance of throughput on MCS level, the simulation was run without URLLC data at increasing levels of hard-coded signal to noise ratio. The result, shown in Fig. 4, displays the expected upward trend with SNR, which indicates that the simulation components are working as desired.

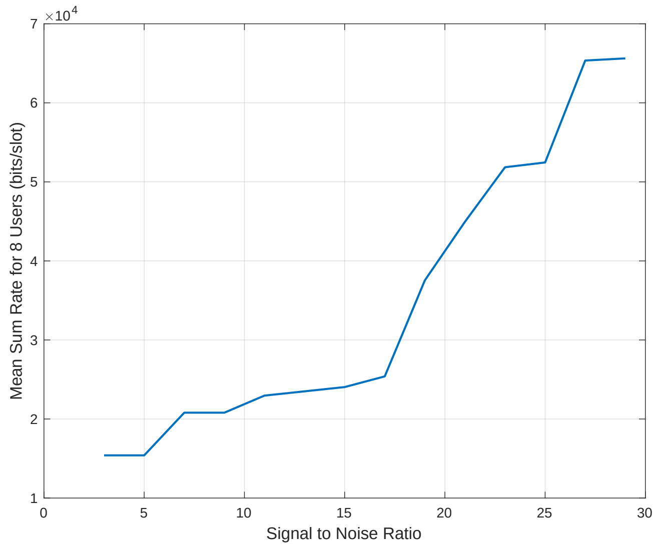

Fig. 4. Sum data rate at various levels of channel signal to noise ratio. This shows an increasing trend, as expected, meaning the MCS selection process and its effect on the throughput in the simulation work.

To evaluate the effectiveness of the various schedulers, several tests were carried out. Recall that the schedulers in question are the RP and TP [4], the CCCP and Greedy [9], and the MTP and PMPF (Section III-D). Every scheduler was simulated 100 times each at several levels of URLLC traffic. The sum data rate of all eMBB users, the block error rate, and the bit error rate were each averaged over the 100 runs at every traffic level, and then plotted. The results are shown in Section IV.

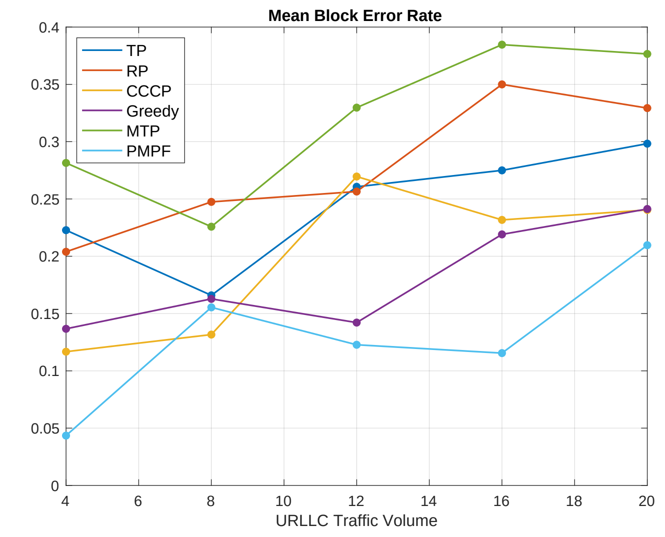

Fig. 5. Block Error Rate versus URLLC Traffic Comparison between Joint Scheduling Algorithms

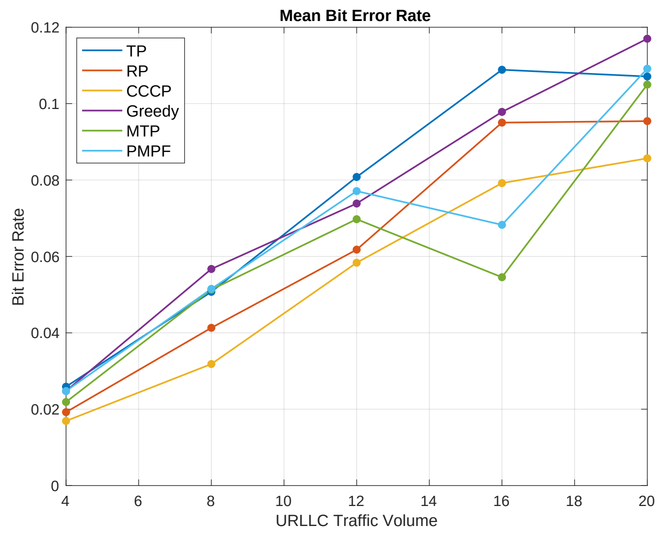

Fig. 6. Bit Error Rate versus URLLC Traffic Comparison between Joint Scheduling Algorithms

The block error rate describes the ratio of RBs that contain a CRC error versus the total number of RBs in a slot, as described in Section II-A6. Since the CRC error can be caused by a single corrupt bit, however, this is an overly sensitive test and gives rise to very coarse results since each user only has a few allocated RBs. Thus, the final signal is also compared bitwise to the original signal emitted by the gNB before preemption or channel fading to find the actual bit error rate in this simulation, which is often very low even though the CRC failed.

This bit error rate and the block error rate are used to evaluate the performance of the URLLC traffic scheduler, and are plotted in Fig. 6 and 5 respectively. The primary performance measure, however, is the sum data rate achieved by the users (see Fig. 7). This is strongly affected by the error rates as errors lead to entire blocks of data being dropped.

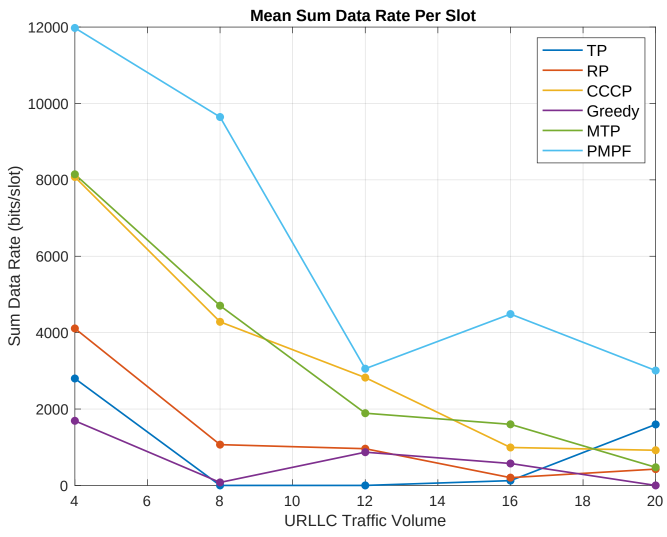

Fig. 7. Sum Data Rate versus URLLC Traffic Comparison between Joint Scheduling Algorithms

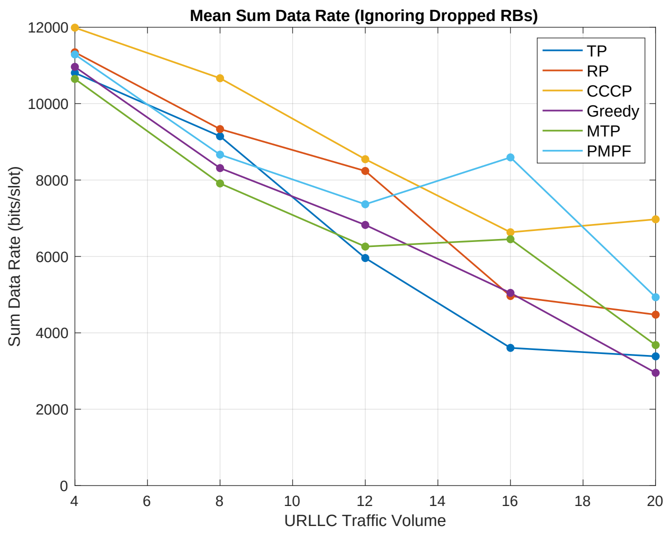

Fig. 8. Sum Data Rate versus URLLC Traffic Comparison between Joint Scheduling Algorithms: Here, RBs with corrupt bits are kept, and only the error bits themselves are thrown out when calculating the rate.

On the other hand, considering that eMBB traffic may often consist of applications such as high-quality video streaming where individual corrupted bits may cause a single pixel to be wrong, it is conceivable that small bit error rates are actually acceptable. Thus, a second sum data rate is also considered in which all RBs are kept, and only the corrupt bits themselves are thrown out (see Fig. 8). This results in much more linear loss in rate as URLLC traffic increases, and is perhaps a more precise description of how many errors are actually occurring.

IV. Results

The results of the tests are shown here. Fig. 6 shows the bit error rate, Fig. 5 shows the block error rate, Fig. 7 is the sum data rate, and Fig. 8 is the sum data rate without dropping entire RBs in which the CRC failed.

It is worth noting first that in Figures 5 and 6, which show error, low numbers indicate good performance whereas in Figures 7 and 8, which show throughput, higher numbers are better. As expected, we see an increase in error and a decrease in eMBB data rate as URLLC traffic is increased. Looking at the plots, it is difficult to pick out a clear best performer, but the PMPF method has the highest sum data rate, some of the highest bitwise data rates at higher URLLC traffic levels, and the lowest block error rate. The CCCP method, which follows a very similar algorithm, has similar performance, with higher bit-wise data rate and lower bit error rate. The greedy approach and the two threshold proportional ones, on the other hand, have higher errors and lower rates.

V. Conclusion

This paper has compared the effectiveness of several joint eMBB/URLLC schedulers for the 5G NR through the implementation of an accurate simulation of the 5G PHY and MAC layers. Comparable performance was found for most of the methods, though a packetized MCS-aware proportional fair algorithm proposed in Section III-D had the highest sum data rate and lowest block error rate at most volumes of URLLC traffic. A similar approach proposed by Yin et al. and based on a convex relaxation of the proportional fair allocation problem [9] was perhaps second best.

In further studies, the performance of deep reinforcement learning-based methods should be compared to these results. In addition, this work and those cited in it have considered only an orthogonal multiple access (OMA) scheme with URLLC puncturing which completely overwrites that section of eMBB data. However, future work could explore a non-orthogonal multiple access (NOMA) approach where the URLLC data is superposed on the eMBB data and each is decoded using successive interference cancellation (SIC) to avoid packet loss.

References

[1] C. She, C. Sun, Z. Gu, Y. Li, C. Yang, H. V. Poor, and B. Vucetic, "A tutorial on ultrareliable and low-latency communications in 6g: Integrating domain knowledge into deep learning," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 204–246, March 2021.

[2] D. Black, Y. O. Yazdi, A. H. Hadi Hosseinabadi, and S. Salcudean, "Human teleoperation - a haptically enabled mixed reality system for teleultrasound," Aug 2021.

[3] X. Jiang, H. Shokri-Ghadikolaei, G. Fodor, E. Modiano, Z. Pang, M. Zorzi, and C. Fischione, "Low-latency networking: Where latency lurks and how to tame it," *Proceedings of the IEEE*, vol. 107, no. 2, pp. 280–306, February 2019.

[4] A. Anand, G. D. Veciana, and S. Shakkottai, "Joint scheduling of urllc and embb traffic in 5g wireless networks." IEEE, 2018 April 2018, pp. 1970–1978.

[5] T. 3rd Generation Partnership Project (3GPP), "Physical channels and modulation," *Document TR38.211*, Jan. 2020.

[6] 3GPP, "3gpp tsg ran wg1 meeting 87," Tech. Rep., November 2016.

[7] C.-P. Li, J. Jiang, W. Chen, T. Ji, and J. Smee, "5g ultra-reliable and low-latency systems design," June 2017, pp. 1–5.

[8] Y. Huang, S. Li, C. Li, Y. T. Hou, and W. Lou, "A deep-reinforcement-learning-based approach to dynamic embb/urllc multiplexing in 5g nr," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6439–6456, July 2020.

[9] H. Yin, L. Zhang, and S. Roy, "Multiplexing urllc traffic within embb services in 5g nr: Fair scheduling," *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 1080–1093, 2020.

[10] A. Karimi, K. I. Pedersen, N. H. Mahmood, G. Pocovi, and P. Mogensen, "Efficient low complexity packet scheduling algorithm for mixed urllc and embb traffic in 5g," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–6.

[11] M. Alsenwi, N. H. Tran, M. Bennis, S. R. Pandey, A. K. Bairagi, and C. S. Hong, "Intelligent resource slicing for embb and urllc coexistence in 5g and beyond: A deep reinforcement learning based approach," *IEEE Transactions on Wireless Communications*, 2021.

[12] F. Saggese, L. Pasqualini, M. Moretti, and A. Abrardo, "Deep reinforcement learning for urllc data management on top of scheduled embb traffic," *arXiv preprint arXiv:2103.01801*, 2021.

[13] T. Richardson and S. Kudekar, "Design of low-density parity check codes for 5g new radio," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 28–34, March 2018.

[14] T. Girici, C. Zhu, J. R. Agre, and A. Ephremides, "Proportional fair scheduling algorithm in ofdma-based wireless systems with qos constraints," *Journal of communications and networks*, vol. 12, no. 1, pp. 30–42, 2010.

[15] X. Qiu and K. Chawla, "On the performance of adaptive modulation in cellular systems," *IEEE transactions on Communications*, vol. 47, no. 6, pp. 884–895, 1999.

[16] . W. G. of the 802 Committee, "Ieee 802.11n part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications," 2009.

[17] C. Xiao, Y. R. Zheng, and N. C. Beaulieu, "Statistical simulation models for rayleigh and rician fading," in *IEEE International Conference on Communications, 2003. ICC'03.*, vol. 5. IEEE, 2003, pp. 3524–3529.

[18] V. K. Lau, "Proportional fair space-time scheduling for wireless communications," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1353–1360, 2005.