Spectrum Sharing For Delay-Sensitive Applications With Continuing QoS Guarantees

Yuanzhang Xiao, Kartik Ahuja, and Mihaela van der Schaar
Department of Electrical Engineering, UCLA
Emails: yxiao@seas.ucla.edu, ahujak@ucla.edu, mihaela@ee.ucla.edu

***Abstract*—We study a wireless network in which multiple users stream delay-sensitive applications such as video conferencing and video streaming. Existing spectrum sharing policies, which determine when users access the spectrum and at what power levels, are either *constant* (i.e. users transmit simultaneously, at constant power levels) or weighted *round-robin* time-division multiple access (TDMA) (i.e. users access the spectrum in turn, one at a time). Due to multi-user interference, constant policies have low spectrum efficiency. We show that round-robin policies are inefficient for delay-sensitive applications because the various "positions" (i.e. transmission opportunities) in a cycle are not created equal: earlier transmission opportunities are more desirable since they enable users to transmit with lower delays. Specifically, we show that (weighted) round-robin TDMA policies cannot simultaneously achieve high network performance and low transmission delays. This problem is exacerbated when the number of users is large.**

We propose a novel framework for designing optimal TDMA spectrum sharing policies for delay-sensitive applications, which can guarantee their continuing QoS (CQoS), i.e. the desired throughput (and the resulting transmission delay) starting from *every moment in time* is guaranteed for each user. We prove that the fulfillment of CQoS guarantees provides strict upper bounds on the transmission delays incurred by the users. We construct the optimal TDMA policy that maximizes the desired network performance (e.g. max-min fairness or social welfare) subject to the users' CQoS guarantees. The key feature of the proposed policy is that it is *not* cyclic as in (weighted) round-robin policies. Instead, it adaptively determines which user should transmit next, based on the users' remaining amounts of transmission opportunities needed to achieve the desired performance. We also propose a low-complexity algorithm, which is run by each user in a distributed manner, to construct the optimal policy. Simulation results demonstrate that our proposed policy significantly outperforms the optimal constant policy and round-robin policies by up to 6 dB and 4 dB in peak signal-to-noise ratio (PSNR) for video streaming.

I. Introduction

A variety of bandwidth-intensive and delay-sensitive applications, such as multimedia streaming, gaming, and teleconferencing, are increasingly deployed over wireless networks. Such applications impose huge challenges when deployed over wireless networks, in which the users share the spectrum and cause interference to each other. Hence, it is crucial to design spectrum sharing policies that provide delay-sensitive users with both high rates and low delays.

The spectrum sharing policies studied in earlier works [1]–[5] require the users to transmit at *constant* power levels all the time[1]. We call them *constant* (spectrum sharing) policies. Constant policies are inefficient in many spectrum sharing scenarios with strong multi-user interference. Under strong multi-user interference, increasing one user's power level significantly degrades the other users' throughput, which results in low spectrum efficiency.

The optimal way to share the spectrum should take into account the users' cross interference. Ideally, we should assign the users to several (perhaps overlapping) subsets based on their cross interference, such that the users in the same group have low cross interference. Then the different subsets of users can transmit in turn. However, the optimal assignment of subsets requires knowledge about the cross channel gains among the users, which is hard to get in a decentralized spectrum sharing scenario. In addition, since the number of subsets grows exponentially with the number of users, even with the complete knowledge of cross channel gains, the computational complexity of searching for the optimal partitioning is prohibitively high. One way to manage interference while limiting the informational and computational costs is to simply let one user access the spectrum at one time, as in e.g. 802.11e MAC wireless networks [6]. Such policies are commonly known as time-division multiple access (TDMA) policies. Our focus in this paper is on designing optimal TDMA policies for delay-sensitive users.

All the existing TDMA policies are round-robin policies or their variants (e.g. weighted round-robin policies) [6]–[9]. In round-robin policies, time slots are divided into cycles of a fixed predetermined length, and each user transmits in fixed predetermined positions within each cycle. The cyclic nature of round-robin policies simplifies the implementation, but imposes restrictions that render round-robin policies inefficient for delay-sensitive applications. For delay-sensitive application, not all the transmission opportunities (i.e. positions) in a cycle are created equal: the earlier transmission opportunities (TXOPs) are more desirable because they result in higher chances to deliver packets on time, prior their delay deadlines [6]–[10]. To ensure that the user's rate and delay constraints are met, round-robin policies need a long cycle, and a careful sharing of TXOPs in a cycle. First, a long cycle is necessary. Suppose that the cycle length is the shortest possible, namely equal to the number of users (as in standard round-robin policies). Then the user allocated to the last TXOP suffers severely from delay. We can compensate this user for its delay by having a longer cycle and allocating some of the extra TXOPs to it. However, a long cycle results in an exponentially increasing (in the cycle length) number of possible policies to choose from. Second, a careful sharing of TXOPs is necessary (see Fig. 1 for an illustration of the following discussion). Suppose that the cycle length is twice the number of users, and that each user gets two positions in a cycle. For fairness, no user should get two advantageous (i.e. earlier) TXOPs. A

[1]Although some spectrum sharing policies go through a transient period of adjusting the power levels before the convergence to the optimal power levels, the users maintain constant power levels after the convergence.

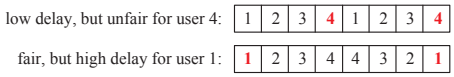

Fig. 1. Two simple round-robin schedules with cycle length 8 for 4 users. The first one has low delay of 4 for all 4 users, but unfair sharing of transmission opportunities (TXOPs) (i.e. user 4 gets later TXOPs). The second one has a fair sharing of TXOPs, but incurs high maximum delay of 7 for user 1.

TABLE I. COMPARISON WITH RELATED WORKS.

	Spectrum efficiency	CQoS guarantee	Delay guarantee	System or individual performance achieved
Constant [1]–[5]	Low	No	Yes	Neither
Round-robin [6]–[9]	High	No	Yes	Either for small # of users Neither for large # of users
Proposed	High	No	Yes	Both

possible fair sharing may ensure that the user who gets an earlier TXOP is also allocated to a later TXOP (e.g. one user should get the first and the last TXOPs in a cycle). However, such a fair sharing is inefficient in terms of individual performance: the user who gets the first and the last TXOPs in a cycle will experience high delay between consecutive transmissions. As we will illustrate in our motivating example (Section IV) and by simulations (Section VII), round-robin policies cannot simultaneously achieve high system performance (e.g. max-min fairness) and fulfill the guarantees in terms of transmission delays required by the delay-sensitive users. This becomes even more difficult to do within reasonable computational complexity when the number of users is large.

In this paper, we propose a framework for designing optimal TDMA spectrum sharing policies for delay-sensitive applications. We define a novel quality-of-service (QoS) metric, called continuing QoS (CQoS) guarantees. CQoS guarantees require a user's average throughput starting from *every point in time* to be higher than a threshold. CQoS guarantees are stricter requirements than conventional QoS guarantees which only guarantee the average throughput starting from the beginning. We will prove that fulfilling CQoS guarantees results in upper bounds on transmission delays. We propose a systematic design methodology, which constructs the optimal TDMA policy that maximizes the system performance (e.g. fairness) subject to the users' CQoS guarantees. The key feature of the proposed policy is that it is not cyclic as in round-robin policies. Instead, it adaptively determines which user should transmit according to the users' remaining amounts of TXOPs needed to achieve the target throughput. We propose a low-complexity algorithm, which can be run by each user in a distributed manner, to construct the optimal policy. Simulation results show that our proposed policy significantly outperforms the optimal constant policy [1]–[5] and round-robin policies in peak signal-to-noise ratio (PSNR) for video streaming, by up to 6 dB and 4 dB respectively, .

Finally, we summarize the comparison of our work with the existing works in Table I.

The rest of the paper is organized as follows. In Section II, we describe the system model. Then in Section IV, we motivate our proposed policy by showing the inefficiency of round-robin policies through a motivating example. We formulate the policy design problem in Section V, and solve it in Section VI. Simulation results are presented in Section VII. Finally, Section VIII concludes the paper.

II. SYSTEM MODEL

We consider a wireless network with N users. The set of users is denoted by $\mathcal{N} \triangleq \{1, 2, \ldots, N\}$. Each user has a transmitter and a receiver. The channel gain from user i's transmitter to user j's receiver is g_{ij}. Each user i chooses a power level p_i from a compact set $\hat{\mathcal{P}}_i$. We assume that $0 \in \hat{\mathcal{P}}_i$, namely user i can choose not to transmit. We also assume that the users need to comply with some interference temperature constraints (ITCs) measured at K locations in the network. Depending on different scenarios, the ITCs can be imposed by primary users in a cognitive radio network or the base station in a femtocell network. The channel gain from user i's transmitter to the kth location is g_{i0_k}. Each user i knows the channel gain $\{g_{i0_k}\}_{k=1}^K$ to each measurement location and the interference temperature limit $\{I_k\}_{k=1}^K$ at each location. Hence, each user i's set of *admissible* power levels is

$$\mathcal{P}_i = \{p_i \in \hat{\mathcal{P}}_i : g_{i0_k} \cdot p_i \leq I_k, \forall k = 1, \ldots, K\}. \quad (1)$$

For convenience, we also define user i's maximum admissible power level as $P_i^{max} \triangleq \max_{p_i \in \mathcal{P}_i} p_i$.

Remark 1: Our system model is general enough to model many wireless communication networks. It can model wireless ad hoc networks where N users transmit in the unlicensed spectrum (e.g. the 2.4 GHz frequency band) without ITCs ($K = 0$). It can also model the uplink (the receivers are co-located) and the downlink (the transmitters are co-located) of a cellular network with possible ITCs imposed by base stations in nearby cells. It can also model cognitive radio networks (or femtocell networks) with N secondary users (or femtocells) sharing the spectrum with K primary users (or $K = 1$ base station) imposing ITCs at their receivers.

We denote the joint power profile of all the users by $\boldsymbol{p} = (p_1, \ldots, p_N)$. Since the users cannot jointly decode their messages and can only treat other users' interference as noise, each user i's instantaneous throughput under the joint power profile $\boldsymbol{p}$ is [1]–[5]

$$r_i(\boldsymbol{p}) = \log_2\left(1 + \frac{p_i g_{ii}}{\sum_{j \in \mathcal{N}, j \neq i} p_j g_{ji} + \sigma_i}\right), \quad (2)$$

where σ_i is the noise power at user i's receiver. We write each user i's maximum throughput as $r_i^{max} \triangleq \log_2(1 + p_i g_{ii}/\sigma_i)$, which is achieved when user i transmits at the maximum power level and the other users do not transmit.

The system is time slotted at $t = 0, 1, \ldots$. We assume that the users are synchronized as in [1]–[5] (e.g. by using a global clock from the global positioning system (GPS)). We write each user i's transmission policy as $\pi_i : \mathbb{N}_+ \to \mathcal{P}_i$, where $\pi_i(t)$ is user i's transmit power level at time t. The spectrum sharing policy is then the collection of all the users' transmission policies, denoted by $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)$. In a constant policy, we have $\boldsymbol{\pi}(t) = \boldsymbol{p}^{const}$ for all $t \in \mathbb{N}_+$. In a TDMA policy, we have $\|\boldsymbol{\pi}(t)\|_0 = 1$, where $\|\cdot\|_0$ is the ℓ-0 norm that represents the number of nonzero elements in a vector.

Remark 2: In this paper, we will focus on TDMA policies. In other words, we do not consider constant policies, and

some non-constant policies that are not TDMA. Constant policies are known to be inferior to TDMA polices when the multi-user interference is strong. However, non-constant, non-TDMA policies may achieve better performance than TDMA policies, by allowing the subset of users with weak mutual interference to transmit simultaneously. However, determining the optimal subsets requires knowledge about the cross channel gains among the users, which is hard to get in a decentralized spectrum sharing scenario. In addition, since the number of subsets grows exponentially with the number of users, a naive exhaustive searching for the optimal partitioning has prohibitively high complexity. Hence, as in [6]–[9], we will focus on the design of TDMA policies in this paper.

Each user i's (discounted) average throughput is defined as

$$R_i(\boldsymbol{\pi}) = (1-\delta)\sum_{t=0}^{\infty}\delta^t \cdot r_i(\boldsymbol{\pi}(t)), \tag{3}$$

where $\delta \in [0,1)$ is the discount factor that models the delay-sensitivity of a user [5][10]. A more delay-sensitive user discounts the future throughput more (i.e. has a smaller discount factor), because it has more urgency to transmit. We use the discounted average throughput instead of the average throughput, because for delay-sensitive applications, the packets need to be transmitted as soon as possible to avoid missing their deadlines [10]. In [9], we also studied the case where the users have different discount factors but have no CQoS constraints. Extensions to the case with heterogeneous discounting and CQoS constraints are interesting future work.

III. Continuing QoS Guarantees

The widely-used QoS guarantee [1]–[10] is that the average throughput is above some guaranteed fraction γ_i^{avg} of the maximum possible throughput r_i^{max}, namely[2]

$$R_i(\boldsymbol{\pi}) \geq \gamma_i^{avg} \cdot r_i^{max}. \tag{4}$$

The above QoS guarantee does not provide sufficient guarantees for TDMA policies: even if a user's average throughput (starting from the beginning) $R_i(\boldsymbol{\pi})$ is high, it may get a extremely low throughput starting from certain point in time, because it may not get sufficient TXOPs after certain point.[3] Such an intuition will be illustrated in the motivating example in Section IV. In this paper, we propose *continuing QoS guarantees*, which ensure that at every point in time, a user's future throughput is guaranteed to be above some desired minimum requirement. Such continuing guarantees are important for delay-sensitive users. We formally define continuing QoS guarantees as follows.

First, we define the continuation throughput at time t as

$$R_i^t(\boldsymbol{\pi}) = (1-\delta)\sum_{\tau=0}^{\infty}\delta^{\tau-t} \cdot r_i(\boldsymbol{\pi}(\tau)), \tag{5}$$

which is the discounted average throughput starting from time t. Note that $R_i^0(\boldsymbol{\pi}) = R_i(\boldsymbol{\pi})$. Then, the continuing QoS guarantees can be written as

$$\textbf{CQoS: } R_i^t(\boldsymbol{\pi}) \geq \gamma_i^{cont} \cdot r_i^{max}, \ \forall t = 0,1,\ldots. \tag{6}$$

[2]In the case of constant policies [1]–[5], such QoS guarantee reduces to the requirement on the instantaneous throughput, namely $r_i(\boldsymbol{p}^{const}) \geq \gamma_i^{avg} \cdot r_i^{max}$, because the users choose the same power profile at any time.

[3]Constant policies do not have such a problem. If they fulfill the average throughput requirements, the throughput will be high enough starting from every point in time. However, it is difficult for them to fulfill the average throughput requirement in the first place, due to multi-user interference.

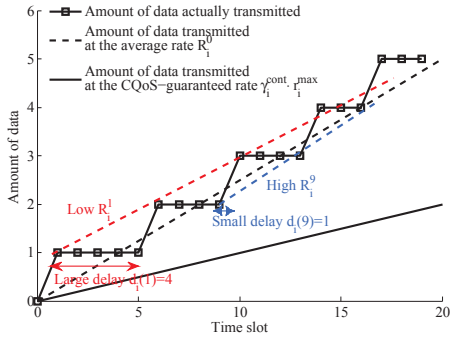

Fig. 2. Relationship of delay and CQoS guarantees of user i. The solid curve with square data points is the amount of data transmitted; each jump in the curve corresponds to a transmission. The two straight lines through the origin are the amount of data transmitted as if the throughput was R_i^0 and $\gamma_i^{cont} \cdot r_i^{max}$, respectively. At each time t, if the continuation throughput R_i^t is higher, the user needs to transmit more after time t. Hence, the corresponding delay $d_i(t)$ is lower.

To differentiate from CQoS, we will call the commonly-used QoS guarantee in (4) as average QoS (AQoS) guarantee in the rest of this paper. We can see that the CQoS guarantee contains the requirement for the average throughput $R_i(\boldsymbol{\pi})$. Hence, we will assume that $\gamma_i^{cont} < \gamma_i^{avg}$. Otherwise, the AQoS guarantee is redundant.

A byproduct of the CQoS guarantees is that once they are satisfied, we can also provide upper bounds on the transmission delays of each user. First, we define user i's transmission delay at any time t as

$$\textbf{Transmission Delay: } d_i^t(\boldsymbol{\pi}) \triangleq \min_{\tau>t}\{\tau - t : \pi_i(\tau) > 0\}.$$

In words, the transmission delay $d_i^t(\boldsymbol{\pi})$ is the minimum wait time until the next transmission. An upper bound on the transmission delays are critical for delay-sensitive applications. As we will prove in Theorem 3, each user's CQoS guarantee leads to an upper bound on its maximum delay $\sup_t d_i^t(\boldsymbol{\pi})$. We also illustrate the relationship of delay and CQoS guarantees in Fig. 2. We can see that the delay is determined by the difference between the traffic arrival rate and the guaranteed rate (i.e. CQoS). A higher CQoS results in lower delay.

IV. A Motivating Example

We provide a motivating example to illustrate the importance and impact of the CQoS guarantees, and to show the advantage of the proposed optimal TDMA policy over round-robin TDMA policies, in terms of both the performance and the computational complexity. Consider a simple network with four symmetric users. They have the same maximum throughput normalized to 1 bits/s/Hz (i.e. $r_i^{max} = 1, \ \forall i$), and the same discount factor of $\delta = 0.83$. The system performance metric is the max-min fairness (i.e. the minimum of all the users' throughput).

A. CQoS Guarantees and System Performance

We first illustrate the tradeoff between CQoS guarantees and the system performance. Intuitively, CQoS guarantees require that a user has sufficiently many transmission opportunities every once in a while. In other words, the transmission

TABLE II. ROUND-ROBIN TDMA POLICIES CANNOT ACHIEVE BOTH GOOD PERFORMANCE AND GOOD CQOS.

Cycle length	$L=4$	$L=5$	$L=6$	$L=7$
Rates (bits/s/Hz)	0.18	0.19	0.20	0.23
CQoS (bits/s/Hz)	0.18	0.13	0.10	0.07

TABLE III. PERFORMANCE LOSS AND COMPLEXITY OF ROUND-ROBIN TDMA POLICIES, UNDER CQOS GUARANTEES $\gamma_i^{cont}=0.1$.

Cycle length	$L=4$	$L=5$	$L=6$	$L=7$	Proposed
Worst user's rate	0.18	0.19	0.20	0.23	0.25
CQoS guarantee	fulfilled	fulfilled	fulfilled	*violated*	fulfilled
Performance loss (compared to proposed)	40%	32%	25%	N/A	–
# of policies	24	240	1560	8400	–

delay at any point in time should be small. For example, for round-robin TDMA policies with cycle length $L=8$, the one that maximizes CQoS guarantees (or minimizes transmission delay) has a cycle of 1234 1234. It is not difficult to see that, any other policies (we consider the policy with a cycle of 4321 4321 as the same since the users are symmetric) with $L=8$ will have a maximum transmission delay higher than 4, and will have a worse CQoS. However, the policy with cycle 1234 1234 is not fair: user 4 always transmits at later positions in the cycle, and hence will experience a very low average rate. The policy that achieves the best max-min fairness (i.e. the worst user's rate is maximized) has a cycle of 1234 4321, because user 4 will get two positions in the middle of the cycle. However, such a policy has a worst-case transmission delay of 7 (for user 1). In other words, user 1 has a low CQoS (e.g. its throughput starting from time slot 2 is very low, because it needs to wait until time slot 8 to transmit).

We illustrate the tradeoff between the CQoS guarantees and the system performance (i.e. max-min fairness) for round-robin TDMA policies in Table II. We can see that with the increase of the cycle length, round-robin TDMA policies achieve better performance, but will have worse CQoS guarantees.

In Table III, we illustrate the performance loss of round-robin TDMA policies compared to the proposed optimal TDMA policy. We find the optimal round-robin policies of different cycle lengths subject to a CQoS guarantee of 0.1 bits/s/Hz. The proposed policy achieves the optimal fairness (i.e. 0.25 bits/s/Hz for all 4 users), and outperforms round-robin policies by at least 20%.

B. Computational Complexity

Remarkably, not only is the proposed optimal policy much more efficient than round-robin policies, it is much easier to compute. To get a hint of why this is so, note that in a round-robin policy, the user's performance is determined not only by the number of slots in a cycle but also by the positions of the slots since users are discounting their future throughput (due to delay sensitivity). For a given number of users N and a given cycle length L, the number of non-trivial round-robin schedules[4] is greater than N^{L-N}. So searching among these schedules will be totally impractical even if L is moderately larger than N - but in order to achieve efficiency close to the optimal policy, the cycle length L must be much larger than N. For instance, for the 4-user case above, achieving energy efficiency within 10% of the optimal policy requires that the cycle length be at least 7, and requires searching among the thousands (8400) of different nontrivial schedules of cycle length 7. Even this small problem is computationally intensive. For a moderate number of users - say 10 - and a cycle length of 20 - we need to search more than ten billion (i.e. 10^{10}) schedules, which is completely intractable. However, we will propose a simple algorithm to compute the optimal policy whose complexity grows *linearly* with the number of users.

We will compare the complexity formally in Section VI-C and Table IV. Throughout this paper, we discuss the complexity of *designing* round-robin policies (e.g. how many policies to search), instead of the complexity of *implementing* them. It is easy to implement round-robin policies; however, it is computationally complex to determine the optimal policy before the implementation.

[4]Non-trivial schedules are the ones in which each user gets at least one time slot in a cycle.

V. FORMULATION OF THE POLICY DESIGN PROBLEM

We aim to design a TDMA spectrum sharing policy $\boldsymbol{\pi}$ that maximizes the system performance, defined as a function of the users' throughput, $W(R_1(\boldsymbol{\pi}),\ldots,R_1(\boldsymbol{\pi}))$. We assume that $W(\cdot)$ is increasing and strictly concave in each argument R_i. Such a definition of system performance is general enough to include the objective functions adopted in most existing works [1]–[10] as special cases. One example of system performance, which will be used in our simulations, is the (normalized) max-min fairness defined as $W(R_1(\boldsymbol{\pi}),\ldots,R_1(\boldsymbol{\pi}))=\min_i \frac{R_i(\boldsymbol{\pi})}{r_i^{max}}$. In addition, we will impose the AQoS and CQoS guarantees. To sum up, we can formally define the policy design problem as follows

$$\textbf{Design Problem:}\quad \max_{\boldsymbol{\pi}}\quad W(R_1(\boldsymbol{\pi}),\ldots,R_1(\boldsymbol{\pi})) \qquad (7)$$
$$s.t.\quad \textbf{AQoS: } R_i(\boldsymbol{\pi})\geq \gamma_i^{avg}\cdot r_i^{max},\ \forall i,$$
$$\textbf{CQoS: } R_i^t(\boldsymbol{\pi})\geq \gamma_i^{cont}\cdot r_i^{max},\ \forall i,t.$$

For the policy design problem (7) to be feasible, we require that $\sum_{i\in\mathcal{N}}\gamma_i^{avg}\leq 1$.

VI. SOLVING THE POLICY DESIGN PROBLEM

In this section, we solve the policy design problem (7). Our proposed solution (illustrated in Fig. 3) has two phases: an offline phase implemented before run-time, which determines the optimal operating point (i.e. each user's target average throughput), and a low-complexity online phase implemented at run-time, which determines the transmission schedule that achieves the optimal operating point while fulfilling AQoS and CQoS guarantees.

A. Offline Phase – The Optimal Operating Point

Before run-time, the users solve the following problem in a distributed manner to determine the optimal operating point:

$$\boldsymbol{r}^\star=\ \arg\max_{\boldsymbol{r}\geq\mathbf{0}}\quad W(r_1,\ldots,r_N) \qquad (8)$$
$$s.t.\quad \sum_{i\in\mathcal{N}} r_i/r_i^{max}=1,$$
$$r_i\geq\gamma_i^{avg}\cdot r_i^{max},\ \forall i.$$

In (8), the linear equality $\sum_{i\in\mathcal{N}} r_i/r_i^{max}=1$ comes from the requirement that the policy is TDMA. Intuitively,

it ensures that the total fraction of all the users' transmission opportunities sum up to 1. Note that the CQoS guarantees are not present in (8). They will be taken care of in our scheduling policy described in the next subsection.

Algorithm 1 The Optimal Operating Point Selection (OOPS) algorithm run by user i.

Require: AQoS γ_i^{avg}, precision ε
Set $\underline{\lambda}=0$, $\bar{\lambda}=1$, $\lambda=\bar{\lambda}$.
Solve $\frac{\partial W}{\partial r_i}=-\frac{\lambda}{r_i^{max}}$ for r_i^*, set $r_i^* \leftarrow \max\{r_i^*, \gamma_i^{avg}\cdot r_i^{max}\}$
Broadcast r_i^*/r_i^{max}, and receive r_j^*/r_j^{max} from users $j\neq i$
while $\sum_{j\in\mathcal{N}} r_j^*/r_j^{max} > 1$ **do**
  $\bar{\lambda}\leftarrow 2\cdot\bar{\lambda}$, $\lambda\leftarrow\bar{\lambda}$
  Solve $\frac{\partial W}{\partial r_i}=-\frac{\lambda}{r_i^{max}}$ for r_i^*, set $r_i^* \leftarrow \max\{r_i^*, \gamma_i^{avg}\cdot r_i^{max}\}$
  Broadcast r_i^*/r_i^{max}, and receive r_j^*/r_j^{max} from users $j\neq i$
end while
while $\left|\sum_{j\in\mathcal{N}} r_j^*/r_j^{max}-1\right| > \varepsilon$ **do**
  $\lambda\leftarrow\frac{\underline{\lambda}+\bar{\lambda}}{2}$
  Solve $\frac{\partial W}{\partial r_i}=-\frac{\lambda}{r_i^{max}}$ for r_i^*, set $r_i^* \leftarrow \max\{r_i^*, \gamma_i^{avg}\cdot r_i^{max}\}$
  Broadcast r_i^*/r_i^{max}, and receive r_j^*/r_j^{max} from users $j\neq i$
  if $\sum_{j\in\mathcal{N}} r_j^*/r_j^{max} < 1$ **then**
    $\bar{\lambda}\leftarrow\lambda$
  else
    $\underline{\lambda}\leftarrow\lambda$
  end if
end while
Normalize $r_i^*\leftarrow r_i^*/\left(\sum_{j\in\mathcal{N}} r_j^*/r_j^{max}\right)$

We propose a distributed optimal operating point selection (OOPS) algorithm (described in Algorithm 1) to solve (8).

We can prove that the distributed Algorithm 1 converges to the optimal operating point linearly[5] at rate $\frac{1}{2}$.

Theorem 1: The optimal operating point $\boldsymbol{r}^\star$ that solves (8) can be found by each user running the distributed OOPS algorithm (Algorithm 1), which converges linearly at rate $\frac{1}{2}$.

Proof: See [12, Appendix A]. ■

B. Online Phase – The Optimal Transmission Schedule

After finding the optimal operating point $\boldsymbol{r}^\star$, we need to determine the transmission schedule that achieves it. Importantly, the transmission schedule should fulfill the CQoS guarantees, which is the major challenge of our solution. We propose a distributed online longest-distance-first (LDF) scheduling algorithm (described in Algorithm 2).

Algorithm 2 has a nice interpretation of longest distance first scheduling. At each time slot t, the user with the largest "distance to target" (i.e. $\alpha_j(t)$ in Algorithm 2) transmits in this time slot[6]. The algorithm updates the distances in the "correct" way, such that the optimal operating points are achieved. Theorem 2 proves the desirable properties of the proposed LDF scheduling algorithm.

[5]Following [11, Sec. 9.3.1], we define linear convergence as follows. Suppose that the sequence $\{x_k\}$ converges to x. We say that this sequence converges linearly at rate c, if we have $\lim_{k\to\infty}\frac{|x_{k+1}-x|}{|x_k-x|}=c$.

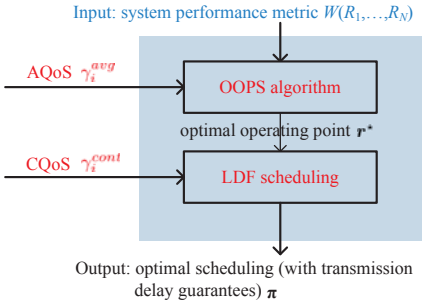

Fig. 3. Illustration of our proposed design framework. The operations in blue and in red are done by the policy designer and the decentralized users, respectively.

Algorithm 2 The Longest-Distance-First (LDF) scheduling algorithm run by user i.

Require: normalized operating points $\{r_j^\star/r_j^{max}\}_{j\in\mathcal{N}}$, discount factor δ
Initialization: $t=0$, "distances" $\alpha_j(0)=r_j^\star/r_j^{max}$, $\forall j\in\mathcal{N}$
repeat
  Find the user with the largest distance
$$i^* \triangleq \min\left\{\arg\max_{j\in\mathcal{N}}\alpha_j(t)\right\}$$
  if $i=i^*$ **then**
    Transmit at power level P_i^{max}
  end if
  Updates distances $\alpha_j(t+1)$ for all $j\in\mathcal{N}$ as follows:
  $\alpha_{i^*}(t+1)=\frac{\alpha_{i^*}(t)}{\delta}-(\frac{1}{\delta}-1)$, $r_j'(t+1)=\frac{\alpha_j(t)}{\delta}$, $\forall j\neq i^*$
  $t\leftarrow t+1$
until $\varnothing$

Theorem 2: For any discount factor $\delta \geq \frac{N-1}{N-\sum_{j\in\mathcal{N}}\gamma_j^{cont}}$, if each user $i\in\mathcal{N}$ runs the distributed LDF scheduling algorithm, we have

- each user i's average throughput up to time t converges to its optimal operating point linearly at rate δ, namely $|(1-\delta)\sum_{\tau=0}^{t}\delta^\tau\cdot r_i^\tau - r_i^\star| \leq r_i^{max}\cdot\delta^{t+1}$;
- each user i fulfills its CQoS guarantee, namely $R_i^t(\boldsymbol{\pi})\geq\gamma_i^{cont}\cdot r_i^{max}$, $\forall t=0,1,2,\ldots$

Proof: See [12, Appendix B]. ■

Remark 3: Although the CQoS guarantees do not directly appear in Algorithm 2, they impose a constraint on the discount factor δ used in Algorithm 2. Theorem 2 proves that the algorithm, given a proper input of discount factor (namely $\delta \geq \frac{N-1}{N-\sum_{j\in\mathcal{N}}\gamma_j^{cont}}$), will construct a policy that fulfill the CQoS guarantees.

[6]Ties can be broken arbitrarily. In Algorithm 2, we choose the user with the smallest index. Specifically, when $\arg\max_{j\in\mathcal{N}}\alpha_j(t)$ returns a set of indices, we choose the minimum one.

TABLE IV. COMPARISON OF COMPUTATIONAL COMPLEXITY.

Policy	Computational complexity
Constant policies [1]–[5]	NP-hard to find the optimal $\boldsymbol{p}^{const}$
Round-robin TDMA (cycle length L)	Offline: $\geq N^{L-N}$ policies to search Online: 0
Proposed	Offline: $N \cdot O(\log_2 1/\varepsilon)$ Online: $O(N)$

Theorems 1 and 2 establish the convergence results of our proposed scheme. Theorem 1 proves that the process of finding the optimal operating points converges in logarithmic time, and Theorem 2 proves that the LDF scheduling achieves the optimal operating points in logarithmic time. Hence, the overall convergence speed is fast. Moreover, Theorem 2 ensures that the CQoS guarantees are fulfilled.

As we have discussed before, a byproduct of the CQoS guarantees is the upper bounds on the transmission delays, which are provided in Theorem 3

Theorem 3: For any discount factor $\delta \geq \frac{N-1}{N-\sum_{j\in\mathcal{N}}\gamma_j^{cont}}$, if each user $i \in \mathcal{N}$ runs the distributed LDF scheduling algorithm, we have

- each user i's maximum transmission delay is upper bounded, namely
$$\sup_{t\geq 0} d_i^t(\boldsymbol{\pi}) \leq \frac{\log \gamma_i^{cont}}{\log \delta};$$
- at each time t, each user i's transmission delay is upper bounded, namely
$$d_i^t(\boldsymbol{\pi}) \leq \frac{\log \alpha_i(t)}{\log \delta},$$
where $\alpha_i(t)$ is user i's distance from target at time t calculated in Algorithm 2.

Proof: See [12, Appendix C]. ■

Theorem 3 gives us the upper bound of the maximum transmission delay, as well as finer upper bounds of transmission delays at each time t based on the user's distances from target $\alpha_i(t)$ (calculated in Algorithm 2). Note that the upper bound on the maximum delay, namely $\log \gamma_i^{cont}/\log\delta$, is decreasing in the CQoS, because we have $\gamma_i^{cont} < 1$ and $\delta < 1$.

C. Computational Complexity and Message Exchange

We compare the computational complexity of the existing solutions and our proposed solution, and discuss the amount of message exchange in our solution.

1) Computational Complexity: For constant policies, finding the optimal power profile $\boldsymbol{p}^{const}$ is NP-hard in general [1]. This is due to the nonconvexity of the problem: the throughput function is not jointly concave in the power profile because of the interference. For round-robin TDMA policies, the number of policies to search is lower bounded by N^{L-N}. To ensure a good performance, the cycle length needs to be large, which means that the number of policies grows exponentially with the number of users N. Hence, it may take a long time to find the optimal round-robin TDMA policy before run-time, although they are easy to implement at run-time. In contrast, in our proposed solution, the OOPS algorithm converges in logarithmic time before run-time, and the complexity of the online LDF scheduling is low (i.e. each user only needs to update the distances based on simple analytical formula).

2) Message Exchange: In our proposed solution, the message exchange happens only before run-time. The total amount of message exchange (i.e. the broadcast of r_i^*/r_i^{max}) is $N \cdot O(\log_2 1/\varepsilon)$. There is no message exchange at run-time.

VII. SIMULATION RESULTS

We demonstrate the performance gain of our proposed TDMA policy over existing policies. Throughout this section, we use the following system parameters. The noise powers at all the users' receivers are normalized as 0 dB. The maximum transmit powers of all the users are 20 dB. Without loss of generality, we normalize the direct channel gains to 1, namely $g_{ii} = 1, \forall i$, and generate the cross channel gains randomly according to the distribution $g_{ij} \sim \mathcal{CN}(0, 0.5), \forall i \neq j$. The system performance is measured by the (normalized) max-min fairness $\min_i R_i/r_i^{max}$, namely we aim to maximize the worst user's (normalized) throughput. At the optimal max-min fairness, each user's normalized average throughput $R_i(\boldsymbol{\pi})/r_i^{max}$ cannot exceed $\frac{1}{N}$. Hence, we let each user's AQoS guarantee to be within 10% of its maximum normalized throughput, namely $\gamma_i^{avg} = \frac{0.9}{N}, \forall i$. In most simulations, we will vary the CQoS guarantees (which are equal cross users). Given each CQoS guarantee, we choose the minimum discount factor specified by Theorem 2, namely $\delta = \frac{N-1}{N(1-\gamma_i^{cont})}$. In other words, we evaluate the performance of the most delay-sensitive applications.

A. Performance Under Different CQoS Guarantees

We first fix the number of users to be $N = 4$, and increase the CQoS guarantees from 0.1 to 0.22. Note that the AQoS guarantee under $N = 4$ is 0.225. Hence, a CQoS guarantee of 0.22 is close to the AQoS guarantee. In Fig. 4, we show the optimal max-min fairness (i.e. $\min_i R_i/r_i^{max}$) achieved by different policies. We can see that under all CQoS guarantees, our proposed policy can achieve the optimal max-min fairness of 0.25. In contrast, the optimal constant policy achieves at least 50% away from the optimal max-min fairness when the CQoS guarantee is small, and becomes infeasible when the CQoS guarantee exceeds 0.13. For round-robin policies, we search all the policies up to cycle length 9 (there are 186480 non-trivial policies with cycle length 9) and choose the optimal one under each CQoS guarantee. We can see that the performance of round-robin policies decreases to 20% away from the optimal performance before it becomes infeasible at CQoS of 0.19.

Next, we investigate the maximum number of users that can be supported by each policy under different CQoS guarantees. We increase the CQoS guarantees from 0.05 to 0.20. Note that theoretically, the maximum number of users that can possibly be supported is $\lfloor \frac{1}{\gamma_i^{cont}} \rfloor$ (because we need $N \cdot \gamma_i^{cont} \leq 1$). In Fig. 5, we can see that the maximum numbers of users supported by our proposed policy are the same as the theoretical upper bounds at most of the time. At certain CQoS guarantees (e.g. $\gamma_i^{cont} = 0.05$), the theoretical upper bound (e.g. 20 users when $\gamma_i^{cont} = 0.05$) can be achieved only when

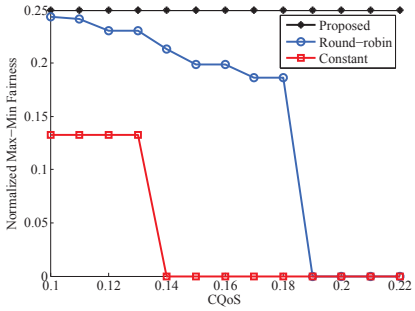

Fig. 4. Comparison of the max-min fairness achieved by different policies under different CQoS guarantees.

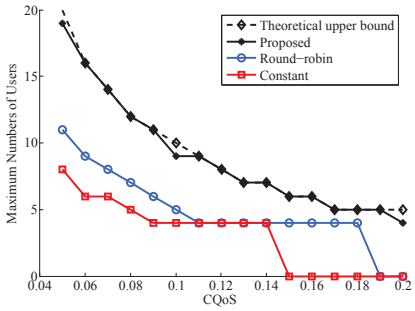

Fig. 5. Comparison of the maximum number of users that can be supported by different policies under different CQoS guarantees.

the discount factor is 1. However, since we consider delay-sensitive applications (i.e. $\delta < 1$), we can support slightly fewer users (e.g. 19 users when $\gamma_i^{cont} = 0.05$). In contrast, the other two polices can support much fewer users. At a CQoS guarantee of 0.05, we roughly double the number of users accommodated, compared to the other two policies. In other words, we can utilize the spectrum much more efficiently while fulfilling CQoS guarantees.

B. Performance for Video Transmission

Finally, we evaluate the performance of different polices for wireless video transmission. In the performance evaluation, we use the PSNR, which is commonly-used as performance metric for video quality. In the experiment, We consider a network with 4 users, and use the classic "Foreman" and "Coastguard" video sequences. In Table V (for "Foreman" video sequence) and Table VI (for "Coastguard" video sequence), we show the worst-case PSNR achieved by different policies under different CQoS guarantees. We can see that our proposed policy improves the PSNR of the constant policy and the round-robin policy by up to 6 dB and 4 dB, respectively. Moreover, when the CQoS guarantees increase, the other two policies become infeasible.

VIII. Conclusion

In this paper, we studied spectrum sharing among users with delay-sensitive applications. We proposed a novel performance metric, namely continuing QoS guarantees, to ensure the performance of delay-sensitive applications. We designed the optimal TDMA policy that maximizes the system performance subject to the CQoS guarantees, and proposed low-complexity distributed algorithms for the users to construct the optimal policy. Our proposed policy significantly outperforms existing constant policies and round-robin policies, in terms of the system performance (e.g. max-min fairness), the number of users accommodated while fulfilling their QoS guarantees, as well as the computational complexity of designing the optimal policies. When applied to video streaming, our proposed policy can achieve performance improvement of up to 6 dB and 4 dB, compared to constant policies and round-robin policies.

TABLE V. Improvement of PSNR in "Foreman" sequence over constant and round-robin policies under different CQoS guarantees.

CQoS guarantee	0.12	0.15	0.18	0.20
Constant	31 dB	infeasible	infeasible	infeasible
Round-robin	37 dB	35 dB	34 dB	infeasible
Proposed	38 dB	38 dB	38 dB	38 dB
Improvement over Constant	6 dB	–	–	–
Improvement over Round-robin	1 dB	3 dB	4 dB	–

TABLE VI. Improvement of PSNR in "Coastguard" sequence over constant and round-robin policies under different CQoS guarantees.

CQoS guarantee	0.12	0.15	0.18	0.20
Constant	29 dB	infeasible	infeasible	infeasible
Round-robin	34 dB	32 dB	32 dB	infeasible
Proposed	36 dB	36 dB	36 dB	36 dB
Improvement over Constant	6 dB	–	–	–
Improvement over Round-robin	1 dB	3 dB	4 dB	–

References

[1] C. W. Tan and S. H. Low, "Spectrum management in multiuser cognitive wireless networks: Optimality and algorithm," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 2, pp. 421-430, 2011.

[2] J. Huang, R. A. Berry, and M. L. Honig, "Distributed interference compensation for wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 5, pp. 1074-1084, May 2006.

[3] N. Gatsis, A. G. Marques, G. B. Giannakis, "Power control for cooperative dynamic spectrum access networks with diverse QoS constraints," *IEEE Trans. Commun.*, vol. 58, no. 3, pp. 933-944, Mar. 2010.

[4] S. Sorooshyari, C. W. Tan, M. Chiang, "Power control for cognitive radio networks: Axioms, algorithms, and analysis," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 878–891, 2012.

[5] R. Etkin, A. Parekh, and D. Tse, "Spectrum sharing for unlicensed bands," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 3, pp. 517–528, 2007.

[6] M. van der Schaar, Y. Andreopoulos, and Z. Hu, "Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 755–768, June 2006.

[7] D. Pradas, M. A. Vazquez-Castro, "NUM-based fair rate-delay balancing for layered video multicasting over adaptive satellite networks," *IEEE J. Sel. Areas in Commun.*, vol. 29, no. 5, May 2011.

[8] P. Dutta, A. Seetharam, V. Arya, M. Chetlur, S. Kalyanaraman, J. Kurose, "On managing quality of experience of multiple video streams in wireless networks," in *Proc. IEEE Infocom*, 2012.

[9] Y. Xiao and M. van der Schaar, "Spectrum sharing policies for heterogeneous delay-sensitive users: A novel design framework," in *Proc. Allerton*, 2013.

[10] M. van der Schaar and F. Fu, "Spectrum access games and strategic learning in cognitive radio networks for delay-critical applications," *Proc. of IEEE, Special issue on Cognitive Radio*, vol. 97, no. 4, pp. 720–740, Apr. 2009.

[11] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York: Cambridge Univ. Press, 2004.

[12] Y. Xiao and M. van der Schaar, "Appendix," Available at: http://www.seas.ucla.edu/~yxiao/AppendixGlobecom14a.pdf