

Periodic Bandits and Wireless Network Selection

Shunhao Oh

Department of Computer Science, National University of Singapore
ohoh@u.nus.edu

Anuja Meetoo Appavoo

Department of Computer Science, National University of Singapore
anuja@comp.nus.edu.sg

Seth Gilbert

Department of Computer Science, National University of Singapore
seth.gilbert@comp.nus.edu.sg

Abstract

Bandit-style algorithms have been studied extensively in stochastic and adversarial settings. Such algorithms have been shown to be useful in multiplayer settings, e.g. to solve the wireless network selection problem, which can be formulated as an adversarial bandit problem. A leading bandit algorithm for the adversarial setting is EXP3. However, network behavior is often repetitive, where user density and network behavior follow regular patterns. Bandit algorithms, like EXP3, fail to provide good guarantees for periodic behaviors. A major reason is that these algorithms compete against fixed-action policies, which is ineffective in a periodic setting.

In this paper, we define a periodic bandit setting, and periodic regret as a better performance measure for this type of setting. Instead of comparing an algorithm’s performance to fixed-action policies, we aim to be competitive with policies that play arms under some set of possible periodic patterns F (for example, all possible periodic functions with periods $1, 2, \dots, P$). We propose Periodic EXP4, a computationally efficient variant of the EXP4 algorithm for periodic settings. With K arms, T time steps, and where each periodic pattern in F is of length at most P , we show that the periodic regret obtained by Periodic EXP4 is at most $O(\sqrt{PKT \log K + KT \log |F|})$. We also prove a lower bound of $\Omega(\sqrt{PKT + KT \frac{\log |F|}{\log K}})$ for the periodic setting, showing that this is optimal within log-factors. As an example, we focus on the wireless network selection problem. Through simulation, we show that Periodic EXP4 learns the periodic pattern over time, adapts to changes in a dynamic environment, and far outperforms EXP3.

2012 ACM Subject Classification Theory of computation → Online learning algorithms

Keywords and phrases multi-armed bandits, wireless network selection, periodicity in environment

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.149

Category Track C: Foundations of Networks and Multi-Agent Systems: Models, Algorithms and Information Management

Related Version A full version of the paper is available at [21] <https://arxiv.org/abs/1904.12355>.

Supplement Material Source code: <https://github.com/Ohohcakester/PeriodicEXP4-Source>

Acknowledgements This project was funded by Singapore Ministry of Education Grant MOE2018-T2-1-160 (Beyond Worst-Case Analysis: A Tale of Distributed Algorithms).

1 Introduction

The *multi-armed bandit* problem is an online learning problem in which a player has access to a set of choices (i.e., “arms”) each of which provides some reward (i.e., “gain”). At each time step, the player chooses an arm and gets some reward. In stochastic variants, rewards are determined by some probabilistic distribution. In adversarial variants, an adversary specifies



© Shunhao Oh, Anuja Meetoo Appavoo, and Seth Gilbert;
licensed under Creative Commons License CC-BY

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).

Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;

Article No. 149; pp. 149:1–149:15



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



the rewards. Amazingly, even when rewards are adversarially chosen, the player can do fairly well! For example, the EXP3 algorithm [6] minimizes the player’s “regret”, ensuring that the player does almost as well as if she had selected the single fixed best arm throughout. Another fascinating property of bandit algorithms is that they work well in multi-player settings [27, 16], converging to close variants of a Nash equilibrium.

Recently, it has been shown that bandit-style algorithms can efficiently solve the *wireless network selection problem*, yielding good performance both in theory and in practice [1, 2, 7]. In this problem, each user has access to a collection of networks (e.g., a few different WiFi networks and a 4G connection); the goal is to pick networks with higher data rates. Selecting the best network is challenging, especially in dynamic environments where the “best” network changes over time, as users move and network bandwidth fluctuates. This can be modeled as an adversarial bandit problem and solved with EXP3 and its variants.

Bandit algorithms have one major weakness in dynamic settings (such as wireless network settings): they are designed to learn the average payoff of each arm, and to converge to the arm that provides the best average performance. In the stochastic case, this is exactly what you want. In the adversarial case, it leads to minimum regret, i.e., the user does almost as well as if they knew the best network in advance. If, however, the situation is changing over time, and especially if it is changing in some predictable manner, then learning the average payoff of each arm is not productive.

Periodic, repetitive patterns are a particularly common type of dynamic behavior. Take, for example, the problem of network selection. Network behavior is often repetitive, with user density and network quality following regular patterns: for example, office WiFi networks have no users at night, their performance drops when workers arrive in the morning, and the performance improves again during lunch hour. Other networks are clogged with streaming video during lunch hour and in the evenings. Periodic patterns are ubiquitous.

Unfortunately, bandit algorithms will fail badly in the case of periodic behavior. As an example, suppose a player is playing a slot machine with two arms. The first arm gives a reward of 1 when pulled on odd-numbered hours and 0 otherwise, while the second arm does the reverse, with a reward of 1 on even-numbered hours and 0 otherwise. In this simple case, a bandit algorithm will never learn this pattern, instead converging to the best single-action policy; and the best policy can only reap half of the maximum reward. The player will receive an average payout of only $1/2$ per selection, despite a very predictable pattern. And when this case is extended to cycle among K arms, the best fixed choice of arm gives only $1/K$ of the total obtainable reward. Thus, algorithms like EXP3 that minimize the regret do not guarantee good performance on periodic problems.

1.1 Contributions

Our goal in this paper is to develop an efficient adversarial bandit algorithm for periodic settings, and to demonstrate the effectiveness of this algorithm in the context of the wireless network selection problem, yielding a new approach to network selection in dynamic, periodic environments. The first step is to establish the right metric by which to evaluate bandit algorithms. The performance of an adversarial bandit algorithm is heavily characterized by the definition of “regret,” which forms the baseline that it competes against. And traditionally, the regret is computed with respect to the best fixed strategy.

For the periodic bandit setting, we define a better performance measure, “periodic regret”, which compares an algorithm’s performance against the best periodic choice of arms. No choice of period may match the input data perfectly, but the goal of periodic regret is to compare against the best choice. Moreover, we provide a generalized notion of periodicity, so that this notion of periodic regret can capture different types of patterned behavior.

Next, we develop an algorithm that minimizes periodic regret, Periodic EXP4, a computationally efficient variant of EXP4 (Exponential-weight algorithm for Exploration and Exploitation using Expert advice) [6]. We show that the algorithm minimizes periodic regret in the following sense: with K arms, $|F|$ possible periods, with each possible period of at most length P , then in an execution of length T the periodic regret is at most $O(\sqrt{PKT \log K + KT \log |F|})$. We also prove a lower bound of $\Omega(\sqrt{PKT + KT \frac{\log |F|}{\log K}})$ on periodic regret in an adversarial setting, showing that this is optimal within log-factors. An important aspect of Periodic EXP4 is that it is a polynomial time algorithm: we leverage the structure provided by the target periodic patterns to reduce the computational complexity. This is in contrast to EXP4 which requires exponential time and space in this context.

The other major contribution of this paper is a new algorithm for network selection that is especially optimized for environments with periodic, patterned behaviors. We simulate the network selection problem, comparing Periodic EXP4 to EXP3 and to a “randomized optimal” omniscient solution. (We have previously seen in [1] that these types of simulations are reasonably predictive of real-world behavior.)

Our first observation is that Periodic EXP4 does in fact efficiently learn periodic patterns and adapts relatively quickly to changes in network data rates (both discrete and continuous). We also see that Periodic EXP4 does indeed outperform EXP3 in periodic settings, as expected, potentially yielding significant real-world improvements.

Our second question involved the robustness of Periodic EXP4 to noisy patterns. Real-world periodic patterns are rarely perfectly periodic, suffering noise and variance. We experiment with noisy patterns, and see that Periodic EXP4 continues to work well.

Finally, our third set of experiments looked at the performance of Periodic EXP4 in the context of user mobility. We simulate several scenarios where users change location over time, leading to changes in which networks they can access (and hence changes in the load on those networks). For example, we imagine a typical office scenario where users arrive at the office in the morning, take a break for lunch, return to work, and then head home at the end of the day. We observe that Periodic EXP4 can also learn this type of periodic behavior, again, learning to adapt the users’ network selection in a near-optimal fashion. In fact, we compare two versions of the algorithm: one in which the algorithm is notified when networks become unavailable, and one in which it is not – we observe that even in the latter case where it is completely oblivious to the changes, the user strategy converges to near-optimal choices.

Overall, we conclude that periodic adversarial bandit algorithms may have significant value, that Periodic EXP4 is an efficient algorithm for the problem, and that it yields a potentially interesting and useful approach to network selection.

2 Related work

In this section, we discuss relevant work done on bandit algorithms, and state-of-art wireless network selection approaches. Multi-armed bandit techniques have been successfully applied to wireless network selection [1, 2, 7]. They have also been considered for other resource selection problems, such as channel selection [13, 27], selection of the right sensors to query in a sensor network [14], and selection of replica server for content distribution networks [28].

Many variations of bandit problems have been studied, in both stochastic and adversarial settings. EXP3 is the most well-known algorithm for the standard adversarial bandit problem. With K arms and T time steps, it establishes a pseudo-regret upper bound of $O(\sqrt{KT \log K})$, which almost matches the lower bound of $\Omega(\sqrt{KT})$ [6]. The $\log K$ gap in the bounds has been recently closed by [5] bringing the upper bound down to $O(\sqrt{KT})$. But, these bound the regret against the best single-action policy, limiting their usefulness in a periodic setting.

A related problem is that of bandits with expert advice, defined in the same paper [6]. It defines a more general notion of regret, by competing against the best policy from a set. With K arms, T time steps and N experts, the EXP4 algorithm gives a pseudo-regret bound of $O(\sqrt{KT \log N})$. However, its possibly high running time and memory cost limit its use in practice. There are other algorithms for bandits with expert advice, like Context-FTPL. The latter is more computationally efficient, but has a weaker regret bound [26]. A lower bound of $\Omega(\sqrt{KT \frac{\log N}{\log K}})$ [23] has been shown, but the $\log K$ gap in bounds has not been closed.

An equivalent formulation of our generalized periodic regret (explained later in Section 4.2) has been briefly discussed in [10, Chapter 4.2.1], phrased as a contextual bandit problem where the algorithm competes against the best context set from a class of context sets. The possible use of EXP4 is mentioned, but an alternative algorithm with a weaker regret bound is instead discussed as it has a reasonable polynomial-time performance unlike EXP4.

While much of the existing literature assume a single best arm, there are other efforts to look beyond this. One approach to the stochastic version of the problem is to allow reward distributions of the arms to occasionally change [9, 22]. Our work on the other hand is fully adversarial, and makes no assumptions on the rewards produced by the adversary.

Numerous wireless network selection approaches have been proposed. Some are centralized [3, 8, 18, 25]; hence, not scalable and limited to managed networks. A number of distributed approaches have been proposed, with various limitations. Some rely on coordination from networks [15], while others require cooperation of wireless devices [12]. Others assume global knowledge [20, 4, 19], or availability of some information [30, 11]. A continuous-time multi-armed bandit approach in a stochastic setting has been considered in [29]. A similar setting to ours, though non-periodic and in the stochastic setting, is considered in [7].

3 Wireless Network Selection

Here, we describe the wireless network selection problem, discuss the periodicity of events in wireless environments, and formulate the network selection problem as a bandit problem.

3.1 Wireless network selection problem.

We consider an environment with multiple wireless devices and heterogeneous wireless networks, such as the one depicted in Figure 1. The latter illustrates four mobile users with their (active) mobile devices, and five wireless networks, namely four WiFi networks and a cellular network (represented using 3 cellular base stations). The wireless networks have limited areas of coverage. Hence, each mobile device may have access to a different set of wireless networks depending on their location, e.g. different networks are available at home and at the office. The bandwidths of wireless networks may also vary with time. Each mobile device aims to quickly identify and associate with the best network, which may vary over time, to maximize their data rates.

Mobile users tend to have daily routines that follow repetitive patterns - going to the office each morning, lunch at noon, returning home in the evening; these activities are performed at fixed times each weekday. Figure 1 broadly depicts the daily routine of a mobile user, Alice. Network behavior, which is affected by user density, is also often repetitive and follows a regular pattern. For example, the available bandwidth of office WiFi networks is likely to be higher during lunch hours, where the office is nearly empty. A good network selection protocol learns and adapts to periodic patterns in network quality for better performance.



■ **Figure 1** Mobile devices with access to a different set of wireless networks as the user moves.

3.2 Wireless network selection as a bandit problem

A device must be aware of the bit rate it can observe from each network to perform an optimal network selection. While this information is unknown at the time of selection, the device can estimate the achievable bit rate by exploring the networks. The network selection problem can be seen as a multi-armed bandit problem in a multi-player setting. A mobile device is a player, and each network can be considered as an arm. Every so often (e.g. once per minute), a device selects a network (analogous to pulling an arm) and observes a bit rate (gain) for that network. The gain from other networks is unknown to the device. Given that mobile devices operate in a dynamic environment, they must continuously explore and adapt to changes, by deciding which networks to select in sequence. The goal of each device is to maximize its cumulative gain over time. Since the quality of a wireless network is affected by its number of clients, other mobile devices in the environment may be considered to be adversaries. We hence use the adversarial setting. A leading bandit algorithm in this setting is EXP3.

4 Periodic Bandit Problem

In this section, we introduce the periodic bandit problem and discuss periodic regret.

We consider a general bandit problem. On each time step, an algorithm is allowed to pick any one out of K possible arms, and each arm produces a certain amount of reward. These rewards are unknown to the algorithm, which can only observe the reward of the arm it picked. We aim to maximize the total reward obtained by the algorithm. We study the adversarial setting with a possibly adaptive adversary, which decides on the distribution of rewards at each time step, taking into consideration the outcomes of past random events.

Let K be the number of arms. The set of arms is $[K] := \{1, 2, \dots, K\}$. Let $x_i(t) \in [0, 1]$ be the reward earned by arm $i \in [K]$ at time step t . Let $a(t) \in [K]$ be the arm played by the algorithm at time t . Let T be the total number of time steps. The set of time steps is $[T] := \{1, 2, \dots, T\}$. Thus, the total reward earned by the algorithm after T iterations is $\sum_{t=1}^T x_{a(t)}(t)$. The commonly used performance measure for bandit algorithms is regret. Regret compares the total reward obtained by the algorithm against a “best possible” reward “OPT” after some number of time steps T . Different types of regret compare the algorithm’s result to different notions of the optimal result.

We can define a form of regret where OPT is allowed to pick any arm in $[K]$ at each time step. For later reference we will refer to this as full regret, defined as follows:

$$R_{full}(T) = \sum_{t=1}^T \max_{i \in [K]} E[x_i(t)] - E\left[\sum_{t=1}^T x_{a(t)}(t)\right]$$

The above definition uses what is commonly known as pseudo-regret, rather than expected regret. For the rest of this paper, we will often refer to pseudo-regret as simply “regret”. Expectations are taken over the possible randomness of the algorithm and adversary.

In most studies of adversarial bandits, a weaker definition of regret is used. This is because full regret uses too powerful an adversary, and it is impossible to achieve better than linear expected full regret in the worst case (we include a proof in the full version of the paper [21]). Therefore, it is common to define a notion of regret where OPT is required to use the same arm for all T time steps. We refer to this as weak regret, defined as follows:

$$R_{weak}(T) = \max_{i \in [K]} \sum_{t=1}^T E[x_i(t)] - E\left[\sum_{t=1}^T x_{a(t)}(t)\right]$$

Weak regret however, severely limits what OPT can do, and being competitive with an algorithm that can only pick one arm and stick to it may not be a very strong result.

4.1 Periodic Regret

We can bridge the two with a periodic definition of regret. Taking the idea that a periodic choice of arms is likely to perform well in situations with periodic patterns, we can define a regret function which measures how competitive an algorithm is with the best periodic choice of arms. For example, we can say OPT is forced to play the same arm every $\tau \in \mathbb{N}$ steps. This defines a regret function as follows,

$$R_\tau(T) = \sum_{\ell=1}^{\tau} \max_{i \in [K]} \sum_{\bar{t}=0}^{\lfloor \frac{T-\ell}{\tau} \rfloor} E[x_i(\bar{t}\tau + \ell)] - E\left[\sum_{t=1}^T x_{a(t)}(t)\right]$$

As OPT may optionally still pick the same arm on all time steps, this is a generalization of weak regret. This makes for a regret value in between weak regret and full regret.

If we were competing against the regret for a specific, known value of τ , this would be equivalent to playing τ independent instances of the adversarial bandits problem over approximately T/τ time steps each. By playing τ separate instances of an algorithm for weak regret, and by Theorem 2 in Section 6.1, we have an upper/lower bound of $\Theta(\sqrt{\tau KT})$.

However, if we were to consider that the “best possible” period τ may not be known (for example, if OPT were to consist of the best periodic function for any of the possible periods $\tau \in \{1, \dots, P\}$), these bounds do not apply as easily.

4.2 Generalized Periodic Regret

A generalization of the periodic case is the use of partition functions. Fix a maximum number of labels P . We define this upper bound P for use in our analysis later on. A partition function $f : [T] \rightarrow [P]$ is a function that assigns every time step a label from 1 to P . We consider two partition functions the same if their choice of label assignments are permutations of each other. The regret under function f would be when OPT is forced to play the same arm for all timesteps with the same label as assigned by f .

$$R_f(T) = \sum_{\ell \in [P]} \max_{i \in [K]} \sum_{\bar{t} \in f^{-1}(\ell)} E[x_i(\bar{t})] - E\left[\sum_{t=1}^T x_{a(t)}(t)\right] \quad (1)$$

Consider a set of partition functions $F \subseteq \{f : [T] \rightarrow [P]\}$ for some $P \in \mathbb{N}$. F is necessarily finite. The regret under the function set f would be when OPT can choose to play using any of the partition functions in F . This gives the following regret definition:

$$R_F(T) = \max_{f \in F} \sum_{\ell \in [P]} \max_{i \in [K]} \sum_{\bar{t} \in f^{-1}(\ell)} E[x_i(\bar{t})] - E\left[\sum_{t=1}^T x_{a(t)}(t)\right] \quad (2)$$

This definition (2) of periodic regret gives us more choice in how we want to define our potential periodic patterns to learn, through deciding on the labels on each time step for each function. We demonstrate this with our choice of partition functions in Section 7.

To model the example described earlier with periods $\tau \in \{1, 2, \dots, P\}$, we can use the set of partitions $F = \{f_1, f_2, \dots, f_P\}$, where $f_\tau(t) := (t \bmod \tau) + 1$ for each $t \in [T]$, $\tau \in [P]$.

5 The Periodic EXP4 Algorithm

We discuss the relationship between our generalized periodic setting and the problem of bandits with expert advice, and hence the applicability of EXP4 [6] to the problem. We use this to introduce Periodic EXP4, an efficient algorithm for generalized periodic regret.

5.1 Applying Bandits with Expert Advice to Periodic Bandit Problems

Periodic bandit problems can be reduced to the problem of bandits with expert advice. In the problem of bandits with expert advice, we are given a set Π of N experts. Each expert predicts an arm on each time step. We fix the number of time steps T . Thus an expert can be seen as a function $\pi : [T] \rightarrow [K]$. An algorithm to solve this problem would make use of each expert's predictions on each time step, to obtain a reward competitive with the best expert in the set. This gives us the following regret definition:

$$R_\Pi(T) = \max_{\pi \in \Pi} \sum_{t=1}^T x_{\pi(t)}(t) - E\left[\sum_{t=1}^T x_{a(t)}(t)\right]$$

This can be used to model all of the above notions of regret. For full regret, we have $\Pi := \{\pi : [T] \rightarrow [K]\}$, the set of all possible functions from $[T]$ to $[K]$. For weak regret, Π is the set of all constant functions from $[T]$ to $[K]$.

In the generalized periodic setting, let F be the set of partition functions $f : [T] \rightarrow [P]$. For each function $f \in F$, let Θ_f be the set of all possible mappings $\theta : f([T]) \rightarrow [K]$ from the image set $f([T])$ of f to the set of arms $[K]$ (thus $|\Theta_f| = K^{|f([T])|}$). Each composition $\theta \circ f$, $f \in F$, $\theta \in \Theta_f$ thus represents a possible mapping of the time steps $[T]$ to arms. Thus, for the generalized periodic setting, $\Pi = \{\theta \circ f \mid f \in F, \theta \in \Theta_f\}$.

We note that when $\Pi_1 \subseteq \Pi_2$, we will have $R_{\Pi_1}(T) \leq R_{\Pi_2}(T)$. Let Π_{full} , Π_{weak} and Π_F be the sets of functions corresponding to full regret, weak regret and generalized periodic regret under some function set F respectively. Thus, for any nonempty set F of partition functions, we have $R_{\Pi_{weak}}(T) \leq R_{\Pi_F}(T) \leq R_{\Pi_{full}}(T)$.

An existing algorithm for this problem is the EXP4 algorithm [6], which achieves a regret upper bound of $O(\sqrt{KT \log N})$, where $N := |\Pi|$. We can thus apply EXP4 directly to our problem. However, a commonly cited drawback of the EXP4 algorithm is that its running time and memory cost are at least linear in N . This is an issue as N is often very large. For example, in the generalized periodic setting, the size of N could easily be on the order of

Algorithm 1 Periodic EXP4.

```

1: procedure INITIALIZATION
2:   for each  $f \in F$  do
3:     for each  $\ell \in f([T])$  do
4:       for each  $i \in [K]$  do
5:         Initialize  $b_i^{\ell,f}(1) = 1$ 
6: procedure ALGORITHM
7:   for each time step  $t = 1, 2, \dots, T$  do
8:     for each  $i \in [K]$  do
9:        $r_i(t) := \sum_{f \in F} \left( b_i^{f(t),f}(t) \prod_{\ell \in f([t]) \setminus \{f(t)\}} \sum_{j=1}^K b_j^{\ell,f}(t) \right)$ 
10:    for each  $i \in [K]$  do
11:       $p_i(t) = \frac{r_i(t)}{\sum_{j=1}^K r_j(t)}$ 
12:    Play arm  $i_t \in [K]$  from the probabilities  $p_1(t), p_2(t), \dots, p_K(t)$ 
13:    Obtain reward  $x_{i_t}(t)$ 
14:    for each  $f \in F$  do
15:      for each  $\ell \in f([T])$  do
16:        for each  $i \in [K]$  do
17:          if  $i = i_t$  and  $\ell = f(t)$  then
18:             $b_i^{\ell,f}(t+1) = b_i^{\ell,f}(t) \exp(\frac{\gamma}{K} x_i(t)/p_i(t))$ 
19:          else
20:             $b_i^{\ell,f}(t+1) = b_i^{\ell,f}(t)$ 

```

$|F|K^P$, which is exponential in P . However, we show below that in the generalized periodic setting, we can devise an algorithm that is distributionally equivalent to EXP4 and can be made to run in time polynomial in $|F|$, K and P .

The EXP4 algorithm works by assigning a weight w_π (with initial value 1) to each expert $\pi \in \Pi$. The probability $p_i(t)$ of playing an arm $i \in [K]$ would then be $\sum_{\pi(t)=i} w_\pi(t) / \sum_{\pi} w_\pi(t)$, the ratio of the combined weights of the experts agreeing to play arm i to the total weight of the experts. Whenever an arm $i \in [K]$ is played, each expert who suggested arm i will have their weight adjusted by some factor $\exp(\frac{\gamma}{K} x_i(t)/p_i(t))$. More details on EXP4 are given in [6]. Note that it discusses a more general form of expert advice where each expert suggests a probability vector on the arms. However, we only require the case where at each time step, each expert suggests one arm with probability 1, and all other arms with probability 0.

5.2 Periodic EXP4, Memory and Running Time Costs

Periodic EXP4 (Algorithm 1) is distributionally equivalent to the EXP4 algorithm when run with the set of experts $\Pi = \{\theta \circ f \mid f \in F, \theta \in \Theta_f\}$. The key intuition behind this algorithm is that the generalized periodic setting produces many symmetries in the weight computation for each expert. Specifically, we take advantage of how for each partition function f , the set of experts contains every possible combination of arm assignments to labels in the image set $f([T])$. This allows us to compute the probabilities that EXP4 would play each arm at each time step without computing the individual weights of every expert.

For brevity, let $P_f := |f([T])|$ be the number of labels used by the function f . Necessarily $P_f \leq P$. The memory requirement is $O(K \sum_{f \in F} P_f)$, which is at most $O(KP|F|)$. A naive implementation of the algorithm gives a running time of $O(K^2 \sum_{f \in F} P_f)$ per time step, but with some pre-computation, the running time can be lowered as shown in the full paper [21].

5.3 Correctness of Periodic EXP4

To show correctness, we show that our algorithm produces the same probability distribution over arms as EXP4 in every time step. Define $\pi_{\theta,f}$ as the expert which at time t recommends arm $\theta \circ f(t)$ with probability 1 and all other arms with probability 0. We show this algorithm is distributionally equivalent to EXP4, where $\Pi = \{\pi_{\theta,f} | f \in F, \theta \in \Theta_f\}$. In EXP4, each expert $\pi_{\theta,f}$ would have some weight $w_{\theta,f}(t)$ at time step t . At time step t , EXP4 plays arm i with probability $p_i(t)$ represented by the following expression:

$$p_i(t) = \frac{\sum_{f \in F, \theta \in \Theta_f, \theta \circ f(t) = i} w_{\theta,f}(t)}{\sum_{f \in F, \theta \in \Theta_f} w_{\theta,f}(t)}$$

Thus, to show that the two algorithms are distributionally equivalent, as $p_i(t) := r_i(t) / \sum_{j=1}^K r_j(t)$ in our algorithm, for each successive time step t , we only need to show the following:

$$r_i(t) = \sum_{f \in F, \theta \in \Theta_f, \theta \circ f(t) = i} w_{\theta,f}(t)$$

The details of this derivation is given in the full paper [21]. We can thus formally state a regret upper bound as follows (Theorem 1). This upper bound comes directly from EXP4's regret bound of $O(\sqrt{KTN})$, where the number of experts $N = \sum_{f \in F} K^{|f([T])|} \leq |F|K^P$.

► **Theorem 1.** *With K arms, T time steps, $|F|$ partition functions, with every function having at most P labels, Periodic EXP4 gives a regret upper bound of $O(\sqrt{PKT \log K} + KT \log |F|)$.*

6 Lower Bounds

In this section, we provide lower bounds for the case of a single partition and for a set of partitions. We demonstrate that the upper and lower bounds differ by a factor of $\log K$.

The existing regret lower bound for the problem of bandits with expert advice [23] is $\Omega(\sqrt{KT \frac{\log N}{\log K}})$. This lower bound is derived by dividing the time steps $[T]$ into $\frac{\log N}{\log K}$ equal parts. For the generalized periodic setting, as this lower bound uses an instance that can be modeled with a single partition function, it does not give immediate insight into whether having multiple different periods or partition functions increases the difficulty of the problem.

6.1 Lower Bound for a Single Partition

We consider the case with only a single partition function $f : [T] \rightarrow [P]$, which partitions the time steps into P labels $1, 2, \dots, P$. The sizes of the partitions are $|f^{-1}(1)|, |f^{-1}(2)|, \dots, |f^{-1}(P)|$ respectively. It seems like intuitively, by seeing this as P separate instances of the weak regret setting, and by the existing $\Theta(\sqrt{KT})$ upper/lower bounds on weak regret [6, 5], we would have an upper/lower bound of $\Theta(\sum_{\ell=1}^P \sqrt{K|f^{-1}(\ell)|})$. For equally sized partitions of size approximately $\frac{T}{P}$ each, this bound would be $\Theta(\sqrt{PKT})$.

However, while the upper bound is clearly met by running P independent instances of an algorithm for weak regret, the lower bound is less clear. Even when considering it as P separate instances, there is a possibility of an algorithm “reacting” to losses in other instances to play differently in the current instance, obtaining a higher total reward as a result. For completeness, we include a proof for the lower bound (Theorem 2) in the full paper [21].

► **Theorem 2.** *Fix a partition function $f : [T] \rightarrow [P]$ which assigns a label to each time step. Assume that for each $\ell \in f([T])$, there are at least $K/(4 \ln \frac{4}{3})$ time steps with label ℓ . Then the minimax pseudo-regret (1), over all algorithms a and adversaries R , has a lower bound as follows, for some positive constant c :*

$$\inf_a \sup_R \left(\max_{\theta \in \Theta_f} E \left[\sum_{t \in [T]} x_{\theta \circ f(t)}(t) \right] - E \left[\sum_{t \in [T]} x_{a(t)}(t) \right] \right) \geq \sum_{\ell \in [P]} \sqrt{cK|f^{-1}(\ell)|}$$

If we consider the simple case where OPT may play only periodic functions from any period $\tau \in \{1, 2, \dots, P\}$, it can do no worse than if it were only allowed to play at period P . We thus obtain a lower regret bound of \sqrt{PKT} .

6.2 Lower Bound for the Generalized Periodic Setting

Let F be the set of partitions, so $|F|$ is the number of partitions. Let P be the maximum number of labels of any partition in F . For sufficiently large T and $K \leq P$, we obtain a pseudo-regret(2) lower bound of $\Omega(\sqrt{PKT} + \sqrt{KT \frac{\log |F|}{\log K}})$. It is proved in the full paper [21].

If $P < K$ instead, a simple lower bound can be obtained by using only P out of the K arms, so we obtain a problem with P arms and maximum partition size P . This gives us a lower bound of $\Omega(\sqrt{PKT} + \sqrt{PT \frac{\log |F|}{\log P}})$. We can then merge these two lower bounds into a single expression $\Omega(\sqrt{PKT} + \sqrt{\min(P, K)T \frac{\log |F|}{\log \min(P, K)}})$.

6.3 Analysis of Bounds

A conclusion we can make from Section 6.2 is that having multiple periods indeed increases the difficulty of the problem - we have obtained a lower bound higher than the known upper bound of $O(\sqrt{PKT})$ had only one partition function of the maximum period P been used.

With K arms, T time steps, $|F|$ partition functions, with every function having at most P labels, Periodic EXP4 gives an upper bound of $O(\sqrt{PKT \log K} + KT \log |F|)$. On the other hand, we have a lower bound of $\Omega(\sqrt{PKT} + KT \frac{\log |F|}{\log K})$ in the case where $K \leq P$. This gives a gap of $\sqrt{\log K}$ between the two bounds. Interestingly, this log-factor is the same as the current gap between the upper and lower bounds in the problem of bandits with expert advice. This is possibly because we use a similar lower bound proof to the problem of bandits with expert advice [23], as well as a similar algorithm for the upper bound.

7 Experimental Evaluation

In this section, we discuss the implementation details of Periodic EXP4 and parameter values chosen, evaluate the algorithm via simulation, and compare its performance to EXP3 [6]. We show how Periodic EXP4 (a) learns periodic patterns over time under both discrete and continuous changes in network data rates, (b) outperforms EXP3, (c) is robust to noisy patterns, and (d) adapts to changes due to mobility of users.

We benchmark against “Optimal Random”, a player with prior knowledge of the actual bandwidths of each network. In each time slot, it picks a network from a probability distribution equal to the ratios of the bandwidths. For example, with network bandwidths 4, 10 and 6, the probability of picking the networks will be 0.2, 0.5 and 0.3, respectively.

All the algorithms are implemented in Python, using SimPy [24], while the core algorithm is written in C++. We use a time-varying learning rate $\gamma = t^{-\frac{1}{10}}$ [17] for both Periodic EXP4 and EXP3; γ slowly tends to zero to ensure convergence [27] while at the same time ensures that the algorithm does not take too long to learn (it learns slowly when γ is very small). Although they are not pre-requirements of Periodic EXP4, for simplicity, we assume that (a) a network’s bandwidth is equally shared among its clients, and (b) devices are time-synchronized. To reduce numerical error in our simulations, we substitute computations of $\sum_{x \in Y} \exp(x)$ with $\exp(\max_{x \in Y} x)$. In nearly all cases, sums of exponentials in our algorithm are heavily dominated by a single term, making the values of the two expressions approximately equal. Experimentally, we find that this has negligible effects on the values computed within the algorithm.

We do simulations on synthetic data. We consider setups with 20 mobile devices and 3 wireless networks, unless otherwise specified. While the number of devices remain constant throughout the simulation run, the data rates and availability of networks may change. We assume that a network selection is performed once every minute; hence, 1440 time slots is one simulated day. All results presented are from 20 simulation runs, of 86,400 time slots each (i.e., 2 simulated months). The pattern of network behavior and/or user mobility over the first 1440 time slots is repeated 60 times; we refer to each repetition as an “iteration”.

We apply Periodic EXP4 in the generalized periodic setting. We define a partition function of period τ as one which divides each iteration of 1440 time slots into τ equal contiguous segments, labeled 1 to τ in chronological order. The same labels are used for each successive repetition. Unless otherwise specified, we use the period set $\{1, \dots, 24\}$. This refers to using 24 partition functions, of periods 1 to τ respectively.

7.1 Evaluation Criteria

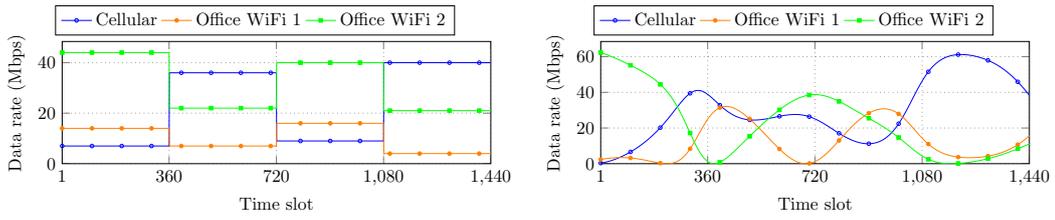
Good assignments of devices to networks divide the available bandwidth evenly among the devices. We thus evaluate the performance of the algorithms based on the lowest data rate observed by any of the devices. We compare this to the optimal allocation of devices, which maximizes the lowest data rate observed by any device. If a device with the lowest data rate observes 3Mbps, but the optimal’s lowest is 5Mbps, we say it loses 40% of its achievable gain. We refer to this percentage loss as the “distance to optimal minimum” in our results.

We do not use average cumulative gain as a performance measure because in our problem setting, average gain is maximized as long as there is at least one user in each network.

7.2 Performance Comparison of Algorithms

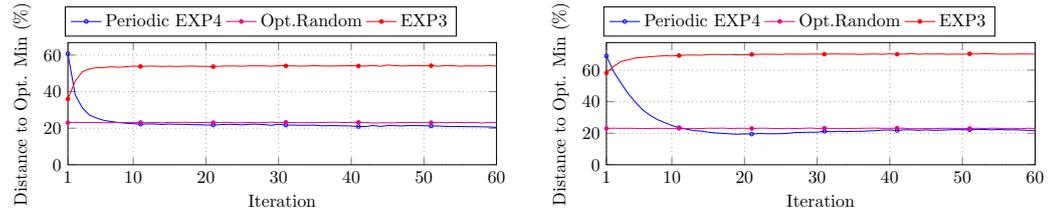
We consider two setups, both at an office with two WiFi networks and a cellular network. The data rates of these networks vary over time. The first setup involves discrete changes in network bandwidths at fixed time intervals (Figure 2a). In the second setup, the data rates vary continuously with time (Figure 2b). Figures 3a and 3b show that in both setups, the distance to optimal minimum of Periodic EXP4 drops over time while EXP3 shows no noticeable improvement with time.

Figure 4 for the continuous setup explains this improvement. The figure for the discrete setup is in the full paper [21]. At each time step, each user has a probability of picking each



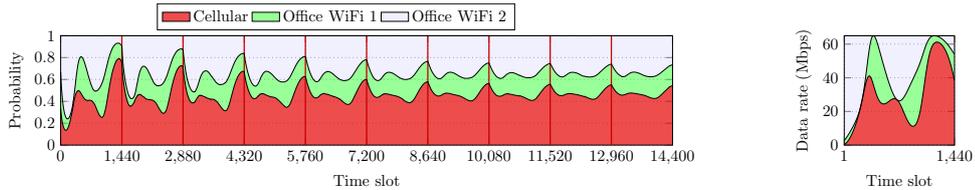
(a) Discrete changes in network data rates. (b) Continuous changes in network data rates.

■ **Figure 2** Changes in network data rates over one iteration (this is repeated 60 times).

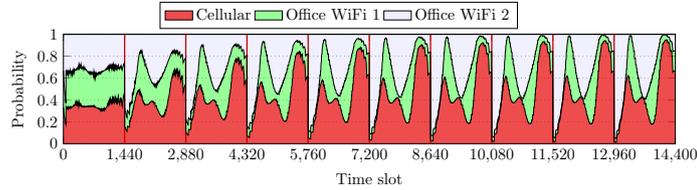


(a) Performance under discrete setup. (b) Performance under continuous setup.

■ **Figure 3** Distance to optimal minimum of Periodic EXP4 and EXP3 over 60 iterations.



(a) EXP3: Combined probabilities for each network over first 10 iterations. (b) Bandwidth ratio.

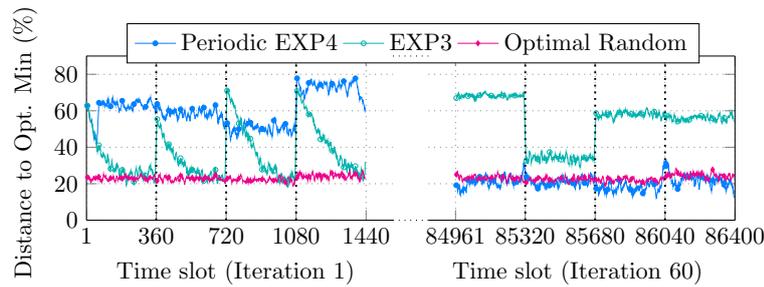


(c) Periodic EXP4: Combined probabilities for each network over first 10 iterations.

■ **Figure 4** Area chart showing the time variation of combined probabilities in the continuous setup. Figure 4b shows the actual ratio of the bandwidths of the three networks within any one iteration.

of the networks. If we consider the combined probability of picking each network, we can see that in Periodic EXP4, these probabilities converge towards the ratios of the bandwidths of the networks (Figures 4c). This is despite the continuous setup having no obvious best period. On the other hand, EXP3’s probabilities slowly flatten out (Figure 4a). This is consistent with what we would expect, as EXP3 seeks to be competitive with the best fixed-action policy, meaning that it only seeks out the best fixed arm to play.

Figure 5 shows that while EXP3 initially learns more quickly, Periodic EXP4 eventually outperforms EXP3 (which converges to the network with the best average performance), with a performance similar to Optimal Random. From our experiments, we find that while all algorithms have similar total cumulative gains, we may note that Periodic EXP4 is fairer than EXP3, with significantly lower variance. We present these results in the full paper [21].



■ **Figure 5** Distances to Optimal minimum in the first and last repetitions of the discrete setting in Figure 2a. Vertical lines indicate points where data rates change.

7.3 Other Experiments

In the full version of the paper [21], we discuss a few more experiments, the results of which are briefly summarized as follows:

1. **Performance in Noisy Settings:** On each time step, we apply a 10% Gaussian noise to each of the networks' data rates. We find that our algorithms are largely unaffected by noise in the data, giving similar results with and without noise.
2. **Comparison of Period Sets:** We do a comparison between different possible period sets F . We find that the algorithm learns more slowly with larger period sets (e.g. $\{1, 2, \dots, 45\}$, as compared to $\{1, 2, \dots, 15\}$), but can converge to better results on more complex instances (instances where the bandwidth may fluctuate more wildly).
3. **Mobility of Users:** We consider a setup where users move around and have access to different sets of networks at different times. We compare Vanilla Periodic EXP4, which is oblivious to networks possibly becoming unavailable, against an optimized version, which selects only from the set of currently available networks. While the optimized version initially yields a better performance, they eventually perform equally well when the Vanilla Periodic EXP4 algorithm learns the pattern.

8 Conclusion

In this paper, we develop an efficient variant of EXP4 for the periodic bandit problem, give nearly matching upper and lower bounds for it, and demonstrate its advantages in learning periodic behavior in the context of the network selection problem.

An interesting issue raised in contrasting this paper and [9, 22] is whether non-stationary bandit problems are better modeled stochastically or adversarially. While these papers address non-stationary rewards primarily in a stochastic setting with some adversarial aspects, we tackle the periodic bandit problem in a fully adversarial setting. Using the adversarial setting has the benefit of not placing any constraints on the adversary; we adapt to the periodic setting only through our definition of regret. A proper comparison of stochastic and adversarial methods for network selection is a possible future line of work.

References

- 1 Anuja Meetoo Appavoo, Seth Gilbert, and Kian-Lee Tan. Shrewd Selection Speeds Surfing: Use Smart EXP3! In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 188–199. IEEE, 2018.
- 2 Anuja Meetoo Appavoo, Seth Gilbert, and Kian-Lee Tan. Cooperation Speeds Surfing: Use Co-Bandit! *arXiv preprint*, 2019. [arXiv:1901.07768](https://arxiv.org/abs/1901.07768).

- 3 E. Aryafar, A. Keshavarz-Haddad, C. Joe-Wong, and M. Chiang. Max-Min Fair Resource Allocation in HetNets: Distributed Algorithms and Hybrid Architecture. In *ICDCS, 2017*, pages 857–869. IEEE, 2017.
- 4 E. Aryafar, A. Keshavarz-Haddad, M.I Wang, and M. Chiang. RAT selection games in HetNets. In *INFOCOM*, pages 998–1006. IEEE, 2013.
- 5 Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.
- 6 P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.
- 7 O. Avner and S. Mannor. Multi-user lax communications: A multi-armed bandit approach. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9, April 2016. doi:10.1109/INFOCOM.2016.7524557.
- 8 Y. Bejerano, S-J. Han, and L. E. Li. Fairness and load balancing in wireless LANs using association control. In *MobiCom*, pages 315–329. ACM, 2004.
- 9 Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic Multi-Armed-Bandit Problem with Non-stationary Rewards. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 199–207. Curran Associates, Inc., 2014. URL: <http://papers.nips.cc/paper/5378-stochastic-multi-armed-bandit-problem-with-non-stationary-rewards.pdf>.
- 10 Sébastien Bubeck, Nicolò Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.
- 11 M. H. Cheung, F. Hou, J. Huang, and R. Saththarajah. Congestion-Aware Distributed Network Selection for Integrated Cellular and Wi-Fi Networks. *arXiv preprint*, 2017. arXiv:1703.00216.
- 12 S. Deng, A. Sivaraman, and H. Balakrishnan. All your network are belong to us: A transport framework for mobile network selection. In *HotMobile*. ACM, 2014.
- 13 Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation. In *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, pages 1–9. IEEE, 2010.
- 14 D. Golovin, M. Faulkner, and A. Krause. Online distributed sensor selection. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 220–231. ACM, 2010.
- 15 B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot. Measurement-based self organization of interfering 802.11 wireless access networks. In *INFOCOM 2007*, pages 1451–1459. IEEE, 2007.
- 16 R. Kleinberg, G. Piliouras, and E. Tardos. Multiplicative updates outperform generic no-regret learning in congestion games. In *ACM STOC*, pages 533–542. ACM, 2009.
- 17 S. Maghsudi and S. Stanczak. Relay selection with no side information: An adversarial bandit approach. In *WCNC*, pages 715–720. IEEE, 2013.
- 18 A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. A. Arbaugh. A Client-Driven Approach for Channel Management in Wireless LANs. In *Infocom*, 2006.
- 19 E Monsef, A. Keshavarz-Haddad, E. Aryafar, J. Saniie, and M. Chiang. Convergence properties of general network selection games. In *INFOCOM*, pages 1445–1453. IEEE, 2015.
- 20 D. Niyato and E. Hossain. Dynamics of network selection in heterogeneous wireless networks: An evolutionary game approach. *TVT*, 58(4):2008–2017, 2009.
- 21 Shunhao Oh, Anuja Meetoo Appavoo, and Seth Gilbert. Periodic Bandits and Wireless Network Selection [full version of paper]. *arXiv preprint*, 2019. arXiv:1904.12355.
- 22 Allesiaro Robin, Raphaël Feraud, and Odalric-Ambrym Maillard. The Non-stationary Stochastic Multi-armed Bandit Problem. *International Journal of Data Science and Analytics*, March 2017. doi:10.1007/s41060-017-0050-5.
- 23 Yevgeny Seldin and Gábor Lugosi. A lower bound for multi-armed bandits with expert advice. In *13th European Workshop on Reinforcement Learning (EWRL)*, 2016.

- 24 SimPy. SimPy - Event discrete simulation for Python, 2016. , accessed 2018-19-12. URL: <https://simpy.readthedocs.io/>.
- 25 K. Sui, M. Zhou, D. Liu, M. Ma, D. Pei, Y. Zhao, Z. Li, and T. Moscibroda. Characterizing and improving wifi latency in large-scale operational networks. In *MobiSys*, pages 347–360. ACM, 2016.
- 26 Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert Schapire. Efficient algorithms for adversarial contextual learning. In *International Conference on Machine Learning*, pages 2159–2168, 2016.
- 27 C. Tekin and M. Liu. Performance and Convergence of Multi-user Online Learning. In *GAMENETS*, pages 321–336. Springer, 2011.
- 28 H. A. Tran, S. Hoceini, A. Mellouk, J. Perez, and S. Zeadally. QoE-based server selection for content distribution networks. *IEEE Transactions on Computers*, 63(11):2803–2815, 2014.
- 29 Q. Wu, Z. Du, P. Yang, Y.-D. Yao, and J. Wang. Traffic-aware online network selection in heterogeneous wireless networks. *TVT*, 65(1):381–397, 2016.
- 30 K. Zhu, D. Niyato, and P. Wang. Network selection in heterogeneous wireless networks: Evolution with incomplete information. In *WCNC*, pages 1–6. IEEE, 2010.