Minimizing File Transfer Time in Opportunistic Spectrum Access Model

Jie Hu, Vishwaraj Doshi, and Do Young Eun

Abstract—We study the file transfer problem in opportunistic spectrum access (OSA) model, which has been widely studied in throughput-oriented applications for max-throughput strategies and in delay-related works that commonly assume identical channel rates and fixed file sizes. Our work explicitly considers minimizing the file transfer time for a given file in a set of heterogeneous-rate Bernoulli channels, showing that max-throughput policy doesn't minimize file transfer time in general. We formulate a mathematical framework for static extend to dynamic policies by mapping our file transfer problem to a stochastic shortest path problem. We analyze the performance of our proposed static and dynamic optimal policies over the max-throughput policy. We propose a mixed-integer programming formulation as an efficient alternative way to obtain the dynamic optimal policy and show a huge reduction in computation time. Then, we propose a heuristic policy that takes into account the performance-complexity tradeoff and consider the online implementation with unknown channel parameters. Furthermore, we present numerical simulations to support our analytical results and discuss the effect of switching delay on different policies. Finally, we extend the file transfer problem to Markovian channels and demonstrate the impact of the correlation of each channel.

Index Terms—Opportunistic spectrum access, file transfer problem, minimum transfer time, shortest path problem

1 Introduction

In recent years, there has been an explosion in demand for wireless services due to the rapid growth in the number of wireless devices, including mobile devices and Internet of Things (IoT) devices. This demand further exacerbates the scarcity of allocated spectrum, which is ironically known to be underutilized by licensed users [2]. The *Opportunistic spectrum access* (OSA) model has been proposed to reuse the licensed spectrum in an opportunistic way otherwise wasted by licensed users [2]. Recently, the FCC has released a new guidance in 2020, which would expand the ability of the unlicensed devices (especially IoT devices) to operate in the TV-broadcast bands [3]. Besides, the related IEEE 802.22 family has been developed to enable spectrum sharing [4] to bring broadband access to rural areas.

In the OSA model, a *secondary user* (SU) aims to opportunistically access the spectrum when it is not used by any other users, while also prioritizing the needs of the *primary user* (PU). The SUs need to periodically sense the spectrum to avoid interfering with PUs. We call an SU's behavior *static* if it adheres to only one channel, and *dynamic* if it is free to switch channels. While the concept of this model is simple, the design of spectrum sensing strategy faces various challenges: the interaction among multiple secondary users [5]–[7], spectrum sensing policy in the Markovian channels [8], [9], the trade-off between the cost of sequential sensing (when permitted) and the expected reward [10], [11], channel selection under resource constraints [12]–[14], to list a few.

1.1 Motivation: Throughput v.s. Latency

Nowadays, low latency has become one of the main goals for 5G wireless networks [15] and other time-sensitive applications with guaranteed delay constraints. In many applications, data is valid only for a limited duration and should be delivered before it expires, and vehicular communication is one such scenario. The increased demand for intelligent vehicular traffic (e.g., autonomous car development [16]) has led to the need for vehicular communication to explore spectrum holes for offloading vehicular users in device-to-device mode [17]. In addition, delay-sensitive safety messages (i.e., speed and position of the vehicles) require low latency (as low as 100 ms) [18]. Another example is medical body sensor networks [19], where the cognitive radio is implemented in body sensor network for life-critical monitoring, e.g., packets indicating a patient's health abnormality should be sent to a doctor as soon as possible, especially when the patient is out of the hospital network and needs to temporarily borrow vacant spectrum resources.

In the OSA literature, *throughput* is one of the most commonly used performance metrics. Recent studies [20]–[23], by utilizing the multi-armed-bandit (MAB) techniques, have focused on finding max-throughput channel while the SU needs to learn the unknown channel parameters on the fly. In order to transmit a file as quickly as possible, common folklore might assume that the max-throughput policy would also suggest the minimum expected file transfer time. For example, Wald's equation implies that the file download time in an *i.i.d* (over time) channel is equal to the file size divided by the average throughput of that channel, implicitly favoring the max-throughput channel for minimal

• *A subset of the material in this paper was presented in [1].*
Jie Hu and Do Young Eun are with the Department of Electrical and Computer Engineering, and Vishwaraj Doshi is with the Operations Research Graduate Program, North Carolina State University, Raleigh, NC. Email: {jhu29, vdoshi, dyeun}@ncsu.edu. *This work was supported in part by National Science Foundation under Grant CNS-1824518 and IIS-1910749.*

download time. However, as will be explained in Section 3, we find that this is *not* the case in general.

On the other hand, most *delay*-related works consider average queuing delay of a large number of packets (fixed size) following Poisson arrivals [24]–[28]. However, focusing on individual file transfers is important for small files when the SU needs to transmit each file as soon as possible. For instance, IEEE 802.11p protocol requires each car to generate and send safety messages continuously at 100 ms intervals [18], making Poisson arrivals unsuitable to model this situation. In addition, the file size can vary depending on the application [29]. Same channel data rate across all channels is another implicit assumption in those delay-related works [24]–[28],[1] but it doesn't reflect the realistic heterogeneous channel environment assumed in the throughput-oriented studies [20]–[23]. Clearly, allowing the SU to *switch* over such channels during instances of PU's interruption can further reduce the file transfer time, but to the best of our knowledge, this issue has not been fully explored.

1.2 Related Works and Their Limitations

Throughput and *delay* are two performance metrics commonly used in the OSA literature to evaluate the quality of service (QoS) in the wireless network. For throughput-oriented works, the PU's behavior can be modeled as a two-state Markov chain (thus correlated over time), for which partially observable Markov decision processes (POMDPs) are typically employed to formulate the spectrum sensing strategy in order to maximize the long-term throughput [30]. These POMDPs do not possess known structured solutions in general and they are known to be Polynomial-Space-complete (PSPACE-complete) even if all the channel statistics are known a priori [31]. To achieve near maximum throughput, computationally efficient yet sub-optimal policies, such as myopic policy [32] and Whittle's index policy [8], have been proposed for the *offline* OSA setting (known channel parameters). In particular, both [32] and [8] introduced a concept of "belief vector" to guess the available probability of each channel and updated the vector after each observation of the channel state. In each time slot, the myopic policy [32] selected the channel with the maximum "guess" throughput, while the Whittle's index policy [8] selected the channel with the highest value according to the Whittle's index and the belief vector.

Recently, machine learning techniques have emerged in the *online* setting (unknown channel parameters) that the SU needs to learn the unknown channel environment in order to find the max-throughput policy. For example, model-based MAB techniques [33]–[35] and model-free deep neural networks [14], [36], are utilized to obtain the max-throughput policy over Markovian channels. Explicitly, single-channel online policies have been developed in [33], [34] to find the channel with maximum long-term throughput. [35] utilized thompson sampling to estimate the parameters of each channel and employed the *offline* policies from [8], [32]. With well-trained neural networks, [36] showed better performance than the policies in [8], [32]. [14] tackled the coordination problem among multiple SUs with deep Q-learning. On the other hand, MAB techniques have been extensively studied for heterogeneous channels, each with *i.i.d* Bernoulli distribution, in order to find the channel with maximum throughput. The widely used MAB techniques include the Bayesian approach [37], upper confidence bounds [23], thompson sampling [20] and its improvement from efficient sampling [21], and coordination approach among multiple SUs [5], [6]. In both two channel models studied in the throughput-oriented works, i.e., Bernoulli channels and Markovian channels, we will later show that "the max-throughput channel does not always minimize the file transfer time".

For delay-sensitive applications, packet delay in cognitive radio networks has been extensively studied using queuing theory to derive delay-efficient spectrum scheduling strategies. In this setting, a stream of packet arrivals (of the same size) modeled as a Poisson process with a constant rate is a common assumption in delay related works [24]–[28], and the goal is often to minimize the average packet delay in the steady state. Specifically, a spectrum sensing strategy (including queuing delay, packet priority and interruption by PU's) was studied in the multimedia applications [24]. A dynamic load-balancing spectrum decision scheme was proposed in [25], where an R-learning algorithm was introduced to deal with unknown channels and queuing statistics. [26] controlled the transmission probability of each SU in the random access network and proposed a random-access strategy for two-and-three-SUs cases to minimize the queuing delay. Later, a model-free reinforcement-learning based strategy was studied in [27] to predict the channel accessing schemes without information exchanges among SUs and maximize the Quality-of-Service performance of the target SU. [28] focused on the hybrid spectrum access strategy with interweave and underlay spectrum access techniques for multiple SUs in order to obtain both good throughput and lower packet delay. However, as discussed Section 1.1, fixed packet length with Poisson arrivals and the same channel rate assumed in [24]–[28] is not realistic in some delay-sensitive applications. The file transfer problem considered in this paper allows for arbitrary file sizes and heterogeneous channel rates, and we can tackle the file transfer problem for each single file.

1.3 Our Contributions

In this paper, we study the OSA model with the aim of minimizing the transfer time of a single file over the heterogeneous Bernoulli channels with different channel rates and channel available probabilities. We also provide practical implementations that take into account computational costs and unknown channel environments, as well as the extension to Markovian channels. Our main contributions can be summarized as follows:

Theoretical Analysis of the File Transfer Problem.

- We first analyze the expected file transfer time of a single file for *static* policies.
- By using the analysis of the static policy as a stepping stone, we interpret the file transfer problem under *dynamic* policies as a stochastic shortest path problem, and obtain the dynamic *optimal* policy.

1. Channel data rate differs from the service rate in [24]–[27] because service rate is related to the length of time the channel is available, while data rate refers to the speed of data transfer through a channel.

- We show that both static and dynamic optimal policies reduce the transfer time compared to the baseline max-throughput policy, and this reduction is even more significant in delay-sensitive applications, where files are relatively small.

Practical Considerations.

- By formulating a mixed-integer programming method, we present an alternative technique to solve the shortest path problem, which speeds up the computation of dynamic optimal policy.
- We propose a lightweight heuristic policy to further reduce the computational cost while maintaining good performance compared to the max-throughput policy.
- In the online setting, where channel parameters are unknown to the SU, we modify an MAB algorithm proposed in [38] and show its gap-dependent regret bound, guaranteeing the learning of the optimal policy that minimizes the file transfer time.

Simulations.

- We empirically show that the max-throughput policy is not the best when it comes to achieving the minimum file transfer time in both known and unknown channel environments.
- When channel switching delay is taken into account, our lightweight heuristic policy can even outperform the dynamic optimal policy.

Extension to Markovian Channels.

- We extend the theoretical analysis to Markovian channels, where each channel is modeled as a two-state Markov chain, and obtain the expected file transfer time of a single file in each channel and show the effect of correlation on the file transfer time.

The rest of the paper is organized as follows: In Section 2 we introduce the OSA model and characterize the file transfer problem and its policy under the OSA framework. In Section 3, we show the expected transfer time for static policy and it's performance analysis. Then, we extend from the static policy to the dynamic policy in Section 4. The practical concerns are discussed in Section 5. In Section 6, we evaluate different policies in the numerical setting. We provide additional analysis on the file transfer problem over Markovian channels in Section 7.

2 Model Description

2.1 The OSA Model

Consider a set of N heterogeneous channels $\mathcal{N} \triangleq \{1, 2, \cdots, N\}$ available for use and each channel $i \in \mathcal{N}$ offers a stable rate of $r_i > 0$ bits/s if successfully utilized [8], [34]. In our setting, a SU wishes to transfer a file of size F bits using one of these N channels via opportunistic spectrum access. The SU can only access one channel at any given time, and can maintain this access for a fixed duration of Δ seconds, after which it has to sense available channels again (even the same channel) in order to avoid the interference to the active PU (or other SUs) in the current channel. At this point, the SU can decide which channel to sense and access that channel for the next Δ second interval if the channel is *available*. Or the SU has to wait for Δ seconds to sense again if that channel is *unavailable*, thereby unable to transfer data for this duration. This pattern is known as the *constant access time* model, and has been commonly adopted for the SU's behavior as a collision prevention mechanism in the OSA literature [2], [37]. The cycle repeats itself until the SU transmits the entire file size F, then it immediately exits the channel in use. We omit the channel switching delay in our OSA model and the duration Δ seconds are fully used for file transmission, which is typically assumed in order to simplify the mathematical model and design a throughput-optimal policy in the OSA literature [8], [20]–[23], [39]. Note that the duration Δ seconds is not a randomly chosen number. For example, Δ is recommended as 100 ms because the SU needs to vacate the current channel within 100 ms once the PU shows up, as defined in IEEE 802.22 standard [4]. The SU can transmit up to 3.1 Mb in each Δ seconds with highest channel rate 31 Mbps in IEEE 802.22 standard and many small files (e.g, 5 KB text-only email, 800 KB GIF image and 3 MB YouTube short video) need just a few slots to transmit.

Remark 2.1. The duration Δ does not include the sensing time for the fair comparison between our policies in Section 3 to 5 and the max-throughput policy in the same OSA model [20]–[23]. In addition, as simulated in [40], if the time duration is set to 100 ms (which is the duration of our Δ) and the target probability of accurate sensing is around 90%, the sensing time for a cognitive radio network is typically chosen to be 6 ms. This sensing time is negligible compared to the whole time duration and is omitted in our mathematical model.

We say a channel is unavailable (or *busy*) if it is currently in use by the primary users (PUs) or other SUs, while it is available (or *idle*) if it is not in use by any other users. The state of a channel (idle or busy) is assumed to be independent over all channels $i \in \mathcal{N}$, and *i.i.d.* over the time instants $\{0, \Delta, 2\Delta, \cdots\}$ following Bernoulli distribution with parameter $p_i \in (0, 1]$, in line with the widely used discrete-time channel model [2], [6], [37].[2] Specifically, for each $i \in \mathcal{N}$, $\{Y_i(k)\}_{k\in\mathbb{N}}$ is a Bernoulli process with $p_i = P[Y_i(k) = 1] = 1 - P[Y_i(k) = 0]$ for all $k \in \mathbb{N}$. Then, we can define $X_i(t)$, the state of channel $i \in \mathcal{N}$ at any time $t \in \mathbb{R}_+$, as a piecewise constant random process $X_i(t) \triangleq Y_i(\lfloor \frac{t}{\Delta} \rfloor)$, where $\lfloor \cdot \rfloor$ denotes the floor function. This way, we write $X_i(t) = 1$ (or 0) if channel $i \in \mathcal{N}$ is available (or unavailable) for the SU with probability p_i (or $1 - p_i$).

Remark 2.2. Inaccurate sensing, including mis-detections with probability v_i for channel i in each time slot, has been studied in the OSA literature [40], [41]. However, this does not affect our theoretical analysis. With probability $p_i' \triangleq p_i(1 - v_i)$, the SU can successfully transmit data in the current time slot, otherwise the data transmission is zero. Thus, we can take the inaccurate sensing

2. We also extend our theoretical analysis of the file transfer problem over Markovian channels in Section 7, i.e., each channel is modeled as a two-state Markov chain that will change its state accordingly every Δ seconds.

into account and replace p_i with p_i' in the analysis in Section 3 to 5 without affecting the conclusions.

The rate at which the SU can transmit files through channel $i \in \mathcal{N}$ at any time instant $t \geq 0$, also termed as the *instantaneous throughput* of the channel i, is given by $r_i X_i(t)$, with its *throughput* [21] by $\mathbb{E}[r_i X_i(t)] = r_i p_i$. We denote by $i^* \triangleq \arg\max_{k\in\mathcal{N}} r_k p_k$ the channel with the *maximum throughput*. For simplicity, we assume that this channel is unique, i.e., $r_{i^*} p_{i^*} > r_k p_k$ for all $k \in \mathcal{N}\setminus\{i^*\}$. In the next section, we take a closer look at the max-throughput policy and static policies in general.

2.2 Policies for File Transfer

We define a *policy* at time t to be a mapping $\pi : \mathbb{R}_+ \mapsto \mathcal{N}$ where $\pi(t) = i$ indicates that the SU has chosen channel i to access during the time period $\left[\lfloor \frac{t}{\Delta} \rfloor \Delta, \left(\lfloor \frac{t}{\Delta} \rfloor + 1\right)\Delta\right)$. From our standing assumption, a policy therefore only changes at $t \in \{0, \Delta, 2\Delta, \cdots\}$, and all policies ensure that the file transfer for any finite size F will eventually be completed. This way, the policy $\pi(t)$ is a piecewise constant function (mapping), defined at all time $t \geq 0$. For a given policy π, let $T(\pi, F)$ denote the *transfer time* of a file of size F — the entire duration of time to complete the file transfer, which is written as

$$T(\pi, F) = \min_{T\geq 0} \left\{ \int_0^T r_{\pi(t)} X_{\pi(t)}(t) dt \quad \geq F \right\}. \qquad (1)$$

Figure 1 explains the file transfer progress via OSA model.

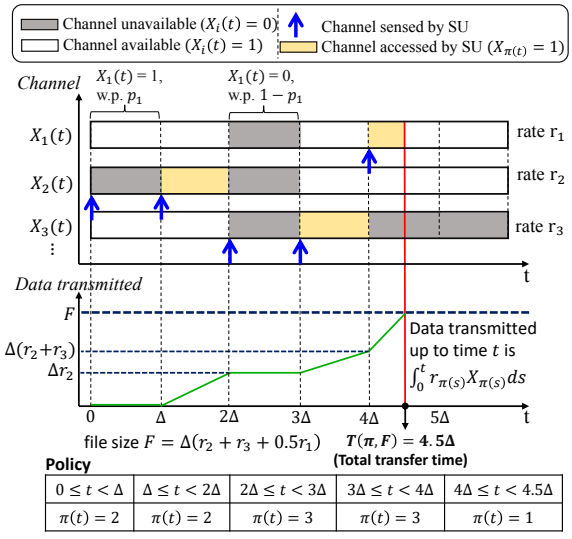


Fig. 1. File transfer via the OSA framework. The SU senses channels according to its policy, and accesses the channel if it is available. Upon gaining access, it begins transmitting data at the corresponding channel rate. Transmission ends (red line) as soon as the amount of data transmitted (green line) equals the file size F.

The objective of our OSA framework is to minimize the expected transfer time $\mathbb{E}[T(\pi, F)]$ over the set of all policies π. Policies can be *static*, where the SU only senses and transmits via one (pre-determined) channel i throughout the file transfer, that is, $\pi(t) = i$ for all $t \geq 0$. For such static policies, we denote by $T(i, F)$ their transfer time for file size F. The channel that provides the minimum expected transfer time is then called *static optimal* given by

$$i_{so}(F) = \arg\min_{k\in\mathcal{N}} \mathbb{E}[T(k, F)], \quad \text{(static optimal)} \qquad (2)$$

and we denote $T(i_{so}, F)$ the corresponding transfer time for this static optimal policy. Note that the static optimal channel $i_{so}(F)$ depends on the file size F and can vary for different file sizes. Policies can also be *dynamic*, in which an SU is allowed to change the channels it chooses to sense throughout the course of the file transfer. Given a file size F, the policy with the minimum expected transfer time over the set of all policies $\Pi(F)$ is called the *dynamic optimal* policy given by

$$\pi^*(F) = \arg\min_{\pi\in\Pi(F)} \mathbb{E}[T(\pi, F)]. \text{ (dynamic optimal)} \quad (3)$$

Lastly, we define the *max-throughput policy* as the static policy with the channel i^*, which maximizes the long-term throughput. In the next section, we take a closer look at the max-throughput policy and static policies in general.

3 Static Optimal Policy

Recent works in the OSA literature focus on estimating channel parameters p_i's, with the goal of eventually converging to the policy $i^* = \arg\max_{k\in\mathcal{N}} r_k p_k$ which provides the maximum throughput [20]–[23], [34], [37].[3] They focus on minimizing the 'regret' in the MAB model, defined as the difference between the cumulative reward obtained by the online algorithm and the max-throughput policy (the optimal policy in hindsight).

The essential assumption behind all these approaches is that the SU always fully dedicates Δ seconds in each time interval for file transfer. Channel i^* appears as a good candidate since it provides the largest expected data transfer $\Delta r_{i^*} p_{i^*}$ across every time interval. This is further supported by the well-known Wald's equation with the *i.i.d* reward assumption at each time interval, suggesting that $\mathbb{E}[T(i, F)] = F/r_i p_i$ for each channel $i \in \mathcal{N}$, which is then minimized by i^*. When policies are dynamic, however, the rewards are not identically distributed since the transfer rates of the dynamically accessed channels can be different, making Wald's equation inapplicable. Surprisingly, it is not applicable for static policies either. As typically is the case in delay-sensitive applications [24], [26]–[28], the file sizes are often not that large, rendering their transfer times small enough that an SU may not need to utilize the whole Δ seconds for data transfer in each time interval. The reward summands are still not identically distributed, causing Wald's equation to be inapplicable in general.

Our key observation in this paper is that choosing channel i^* may not be the best option to minimize the expected transfer time. In this section, we limit ourselves to the set of static policies of the form shown in (2) and analyze the resulting expected transfer time in the OSA network. We use this to compare the performance gap between the max-throughput policy and the static optimal policy, and show that for a reasonable choice of channel statistics and file sizes, the static optimal policy performs significantly better than the max-throughput policy. We derive a closed-form

3. While [20], [21] deal with link rate selection problem to select best rate in one channel to maximize the expected throughput, the mathematical model of link rate selection problem is essentially the same as the standard OSA setting for choosing the max-throughput channel, as considered in our setting.

expression of the expected transfer time of a file of size F in each fixed channel by the following proposition.

Proposition 3.1. Given a file of size F, the expected transfer time $\mathbb{E}[T(i,F)]$ of the static policy for channel $i \in \mathcal{N}$ is

$$\mathbb{E}[T(i,F)] = \Delta\left(k_i/p_i + \mathbb{1}_{\{\alpha_i>0\}}(1-p_i)/p_i + \alpha_i\right), \quad (4)$$

where $k_i \triangleq \lfloor F/\Delta r_i \rfloor \in \mathbb{Z}_+$ and $\alpha_i \triangleq F/\Delta r_i - k_i \in [0,1)$.

Proof: Observe that any file size F transmitted in channel i can be written as

$$F = k_i \Delta r_i + \alpha_i \Delta r_i \quad (5)$$

with k_i being the number of intervals fully utilized for successful transmission, and α_i being the fraction of the Δ second interval utilized for file transfer toward the end. After choosing channel i, the SU first spends a random amount of time, denoted by T^n_{wait} $(n = 1, 2, \cdots, k_i)$, waiting for channel i to become available and starts the n-th transmission in that channel for Δ seconds. If the remaining portion $\Delta\alpha_i$ is not zero, the SU needs additional random waiting time $T^{k_i+1}_{wait}$ to complete the transfer. These random variables $\{T^n_{wait}\}$ are geometrically distributed and i.i.d over $n = 1, 2, \cdots, k_i + \mathbb{1}_{\{\alpha_i>0\}}$ with mean $E[T^n_{wait}] = \Delta(1-p_i)/p_i$. Let the constant $T_{tran} \triangleq F/r_i = \Delta(k_i + \alpha_i)$ be the total successful transmission time. Then the transfer time can be written as

$$\begin{aligned} T(i,F) &= \sum_{n=1}^{k_i} (T^n_{wait} + \Delta) + \mathbb{1}_{\{\alpha_i>0\}} T^{k_i+1}_{wait} + \Delta\alpha_i \\ &= T_{tran} + \sum_{n=1}^{k_i+\mathbb{1}_{\{\alpha_i>0\}}} T^n_{wait}. \end{aligned}$$

Taking the expectation of the equation above yields (4). □

From Proposition 3.1, the expected transfer time $\mathbb{E}[T(i,F)]$ for any file size F under the static policy on channel $i \in \mathcal{N}$ can be explicitly written in terms of file size F, time duration Δ and channel statistics r_i and p_i of the chosen channel i. Substituting $k_i = F/\Delta r_i - \alpha_i$ in (4) gives

$$\mathbb{E}[T(i,F)] = \frac{F}{r_i p_i} + \Delta \mathbb{1}_{\{\alpha_i>0\}}(1-\alpha_i)\frac{1-p_i}{p_i} \geq \frac{F}{r_i p_i}. \quad (6)$$

The inequality in (6) shows the expected transfer time of any static policy is no smaller than that given by Wald's equation.

We use Figure 2 to illustrate the results in Proposition 3.1, where each line represents the expected transfer time via one channel over a range of file sizes from (4). We observe that the expected transfer time of channel 1 (red line) is always above the Wald's equation of channel 1 (purple dot-line). As shown in (4), the slope of each line (channel i) is $1/r_i$ and the "jump size" $\Delta(1-p_i)/p_i$ is equal to the expected waiting time till the channel is available. The jumps in the plot for each channel i, representing the waiting times, occur at exactly the instances where file size is an integer multiple of Δr_i, and come into play especially when there is still a small amount of remaining file to be transferred at the end of a Δ time interval.

By definition, the static optimal policy provides the minimum expected transfer time over all static policies including the max-throughput policy itself. While it is true for all file sizes, in some cases with certain file sizes, these two policies may coincide.

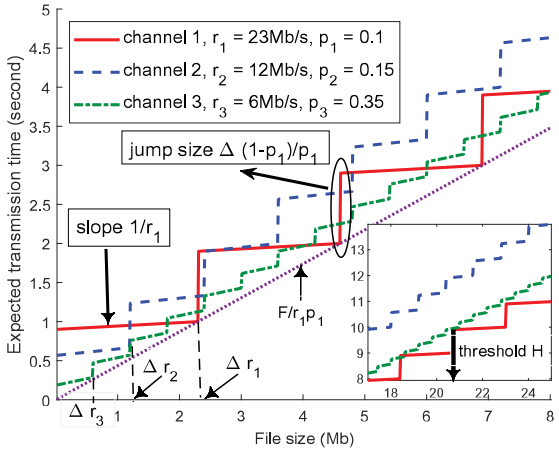


Fig. 2. Expected transfer time from (4) with duration $\Delta = 100$ ms (IEEE 802.22 standard). Channel 1 has the maximum throughput. The purple dot-line $E[T] = F/r_1p_1$ corresponds to the Wald's equation for channel 1. Downward arrow in the inset is the threshold H in Proposition 3.2.

Proposition 3.2. The max-throughput policy coincides with the static optimal policy, that is, $i_{so}(F) = i^*$, for any file size F satisfying at least one of the two conditions below:

1) F exceeds a threshold H, where
$$H = \frac{\Delta(1-p_{i^*})/p_{i^*}}{1/r_h p_h - 1/r_{i^*}p_{i^*}}, \quad (7)$$
and $h = \arg\max_{j\in\mathcal{N}\setminus\{i^*\}} r_j p_j$ is the channel with the second largest throughput.
2) F is an integer multiple of Δr_i^*, i.e., $F = k\Delta r_{i^*}$ for some $k \in \mathbb{Z}_+$.

Proof: From $\mathbb{1}_{\{\alpha_i>0\}}(1-\alpha_i) \in [0,1]$ and (6), we have the upper bound and the lower bound of $E[T(i,F)]$ as follows:

$$\begin{aligned} \frac{F}{r_i p_i} \leq \mathbb{E}[T(i,F)] &= \frac{F}{r_i p_i} + \Delta \mathbb{1}_{\{\alpha_i>0\}}\frac{(1-\alpha_i)(1-p_i)}{p_i} \\ &\leq \frac{F}{r_i p_i} + \Delta\frac{1-p_i}{p_i}. \end{aligned} \quad (8)$$

To ensure $\mathbb{E}[T(i^*,F)] \leq \mathbb{E}[T(j,F)]$ for all $j \in \mathcal{N}\setminus\{i^*\}$, it suffices to consider the upper bound of $E[T(i^*,F)]$ to be always smaller than the lower bound of $E[T(j,F)]$ for all $j \in \mathcal{N}\setminus\{i^*\}$ from (8), that is

$$\frac{F}{r_{i^*}p_{i^*}} + \Delta\frac{1-p_{i^*}}{p_{i^*}} \leq \min_{j\in\mathcal{N}\setminus\{i^*\}} \frac{F}{r_j p_j}. \quad (9)$$

By definition of channel $h = \arg\max_{j\in\mathcal{N}\setminus\{i^*\}} r_j p_j$ we have $F/r_h p_h = \min_{j\in\mathcal{N}\setminus\{i^*\}} F/r_j p_j$. Then rearranging the second inequality in (9) yields $F \geq H$ in (a).

When $F = k\Delta r_{i^*}$, we have $\alpha_{i^*} = 0$. Then from (4), the expected transfer time is simply $\mathbb{E}[T(i^*,F)] = \Delta(k/p_{i^*}) = F/r_{i^*}p_{i^*}$. Since $r_{i^*}p_{i^*} \geq r_j p_j$ for any $j \in \mathcal{N}$, we have

$$\mathbb{E}[T(i^*,F)] = \frac{F}{r_{i^*}p_{i^*}} \leq \frac{F}{r_j p_j} \leq \mathbb{E}[T(j,F)]$$

for any $j \in \mathcal{N}$, where the second inequality is from (8). Hence i^* is the static optimal channel, that is, $i^* = i_{so}(F)$. This establishes (b), completing the proof. □

Outside of Proposition 3.2, however, there are many instances where the max-throughput channel is not static optimal and other channels can perform better for smaller file sizes. In such cases, we would like to discuss how much time the static optimal policy can save against the max-throughput policy.

Corollary 3.3. Let $m_i \triangleq p_i/p_{i^*}$ for $i \in \mathcal{N} \setminus \{i^*\}$. Consider a file of size $F \in (k\Delta r_{i^*}, (k+1)\Delta r_{i^*})$ for some $k \in \mathbb{N}$. Then, we have

$$\frac{\mathbb{E}[T(i_{so}, F)]}{\mathbb{E}[T(i^*, F)]} \leq \min_{i \in \mathcal{N}\setminus\{i^*\}} \left\{1, \frac{F/\Delta r_i p_i + (1-p_i)/p_i}{(k+1)(m_i/p_i - 1)}\right\}. \quad (10)$$

Proof: For the file of size $F \in (k\Delta r_{i^*}, (k+1)\Delta r_{i^*})$ with $k \in \mathbb{N}$, we have $r_{i^*}p_{i^*} > r_i p_i$ and $m_i \triangleq p_i/p_{i^*}$ for $i \in \mathcal{N} \setminus \{i^*\}$. From (8) in the proof of Proposition 3.2, we have $\mathbb{E}[T(i, F)] \leq F/r_i p_i + \Delta(1-p_i)/p_i$. Moreover, from (4) we have

$$\mathbb{E}[T(i^*, F)] > \Delta(k+1)(1/p_{i^*} - 1) = \Delta(k+1)(m_i/p_i - 1). \quad (11)$$

Therefore, the upper bound of the time ratio between channel i and channel i^* is shown as follows:

$$\frac{\mathbb{E}[T(i, F)]}{\mathbb{E}[T(i^*, F)]} < \frac{F/\Delta r_i p_i + (1-p_i)/p_i}{(k+1)(m_i/p_i - 1)}. \quad (12)$$

By definition of the static optimal channel and $\mathbb{E}[T(i_{so}, F)] \leq \mathbb{E}[T(i^*, F)]$, we can get the result (10) by lower bounding (12) for channel $i \in \mathcal{N} \setminus \{i^*\}$. □

Note that by definition $m_i/p_i = 1/p_{i^*} > 1$ so that the upper bound on the ratio $\mathbb{E}[T(i_{so}, F)]/\mathbb{E}[T(i^*, F)]$ in (10) is always in the interval $(0, 1]$. Moreover, smaller ratio means better performance of the static optimal policy against the max-throughput policy. To gauge how the parameters of the max-throughput channel could affect the performance of the static optimal policy, suppose we fix F, Δ, r_i, p_i for all $i \in \mathcal{N} \setminus \{i^*\}$ and the maximum throughput $r_{i^*}p_{i^*}$, while treating p_{i^*} as a variable. The upper bound in (10) is then monotonically decreasing in $m_i = p_i/p_{i^*}$, and can even approach to 0 if at least one of m_i is really large, resulting in the huge performance gain of the static optimal channel compared to that of the max-throughput channel. This implies that accessing channel i^* can take much longer time to transmit a file than other channels if its available probability p_{i^*} is very small, which is common in outdoor networks where the max-throughput channel i^* has very high rate but with low available probability [42].

Our static optimal policy shows better performance against the max-throughput policy for small files and small p_{i^*}. Since the static optimal channel depends on the file size, choosing channels *dynamically* according to its remaining file size can further reduce the expected transfer time. We next formulate the file transfer problem as an instance of the stochastic shortest path (SSP) problem and analyze the performance of the dynamic optimal policy.

4 Dynamic Optimal Policy

Now that we have analyzed the static policies, we turn our attention to feasible dynamic policies for our file transfer problem. We start by first formulating the file transfer problem as a stochastic shortest path (SSP) problem, in which the agent acts dynamically according to the stochastic environment to reach the predefined destination as soon as possible. Then, we translate this SSP problem into an equivalent shortest path problem, which helps us derive the closed-form expression of the expected transfer time for any given dynamic policy, and we utilize this to obtain the performance analysis of the dynamic optimal policy against the max-throughput policy.

4.1 Stochastic Shortest Path Formulation

The SSP problem is a special case of the infinite horizon Markov decision process [43]. To make this section self-contained, we explain our problem as a SSP problem.

State Space and *Action Space:* We define the state $s \in \mathcal{S} \triangleq \mathbb{R}_+$ of our SSP as the remaining file size yet to be transmitted. The action $i \in \mathcal{N}$ is the channel chosen to be sensed at the beginning of each time interval. The objective of our problem is to take the optimal action at each state s which minimizes the expected time to transmit the file of size F.

State Transition: Denote by $P_{s,s'}(i)$ the transition probability that the SU moves to state s' after taking action i at state s. From any given state $s \in \mathcal{S} \setminus \{0\}$, the next state under any action $i \in \mathcal{N}$ depends on the availability of channel i. Since the channel is available or unavailable according to an *i.i.d* (over time) Bernoulli distribution, the next state is either the same as the current one if channel i is unavailable, i.e. $P_{s,s}(i) = 1 - p_i$; or the next state is $(s - \Delta r_i)^+ \triangleq \max\{0, s - \Delta r_i\}$ if channel i is available, i.e. $P_{s,(s-\Delta r_i)^+}(i) = p_i$. State 0 is a termination state since there is no file transmission remaining.

Cost Function: The cost $c(s, i, s')$ is the amount of time spent in transition from state s to s' after sensing channel i. Since the SU can only sense channels at intervals of size Δ, sensing an unavailable channel costs a Δ second waiting period until the SU can sense next, that is, $c(s, i, s) = \Delta$ for all $s \in \mathcal{S} \setminus \{0\}, i \in \mathcal{N}$. Similarly, if the sensed channel is available, the time spent in transmitting is also Δ seconds, unless the SU finishes transmitting the file early. In the latter case the cost of transmission is $c(s, i, 0) = s/r_i$. Overall, the cost of a successful transmission can be written as $c(s, i, (s - \Delta r_i)^+) = \min\{\Delta, s/r_i\}$ for all $s \in \mathcal{S} \setminus \{0\}, i \in \mathcal{N}$. Once the remaining file size reduces to 0, the SU will end this file transmission immediately with no additional cost incurred, so that $c(0, i, 0) = 0$ for any $i \in \mathcal{N}$.

Table 1 summarizes the state transition and cost function for our file transfer problem. All other cases except the two cases in Table 1 have zero transition probability and zero cost.

TABLE 1
SSP SETTING OF OUR FILE TRANSFER PROBLEM

current state	action	next state	transition	cost
$s > 0$	i	s	$1 - p_i$	Δ
$s > 0$	i	$(s - \Delta r_i)^+$	p_i	$\min\{\Delta, s/r_i\}$

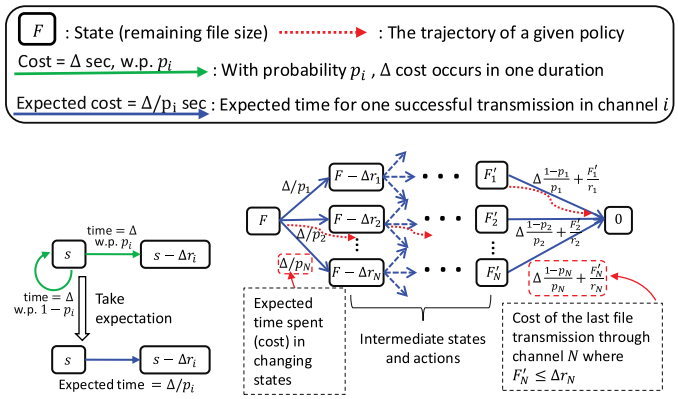
Our dynamic policy[4] is written as a mapping $\pi : S \to \mathcal{N}$, where $\pi(s) \in \mathcal{N}$ denotes the channel chosen for sensing when the current state (remaining file size) is s. For any

4. There always exists an optimal policy π^* to be deterministic in the SSP problem, as proved in Proposition 4.2.4 [43]. Thus, we restrict ourselves to the class of deterministic policies in this paper.

policy π, we have $T(\pi,0)=0$ at the termination state. Our goal in this SSP problem is to find the dynamic optimal policy $\pi^*(F)$ that minimizes the expected transfer time for the file size F, which can be derived from a variety of methods such as value iteration, policy iteration and dynamic programming [43].

4.2 Performance Analysis

For ease of exposition, we introduce additional notations here. By a *successful transmission*, we refer to state transitions of the form $s \to s'$. This is denoted by the horizontal green line in Figure 3(a) connecting states s and $s' \triangleq s - \Delta r_i > 0$, and should be distinguished from the self-loop $s \to s$, which implies the sensed channel was unavailable. As shown in Figure 3(a), taking expectation helps get rid of these self-loops by casting the original SSP to a deterministic shortest path problem in expectation. The cost associated with each link is then the expected time it takes to transit between the states. Figure 3(b) shows the underlying network for the shortest path problem, where each link is a channel chosen to be sensed and each path from *source* F to *destination* 0 corresponds to a policy $\pi \in \Pi(F)$. The path-length or the number of links traversed from F to 0 under any given policy π then becomes the total number of successful transmissions needed by that policy to complete the file transfer, which we denote by $|\pi|$.



(a) Transformation of the SSP problem. (b) Equivalent shortest path diagram.

Fig. 3. Illustration to translate the file transfer problem into an equivalent shortest path problem.

For any policy $\pi \in \Pi(F)$ and $n \in \{1,\cdots,|\pi|\}$, let F_n denote the remaining file size right before the n-th successful transmission. Then for all $n \in \{2,\cdots,|\pi|\}$, we have the recursive relationship: $F_n = F_{n-1} - \Delta r_{\pi(F_{n-1})}$, starting with $F_1 = F$ and ending with $F_{|\pi|+1} = 0$. Given a file size F, each policy $\pi \in \Pi(F)$ can then be written in a vector form as $\boldsymbol{\pi} = [\pi(F_1), \pi(F_2), \ldots \pi(F_{|\pi|})]^T$. With this in mind, we can derive a closed-form expression of the expected transfer time for any dynamic policy in the following proposition.

Proposition 4.1. Given a file of size F, the expected transfer time of a dynamic policy π is written as

$$\mathbb{E}[T(\pi,F)] = \Delta\left(\sum_{n=1}^{|\pi|-1}\frac{1}{p_{\pi(F_n)}} + \frac{1-p_{\pi(F_{|\pi|})}}{p_{\pi(F_{|\pi|})}}\right) + \frac{F_{|\pi|}}{r_{\pi(F_{|\pi|})}}. \tag{13}$$

Proof: With our notation $E[T(\pi,s)]$ in mind, the Bellman equation for any fixed policy $\pi \in \Pi(F)$ (Proposition 4.2.3 in [43]) is shown as

$$E[T(\pi,s)] = \sum_{s' \in \mathcal{S}} P_{s,s'}(\pi(s))\left[c\left(s,\pi(s),s'\right) + E[T(\pi,s')]\right]. \tag{14}$$

The transition probability and cost function in section 4.1 are defined as $P_{s,s}(i) = 1-p_i, P_{s,(s-\Delta r_i)^+}(i) = p_i, \forall s \in \mathcal{S}\backslash\{0\}, i \in \mathcal{N}$, and

$$c\left(s,i,(s-\Delta r_i)^+\right) = \begin{cases} \min\{\Delta, s/r_i\} & s \in \mathcal{S}\backslash\{0\}, i \in \mathcal{N} \\ 0 & s = 0, i \in \mathcal{N}. \end{cases}$$

Then, by substituting $P_{s,s'}(i)$ and $c(s,i,s')$ with our transition probability and cost function defined above, (14) can be written as

$$\begin{aligned}\mathbb{E}[T(\pi,F)] =& (1-p_{\pi(s)})\left(\Delta + \mathbb{E}[T(\pi,F)]\right) \\ &+ p_{\pi(s)}\left(\min\left\{\Delta, \frac{s}{r_{\pi(s)}}\right\} + \mathbb{E}[T(\pi,F_2)]\right).\end{aligned}$$

Recall that F_n is the remaining file size right before the n-th successful file transmission given a policy π and $F_{n+1} = F_n - \Delta r_{\pi(F_n)}$ for $n = 1,2,\cdots,|\pi|-1$, we can generalize this recursion to two adjacent states $F_n, F_{n+1} \in \mathcal{S}$ in policy π such that

$$\begin{aligned}\mathbb{E}[T(\pi,F_n)] =& \Delta\frac{1-p_{\pi(F_n)}}{p_{\pi(F_n)}} + \min\left\{\Delta, \frac{F_n}{r_{\pi(F_n)}}\right\} \\ &+ \mathbb{E}[T(\pi,F_{n+1})].\end{aligned} \tag{15}$$

When $n = 1,2,\cdots,|\pi|-1$, the player fully spends Δ time in each successful transmission and the file transfer task has not been done yet ($F_n > \Delta r_n$), so that $\min\{\Delta, F_n/r_n\} = \Delta$. For the last successful transmission $n = |\pi|$, we have

$$\mathbb{E}[T(\pi,F_{|\pi|})] = \Delta\frac{1-p_{\pi_{|\pi|}}}{p_{\pi_{|\pi|}}} + \frac{F_{|\pi|}}{r_{\pi_{|\pi|}}}.$$

Thereby recursively solving the above equation gives (13). □

In (13), the first summation is the cumulative expected transmission time, or the cost, to the $(|\pi|-1)$-th successful transmission, with the last two terms being the expected transmission time of the last successful transmission. Proposition 4.1 also includes the expected transfer time of the static policy as a special case. Recall that $k_i = \lfloor F/\Delta r_i \rfloor$ and $\alpha_i = F/\Delta r_i - k_i$ in Proposition 3.1. When applied to a static policy for any channel $i \in \mathcal{N}$, we have $|\pi| = k_i + \mathbb{1}_{\{\alpha_i > 0\}}$ and $p_{\pi(F_n)} = p_i$ for all $n = 1,2,\cdots,|\pi|$. The recursive relationship becomes: $F_n = F_{n-1} - \Delta r_i$, implying that $F_n = F - (n-1)\Delta r_i$. Then, we have $F_{|\pi|} = F - (|\pi|-1)\Delta r_i = \alpha_i \Delta r_i$ if $\alpha_i > 0$. Otherwise, $F_{|\pi|} = \Delta r_i$. Substituting these into (13) gets us (4).

The common folklore around the max-throughput policy is that it would lead to the minimal file transfer time of $F/r_{i^*}p_{i^*}$. Our next result shows this is too optimistic and not achieved in general even under the dynamic optimal policy.

Proposition 4.2. For any file size F and any dynamic policy $\pi \in \Pi(F)$, we have $\mathbb{E}[T(\pi,F)] \ge \mathbb{E}[T(\pi^*,F)] \ge F/r_{i^*}p_{i^*}$. Moreover, $i^* = \pi^*(F)$ for $F = k\Delta r_{i^*}, k \in \mathbb{Z}_+$.

Proof: From (13) we have

$$\begin{aligned}
&\mathbb{E}[T(\pi,F)]\\
&=\sum_{n=1}^{|\pi|-1}\frac{\Delta r_{\pi(F_n)}}{r_{\pi(F_n)}p_{\pi(F_n)}}+\frac{\Delta\left(1-p_{\pi\left(F_{|\pi|}\right)}\right)r_{\pi\left(F_{|\pi|}\right)}+F_{|\pi|}p_{\pi\left(F_{|\pi|}\right)}}{r_{\pi\left(F_{|\pi|}\right)}p_{\pi\left(F_{|\pi|}\right)}}\\
&\geq\sum_{n=1}^{|\pi|-1}\frac{\Delta r_{\pi(F_n)}}{r_{i^*}p_{i^*}}+\frac{\Delta\left(1-p_{\pi\left(F_{|\pi|}\right)}\right)r_{\pi\left(F_{|\pi|}\right)}+F_{|\pi|}p_{\pi\left(F_{|\pi|}\right)}}{r_{i^*}p_{i^*}}\\
&\geq\frac{1}{r_{i^*}p_{i^*}}\left(\sum_{n=1}^{|\pi|-1}\Delta r_{\pi(F_n)}+F_{|\pi|}\right)=\frac{F}{r_{i^*}p_{i^*}},
\end{aligned}$$

where the first inequality comes from the fact that $r_{i^*}p_{i^*}\geq r_ip_i$ for all $i\in\mathcal{N}$. The second inequality is from our definition of $F_{|\pi|}$ which implies that $F_{|\pi|}\leq\Delta r_{\pi_{|\pi|}}$.

When file size F is an integer multiple of Δr_{i^*}, i.e., $F=k\Delta r_{i^*}$ for some $k\in\mathbb{Z}_+$, we have $\mathbb{E}[T(i^*,F)]=F/r_{i^*}p_{i^*}\leq\mathbb{E}[T(\pi^*,F)]\leq E[T(\pi,F)]$. Since $\mathbb{E}[T(i^*,F)]\geq\mathbb{E}[T(\pi^*,F)]$, we have $\mathbb{E}[T(\pi^*,F)]=F/r_{i^*}p_{i^*}$, and the max-throughput policy coincides with the dynamic optimal policy. □

As shown in (6), $F/r_{i^*}p_{i^*}$ is always the lower bound on the transfer time for any static policy. Proposition 4.2 strengthens this by showing that the same is true even for the dynamic optimal policy. Similar to condition (b) in Proposition 3.2 for the static optimal policy, the dynamic optimal policy $\pi^*(F)$ also coincides with the max-throughput policy i^* when the file size is an integer multiple of Δr_{i^*}, while we no longer have the finite threshold H as in Proposition 3.2(a). We next give bounds to quantify the performance of the dynamic optimal policy with respect to the max-throughput policy.

Corollary 4.3. Let $m_i\triangleq p_i/p_{i^*}$ for $i\in\mathcal{N}\setminus\{i^*\}$. Consider a file of size $F\in(k\Delta r_{i^*},(k+1)\Delta r_{i^*})$ for some $k\in\mathbb{N}$. Then, we have

$$\begin{aligned}
&\frac{F}{F+\Delta\mathbb{1}_{\{\alpha_i>0\}}(1-p_{i^*})r_{i^*}}\leq\frac{\mathbb{E}[T(\pi^*,F)]}{\mathbb{E}[T(i^*,F)]}\\
&\leq\min_{i\in\mathcal{N}\setminus\{i^*\}}\left\{1,\frac{\frac{F}{\Delta r_i}+1-p_i-km_i\frac{r_{i^*}p_{i^*}-r_ip_i}{r_ip_i}}{(k+1)(m_i-p_i)}\right\}.
\end{aligned}$$

Proof: We first define a suboptimal policy π_{heu} and then use $E[T(\pi^*,F)]\leq E[T(\pi_{heu},F)]$ for our proof. The suboptimal policy π_{heu} is defined as sensing and accessing the max-throughput channel i^* to transmit the file of size $F_1=n\Delta r_{i^*}$, then following a static optimal policy for the remaining file of size $F_2=F-n\Delta r_{i^*}$. n is an integer chosen from 0 to $k=\lfloor F/\Delta r_{i^*}\rfloor$. Then, the expected transfer time of the dynamic optimal policy is always smaller than that of the dynamic suboptimal policy, that is,

$$\begin{aligned}
\mathbb{E}[T(\pi^*,F)]&\leq\mathbb{E}[T(\pi_{heu},F)]\\
&=\mathbb{E}[T(i^*,n\Delta r_{i^*})]+\mathbb{E}[T(i_{so},F-n\Delta r_{i^*})]\\
&\leq\Delta n/p_{i^*}+\min_{i\in\mathcal{N}\setminus\{i^*\}}\left\{\frac{F-n\Delta r_{i^*}}{r_ip_i}+\Delta\frac{1-p_i}{p_i}\right\}\\
&=\min_{i\in\mathcal{N}\setminus\{i^*\}}\left\{\frac{F}{r_ip_i}+\Delta\left(\frac{1-p_i}{p_i}-n\frac{r_{i^*}p_{i^*}-r_ip_i}{r_ip_ip_{i^*}}\right)\right\},
\end{aligned}$$

where the second inequality comes from (4), and (8) in the proof of Proposition 3.2. It shows monotonically decreasing in n such that we can choose $n=k$ to get the smallest upper bound for $\mathbb{E}[T(\pi^*,F)]$. Moreover, we have $E[T(i^*,F)]>\Delta(k+1)(m_i/p_i-1)$ from (11). Hence, the upper bound of the ratio $\mathbb{E}[T(\pi^*,F)]/\mathbb{E}[T(i^*,F)]$ in Corollary 4.3 is proved.

For the lower bound of the ratio, by using Proposition 4.2 we have $E[T(\pi^*,F)]\geq F/r_{i^*}p_{i^*}$. Together with (6), we have

$$\begin{aligned}
\frac{\mathbb{E}[T(\pi^*,F)]}{\mathbb{E}[T(i^*,F)]}&\geq\frac{F/r_{i^*}p_{i^*}}{F/r_{i^*}p_{i^*}+\Delta\mathbb{1}_{\{\alpha_{i^*}>0\}}(1-\alpha_{i^*})(1-p_{i^*})/p_{i^*}}\\
&\geq\frac{1}{1+\Delta\mathbb{1}_{\{\alpha_{i^*}>0\}}(1-p_{i^*})r_{i^*}/F},
\end{aligned}$$

where the second inequality comes from $1-\alpha_{i^*}\leq1$. This completes the proof. □

To better understand Corollary 4.3, we analyze how the parameters of the max-throughput channel could impact the performance of the dynamic optimal policy. Similar to Corollary 3.3, small value of $\mathbb{E}[T(\pi^*,F)]/\mathbb{E}[T(i^*,F)]$ implies that the dynamic optimal policy offers significant saving in time over the max-throughput policy. We note that Corollary 4.3 tightens the upper bound with an extra negative term in the numerator, compared to that in Corollary 3.3, potentially providing greater savings in time as we extend the policy from static optimal to dynamic optimal.

In contrast to Proposition 3.2 that max-throughput policy is good enough for $F\geq H$, Corollary 4.3 tells us that there is always some reduction in file transfer time even for large file size F under the dynamic optimal policy. This is because the extra negative term in the numerator can be large, since $k=\lfloor F/\Delta r_{i^*}\rfloor$ could be big for large F, implying that the second argument in the $\min\{\cdot,\cdot\}$ function may no longer be increasing in F. Note however that the reduction in transfer time would be minimal for large file sizes since the lower bound in Corollary 4.3 will rise to 1 as F goes to infinity.

5 Practical Considerations

While the dynamic optimal policy gives a smaller expected transfer time, the computational cost of solving the shortest path problem is still a concern. Besides, we face a scaling problem when the file size differs from each, effectively changing the underlying "graph" in the corresponding shortest path problem. This warrants re-computation of the dynamic optimal polity for each file size, which would be unacceptable in reality. In this section, we discuss a mixed-integer programming formulation and propose a heuristic policy to balance the performance and the computational cost. We also consider the case where the SU doesn't know about the channel parameters beforehand and it must sense and access channels *on the fly* to find the optimal policy.

5.1 Mixed-Integer Programming Formulation

Dynamic programming problems often have equivalent integer or mixed integer programming formulations as well [44]. For our shortest path problem, however, we can leverage the fact that the cost of an action is the same for each state (before the last successful transmission) to notably reduce the size of the solution space of the mixed integer formulation, especially for large file sizes where the curse of dimensionality is most felt.

Before putting forward our equivalent mixed integer programming problem, We first provide an alternate expression for (13) as below.

Proposition 5.1. Given a policy $\pi \in \Pi(F)$ for any file of size F, let $x_i \triangleq \sum_{n=1}^{|\pi|-1} \mathbb{1}_{\{\pi(F_n)=i\}}$, and let $y_i \triangleq \mathbb{1}_{\{\pi(F_{|\pi|})=i\}} F_{|\pi|}/\Delta r_{\pi(F_{|\pi|})}$. Then, we can rewrite (13) as

$$\begin{aligned} E[T(\pi,F)] =& \Delta \sum_{i=1}^{N} \Bigg[\sum_{n=1}^{|\pi|-1} \frac{\mathbb{1}_{\{\pi(F_n)=i\}}}{p_i} \\ &+ \mathbb{1}_{\{\pi(F_{|\pi|})=i\}} \left(\frac{1-p_{\pi(F_{|\pi|})}}{p_{\pi(F_{|\pi|})}} + \frac{F_{|\pi|}}{\Delta r_{\pi(F_{|\pi|})}} \right) \Bigg] \\ =& \Delta \sum_{i=1}^{N} \frac{x_i}{p_i} + \mathbb{1}_{\{y_i>0\}} \frac{1-p_i}{p_i} + y_i. \end{aligned}$$ □

In the above, x_i counts the total number of successful transmissions through any channel $i \in \mathcal{N}$ except for the last ($|\pi|$-th) transmission. On the other hand, y_i is zero if channel i is not the last one sensed under policy π, or else is equal to the fraction of the Δ second interval used for the $|\pi|$-th successful transmission.

A combination of $\mathbf{x} = [x_i]_{i\in\mathcal{N}}$ and $\mathbf{y} = [y_i]_{i\in\mathcal{N}}$ can be the same for multiple optimal policies, which have the same expected time although different order in which the channels are sensed. This means there can be $(\mathbf{1}^T\mathbf{x})!/\prod_{i\in\mathcal{N}}(x_i!)$ policies with the same expected transfer time - a number which can be really large for large F. Condensing the state space by preventing the solver from considering these $(\mathbf{1}^T\mathbf{x})!/\prod_{i\in\mathcal{N}}(x_i!)$ many policies individually can significantly reduce computation time. This can be done by solving the following mixed-integer programming problem over the set of all $\mathbf{x}, \mathbf{y}$ that correspond to feasible dynamic policies.

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y}} \quad & \Delta \sum_{i=1}^{N} \left[\frac{x_i}{p_i} + \mathbb{1}_{\{y_i>0\}} \frac{1-p_i}{p_i} + y_i \right] \\ s.t. \quad & \Delta \sum_{i=1}^{N} (x_i r_i + y_i r_i) = F, \\ & y_i y_j = 0, \qquad \forall i \neq j \in \mathcal{N}, \\ & x_i \in \mathbb{Z}_+,\ y_i \in [0,1),\ \ \forall i \in \mathcal{N}. \end{aligned} \tag{16}$$

In the above optimization problem, the first constraint ensures that the choice of $\mathbf{x}, \mathbf{y}$ guarantees the transmission of the entire file by adding up to F when multiplied by Δ and the respective channel rates. The second constraint makes sure that at most one of the y_i's is positive to ensure that the last transmission, if any, is assigned only to one channel. Once an optimal solution $\mathbf{x}^* = [x_i^*]$ and $\mathbf{y}^* = [y_i^*]$ is obtained, we can construct a corresponding dynamic optimal policy π^* by setting the sole channel i for which $y_i > 0$ as the last channel for transmission under the policy, and the first $|\pi^*| - 1$ transmissions can be according to any permutation of assignments[5] from the vector $\mathbf{x}$. In practice, we usually consider transmitting the file in channel i successfully for x_i counts and then switching to the closest channel j for x_j successful transmissions, which can help reduce the switching delay and energy cost. In this way, we expect that searching for a dynamic optimal policy over this condensed space would be much quicker than searching over the entire set of paths for the shortest path.

Now, we formally compare the computational complexity of Dijkstra algorithm and mixed integer programming, which is known to be NP-hard in general [45]. In the following lemma, denote by $r_{\min}$ the minimum channel rate over all channels and $\lceil\cdot\rceil$ the ceiling function, we show that Dijkstra algorithm for the underlying graph in Figure 3 is also an NP-hard problem w.r.t the file size F.

Lemma 5.2. The worst-case computational complexity of Dijkstra algorithm is $O(\lceil F/\Delta r_{\min}\rceil N^{\lceil F/\Delta r_{\min}\rceil}\log(N))$.

Proof: Dijkstra algorithm is known to have the time complexity $O(E\log(V))$ [46], where E is the number of edges and V is the number of nodes of the underlying graph. We now consider the tree-like underlying graph structure, F being the source node and 0 being the destination, as the worst case. The tree-depth is $\lceil F/\Delta r_{\min}\rceil$ and the node in each level contains N child nodes, resulting in $O(N^{\lceil F/\Delta r_{\min}\rceil})$ edges in total. In this case, $E = O(N^{\lceil F/\Delta r_{\min}\rceil})$, $V \approx E$, and we have

$$O(E\log(V)) = O(\lceil F/\Delta r_{\min}\rceil N^{\lceil F/\Delta r_{\min}\rceil}\log(N)).$$

□

Lemma 5.2 shows that the time complexity of Dijkstra algorithm is exponential in the file size F. As a result, both Dijkstra algorithm and mixed integer programming are NP-hard w.r.t the file size F. However, from the practical implementation, we address that the underlying graph of the shortest path problem is not given upfront. Later in Section 6.2, we will point out that the SU needs to first generate the graph of the shortest path problem in the real-world implementation, which includes all possible paths from the source node F to the destination 0 and involves a huge computational overhead, before feeding it into the dynamic programming solver. We observe that generating such graph is the most time-consuming step, while the mixed integer programming solver doesn't need such overhead and runs faster in practice.

5.2 Performance-Complexity Trade Off

Transmitting different-sized files is very common in the real world. For example, a short text-only email only takes up 5KB, one five-page paper is around 100KB and the average size of web page is 2MB, all implying that the file size may vary greatly [47]. However, due to the nature of the shortest path problem, a change in file size induces a change in the underlying graph. If the goal is to always determine the best solution, the only option is to recompute the dynamic optimal policy for every different file size. This would not be scalable in applications where minimal computation is required, and policies that can be promptly modified and reused across different file sizes with performance guarantees would be highly desirable.

To avoid heavy computation for each file size to obtain the dynamic optimal policy, we propose a heuristic policy that utilizes the max-throughput policy and the static optimal policy in order to reduce the computational

5. For example, if $\mathbf{x} = [0, 3, 2]^T$ for a three channel system, then the 5 successful transmissions will be 3 of them via channel 2 and 2 of them via channel 3, in no particular order.

cost, while still maintaining considerable performance gain. Note that the max-throughput policy coincides with the dynamic optimal policy when the file size is an integer multiple of Δr_{i^*} according to Proposition 4.2. We also know that the static optimal policy significantly outperforms the max-throughput policy especially for smaller file sizes. Combining these two policies, by transmitting file through max-throughput channel until the remaining file size becomes 'small' so as to apply the static optimal policy for the rest,[6] will strike the right balance between computational complexity and achievable performance gain. From the computational-cost-saving perspective, max-throughput policy is fixed and known to the SU. The closed-form expression $E[T(i,F)]$ for static policy (channel i) is also known to the SU. Since our heuristic policy includes the "min" function on $\mathbb{E}[T(i,F)]$ for all channel $i \in \mathcal{N}$, its computational complexity is $O(N)$, which is much smaller than that of the dynamic optimal policy whose complexity is polynomial in N as described in Lemma 5.2.

With this motivation in mind, we divide file size $F := F_1 + F_2$ into two parts: $F_1 = n\Delta r_{i^*}$ ($n = 0, 1, \cdots, \lfloor F/\Delta r_{i^*} \rfloor$) and $F_2 = F - F_1$. The heuristic policy π_{heu} is defined as follows: The SU first transmits the file of size F_1 through the max-throughput channel i^* and then sticks to the static optimal policy $i_{so}(F_2)$ for the remaining file of size F_2.[7] We have shown in the proof of Corollary 4.3 that the upper bound of ratio $E[T(\pi_{heu},F)]/E[T(i^*,F)]$ is monotonically decreasing in n. Therefore, $n = k$ potentially gives us the smallest upper bound of the ratio (the same upper bound in Corollary 4.3). Moreover, we have explained after Corollary 4.3 that the upper bound is smaller than that of the static optimal policy in Corollary 3.3. These arguments suggest that the heuristic policy π_{heu} with $n = k$ can potentially offer smaller delay than other candidates with different values of n. Besides, our heuristic policy can further reduce the computational cost for a set of files sharing the same remaining file size F_2, for which the static optimal policy $i_{so}(F_2)$ has already been found and no further recomputation is needed.

5.3 Unknown Channel Environment

We now consider the setting where the SU does not know the available probability p_i for any channel $i \in \mathcal{N}$ and only knows the rate r_i — a commonly analysed setting in the OSA literature [34], [37]. The SU has no alternative but to observe the states of these channels when it tries to access them, and build its own estimations of channel probabilities. In this extended setting, we study our problem as an online shortest path problem, which has been widely studied in [38], [48], [49] for different kinds of cost functions. [38] proposed a Kullback-Leibler source routing (*KL-SR*) algorithm to an online routing problem with geometrically distributed delay in each link, which coincides with our link cost in the underlying graph of the shortest path problem shown in Figure 3(b).

6. We can choose remaining file size to be smaller than Δr_{i^*} to apply the static optimal policy.

7. For F being integer multiple of Δr_{i^*}, we have $F_1 = F$ and $F_2 = 0$, then the heuristic policy coincides with the max-throughput policy, which is also the dynamic optimal policy in view of Proposition 4.2.

For our purpose, we modify *KL-SR* algorithm; key differences being that we let the file size vary across the episodes, allowing a different underlying graph of the shortest path problem for each episode instead of the fixed underlying graph of the shortest path problem in [38]. Algorithm 1 describes our online implementation, where F^k is the file size to be transferred in the k-th episode. $n_i(k)$ denotes the number of times channel i has been sensed before the k-th episode and $\bar{p}_i(k)$ is the empirical average of channel i's available probability throughout the $k-1$ episodes so far. With $n_i(k)$ and $\bar{p}_i(k)$, the estimated available probability $\hat{p}_i(k)$ of channel i is then derived from the KL-based index in [38]. As mentioned in line 1 in Algorithm 1, the SU can choose one of the various policies according to which it wishes to perform the file transfer, i.e., dynamic optimal policy π^*, static optimal policy i_{so}, max throughput policy i^* or the heuristic policy π_{heu}, and then stick to that policy. Let $\tilde{\mathbb{E}}[T(\pi,F^k)]$ be the estimated expected transfer time of policy π at the k-th file by using the estimated parameter $\hat{p}_i(k)$ instead of p_i for all $i \in \mathcal{N}$ in (13). In line 7 in Algorithm 1, π^k will be computed as $\pi^k = \arg\min_{\pi\in\Pi(F^k)} \tilde{\mathbb{E}}[T(\pi,F^k)]$ for dynamic optimal policy; $\pi^k = \arg\min_{i\in\mathcal{N}} \tilde{\mathbb{E}}[T(i,F^k)]$ for static optimal policy and $\pi^k = \arg\max r_i\hat{p}_i(k)$ for max-throughput policy. For heuristic policy π_{heu}, π^k will be computed in the same way as described in Section 5.2 with estimated parameters $\{\hat{p}_i(k)\}_{i\in\mathcal{N}}$.

Algorithm 1 Online file transfer algorithm

1: Choose the type of policy to use: π^*, i_{so}, i^* or π_{heu}.
2: Apply static policy i in the i-th episode to transmit the file of size F^i for $i = 1, 2, \cdots, N$, and update $n_i(N+1)$, $\bar{p}_i(N+1)$ for $i \in \mathcal{N}$ at the end of the N-th episode
3: **for** $k \geq N+1$ **do**
4: Compute the estimated channel statistics $\{\hat{p}_i(k)\}_{i\in\mathcal{N}}$ according to the KL-based index in [38].
5: Given a file of size F^k, compute the policy π^k with $\{\hat{p}_i(k)\}_{i\in\mathcal{N}}$ and observe channel status for the whole file transfer process in this episode.
6: Update $n_i(k+1)$, $\bar{p}_i(k+1)$ for $i \in \mathcal{N}$.
7: **end for**

The performance of Algorithm 1 with varying file sizes (assuming bounded file size) is measured by its *regret* $\mathbb{E}[R(K)]$, which is defined as the cumulative difference of expected transfer time between policy π^k at k-th file and the targeted optimal policy $\pi_{tar} \in \{\pi^*, i_{so}, i^*, \pi_{heu}\}$ up to the K-th file. The regret analysis is nearly the same as Theorem 5.4 in [38]. Let $f(K) = \log(K) + 4\log(\log(K))$ and $H = F_{\max}/r_{\min}$, where $F_{\max}$ is the largest possible file size, $r_{\min} = \min_{i\in[N]} r_i$. Denote by $D = \max_\pi \mathbb{E}[T(\pi, F_{\max})]$ the longest expected transfer time and $\epsilon = (1 - 2^{-\frac{1}{4}})\Delta_{\min}/D$,

$$\Delta_{\min} = \min_{F\in(0,F_{\max}],\pi\in\Pi(F)/\pi_{tar}} \mathbb{E}[T(\pi,F)] - \mathbb{E}[T(\pi_{tar}(F),F)]$$

is the smallest non-zero difference of expected transfer time between any sub-optimal policy π and the targeted optimal policy π_{tar}. Let $p_{\min} = \min_{i\in[N]} p_i$. The regret bound of Algorithm 1 is given in the following theorem.

Theorem 5.3. The gap-dependent regret bound under Algorithm 1 is

$$\mathbb{E}[R(K)] \leq \frac{360NHf(K)}{\Delta_{\min} p_{\min}^2} + 2D\left(4H + s\sum_{i=1}^{n}\frac{1}{\epsilon^2 p_i^2}\right). \quad (17)$$

Proof: The proof is nearly the same as the analysis of Theorem 5.4 in Appendix G.B [38]. Here we only give the main modifications for our setting.

The first modification comes from the definition of 'arm'. In [38], each edge in the graph is treated as a different arm, that is, the status of each edge is observed and estimated separated. In our setting, each edge in the shortest path problem (Figure 3) represents one of N channels such that each policy (path) may observe one channel multiple times. Then, some summation terms in the proof, previously were over all edges (e.g., (12), (13) in [38]), are now over all N channels.

Second, the *KL-SR* algorithm in [38] for dynamic optimal policy works for a fixed source node, which can be interpreted as a fixed file size F. Our algorithm deals with the varying file size. Since file size $F_{\max} < \infty$, we only need to change parameter H to be the longest policy length for maximum file size $F_{\max}$ (instead of fixed file size F), $\Delta_{\min}$ to be the smallest non-zero difference of expected transfer time between any sub-optimal policy and targeted optimal policy for file size in $(0, F_{\max}]$ (instead of fixed file size F) and D to be the longest expected transfer time for file size $F_{\max}$ (instead of fixed file size F). Then, the proof will be carried over. □

The regret (17) scales linearly with the number of channels N, instead of the number of edges in the online shortest path problem [38], because each edge in our setting (see Figure 3) is chosen from one of N channels while each edge in [38] is treated as a different 'arm'.

6 Numerical Results

In this section, we present numerical results for file transfer time under four different policies in both offline setting (known p_i's) and online setting (unknown p_i's), using three different channel scenarios as in [21], [42]. Through these results, we show the significant time reduction achieved by the dynamic optimal, static optimal and heuristic polices over the max-throughput channel, in line with theoretical analysis.

6.1 Simulation Setup

We consider the experimental setup as an IEEE 802.22 system with 8 different channels. The time duration Δ is set to 100 ms, per IEEE 802.22 standard [4]. We use three different channel scenarios: *gradual*, *steep* and *lossy* [21], [42]. *Gradual* refers to a case where the available probability of the max-throughput channel is larger than 0.5. *Steep* is characterized by the available probability of each channel being either very high or very low. *Lossy* means that the available probability of the max-throughput channel is smaller than 0.5. The channel parameters used for simulation in the above three channel scenarios are given in Table 2. All simulations are run on a PC with AMD Ryzen 1700X and 32G RAM.

TABLE 2
Channel parameters in three channel scenarios

channel i	1	2	3	4	5	6	7	8
r_i (Mbps)	1.5	4.5	6	9	12	18	20	23
p_i (*gradual*)	0.95	0.85	0.75	0.65	0.4	0.3	0.2	0.1
p_i (*steep*)	0.9	0.25	0.2	0.18	0.17	0.16	0.15	0.14
p_i (*lossy*)	0.9	0.8	0.7	0.4	0.3	0.25	0.2	0.1

6.2 Computation of Dynamic Optimal Policy

To obtain the dynamic optimal policy in both offline and online cases, we utilize the mixed-integer programming formulation as in Section 5.1 to fasten the simulation speed and use the SCIP solver [50]. For SSP formulation as in Section 4.1, we use policy iteration as a solver. We select 10 files of sizes in the interval $(0, 7]$ (Mb) uniformly at random and compare the total time of computing the dynamic optimal policies of these 10 files from the mixed-integer programming formulation and SSP formulation. The policy iteration takes 77.94 seconds, while the SCIP solver only takes 0.23 seconds. We observe this because any dynamic programming procedure has to effectively first construct the underlying network for the shortest path problem, and then traverse all the possible paths from the source F to destination 0. This underlying network changes for every different F as well, rendering previous computations useless. However, the mixed-integer programming doesn't need the construction of a network to solve it and SCIP solver only needs to compute one possible solution to (16).

6.3 Online File Transfer Simulation

In the online file transfer problem, since the file size needs not be fixed and larger file sizes naturally take more time to transmit, it makes sense to normalize our performance metric across the range of file sizes and use max-throughput policy as our baseline policy. We define our metrics as *average time ratio* and *average throughput*. For an arbitrary sequence of files $\{F^k\}_{k\in\mathbb{Z}_+}$, the average time ratio at the K-th episode is defined as

$$\frac{1}{K}\sum_{k=1}^{K} T(\pi^k, F^k)/\mathbb{E}[T(i^*, F^k)], \quad (18)$$

and the average throughput is represented as

$$\frac{1}{K}\sum_{k=1}^{K} F^k/T(\pi^k, F^k). \quad (19)$$

Here, $T(\pi^k, F^k)$ is the measured transfer time of a file of size F^k applying the policy π^k at the k-th episode. Policy π^k is based on the estimated parameter, which is updated by the SU on the fly, as described in Section 5.3.

In our simulation, we generate 7000 files from $(0, 7]$ (Mb) uniformly at random to be used in Algorithm 1. The simulation is repeated 500 times to ensure stable results. We first observe the bottom row in Figure 4. The max-throughput policy achieves the largest average throughput while, counter-intuitively, has the longest transfer time in all channel cases. The reason is that the max-throughput policy computed by the SU is decided by the estimated parameters and can be the inferior policy, resulting in lower average throughput initially, which is an effect of imperfect knowledge of channel parameters. As time goes on, we can

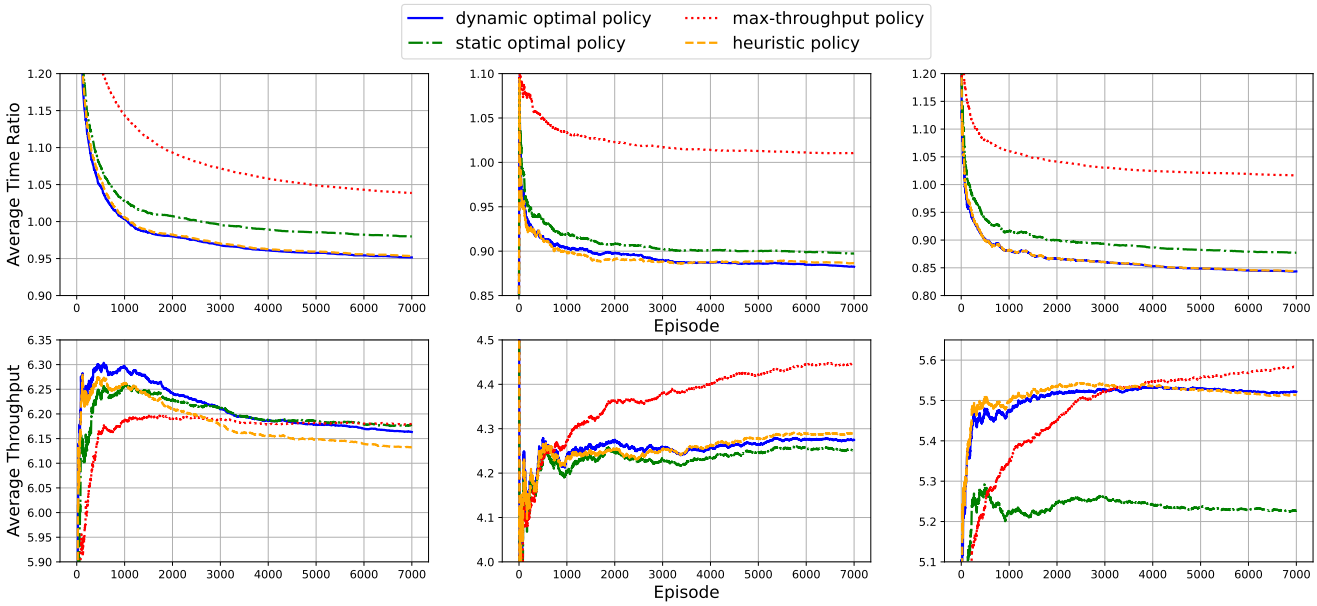
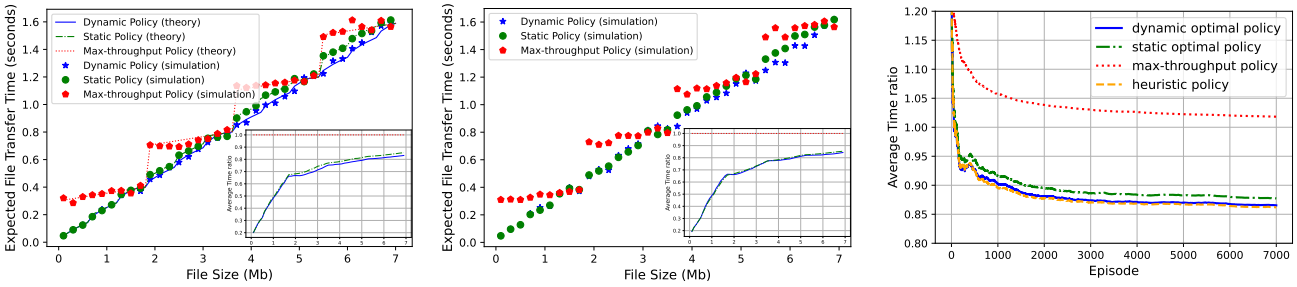


Fig. 4. Average time ratio (top row) and average throughput (bottom row) in the online setting. Channel scenarios from left to the right: Gradual; Steep; Lossy.



(a) Offline setting, no switching delay. (b) Offline setting, with switching delay. (c) Online setting, with switching delay.

Fig. 5. Effect of switching delay on the performance of (1) dynamic optimal, static optimal, and max-throughput policies in the offline setting with or without switching delay; (2) above three policies and heuristic policy in the online setting with switching delay over the lossy channels.

see the red curve eventually exceeds all other curves because the SU will eventually learn all the channel parameters well.

Next we focus on the average time ratio in the top row of Figure 4. We first observe that all curves eventually flatten out, signifying the convergence of Algorithm 1. In the gradual case, the average time ratio is above 95% for all three policies, implying that they don't obtain much reduction in time and the max-throughput channel is good to access when it is available for most of the time. However, as shown in the *steep* and *lossy* cases respectively, the dynamic optimal policy and heuristic policy, as well as the static optimal policy, can save over 10% time on average over the baseline. This observation is in line with Corollary 3.3 and Corollary 4.3 since the available probabilities of the max-throughput channel are very small in *steep* and *lossy* cases. Furthermore, the heuristic policy, in addition to keeping the complexity low, achieves similar transfer time to that of the dynamic optimal policy; at the same time performing better than the static optimal policy, as expected from Section 5.2.

6.4 OSA File Transfer with Switching Delay

In reality, switching from one channel to another also takes some time and could affect the file transfer time if the SU switches too often. In this section, we take the switching delay into consideration. Specifically, we set the switching delay to be 20 ms as in [51] (within the $\Delta = 100$ ms duration) and simulate the file transfer time in both offline and online settings.

We first consider the empirical file transfer time over the lossy channels without switching delay in Figure 5a. It indicates that dynamic optimal policy (blue curve) achieves the best performance (the lowest curve) and the simulation results are in line with the theoretical results. The inset shows the average time ratio (18) of each policy (smaller is better), where the average is taken over the file size from 0.1 Mb to F Mb on the x axis. This is to show the expected file transfer time of each policy compared to the max-throughput policy when the file size falls into a given range that is governed by different applications. Larger F leads to smaller average time ratio for each policy, supporting the discussion after Corollary 4.3. The average time ratio of $F = 7$ Mb in Figure 5a is 0.85 for static optimal policy and 0.83 for dynamic optimal policy, which is consistent with the top-right plot in Figure 4.

On the other hand, switching too frequently penalizes the performance of each policy, and the switching delay may outweigh the time saved by channel switching. The performance gap between dynamic optimal policy and static optimal policy becomes smaller in both offline and online settings when switching delay is taken into account, as shown

in Figure 5b (compared to Figure 5a) and 5c (compared to the top-right plot in Figure 4) over the lossy channels. Additionally, our proposed heuristic policy (orange curve) performs slightly better than the dynamic optimal policy and is the best among the four policies in Figure 5c. This depicts that the advantage of our heuristic strategy is not only in terms of computational complexity but also in terms of less channel switching.

7 Extension to Markovian Channels

The OSA literature has long focused on the throughput-oriented policies for both Bernoulli channels [2], [6], [22] and Markovian channels, i.e., each channel can be modeled as a two-state discrete-time Markov chain [8], [14], [30], [32], [33]. For the file transfer problem, we have presented static and dynamic policies for Bernoulli channels in Section 3 and 4. However, for dynamic policies over Markovian channels, the problem is beyond the SSP framework described in Section 4 since the transition probability to the next state (remaining file size) also depends on the past action (last chosen channel), instead of merely the current state and the current action (as in the SSP problem). Augmenting the state space to accommodate for this extra dependency transforms the problem into a POMDP problem, which generally has no known structured solution and is therefore intractable [31]. For this reason, we extend the file transfer problem to Markovian channels and focus mainly on static policies.

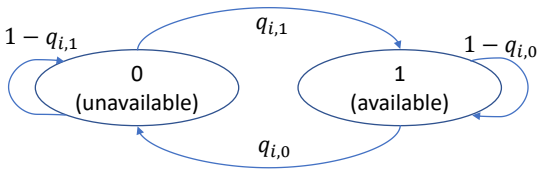


Fig. 6. Diagram of the state transition of channel $i \in \mathcal{N}$ modeled as a two-state Markov chain.

In this section, the state of any channel $i \in \mathcal{N}$ takes the form of a two-state Markov chain $\{Y_i(k)\}_{k\in\mathbb{N}}$. For any time step $k \in \mathbb{N}$, we have $Y_i(k) \in \{0,1\}$, and $P(Y_i(k+1) = 1|Y_i(k) = 0) = q_{i,1}$, $P(Y_i(k+1) = 0|Y_i(k) = 1) = q_{i,0}$, denoting the transition probabilities, as shown in Figure 6. The stationary distribution of channel i is denoted by $\pi_i \triangleq q_{i,1}/(q_{i,1}+q_{i,0})$. Denote by $c_i \in [0,1]$ an arbitrary probability that channel i is available at the beginning of the file transfer process. Similar to the notations used in Proposition 3.1, given a file of size F, we define $k_i \triangleq \lfloor F/\Delta r_i \rfloor \in \mathbb{Z}_+$ and $\alpha_i \triangleq F/\Delta r_i - k_i \in [0,1)$. Then, we show the closed-form expected file transfer time for static policies in correlated channels as follows.

Proposition 7.1. Given a file of size F, the expected transfer time of a static policy in channel $i \in \mathcal{N}$ is

$$\mathbb{E}[T(i,F)] = \Delta\left(\left(k_i + \mathbb{1}_{\{\alpha_i>0\}} - 1\right)\frac{q_{i,0}}{q_{i,1}} + \frac{1-c_i}{q_{i,1}}\right) + \frac{F}{r_i}. \tag{20}$$

Proof: Similar to the proof of Proposition 3.1, we can decompose the file size F as

$$F = k_i\Delta r_i + \alpha_i \Delta r_i \tag{21}$$

such that the SU can fully utilize k_i time slots for the file transmission and α_i is the fraction of the Δ seconds utilized for file transfer toward the end. Initially, the available probability of channel i is c_i. The expected waiting time $\mathbb{E}[T^1_{wait}]$ for the first successful transmission of Δr_i data is given as

$$\mathbb{E}[T^1_{wait}] = 0 \cdot c_i + \sum_{k=1}^{\infty} k(1-c_i)(1-q_{i,1})^{k-1}q_{i,1} = \frac{1-c_i}{q_{i,1}}. \tag{22}$$

For the m-th transmission of the amount of Δr_i data ($m = 2, 3, \cdots, k_i$), the expected waiting time $\mathbb{E}[T^m_{wait}]$ conditioned on channel i being in state 1 is

$$\mathbb{E}[T^m_{wait}] = 0 \cdot (1-q_{i,0}) + \sum_{k=1}^{\infty} kq_{i,0}(1-q_{i,1})^{k-1}q_{i,1} = \frac{q_{i,0}}{q_{i,1}}. \tag{23}$$

Note that $\{T^m_{wait}\}_{m=1,2,\cdots,k_i}$ are mutually independent to each other. Let the constant $T_{tran} \triangleq F/r_i = \Delta(k_i + \alpha_i)$ be the total successful transmission time (from (21)). Then, the transfer time can be written as

$$\begin{aligned} T(i,F) &= \sum_{m=1}^{k_i}(T^m_{wait} + \Delta) + \mathbb{1}_{\{\alpha_i>0\}}T^{k_i+1}_{wait} + \Delta\alpha_i \\ &= T_{tran} + T^1_{wait} + \sum_{m=2}^{k_i+\mathbb{1}_{\{\alpha_i>0\}}} T^m_{wait}. \end{aligned}$$

Taking the expectation of the equation above, along with (22) and (23), yields (20). □

The static optimal policy over Markovian channels is then derived from $i_{so}(F) = \arg\min_{i\in\mathcal{N}} \mathbb{E}[T(i,F)]$. By choosing transition probability $q_{i,1} = 1-q_{i,0}$ and initial state $c_i = q_{i,1}$, the Markovian channel reduces to the Bernoulli channel (i.e., $q_{i,1} \equiv p_i$ and $q_{i,0} \equiv 1-p_i$), and (20) coincides with (4). From (20) we have

$$\begin{aligned} \mathbb{E}[T(i,F)] &= \Delta\left((k_i + \mathbb{1}_{\{\alpha_i>0\}})\frac{q_{i,0}}{q_{i,1}} + \frac{1-c_i-q_{i,0}}{q_{i,1}}\right) + \frac{F}{r_i} \\ &\geq \Delta\left((k_i+\alpha_i)\frac{q_{i,0}}{q_{i,1}} + \frac{1-c_i-q_{i,0}}{q_{i,1}}\right) + \frac{F}{r_i} \\ &= \frac{F}{r_i\pi_i} + \Delta\frac{1-c_i-q_{i,0}}{q_{i,1}}, \end{aligned} \tag{24}$$

where the inequality comes from $\mathbb{1}_{\{\alpha_i>0\}} \geq \alpha_i$. When the initial channel state c_i satisfies the condition $1-c_i-q_i \geq 0$, (24) is lower bounded by $F/r_i\pi_i$, where $r_i\pi_i$ is the standard criterion to choose the max-throughput policy in the single channel [33], [34], i.e., $i^* = \arg\max_{i\in\mathcal{N}} r_i\pi_i$. An example for such condition would be that channel i has positive correlation (i.e., $1-q_{i,1}-q_{i,0} > 0$) and is in the stationary regime from the beginning (i.e., $c_i = \pi_i$). Besides, note that when the file size is an integer multiple of Δr_{i^*}, Proposition 3.1 for Bernoulli channels shows that the expected file transfer time is $F/r_{i^*}p_{i^*}$, while for Markovian channel i^* with positive correlation and $c_{i^*} = \pi_{i^*}$, Proposition 7.1 for Markovian channels indicates that $\mathbb{E}[T(i^*,F)] > F/r_{i^*}\pi_{i^*}$ and the equality never holds. Both imply that the max-throughput channel does not necessarily minimize the expected file transfer time.

Now, we analyze the impact of the correlation on the expected file transfer time in each channel. Consider the case

where two channels i and j share the same stationary distribution $\pi_i = \pi_j$, initial state probability $c_i = c_j$ and channel rate $r_i = r_j$ but with different correlation, i.e., $q_{i,1} = \beta q_{j,1}$ and $q_{i,0} = \beta q_{j,0}$, where $\beta \in (0, 1)$. Then, we know that both channels share the same long-term throughput. However, from Proposition 7.1 we have

$$\mathbb{E}[T(i,F)] - \mathbb{E}[T(j,F)] = \Delta(1-c_i)\left(\frac{1}{q_{i,1}} - \frac{1}{q_{j,1}}\right) > 0, \quad (25)$$

which demonstrates that larger correlation (smaller β) leads to larger expected file transfer time (worse performance).

8 Conclusion and Future Work

In this paper, we have developed a theoretical framework for the file transfer problem, where channels are modeled as independent Bernoulli process, to provide the accurate file transfer time for both static and dynamic policies. We pointed out that the max-throughput channel does not always minimize the file transfer time. We demonstrated in our our analysis that our proposed policies can obtain significant reduction in file transfer time over the max-throughput policy for small file sizes or when the max-throughput channel has very high rate but with low available probability, as typically the case in reality. In addition, we have extended the theoretical analysis to Markovian channels and static polices, showing that greater correlation can compromise the performance.

When the wireless devices work in the outside network, the effect of a propagation environment on a radio signal needs to be considered, i.e., Rayleigh fading, Rician fading and Nakagami fading, which leads to the varying channel environment. Our future work includes the extension to the channels with multiple rates such that the rate will be treated as a general random variable sampled from some probability distribution or finite-state Markov chain, instead of Bernoulli random variable or two-state Markov chain studied in this paper.

References

[1] J. Hu, V. Doshi, and D. Y. Eun, "Opportunistic spectrum access: Does maximizing throughput minimize file transfer time?" in *19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, Philadelphia, PA, USA, 2021.

[2] K. Zaheer, M. Othman, M. H. Rehmani, and T. Perumal, "A survey of decision-theoretic models for cognitive internet of things (ciot)," *IEEE Access*, vol. 6, pp. 22 489–22 512, 2018.

[3] "FCC increases unlicensed wireless operations in tv white spaces," Dec 2020. [Online]. Available: https://www.fcc.gov/document/fcc-increases-unlicensed-wireless-operations-tv-white-spaces-0

[4] "IEEE standard for information technology," *IEEE Std 802.22-2019*, pp. 1–1465, 2020.

[5] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 310–323, 2018.

[6] O. Avner and S. Mannor, "Multi-user communication networks: A coordinated multi-armed bandit approach," *IEEE/ACM Transactions on Networking*, vol. 27, no. 6, pp. 2192–2207, 2019.

[7] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE IEEE Journal on Selected Areas in Communications.*, vol. 37, no. 10, pp. 2239–2250, 2019.

[8] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5547–5567, 2010.

[9] M. A. Yadav, Y. Li, G. Fang, and B. Shen, "Deep q-network based reinforcement learning for distributed dynamic spectrum access," in *2022 IEEE 2nd International Conference on Computer Communication and Artificial Intelligence (CCAI)*, 2022, pp. 1–6.

[10] P. Yang, B. Li, J. Wang, X. Li, Z. Du, Y. Yan, and Y. Xiong, "Online sequential channel accessing control: A double exploration vs. exploitation problem," *IEEE Transactions on Wireless Communications*, vol. 14, no. 8, pp. 4654–4666, 2015.

[11] J. Zuo, X. Zhang, and C. Joe-Wong, "Observe before play: Multi-armed bandit with pre-observations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, New York, USA, 2020.

[12] M. Monemian, M. Mahdavi, and M. J. Omidi, "Optimum sensor selection based on energy constraints in cooperative spectrum sensing for cognitive radio sensor networks," *IEEE Sensors Journal*, vol. 16, no. 6, pp. 1829–1841, 2015.

[13] C. Gan, R. Zhou, J. Yang, and C. Shen, "Cost-aware learning and optimization for opportunistic spectrum access," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 1, pp. 15–27, 2018.

[14] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-learning-based wireless resource allocation with application to vehicular networks," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 341–356, 2019.

[15] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5g: Ran, core network and caching solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018.

[16] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues, and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2018.

[17] A. Paul, P. Kunarapu, A. Banerjee, and S. P. Maity, "Spectrum sensing in cognitive vehicular networks for uniform mobility model," *IET Communications*, vol. 13, no. 19, pp. 3127–3134, 2019.

[18] G. V. Rossi and K. K. Leung, "Optimised csma/ca protocol for safety messages in vehicular ad-hoc networks," in *2017 IEEE Symposium on Computers and Communications (ISCC)*, Heraklion, Greece.

[19] S. Sodagari, B. Bozorgchami, and H. Aghvami, "Technologies and challenges for cognitive radio enabled medical wireless body area networks," *IEEE Access*, vol. 6, pp. 29 567–29 586, 2018.

[20] H. Gupta, A. Eryilmaz, and R. Srikant, "Low-complexity, low-regret link rate selection in rapidly-varying wireless channels," in *IEEE INFOCOM*, Honolulu, HI, USA, 2018.

[21] ——, "Link rate selection using constrained thompson sampling," in *IEEE INFOCOM*, Paris, France, 2019.

[22] P. Zhu, J. Li, D. Wang, and X. You, "Machine-learning-based opportunistic spectrum access in cognitive radio networks," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 38–44, 2020.

[23] M. Almasri, A. Mansour, C. Moy, A. Assoum, D. Le Jeune, and C. Osswald, "Managing single or multi-users channel allocation for the priority cognitive access," in *2020 28th European Signal Processing Conference (EUSIPCO)*, Amsterdam, NL.

[24] Y. Wu, F. Hu, S. Kumar, Y. Zhu, A. Talari, N. Rahnavard, and J. D. Matyjas, "A learning-based qoe-driven spectrum handoff scheme for multimedia transmissions over cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 11, pp. 2134–2148, 2014.

[25] H. Cao, H. Tian, J. Cai, A. S. Alfa, and S. Huang, "Dynamic load-balancing spectrum decision for heterogeneous services provisioning in multi-channel cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5911–5924, 2017.

[26] I. Dimitriou and N. Pappas, "Stable throughput and delay analysis of a random access network with queue-aware transmission," *IEEE Transactions on Wireless Communications*, vol. 17, no. 5, pp. 3170–3184, 2018.

[27] X.-L. Huang, X.-W. Tang, and F. Hu, "Dynamic spectrum access for multimedia transmission over multi-user, multi-channel cognitive radio networks," *IEEE Transactions on Multimedia*, vol. 22, no. 1, pp. 201–214, 2019.

[28] A. Iqbal, R. Hussain, A. Shakeel, I. L. Khan, M. A. Javed, Q. U. Hasan, B. M. Lee, and S. A. Malik, "Enhanced spectrum access for qos provisioning in multi-class cognitive d2d communication system," *IEEE Access*, vol. 9, pp. 33 608–33 624, 2021.

[29] Lifewire, "Ever wonder what makes email files so large?" 2020. [Online]. Available: https://www.lifewire.com/what-is-the-average-size-of-an-email-message-1171208

[30] Q. Zhao, L. Tong, A. Swami, and Y. Chen, "Decentralized cognitive mac for opportunistic spectrum access in ad hoc networks: A

pomdp framework," *IEEE Journal on selected areas in communications*, vol. 25, no. 3, pp. 589–600, 2007.
[31] Y. Liu and M. Liu, "An online approach to dynamic channel access and transmission scheduling," in *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Hangzhou, China, 2015.
[32] Q. Zhao, S. Geirhofer, L. Tong, and B. M. Sadler, "Opportunistic spectrum access via periodic channel sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 785–796, 2008.
[33] C. Tekin and M. Liu, "Online learning in opportunistic spectrum access: A restless bandit approach," in *IEEE INFOCOM*, Shanghai, China, 2011.
[34] W. Dai, Y. Gai, and B. Krishnamachari, "Efficient online learning for opportunistic spectrum access," in *IEEE INFOCOM*, Orlando, FL, USA, 2012.
[35] Y. H. Jung and A. Tewari, "Regret bounds for thompson sampling in episodic restless bandit problems," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2019.
[36] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 257–265, 2018.
[37] A. Mohamedou, A. Sali, B. Ali, M. Othman, and H. Mohamad, "Bayesian inference and fuzzy inference for spectrum sensing order in cognitive radio networks," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, p. e2916, 2017.
[38] M. S. Talebi, Z. Zou, R. Combes, A. Proutiere, and M. Johansson, "Stochastic online shortest path routing: The value of feedback," *IEEE Transactions on Automatic Control*, vol. 63, no. 4, pp. 915–930, 2017.
[39] X. Tan, L. Zhou, H. Wang, Y. Sun, H. Zhao, B.-C. Seet, J. Wei, and V. C. Leung, "Cooperative multi-agent reinforcement learning based distributed dynamic spectrum access in cognitive radio networks," *IEEE Internet of Things Journal*, 2022.
[40] Y. Pei, Y.-C. Liang, K. C. Teh, and K. H. Li, "How much time is needed for wideband spectrum sensing?" *IEEE Transactions on Wireless Communications*, vol. 8, no. 11, pp. 5466–5471, 2009.
[41] O. H. Toma, M. Lopez-Benitez, D. K. Patel, and K. Umebayashi, "Estimation of primary channel activity statistics in cognitive radio based on imperfect spectrum sensing," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2016–2031, 2020.
[42] J. C. Bicket, "Bit-rate selection in wireless networks," Master's thesis, Massachusetts Institute of Technology, 2005.
[43] D. Bertsekas, *Reinforcement Learning and Optimal Control*. Athena Scientific, 2019.
[44] F. Della Croce, F. Salassa, and R. Scatamacchia, "An exact approach for the 0–1 knapsack problem with setups," *Computers & Operations Research*, vol. 80, pp. 61–67, 2017.
[45] M. Conforti, G. Cornuéjols, G. Zambelli *et al.*, *Integer programming*. Springer, 2014, vol. 271.
[46] D. Bertsekas, *Dynamic programming and optimal control: Volume I*. Athena scientific, 2012, vol. 1.
[47] A. Mantuano, "File size basics," 2016. [Online]. Available: https://techdocs.blogs.brynmawr.edu/5523
[48] Y. Gai, B. Krishnamachari, and R. Jain, "Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1466–1478, 2012.
[49] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *International conference on machine learning*, Atlanta, GA, USA, 2013.
[50] S. Maher, M. Miltenberger, J. P. Pedroso, D. Rehfeldt, R. Schwarz, and F. Serrano, "PySCIPOpt: Mathematical programming in python with the SCIP optimization suite," in *Mathematical Software – ICMS 2016*. Springer International Publishing, 2016, pp. 301–307.
[51] M. A. Nezhad, L. Cerdà-Alabern, B. Bellalta, and M. G. Zapata, "A semi–dynamic, game based and interference aware channel assignment for multi–radio multi–channel wireless mesh networks," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 14, no. 3, pp. 200–213, 2013.

Jie Hu received his B.E degree in communication engineering from Wuhan University of Technology, Wuhan, China, and Masters degree in electrical engineering from Northwestern University, Evanston, IL, USA. He is a Ph.D. student in the Department of Electrical and Computer Engineering at North Carolina State University. His current research interests are in the area of machine learning in dynamic spectrum access problem.

Vishwaraj Doshi received his B.E. degree in mechanical engineering from the University of Mumbai, Mumbai, MH, India, and Masters degree in Operations Research from North Carolina State University, Raleigh, NC, USA. He is currently pursuing his Ph.D. degree with the Operations Research Graduate Program at North Carolina State University. His primary research interests include design of randomized algorithms on graphs, and epidemic models on networks.

Do Young Eun (Senior Member, IEEE) received his B.S. and M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, Korea, in 1995 and 1997, respectively, and Ph.D. degree from Purdue University, West Lafayette, IN, in 2003. Since August 2003, he has been with the Department of Electrical and Computer Engineering at North Carolina State University, Raleigh, NC, where he is currently a professor. His research interests include distributed optimization for machine learning, machine learning algorithms for networks, distributed and randomized algorithms for large social networks and wireless networks, epidemic modeling and analysis, graph analytics and mining techniques with network applications. He has been a member of Technical Program Committee of various conferences including IEEE INFOCOM, ICC, Globecom, ACM MobiHoc, and ACM Sigmetrics. He is serving on the editorial board of IEEE Transactions on Network Science and Engineering, and previously served for IEEE/ACM Transactions on Networking and Computer Communications Journal, and was TPC co-chair of WASA'11. He received the Best Paper Awards in the IEEE ICCCN 2005, IEEE IPCCC 2006, and IEEE NetSciCom 2015, and the National Science Foundation CAREER Award 2006. He supervised and co-authored a paper that received the Best Student Paper Award in ACM MobiCom 2007.