Fast Change Identification in Multi-Play Bandits and Its Applications in Wireless Networks

Gourab Ghatak, Member, IEEE

Abstract—Next-generation wireless services are characterized by diverse requirements. To sustain several such applications, the wireless access points need to probe the users in the network periodically in an energy-efficient manner. We study a novel multi-armed bandit (MAB) setting that mandates probing all the arms periodically while keeping track of the best current arm in a piece-wise stationary environment. We develop TS-GE that balances the regret guarantees of the classical Thompson sampling (TS) with the broadcast probing (BP) of all the arms simultaneously in order to actively detect a change in the reward distributions. The main innovation is in identifying the changed arm by an optional indexing subroutine called group exploration (GE) that scales as $\log_2(K)$ for a $K$-armed bandit setting. We characterize the probability of missed detection and the probability of false alarms in terms of environmental parameters. We highlight the conditions for which the regret guarantee of TS-GE outperforms that of the state-of-the-art passively-adaptive and actively-adaptive algorithms, in particular, ADSWITCH. We demonstrate the efficacy of TS-GE by employing it in two wireless system applications - task offloading in mobile-edge computing (MEC) and an industrial Internet-of-Things (I-IoT) network designed for simultaneous wireless information and power transfer (SWIPT).

Index Terms—Multi-armed bandits, Thompson sampling, non-stationarity, online learning.

I. INTRODUCTION

SEQUENTIAL decision-making problems in reinforcement-learning (RL) are popularly formulated using the multi-armed bandit (MAB) framework, wherein, an agent (or player) selects one or multiple options (called arms) out of a set of arms at each time slot [1], [2], [3], [4], [5]. Each time the player selects an arm or a group of arms, it receives a reward characterized by the reward distribution of the played arm/arms. The player performs consecutive action selections based on its current estimate of the reward (or that of its distribution) of the arms. The player updates its belief of the played arms based on the reward received. In case the reward distribution of the arms is stationary, several algorithms have been shown to perform optimally [6].

On the contrary, most real-world applications such as industrial Internet-of-Things (I-IoT) networks [7], wireless communications [8], computational advertisement [9], and portfolio optimization [10] are better characterized by non-stationary rewards. However, non-stationarity in reward distributions is notoriously difficult to handle analytically. Additionally, in several of these applications, the player needs to probe each of the available arms periodically in an energy-efficient manner, e.g., to get timely status updates or to perform wireless power transfer in I-IoT networks.

To address non-stationarity, researchers either i) construct passively-adaptive algorithms that are change-point agnostic and work by discounting the impact of past rewards gradually or ii) derive frameworks to actively detect the changes in the environment. Among the actively-adaptive algorithms, the state-of-the-art solutions, e.g., ADSWITCH by Auer et al. [11] provide a regret guarantee of $O(\sqrt{K N C T \log T})$ for a $K$-armed bandit setting experiencing $N_C$ changes in a time-horizon $T$.

However they do not provide any guarantees on the frequency of probing of each arm. Moreover, algorithms such as ADSWITCH perform $O(n^3)$ computations at the $n$-th time slot which makes it unattractive to be deployed on wireless edge devices.

Contrary to most non-stationary bandit algorithms, in several applications, the agent can possibly select multiple arms simultaneously. In such cases, the agent may either have access to the individual rewards of the arms played or a function of the rewards. For example, authors in [12] explored a wireless communication setup where a user can select multiple channels to sense and access simultaneously. Similarly, in portfolio optimization [13] multiple plays refer to investing in multiple financial instruments simultaneously. However, to the best of our knowledge, no existing work investigates multiple simultaneous plays under a changing environment.

In contrast to this, we propose an algorithm based on grouped probing of the arms. The salient features of this algorithm are i) it guarantees the probing of all the arms periodically, ii) it identifies the exact arm that has undergone a change, and iii) at each time slot it is more computationally efficient than its competitors. We investigate the conditions under which the proposed algorithm achieves superior regret guarantees than the state-of-the-art algorithms. We also report the conditions in which our proposed algorithm provides weaker regret guarantees than the competing algorithms.

A. Related Work

Recently, MAB algorithms have gained a lot of interest from the wireless research community. Some recent applications include edge-assisted computing [14], dynamic spectrum access [15], and caching [16]. In our work, we focus on piece-wise stationary environments for the MAB framework.

1) Non-Stationarity: One popular model for emulating non-stationarity in bandit problems is to consider the
distribution of the rewards to remain constant over epochs and change at slots unknown to the player (e.g., see the work by Garivier and Moulines [17]). The authors in [17] characterized discounted upper confidence bound (UCB) (D-UCB) and sliding window UCB (SW-UCB) and showed that both the algorithms match the lower bound up to a logarthmic factor. Raj and Kalyani [18] proposed discounted TS (dTS) derived from the classical Thompson Sampling (TS) algorithm which works by discounting the effect of past observations. Recently, Rahman et al. [19] proposed a passively adaptive algorithm for computation offloading for edge servers that performed better than dTS and D-UCB. However, their evaluation was not backed by theoretical characterization.

Actively-adaptive algorithms have been experimentally shown to perform better than the passively-adaptive ones [20]. In particular, ADAPT-EVE, detected abrupt changes via the Page-Hinkley statistical test (PHT) [21]. However, their evaluation is empirical without any regret guarantees. Similarly, authors in [10] employed the Kolmogorov-Smirnov statistical test to detect a change in the distribution of the arms. However, there the authors assumed that all the arms in the framework change simultaneously. In case the reward distribution of a single arm changes, their algorithm fails to identify the changed arm. Interestingly, tests such as the PHT have been applied in different contexts in bandit frameworks, e.g., to adapt the window length of SW-UCL [22]. The proposals by [23] and [24], and by Cao et al. [25] detected a change in the empirical means of the rewards of the arms by assuming a constant number of changes within an interval. While the algorithm in [25], called M-UCB achieved a regret bound of \(\mathcal{O}(\sqrt{KN_C T \log T})\), the work by Yu et al. [23] leveraged side-information to achieve a regret of \(\mathcal{O}(K \log T)\). However, for deriving the regret guarantees and consequently, setting the parameters of these proposed algorithms optimally, the authors assumed that the player either knows the exact number of changes in the framework within a time horizon (e.g., see Assumption 1 in [25] and Remark 3 [23]) or the change frequency, i.e., rate of the change arrival process (Lemma 3 in [24]). However, these proposed algorithms assume prior knowledge of either the number of changes or the change frequency. In our work, we relax this assumption. On these lines, recently, Auer et al. [11] proposed ADSWITCH based on the mean-estimation-based checks for all the arms. Remarkably, the authors bounded the regret by order \(\mathcal{O}(\sqrt{KN_C T \log T})\) for ADSWITCH without any condition on the number of changes \(N_C\) for the \(K\)-arm bandit problem. Nevertheless, since both M-UCB and ADSWITCH provide the same regret guarantees, we choose them as the competitor algorithms for our proposal.

2) Multiple Plays: Anantharam et al. [26] introduced the concept of multiple play bandits. Apart from [12], others that have investigated bandits with multiple plays include [27] which considered budgeted settings for multiple plays with random rewards and costs. In the context of distributed channel access, [28] explored the setting where multiple agents access the same arm simultaneously. It is important to note that a multi-player setting is different than a single-player multi-play setting as considered in this paper. Multi-player bandits may lead to collisions that can be resolved in several ways, e.g., equal division of reward. We refer the reader to the work by Besson et al. [29] for a discussion on this. For multi-play bandits that include multiple players, the recent work by Wang et al. [30] considered a per-load reward distribution and proves a sample complexity lower bound for Gaussian distribution. Nevertheless, none of these existing works treat multiple plays under non-stationarity which can dramatically alter the policy of the player.

3) Frequency of Probing: Zhou et al. [31] studied a framework for joint status sampling and updating in Internet-of-Things (IoT) networks. The authors formulated the problem as an infinite horizon average cost-constrained Markov decision process. The policy for a single IoT device is derived and a trade-off is revealed between the average age of information (AoI) and the sampling and updating costs. The work by Stahlbuhk et al. [32] considered a system consisting of a single transceiver pair and a set of communication channels to choose from. They devised a policy to minimize the queue length regret, for which they provide a regret guarantee of the order of \(\mathcal{O}(\log T)\). A rigorous analysis for the regret of age-of-information bandits was presented in [33], which first showed that UCB and TS are order-optimal for AoI bandits. However, to the best of our knowledge, none of these works consider the environment to be non-stationary which limits their applicability in modern wireless systems.

4) Other Approaches: The two case studies presented in this paper are on computation offloading in mobile edge computing (MEC) networks and on simultaneous wireless information and power transfer (SWIPT) in I-IoT networks. These are indeed extensively studied in literature with a variety of optimization and deep learning approaches, e.g., see [34], [35], [36], and [37]. However, unlike bandit algorithms, heuristic approaches and deep learning algorithms are limited in terms of their theoretical guarantees.

B. Motivation and Contribution

Unlike M-UCB but similar to ADSWITCH, we consider a framework where the number of changes \(N_C\) is unknown and least-ways smaller than \(\sqrt{T}\). Furthermore, we target an additional requirement - the agent algorithm should guarantee that the age between two consecutive plays of each arm is bounded. The major innovation in this paper is two-fold - i) by allowing simultaneous probing of multiple arms in a coded manner, we reduce the scaling of changed-arm identification form \(\mathcal{O}(K)\) to \(\mathcal{O}(\log K)\), and ii) by design, TS-GE guarantees that the last sample of each arm is not older than \(\sqrt{T}\). Overall, the main contributions of this paper are:

- We develop and characterize TS-GE, tuned for piece-wise stationary environments with an unknown number of change points with an additional design guarantee of periodic mandatory probing of all the arms. Classical bandit algorithms that focus on regret optimization do so by limiting the number of times sub-optimal arms are played. Hence the requirement of mandatory probing, which is key in several wireless applications, has not been considered in bandit literature. By balancing the regret guarantees of stationary Thompson sampling with grouped probing of all the arms, TS-GE ensures an upper bound of \(\sqrt{T}\) in the sampling age of each arm.
- We propose a coded grouping of the arms based on the arm indices and consequently, derive the
probability of missed detection of change and the probability of false alarm and highlight the conditions to limit these probabilities. Based on this, we show that TS-GE achieves sub-linear regret, \( O \left( K \log T + \sqrt{T \max \{ N_C (1 + \log K), T \}} \right) \). We compare this bound with the best known bound of \( O(\sqrt{KN_C T \log T}) \) and discuss the conditions under which the bound of TS-GE outperforms the latter. A summary of the pros and cons of our proposal is presented in Table I.

- As the first case study, we employ TS-GE as a strategy for wireless nodes to offload their computational tasks to a set of MEC servers. We demonstrate how the grouped probing phase of the algorithm enables the users to keep track of the best server to offload their tasks under a dynamic environment.

- Finally, as the second case study, we consider an I-IoT network where a central controller is required to sustain SWIPT services to the IoT devices. The different phases of TS-GE are mapped to the data-transfer and energy-transfer operations of the network. We demonstrate the performance of TS-GE with respect to the statistical upper-bound derived using stochastic geometry tools.

The rest of the paper is organized as follows. First, in Section II we describe the considered bandit model and its assumptions. The TS-GE algorithm is proposed as a solution to the posed problem in Section III. The mathematical analysis of missed detection, false alarm, and regret analysis are presented in Section IV. Then, the proposed algorithm is applied in two wireless network case studies in Section V-A and Section V-B. Finally, the paper is concluded in Section VI. The notations used in the paper are summarized in Table II.

II. PROBLEM SETUP

Consider a \( K \)-arm bandit setting with arms \( a_i \in \mathcal{K} \), for \( i = 0, 2, \ldots, K - 1 \). Let us assume that \( K = 2^d \) for some \( d \in \mathbb{Z} \). It may be noted that in case the number of arms is not a power of 2, the same can be transformed into one by adding dummy arms which are sub-optimal with probability 1 (e.g., arms with a constant reward of \(-\infty\)). The policy of the player is denoted by \( \pi \), where \( \pi(t) = a_i \) refers to the player pulling arm \( a_i \) at time \( t \). The reward \( R_i(t) \) of an arm \( a_i \) at time \( t \) is assumed to be an instance of a truncated Gaussian random variable \( \mathcal{N} (\mu_i(t), \sigma^2, R_{\text{max}}) \) with mean \( \mu_i(t) \), variance \( \sigma^2 \), and truncation limit 0 and \( R_{\text{max}} \) [38]. This assumption is valid for most practical wireless applications, e.g., upper bound on received power, signal to interference plus noise ratio (SINR), or data rate by the wireless nodes. Note that the regret guarantees derived in this paper hold for any sub-Gaussian distribution of the rewards. The assumption of the truncated Gaussian is mainly due to i) the tractability of the results using the Gaussian Q-function and ii) its widespread use in modeling uncertainty in wireless communication applications. It is worth noting that the variance of all the arms is constant and is the same for all arms,\(^1\) however, the mean is a function of time.

\(^1\)This assumption is simply due to the case of notations and the derivation of the regret bounds. The algorithm can be executed with this assumption does not hold. However, the regret analysis for such a case is much more involved and will be investigated in future work.

Multiple Plays: Finally, we assume that at any time slot, multiple arms can be played by the agent. However, in that case, the agent observes a weighted average of the rewards of the pulled arms. This is formally mentioned below.

Assumption 1: In case at any time slot \( t \) the policy \( \pi \) pulls multiple arms, i.e., \( \pi(t) = \bigcup_{k \in S} \alpha_k \), for some subset \( S \subseteq K \), then the reward observed by the player is \( R_\pi(t) = \frac{1}{\#S} \sum_{a_k \in S} R_{a_k}(t) \).

Note that the player does not have access to the individual arm rewards of the set of arms it has played. This model fits into several application settings, for example, when the bandwidth \( B \) is divided among \( n \) users equally, the network throughput is \( B/n \) times the sum of spectral efficiency experienced by each user. We note that this assumption presents an interesting trade-off for action selection. A larger number of arms pulled simultaneously reveals less information about the individual rewards, but it presents a wider view of the system since the reward is drawn from the sample average of the arms pulled. It is precisely this trade-off that we characterize in this paper. It is important to highlight that such an assumption is not present in [11] as highlighted in Table I.

A. Piece-Wise Stationary Model

1) Mandatory Probing Requirement: The player interacts with the bandit framework in a sequence of \( N_l \) episodes, denoted by \( E_i, i = 1, 2, \ldots, N_l \), each of length \( T_l \). Consequently, the total time-horizon \( T \) can be expressed as \( T = N_l T_l \). It is important to note that, unlike typical bandit-based work, the episodes are a feature of the problem setting and the framework rather than that of the algorithm. The episodes impose a condition on any feasible policy of the player as stated below.

Condition 1: The framework mandates that each arm be probed (either individually or in a group) at least once in each episode.

This condition corresponds to applications such as mandatory status updates or timely wireless power transfer in I-IoT networks.

2) Change Model: We assume a piece-wise stationary environment where changes in the reward distribution occur in a stochastic manner at time slots called change points, denoted by \( t_{c_j}, j = 1, 2, \ldots \), where each \( t_{c_j} \in [T] \). At each change point, exactly one of the arms \( a_i \), uniformly selected from \( K \) experiences a change (increase or decrease) in its mean by an unknown amount \( \Delta_{c_i,j} \). The sample paths of the change process are such that during each episode, at most one change point occurs with probability \( p_C \) and the total number of changes is \( N_C \) within \( T \), which is unknown to the player. In each episode, at the beginning of each time slot, the environment samples a Bernoulli random variable \( C \) with success probability \( p_C \). In case of success, the change occurs in that slot, otherwise the bandit framework does not change. Once a change occurs in an episode, the change framework is paused until the next episode. Thus, \[
P( \text{Episode } E_i \text{ experiences } c_i \text{ changes}) = \begin{cases} p_C; & c_i = 1, \\ 1 - p_C; & c_i = 0, \\ 0; & c_i > 1. \end{cases}
\]
where, \( p_C = \sum_{k=1}^{T_1} (1 - p_b)^{k-1} p_b \). We note that the assumption of at most one change per episode limits the change framework. Although the case of multiple simultaneous changes will be taken up in future work, we highlight here the relevance of the current model. In an edge-computing framework, consider the case of scheduled access of the edge servers to the cloud, wherein, at each phase (e.g., frame) one edge server is selected to send its status updates to the cloud. It attempts the transmission of the update packet over a wireless link characterized by an outage probability. In case of transmission success, the server broadcasts its availability status and waits until the next phase to select a new server. This situation is characterized by a single-arm change in each phase. Here the probability of a successful transmission is \( p_b \).

### III. TS-GE: Algorithm Description and Features

The key features of our proposed algorithm TS-GE are
i) actively detecting the change in the bandit framework,
ii) identifying the arm which has undergone a change, and
iii) modifying its probability of getting selected in the further rounds based on the amount of change. The TS-GE algorithm consists of an initialization phase called explore-then-commit, ETC, followed by two alternating phases: classical TS phase followed by a BP phase to determine a change in the system. In case a change is detected in the BP phase, the arm which has undergone a change is identified using an optional sub-routine called GE. Thus, the GE phase is only triggered if a change is detected in the BP phase. The overall algorithm is presented in Algorithm 1.

#### A. Initialization: ETC for TS-GE

For initialization, the player performs a ETC for TS-GE, wherein each arm is played for \( n_{ETC} \) times and consequently, their mean \( \hat{\mu}_i \) is estimated to be \( \hat{\mu}_i \).
Definition 1: An arm \( a_i \) is defined to be well-localized at time \( t \) if the empirical estimate \( \hat{\mu}_i(t) \) of its mean \( \mu_i(t) \) is bounded as \( |\hat{\mu}_i(t) - \mu_i(t)| \leq \delta \).

Let \( p_l \) be the probability that an arm \( i \) is well-localized. We will characterize the performance of TS-GE in the ETC phase in the next section. The ETC phase is conducted only once in the entire implementation of the algorithm, and in wireless communication, scenarios can be facilitated through historical data or through synthetic data generated through models/equations. For example, in the case of the rewards being the instantaneous received signal-strength indicator (RSSI) of the transmissions, the same can be generated with knowledge of the locations of the devices and by employing fading-channel emulators. At this stage, we do not go into a detailed discussion on specific wireless scenarios and instead, make the following assumption.

Assumption 2: The bandit environment remains stationary during the ETC phase.

We will characterize \( n_{ETC} \) and \( p_l \) at a later stage, where we will discuss the performance of TS-GE under circumstances when this assumption does not hold. After the ETC for TS-GE, the player switches to alternating TS and BP phases as discussed below.

B. Alternating TS and BP Phases

Each episode consists of a TS phase, a BP phase, and an optional GE phase. Since in practical scenarios of interest, the time-horizon \( T \) can be made arbitrary, let us set \( T_i = \sqrt{T} \).

In the TS phase, the player performs the action selection of the choices according to the TS algorithm for \( T_{TS} \) slots as given in [39].

Success/Failure Events: Based on the reward obtained, the player observes a success or failure event \( R_{TS}(t) \) drawn from a Bernoulli distribution with parameter \( p_{max} \). For example, in wireless networks, \( R_{TS}(t) \) may correspond to the RSSI, while \( R_{TS}(t) \) corresponds to the event of successful packet reception which depends not only on \( R_{TS}(t) \) but also on other network parameters such as cell-load. Consequently, Bernoulli distributed \( R_{TS}(t) \) leads to the choice of Beta priors in the TS phase of our algorithm. Furthermore, the version of Agrawal and Goyal [39] can be implemented. Each arm \( a_i \) is characterized by its TS parameters \( \alpha_i \) and \( \beta_i \), all of which are initially set to one. After each play of an arm \( a_i \), its estimated mean \( \hat{\mu}_i \) is updated according to the reward from the truncated Gaussian distribution \( R_{TS}(t) \). Let \( a_i \) be played at time slots \( \{t_{i}\} \subseteq [T] \) and the number of times it is played is \( n_{i}(t) \) by time slot \( t \), then:

\[
\hat{\mu}_i(t) = \frac{1}{n_{i}(t)} \sum_{t \in t_{i}} R_{TS}(t).
\]

Additionally, the TS parameters for the played arm \( a_j \), i.e., \( \alpha_j \) and \( \beta_j \) are updated as per a Bernoulli trial with a success probability \( R_{TS}(t), \forall t \in \{t_j\} \) each time \( a_j \) is played. In particular, let the result of the Bernoulli trial be \( R^* = \text{Bern}(R_{TS}(t)) \). Then, \( \alpha_j \) is incremented by 1 if \( R^* = 1 \), otherwise, \( \beta_j \) is incremented by 1.

Each TS phase is followed by the BP phase for \( T_{BP} \) time slots, where the player samples all the arms simultaneously for \( T_{BP} \) rounds. During this phase, the reward observed by the player is the average of the rewards from all the arms, i.e., \( \pi(t) = K \), for all the time slots in the BP phase. The reward in the BP phase is then compared with the average of the estimates of all the arms to detect whether an arm of the framework has changed its mean. Recall that as per Assumption 1, during the BP phase of the \( m \)-th episode, the player receives the following reward for each play:

\[
R_{BP}(t) \sim N_T \left( \frac{1}{K} \sum_{a_i \in K} \mu_i(t), \sigma^2 \frac{1}{K} \right),
\]

\[
\forall(m-1)T_i + T_{TS} < t \leq (m-1)T_i + T_{TS} + T_{BP},
\]

where \( N_T(x, y) \) represents the truncated Gaussian distribution with mean \( x \) and variance \( y \). At the end of the \( m \)-th BP phase, a change is detected if:

\[
\left| \frac{1}{K} \sum_{a_i \in K} \hat{\mu}_i((m-1)T_i + T_{TS}) - \frac{1}{T_{BP}} \sum_{t=(m-1)T_i+T_{TS}+1}^{(m-1)T_i} R_{BP}(t) \right| \geq 4\delta.
\]

Here the first term is the average of the estimated means of all the arms at the end of the \( m \)-th TS phase, while the second term represents the same evaluated during the \( m \)-th BP phase. In case the change does not occur or goes undetected, the algorithm continues with the next TS phase. However, in case a change is detected or a false alarm is generated, the algorithm moves on to the GE sub-routine as described below.

C. Policy After Change Detection

If a change is detected in the BP phase, the GE phase begins for the identification of the changed arm. The key step in this phase is the creation of \( d \) sets \( \mathcal{B}_k \subseteq K, k = 1, 2, \ldots, d \), called super-arms as shown in Algorithm 2. Recall that \( d \) is a number such that \( d = \log_2(K) \).

Arm Grouping Strategy: The \( i \)-th arm, \( i = 0, 1, 2, \ldots, K-1 \) is added to a super-arm \( \mathcal{B}_k \), \( k = 1, 2, \ldots, d \) if and only if the binary representation of \( i \) has a “1” in the \( k \)-th binary place. In other words, \( a_i \) is added to \( \mathcal{B}_k \) if \( \text{bin2dec} (\text{dec2bin}(i) \text{ AND onehot}(k)) \neq 0 \), where \( \text{bin2dec()} \) and \( \text{dec2bin()} \) are respectively operators that convert binary numbers to decimals and decimal numbers to binary. Additionally, \( \text{onehot}(k) \) is a binary number with all zeros except 1 at the \( k \)-th binary position, and is the bit-wise AND operator. The arm grouping strategy follows similarly to the parity bit generation using Hamming code, which belongs to the family of linear error-correcting codes [40]. It is worth recalling that Hamming codes are perfect codes, that is, they achieve the highest possible rate for codes with their block length and minimum distance of three. Thus, the minimum number of parity bits needed to detect and correct one-bit errors is \( \log_2(K) \) for a \( K \)-bit sequence. As a consequence of this, the minimum groups that are needed to detect a change in a single arm as well as identify the changed arm is \( \log_2(K) \).

In the GE phase, each super-arm is played \( n_{GE} \) times and the player obtains an average of rewards of all the arms that
belong to $B_k$ as per Assumption 1, i.e., each time the super-arm $B_k$ is played, the player gets a reward that is sampled from the distribution $R_{B_k}(t) \sim \mathcal{N}_T \left( \frac{1}{|B_k|} \sum_{i \in B_k} \mu_i(t), \frac{\sigma^2}{|B_k|} \right)$. Let the mean reward of the super arm $B_k$ be denoted by $\mu_{B_k}(t)$. Before each GE phase, $\mu_{B_k}$ is estimated. As an example, let a change be detected after the $m$-th BP phase. Then, the estimate of the mean reward of the super arm $B_k$ is $\hat{\mu}_{B_k}(m(T_{TS} + T_{BP})) = \frac{1}{|B_k|} \sum_{i \in B_k} \hat{\mu}_i(m(T_{TS} + T_{BP}))$. Then, the arm with the changed mean is the one that belongs to the super-arms in which a change of mean is detected. For each super arm $B_k$, the changed arm $a_j$ either is present in $B_k$ or it is present in its complimentary set $B_k^C$. Let us define a new sequence of sets as

$$C_k = \begin{cases} B_k; & \text{If a change is detected in } B_k, \\ B_k^C; & \text{If no change is detected in } B_k. \end{cases}$$

(4)

Then, the changed arm is identified as $a_j$, where $a_j = \bigcap C_k$. In case no change is detected in any of the super-arms, but the BP phase detects a change, the changed arm is identified as $a_0$.

### D. Illustrative Example

Let us elaborate on this further with an illustrative example with $K = 16$. Fig. 1 shows the following grouping for the super-arms -

- $B_1$ - Arms with ‘1’ in the first binary position - $a_1, a_3, a_5, a_7, a_9, a_{11}, a_{13}$, and $a_{15}$.
- $B_2$ - Arms with ‘1’ in the second binary position - $a_2, a_3, a_6, a_7, a_{10}, a_{11}, a_{14}$, and $a_{15}$.
- $B_3$ - Arms with ‘1’ in the third binary position - $a_4, a_5, a_6, a_7, a_{12}, a_{13}, a_{14}$, and $a_{15}$.
- $B_4$ - Arms with ‘1’ in the fourth binary position - $a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}$, and $a_{15}$.

Now, in case a change occurs in an arm, say arm $a_{10}$, this will be detected in super-arms $B_2$ and $B_4$ but not in $B_1$ and $B_3$. Additionally, this combination of changes in the super-arms uniquely corresponds to the arm $a_{10}$. Similarly, a change only in the super-arm $B_3$ and in no other super-arms corresponds to a change in arm $a_4$. Finally, note that no changes in any of the super-arms (but a change detected in the BP phase) correspond to a change in $a_6$.

Once the change is detected and the changed arm $a_j$ is identified, the corresponding mean of $a_j$ is updated as $\hat{\mu}_j = \sum_{k:a_j \in B_k} \hat{\mu}_{B_k} - \sum_{k:a_j \in B_k} a_i \in B_k:\forall i \neq j \hat{\mu}_i$. Furthermore, the TS parameters of the arm are updated. In particular, we set the parameters of $a_j$ to be the same as the arm that has an estimated mean closest to $a_j$ as $\alpha_j = \alpha_k, \beta_j = \beta_k$, where $k = \arg\min_{i \neq j} |\hat{\mu}_i - \hat{\mu}_j|$.

In the next section, we characterize the probability with which TS–GE misses the detection of a change or raises a false alarm in case of no change. This eventually leads to regret.

### IV. Analysis of TS–GE

In this section, we derive some conditional mathematical results for TS–GE.

#### A. Overview of the Regret Analysis

The regret analysis is based on the derivation of two key probabilities. First is the probability of missed detection, i.e., the probability that the change detection framework fails to detect a change. Note that a change can occur either in the TS phase or the BP phase. In case the change occurs in the TS phase, we prove that it is missed with a probability $\leq 1/7$. On the contrary, for a change in the BP phase, the exact location of the change point dictates the efficacy of change detection. We bound the probability of missed detection in the BP phase by $1/7$ conditional on a lower bound on the probability of change in each slot. The second key probability is that of a false alarm. We prove that in case the framework does not undergo a change, the probability that the BP phase reports a change is also upper bounded by $1/7$. Combining these two results in the sub-linear regret for TS–GE.

#### B. Detailed Analysis

**Lemma 1:** In the stationary regime, in order for the arm $a_i$ to be well-localized with probability $1 - p_L$, the arm needs to have been played at least $n_{ETC}$ times, where $n_{ETC} = \frac{R_{max}}{p_L} \ln \frac{1}{p_L}$.

**Proof:** Follows from Hoeffding’s inequality for bounded random variables.

Thus, the ETC phase lasts for at least $T_{ETC} = K n_{ETC} = \frac{K R_{max}}{p_L} \ln \frac{1}{p_L}$ rounds. Naturally, in order to restrict $p_L$, $O(\frac{1}{T})$, $n_{ETC}$ needs to be $O(\ln T)$. Then let us recall that the GE phase is triggered only if a change is detected in the BP phase. Consequently, the algorithm can miss detecting the chase in case the change occurs either in the TS phase or the BP phase, which we analyze below.

#### C. Probability of Missed Detection

Since the only condition on $T_{TS}$ is that it has to be upper bounded by $T_1 = \sqrt{T}$, let us set $T_{TS} = \sqrt{T} - T^\frac{1}{2}$. Let the change occur in the arm $a_i$ at $t_c$ time slots in the $m$-th TS phase, i.e., $T_{ETC} + (m - 1)(T_{TS} + T_{BP}) < t_c \leq T_{ETC} + m T_{TS} + (m - 1)T_{BP}$. The mean is assumed to change from

![Diagram showing the grouping of super-arms](image-url)
Algorithm 1 TS-GE

1: Parameters: 
2: Initialization: $\alpha_k = \beta_k = 1, \forall k = 1, \ldots, K$. 
3: Thompson Sampling Phase: 
4: for $e_i = E_1, \ldots, E_N$ do 
5: for $t = 1, \ldots, T_N$ do 
6: $\theta_i \sim \text{Beta}(\alpha_i, \beta_i)$. \hspace{1em} // Sample the Beta prior. 
7: $a_j \leftarrow a_i | \theta_j = \max(\theta_j)$ \hspace{1em} // Select the best arm. 
8: $R_{\text{TS-GE}}(t) \leftarrow R_{a_j}(t)$ \hspace{1em} // Reward at time $t$. 
9: $R_\pi(t) \leftarrow \frac{R_{a_j}(t)}{R_{\text{max}}}$ \hspace{1em} // Normalize for Beta update. 
10: $R^* = \text{Bern}(R_\pi(t))$ \hspace{1em} // Bernoulli trial for Beta update. 
11: $\alpha_j \leftarrow \alpha_j + R^*$ \hspace{1em} // Update priors. 
12: $\beta_j \leftarrow \beta_j + 1 - R^*$ \hspace{1em} // Update priors. 
13: $n_j \leftarrow n_j + 1$ \hspace{1em} // Count of arm $a_j$. 
14: $\hat{\mu}_j(t) = \frac{\sum_{i=t-1}^{t} R_{a_j}(s)X(a_j(s))}{n_j}$ \hspace{1em} // Update estimated mean of $a_j$. 
15: end for 
16: $p \leftarrow p + 1$. \hspace{1em} //End of the $p$-th TS phase. 
17: Broadcast Probing Phase: 
18: Play all the arms simultaneously for $T_{BP}$ rounds and build the estimate: 
19: \begin{align*} 
\hat{\mu}_{BP} = \frac{1}{T_{BP}} \sum_{t = (e_i-1)T+T_{TS}+1}^{(e_i-1)T+T_{TS}+T_{BP}} R_{BP}(t) 
\end{align*} 
20: if Equation (3) holds then 
21: Change is detected. \hspace{1em} // End of the $p$-th TS phase. 
22: Group Exploration Phase: 
23: Construct super-arms $\{B_k\} = \text{CSA}(a)$. 
24: for $T_{GE}$ slots do 
25: Play $B_k$ for $T_{B_k}$ rounds. 
26: $\{B_k\}$ \hspace{1em} // Update $\mu_k$: 
27: $\hat{\mu}_{B_k}(e_i(T_{TS} + T_{BP})) = \frac{1}{|B_k|} \sum_{a_i \in B_k} \hat{\mu}_i(e_i(T_{TS} + T_{BP}))$ 
28: for $t = 1, \ldots, T_{TS} + T_{BP}$ do 
29: $\hat{\mu}_{B_k}(e_i(T_{TS} + T_{BP})) = \frac{1}{|B_k|} \sum_{a_i \in B_k} \hat{\mu}_i(e_i(T_{TS} + T_{BP}))$ 
30: end for 
31: Identify changed arm. 
32: $\hat{\mu}_j(e_iT_i + 1) = \frac{\sum_{k:a_j \in B_k} \hat{\mu}_{B_k} - \sum_{k:a_j \in B_k} \sum_{a_i \in B_k:j \neq i} \hat{\mu}_i}{\sum_{k:a_j \in B_k}} \hat{\mu}_i$ \hspace{1em} // Update the changed arm. 
33: Update the Beta parameters of the changed arm: 
34: $\alpha_j = \alpha_k$, $\beta_j = \beta_k$ \hspace{1em} // where $k = \arg \min_i |\hat{\mu}_i - \hat{\mu}_j|$. 
35: Continue. \hspace{1em} // When no change is detected. 
36: end if 

The following lemma characterizes the probability of missed detection when the change occurs in the TS phase.

**Lemma 2: Let the arm $a_i$ change its mean from $\mu^*_i$ to $\mu^+_i$, where $\Delta_{C,i} = \mu^+_i - \mu^*_i$ at a time slot $t_c$ in the $m$-th TS phase. Then, if $\Delta_{C,i} \geq 2\sigma$, the probability of missed detection after the $m$-th TS phase following this change is upper bounded by $\mathcal{P}_{TS} \leq \frac{1}{T}$.**

**Proof:** Please see Appendix A.

However, if the change occurs in the BP phase, the probability of missed detection increases as discussed below.

**Lemma 3: Let the arm $a_i$ change its mean from $\mu^*_i$ to $\mu^+_i$, where $\Delta_{C,i} = \mu^+_i - \mu^*_i$ at a time slot $t_c$ in the $m$-th BP phase. Furthermore, let $p_b$ be lower-bounded as**

$$ p_b \geq 1 - \left( \frac{1}{T} \right)^{\frac{1}{T} - \frac{\Delta_{C,i}}{2\sigma}}. \quad (5) $$

**Then the probability of missed detection after the $m$-th BP phase following this change has the following characteristic:**

$$ \mathcal{P}_{BP} = \begin{cases} 
\mathcal{P}_{BP,M,Case 1} & \text{with probability } 1 - \frac{1}{T}, \\
\mathcal{P}_{BP,M,Case 2} & \text{with probability } \frac{1}{T}, 
\end{cases} \quad (6) $$

**Proof:** Please see Appendix B.

First, let us note that since the right-hand side of (5) is a decreasing function, for large values of $T$, it is a fairly mild assumption. This lemma shows that a change in the BP phase results in a different probability of missed detection based on the exact point of change. In particular, from the proof, we note that for cases 2 and 4, the algorithm misses the detection of the change in the BP phase with a high probability. However, these cases themselves occur with low probability due to the condition (5), which enables us to bound the probability of missed detection.

**D. Probability of False Alarm**

Another aspect of the algorithm that needs to be considered for a regret analysis is the fact that the BP phase can raise a false alarm when a change has not occurred in an episode while, the condition (3) holds true simultaneously. However, in case of no change, the test statistic is simply $Z_{NC} \sim N_T(0, \sigma_{NC})$, where $

$$ \sigma^2_{NC} = \frac{\sigma_f^2 \left( \frac{1}{m_{ETC}} + \frac{1}{m_{BP}} + \sum_{a_j \in K} \frac{1}{n_j(m(T_{TS} + T_{BP}))} \right)}{\sigma_f^2}, \quad (7) $
\( n_j(nT_i) \) is the number of times the arm \( a_j \) has been played in all the TS phases. Thus,

\[
P_{FA} = P(|Z_{NC}| \geq 4\delta) \leq Q \left( \frac{4\delta}{\sigma_{NC}} \right) \leq \frac{1}{T_n}. \tag{7}
\]

**E. On the Regret of TS-GE**

Now we have all the necessary results to derive the following regret bound for TS-GE.

**Theorem 1:** The regret for TS-GE under assumption \( \Delta_C \geq 2\sigma \) and Assumption 5 is upper bounded by

\[
R_{TS-GE} \leq O \left( K \log T + \sqrt{T} \max \{ N_C (1 + \log K), T^{\frac{2}{3}} \} \right). \tag{8}
\]

**Proof:** Please see Appendix C

Thus, not only the regret of TS-GE is sub-linear but also as discussed in the next section, it outperforms the known bounds under several regimes.

**F. Discussion on the Derived Bound**

Let us discuss the derived regret bound of TS-GE with respect to the best known bound of ADSWITCH [11] and M-UCB [25]. Let us define three time slots \( T_1, T_2, \) and \( T_3 \) as follows:

- \( T_1 = t : N_C(1 + \log_2 K) = t^2 \).
- \( T_2, T_3 = t : R_{TS-GE}(t) = \sqrt{N_C K T \log t}, T_2 \leq T_3. \)

In other words, \( T_2 \) and \( T_3 \) are the time instants where the regret bound of TS-GE matches the bound of ADSWITCH or M-UCB. We compare the regret bounds for three values of \( K \) relevant for a massive IoT setup: small - \( K = 100 \) arms, medium - \( K = 500 \) arms, and large - \( K = 1000 \) arms. We consider a time horizon of \( T = 10^5 \) number of plays.

For \( K = 500 \), Fig. 4b shows that there are specific regions where TS-GE outperforms M-UCB. Beyond \( T_2 \), M-UCB outperforms TS-GE. The point of interest for our discussion is the exact location of \( T_2 \) for different values of \( K \). For \( K = 100 \), the value of \( T_2 \) is low (see Fig. 4a) and the regret bound of M-UCB is lower than TS-GE for the most part of the time-horizon. However, in case of \( K = 1000 \), the value of \( T_2 \) is beyond the time-horizon, and accordingly, beyond time step \( 5000 \), throughout the time frame of interest, TS-GE outperforms M-UCB. This highlights the fact that the time period of interest in a specific application would dictate the choice of a particular algorithm.

Next, let us study the trade-offs in the competing algorithms in terms of computational complexity and sample complexity. We define them as follows.

**Definition 2:** The computational complexity for an algorithm \( \Pi \) at a time slot \( n \) is defined as the number of computations required at slot \( n \) for selecting the arm at slot \( n + 1 \).

**Definition 3:** We define the sample complexity for an algorithm \( \Pi \) for a time horizon \( T \) as the total number of pulls of the sub-optimal arms to achieve a given regret.

In essence, the computational complexity determines the requirement of the processing power and the energy budget of the devices where the algorithm is deployed. On the contrary, the sample complexity determines the devices’ required sampling frequency of the environment. Recall that the edge devices in a wireless network are typically energy-constrained and accordingly, often, sensing the environment is more energy efficient than carrying our complex on-board computation. The computational complexity reported by Auer et al. [11] for ADSWITCH is \( O(Kn^3) \) for a \( K \)-armed bandit framework in its \( n \)-th time step (although the authors mention that this can be improved upon but it is not investigated). On the contrary, at any time step \( n \), the computational complexity for TS-GE in the worst case is \( O(Kn) \). However, for a time horizon of \( T \) the number of bad pulls of the arms in the worst case is significantly larger \( O(KT^{0.9}) \) for TS-GE as compared to ADSWITCH \( O(K\sqrt{T \log T}) \). We illustrate this in Fig. 2 where the blue and yellow curves respectively represent the computation and sample complexity of the algorithms. We conclude that in case the computational complexity is not a constraint, e.g., in the case of devices connected to a power supply, ADSWITCH is preferable due to its low sampling complexity. While for energy-constrained devices, TS-GE, due to its low computational requirement is more attractive.

**G. On Parameter Tuning**

Recall that \( n_{ETC} \) is the number of time steps necessary for the exploration and initialization of the estimates of the means of the arms. A higher value of \( n_{ETC} \) results in a more accurate estimate of the means of the arms in a stationary environment, which improves the performance of TS-GE in the later stages of TS and GE. However, a large value of \( n_{ETC} \) results in the Assumption 2 being unrealistic, i.e., it is impractical to guarantee the environment to remain stationary for a large stretch of time slots.

Let us study the impact of the value of \( n_{ETC} \) on the regret performance of TS-GE by relaxing the Assumption 2. Fig. 3a shows that for smaller values of \( n_{ETC} \), the algorithm experiences a higher regret due to the inefficient estimation of the mean of the arms. Furthermore, larger values of \( n_{ETC} \) result in the possibility of multiple change points within the ETC which results in a higher regret due to incorrect estimation of the means. Note that the higher the \( N_c \), the larger will be the possibility of multiple change points within the ETC phase, and hence the larger will be the regret. Furthermore, Fig. 3b indicates the possibility of multiple change points within the ETC phase, and hence the larger will be the regret.
buffer which characterizes the pending task load at a server as discussed next.

1) Temporal Evolution of the Workload Buffer: The central controller updates the primary user and server connectivity periodically in epochs. At the beginning of each epoch, the central controller updates the status of at most one server. In particular, for the selected server \(a_i \in K\), the controller admits new primary users with fresh requests and disconnects with the primary users that have already been served by the server. The selection of the server under consideration at each epoch follows the current server load, temporally dynamic traffic density, and channel conditions. In a given epoch, the controller may also decide to not update any server. The server selection policy for updates is unknown to the secondary users. Once the update period is over, it is followed by the service period for the primary and the secondary users simultaneously.

The service period consists of multiple time slots which last until the beginning of the next epoch. While each primary user is allotted a dedicated server with a service guarantee, on the contrary, each secondary user chooses one or multiple servers during the service period to offload their computational task. In case multiple servers are selected by the secondary user, it partitions the task into segments of sub-tasks and sends it to each corresponding server. Let the beginning of the \(i\)-th epoch be denoted by \(t_E(i) \in [T]\), where \(i = 1, 2, 3, \ldots\). For an epoch \(l\), we denote the workload buffer size for server \(a_k \in K\) as \(W_{k, \text{max}}\). An admission control mechanism is assumed wherein a server \(a_k\) accommodates a maximum workload buffer of \(W_{k, \text{max}}\). Let \(m_{l,k}\) denote the number of admitted users in the server \(a_k\) in epoch \(l\). The offload rate for the primary users and the total service rate for the server \(a_k\) are assumed to be \(\eta\) cycles/second and \(C_k\) cycles/second, respectively. In case the \(j\)-th server changes its state in the \(l\)-th epoch, the size of the workload buffers evolves during \(t_E(l) \leq t \leq t_{E}(l)\) as

\[
W_k(t) = \begin{cases} 
W_k(t_E(l)) + (m_{l,k} \cdot \eta - C_k) (t - t_{E}(l)) & k = j, \\
W_k(t_E(l)) - C_k (t - t_{E}(l)) & k \neq j.
\end{cases}
\]

2) Strategy for the Secondary Users: We model the task offloading process of a secondary user in this network as the MAB problem. This problem was investigated in [19] where the authors proposed a passively adaptive algorithm for the non-stationary environment. Although the authors demonstrated the efficacy of the algorithm with numerical experiments, they did not provide any regret guarantees for the same. Recall that the secondary users are not aware of the state changes in the servers. However, they have information about the epochs. We map the TS-GE framework developed in this paper to the strategy of a single secondary user. In particular, the service period is mapped to the TS phase of the algorithm. Then, the update and association period of the next epoch corresponds to the BP and GE of an episode.

Let a secondary user attempt to offload a task that takes \(\delta\) cycles to process. Based on the workload buffer, the time required by server \(a_k\) to generate the output is calculated as:

\[
\zeta_{l,k} = \frac{W_k(t) + \delta}{C_k}.
\]

The framework can also be implemented in dense cognitive radio networks with appropriate wireless propagation models.
We consider that the offloading is successful if the process output is generated within a compute deadline $T_{\text{max}}$ of the secondary user. Thus, the reward for selecting an arm $a_k$ in the TS phase is modeled as $R_{a_k} = T_{\text{max}} - \zeta_{k,t}$ for Algorithm 1. In the BP phase, the edge user partitions its compute task into $K$ equal sub-tasks and attempts to offload a sub-task in each edge server. Naturally, the reward obtained by the user in such a case is the sum of the rewards for each sub-task. This naturally blends into the TS-GE algorithm for change detection and server identification.

3) Numerical Example: We employ the TS-GE framework in a MEC cluster with 128 servers. The maximum workload buffer is taken as 1 Giga-cycles, while in a given realization for a server $a_k$, the service rate is uniformly sampled from $\mathcal{U}(2, 4)$ GHz. The value of $\eta$ is taken to be 8 Mbps and the task deadline is taken as 0.05 seconds. Whenever a server has a completed service, it reports its availability in the next phase. We recall that as compared to bandit frameworks, deep learning models rely on large training data sets. In addition, once trained on a specific model, their performance can suffer in a highly dynamic environment. Let us compare the performance of our proposal with one popular deep learning algorithm for computation offloading - deep Q networks (DQN) [36], [37]. We train the model on synthetic data generated using system-level simulations. Fig. 5 shows that the DQN without the knowledge of the change points performs worst than even the classical bandit algorithms. Thus, to be fair to the DQN framework, we also provide the results for the same when the agent knows the change points exactly. Thus we train the model with new data after a change point. We see that in that case, the DQN outperforms all the other competitors. However, the assumption of the knowledge of the change points is unrealistic in most practical networks.

**B. Case Study 2: SWIPT in an I-IoT Network**

In this section, we employ the TS-GE algorithm to evaluate an I-IoT network where a central controller transmits data packets to the device with the best channel condition and simultaneously performs wireless power transfer to all the devices.

1) Network Model: Let us consider an I-IoT network consisting of a central wireless access point (AP) and $K$ IoT devices. The set of the devices is denoted by $\mathcal{K}$. Typically an industrial environment deals with a large $\mathcal{K}$ that represents multiple sensors and cyber-physical systems. The AP provides two wireless services in the network: i) periodic wireless power transfer (wpt) to all the IoT devices and ii) a unicast broadband data transmission to one selected IoT device.\footnote{The unicast service is relevant in cases when the AP intends to select the best storage-enabled IoT device to transfer large files for caching at the edge. This may then be accessed by the other IoT devices, e.g., using the device-to-device link. However, in this paper, we do not delve deeper into such an analysis.}

Let the scenario of interest be modeled as a two-dimensional disk $B(0, R)$ of radius $R$ centered around the origin similar to [41]. The transmit power of the AP is $P_{\text{t}}$. The location of the devices is assumed to be uniform in $B(0, R)$. Each AP device link may be blocked by roaming blockages in the environment. The probability that a link of length $r$ is in line of sight (LOS) is assumed to be $p_{\text{L}}(r) = \exp(-\omega r)$ [42].
Furthermore, note that due to the presence of a large number of metallic objects, an industrial scenario presents a dense scattering environment. Consequently, we assume that each transmission link experiences a fast-fading $h$ modeled as a Rayleigh-distributed random variable with variance 1. Thus, the received power at an IoT device at a distance of $r$ from the AP is given by $P_t(r) = KP_t hr^{-\gamma}$ with probability $p_t(r)$. Here $K$ and $\gamma$ respectively are the path-loss coefficients and the path-loss exponent. The total transmission bandwidth is assumed to be $B$ which is orthogonally allotted to the users scheduled in one time slot.

At each episode, the controller selects the device with the best channel conditions and executes information transfer in a sequence of time slots using the TS phase of the algorithm. The device-specific transmission can be facilitated by employing techniques such as beamforming. However, we do not consider the details of such procedures. The TS phase information transfer is followed by joint power transfer to all the devices. This is mapped to the BP phase of the algorithm. At the end of the BP phase, the total energy harvested at all the devices at the end of the BP phase is reported back to the controller. Using this total energy harvested at all the devices, this is mapped to the BP phase of the ADSWITCH algorithm. At the end of the BP phase, the total energy experiences a fading-averaged downlink received power $P_t = \left( KP_t r_{N1}^{-\gamma} \right)$ with probability $P_L$ and $KP_t r_{N1}^{-\gamma}$ with probability $1 - P_L$.

Over $T$ slots the throughput of the system is $T = \frac{N_I}{T_{ETC} + T_{ETS} + T_{GE}} B \log_2 \left( 1 + \frac{P_t}{N_0} \right)$, where the expectation taken over $P_t$ as per (11).

4) Multicast/Broadcast Transmission for Power Transfer:
Let us assume that in the multicast transmission phase, the AP transmits data to a subset $\mathcal{J} \subset \mathcal{K}$ of the IoT devices, where $|\mathcal{J}| = N_J$. In this case, the available bandwidth $B$ is shared among the $N_J$ devices. The harvested power experienced by an IoT device of $\mathcal{J}$ is:

\[
T_j = \begin{cases} 
\frac{\theta e N_J}{N} K P r_j^{-\gamma} & \text{with probability } \exp(-\omega_j) \\
\frac{\theta e N_J}{N} B N_0 & \text{with probability } 1 - \exp(-\omega_j)
\end{cases}
\]

Accordingly, the network sum energy is given by: $T_T = \sum_{j \in \mathcal{J}} T_j$, in one slot. In the case of the BP phase, naturally, we have $\mathcal{J} = \mathcal{K}$.

5) Numerical Example: We run the TS–GE algorithm in our I-IoT network for a total of 1000 seconds with time slots of 10 ms [44]. Additionally, we assume slow-moving blockages in which every 30 seconds the visibility state of exactly one IoT device changes from LOS to NLOS or vice-versa. In Fig. 6 we plot the average throughput of the information transfer phase (i.e., to the best device) as well as the minimum harvested energy in a device in the I-IoT network. We observe that in the case of fewer IoT devices, the ADSWITCH algorithm performs better than TS–GE. Indeed resetting all arms does not incur a large exploration loss in ADSWITCH in case $K$ is small. Additionally, ADSWITCH is not constrained by mandatory exploration. Accordingly, it enjoys a higher throughput as compared to TS–GE which needs to transfer energy to all the devices.

Interestingly, as the number of devices in the network increases beyond a threshold, TS–GE outperforms...
ADSWITCH, especially due to rapidly changed state identification in the device. Naturally, as \( K \) increases, the amount of time dedicated to energy transfer decreases. This is reflected in the reduced energy harvested in the worst device.

Several open problems are apparent. For example, the condition that each episode can experience only one change can be too stringent to be applied meaningfully in contexts. Furthermore, change detection in the distributions rather than mean and extension to multiple players are indeed interesting directions of research that will be treated in future work.

VI. CONCLUSION

We have proposed a novel multi-armed bandit algorithm, TS-GE, that not only detects changed arms in a piece-wise stationary environment but also periodically probes all the arms in the framework. Additionally, unlike state-of-the-art algorithms such as ADSWITCH which employs computations of the order of \( \mathcal{O}(n^3) \) in the \( n \)-th time step, TS-GE is computationally efficient. These properties make it attractive to be deployed in wireless systems, especially, energy-constrained edge devices. The key innovation in TS-GE is an index-based grouped probing strategy for fast identification of the changed arm. We have demonstrated the efficacy of TS-GE in two wireless communication applications - edge offloading and edge devices. The key innovation in \( \text{TS-GE} \) algorithms such as ADSWITCH that not only detects changed arms in a piece-wise condition that each episode can experience only one change and extension to multiple players are indeed interesting.

Several open problems are apparent. For example, the mean and extension to multiple players are indeed interesting directions of research that will be treated in future work.

APPENDIX A

PROOF OF LEMMA 2

Let the number of times the arm \( a_i \) is played is \( t_i^+ \) times before \( t_e \) and \( t_i^- \) times after \( t_e \). The test statistic (i.e., the parameter to be compared to \( 2\delta \)) is simply a random variable \( Z_{TS} \):

\[
Z_{TS} = \frac{1}{K} \sum_{q=(i-1)n_{ETC}+1}^{in_{ETC}} X_i^-(q) + \frac{1}{K} \sum_{a_i \neq a_i} j_a n_{ETC} \sum_{q=(j-1)n_{ETC}+1}^{j n_{ETC}} X_j(q) + \frac{1}{(K)n_j(T')} \sum_{a_j \neq a_i, q_j=T_{ETC}+1}^{j n_{ETC}} X_j(q) \sum_{q} X_i^-(q) + \sum_{q} X_j^+(q) - \frac{1}{K T_{BP}} \sum_{q=T'}^{T''} X_j^+(q).
\]

Here \( T' = T_{ETC} + (m-1)(T_{TS} + T_{BP}) + T_{TS} \) and \( T'' = T_{ETC} + m(T_{TS} + T_{BP}) \) and \( n_j(t) \) is the number to times the arm \( a_j \) has been played until time \( t \). Since all the arms except \( a_i \) remain stationary, we have

\[
P \left( \left| \frac{1}{K} \sum_{a_i \neq a_i} j_a n_{ETC} \sum_{q=(j-1)n_{ETC}+1}^{j n_{ETC}} X_j(q) + \frac{1}{(K)n_j(T')} \sum_{a_j \neq a_i, q_j=T_{ETC}+1}^{j n_{ETC}} X_j(q) \sum_{q} X_i^-(q) + \sum_{q} X_j^+(q) - \frac{1}{K T_{BP}} \sum_{q=T'}^{T''} X_j^+(q) \right| \leq 2\delta \right) \leq \mathcal{O} \left( \frac{1}{T} \right).
\]

Consequently, for the decision of change detection it is of interest to consider the following random variable instead:

\[
Z_{TS}' = \frac{1}{K} \sum_{q=(i-1)n_{ETC}+1}^{in_{ETC}} X_i^-(q) + \frac{1}{K} \sum_{a_i \neq a_i} \sum_{q=(i-1)n_{ETC}+1}^{in_{ETC}} X_j(q) + \frac{1}{(K)n_j(T')} \sum_{a_j \neq a_i} \sum_{q_j=T_{ETC}+1}^{j n_{ETC}} X_j(q) - \frac{1}{K T_{BP}} \sum_{q=T'}^{T''} X_j^+(q),
\]

and compare it to a threshold of \( 2\delta \). Note that \( Z_{TS}' \) is Gaussian distributed with mean \( \mu_{Z_{TS}'} = \Delta_{C,i} + \frac{1}{t_i^- + t_i^+} (t_i^- \mu_i^- + t_i^+ \mu_i^+) \) and variance given by \( \sigma_{Z_{TS}'}^2 = \frac{1}{K^2} \left( \frac{1}{n_{ETC}} + \frac{1}{t_i^- + t_i^+} + \frac{1}{K T_{BP}} \right) \).

Consequently, there are two cases of interest:

Case 1 - \( \Delta_{C,i} > 0 \): This is the case where the mean of the arm \( a_i \) increases from \( \mu_i^- \) to \( \mu_i^+ \). Accordingly, the missed detection probability can be written as

\[
P_M^{TS} = P \left( |Z_{TS}'| \leq 2\delta \right) \leq \mathcal{Q} \left( \frac{\mu_{Z_{TS}'} - 2\delta}{\sigma_{Z_{TS}'}^2} \right).
\]

In the above \( \mathcal{Q}(\cdot) \) is the Gaussian-Q function. The inequality (a) follows from the facts that \( \frac{1}{t_i^- + t_i^+} (t_i^- \mu_i^- + t_i^+ \mu_i^+) \geq 0 \) and the \( \mathcal{Q}(\cdot) \) is a decreasing function. Step (b) follows from the AM > HM inequality, while step (c) follows from the assumption that \( \Delta_{C,i} \geq 2\sigma \) and the inequality \( Q(K \sqrt{\frac{2}{5}}) \leq \frac{1}{K} \) for \( K \geq 1 \).

Case 2 - \( \Delta_{C,i} \leq 0 \): This denotes the case where the mean of arm \( a_i \) decreases from \( \mu_i^- \) to \( \mu_i^+ \), i.e., \( \mu_i^+ \leq \mu_i^- \). Accordingly, the missed detection probability follows similarly to the above

\[
P_M^{TS} = P \left( |Z_{TS}'| \leq 2\delta \right) \leq \mathcal{Q} \left( \frac{\mu_{Z_{TS}'} - 2\delta}{\sigma_{Z_{TS}'}^2} \right).
\]

APPENDIX B

PROOF OF LEMMA 3

Let in the BP phase, the number of times all the arms is played simultaneously be \( t_i^- \) times before \( t_e \) and \( t_i^+ \) times after \( t_e \). Given that other arms \( a_j \) where \( j \neq i \) has not changed, the test statistic (i.e., the parameter to be compared to \( 2\delta \)) is simply a random variable \( Z_{BP} \) similar to \( Z_{TS} \). given by

\[
Z_{BP} = \frac{1}{K} \sum_{q=(i-1)n_{ETC}+1}^{in_{ETC}} X_i^-(q) + \frac{1}{K} \sum_{a_i \neq a_i} \sum_{q=(j-1)n_{ETC}+1}^{in_{ETC}} X_j(q) + \frac{1}{(K)n_j(T')} \sum_{a_j \neq a_i, q_j=T_{ETC}+1}^{j n_{ETC}} X_j(q).
\]

In the above, the conditions that each episode can experience only one change and extension to multiple players are indeed interesting directions of research that will be treated in future work.
\[ + \frac{1}{Kn_j(T')} \sum_{a_j} X_j(q) \mathbb{I}(a_{TS-GE}(q) = a_j) \]
\[ + \frac{1}{(t_i + t_i^+)^2} \sum_{q} X_i(q)^2 \]
\[ - \frac{1}{KT_{BP}} \sum_{q=ETC} X_j(q). \tag{15} \]

Since all the arms except \( a_i \) remain stationary, we have
\[ \mathbb{P} \left( \frac{1}{K} \sum_{a_j \neq a_i} \sum_{nETC} X_j(q) \right) \]
\[ + \frac{1}{Kn_j(T')} \sum_{a_j \neq a_i} X_j(q) \mathbb{I}(a_{TS-GE}(q) = a_j) \]
\[ + \frac{1}{KT_{BP}} \sum_{q=ETC} X_j(q) \geq 2\delta \leq \mathcal{O} \left( \frac{1}{T^2} \right). \tag{16} \]

Consequently, for the decision of change detection it is of interest to consider the following random variable instead
\[ Z''_{BP} = \frac{1}{Kn_j(T')} \sum_{q} X_i(q)^2 \]
\[ + \frac{1}{Kn_1(T')} \sum_{q} X_i(q) \mathbb{I}(a_{TS-GE}(q) = a_j) \]
\[ \times \left[ \sum_{q} X_i(q) - \sum_{q} X_i(q)^2 \right]. \tag{17} \]

Note that \( Z''_{BP} \) is Gaussian distributed with mean \( \mu_{Z''_{BP}} = \mu_i - \frac{1}{t_i + t_i^+} \mathbb{I}(t_i < T) \mathbb{I}(t_i^+ < T) \mathbb{I}(a_{TS-GE}(q) = a_j) \) and variance given by \( \sigma^2_{Z''_{BP}} = \frac{\sigma^2}{Kn_j(T')} \left( 1 + \frac{1}{t_i + t_i^+} + \frac{1}{T_{BP}} \right) \).

**Case 1 - \( \Delta_{C,i} > 4\delta \) and \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta) \):**
This case occurs with a high probability. Due to the fact that for this case, we have \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta), \) i.e., \( \mu_{Z''_{BP}} \geq 4\delta, \) thus, similar to the Lemma 2, \( \mathbb{P}_{M, Case 1} \leq \frac{1}{7}. \) Thus, we have \( \mathbb{P}_{M} = \mathbb{P}_{M, Case 1} \cdot \mathbb{P}_{Case 1} \leq \frac{1}{7}. \)

**Case 2 - \( \Delta_{C,i} > 4\delta \) and \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta) \):** Here we have \( \mu_{Z''_{BP}} < 4\delta, \) and accordingly, the probability of missed detection is high. However, let us first observe the probability that the change occurs such that \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta). \) We have:
\[ \mathbb{P}(Case 2) = \mathbb{P} \left( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta) \right) \]
\[ \leq \mathbb{P} \left( t_i^+ \geq 4\delta \mathbb{I}(\Delta_{C,i} - 4\delta) \right) \leq \mathbb{P} \left( t_i^+ \geq \frac{4\delta}{\Delta_{C,i} - 4\delta} \right) \leq \mathcal{O} \left( \frac{1}{T} \right). \]

where step (a) is due to Assumption 5.

**Case 3 - \( \Delta_{C,i} < 4\delta \) and \( t_i^+ \leq T_{BP}(\Delta_{C,i} - 4\delta) \):** This case is similar to Case 1 and hence we skip the detailed proof for brevity. In summary, similar to Case 1, for Case 3 the probability that the change occurs at a time step such that \( t_i^+ \leq T_{BP}(\Delta_{C,i} - 4\delta) \) holds is high. However, the probability of missed detection is bounded by \( \frac{1}{T}. \)

**Case 4 - \( \Delta_{C,i} < 4\delta \) and \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta) \):** This is similar to Case 2, wherein the probability of missed detection is high, while due to the Assumption 5, the occurrence of the change such that the condition \( t_i^+ > T_{BP}(\Delta_{C,i} - 4\delta) \) holds is bounded by \( \frac{1}{T}. \)

**Appendix C: Proof of Theorem 1**

Recall that Each episode either experiences one change or no changes.

**A. Regret in Case of no Change**
The number of such episodes is \( N_1 - N_C. \) Each such episode experiences a mandatory regret bounded by:
\[ R_{no change}(T_i) \leq \mathcal{O} \left[ \log \sqrt{T - T^2} \right] + \mathcal{O} \left( T^2 \right), \]

where the term \( A \) is due to the TS phase (e.g., see [39]) and the term \( B \) is due to the BP phase, where we assume linear regret as the worst case. In case of a false alarm, the algorithm subsequently experiences worst-case regret in all the subsequent phases. This occurs with probability of \( \mathbb{P}_{FA}, \) and hence its contribution to the overall regret is \( R_{no change}(T) \leq \mathbb{P}_{FA} \Delta_{max} T \leq K_1, \) where \( \Delta_{max} \) is the maximum difference between the means of the arms at any given instant. Step (a) follows from (7). Thus, overall, in the case of no change, the regret is \( R_{no change}(T) \leq \mathcal{O} \left[ \log \sqrt{T - T^2} \right] + \mathcal{O} \left( T^2 \right) + K_1. \)

**B. Regret in Case of Change**
The number of such episodes is \( N_C. \) Each such episode experiences a regret bounded by \( R_{change} \leq \mathcal{O} \left( \Delta_{max} \sqrt{T} \right), \) where we assume that in case a change occurs, the player suffers a worst-case linear regret for the rest of the episode. However, in case of missed detection, the algorithm experiences worst-case regret in all the subsequent phases. This occurs with probability of \( \mathbb{P}_{M} = p_{TS}^BP_{TS}^A + p_{BP}^B, \) which varies as \( \frac{1}{T}, \) and hence its contribution to the overall regret is \( R_{change} \leq \mathbb{P}_{M} \Delta_{max} T \leq K_2. \) Thus, overall, for the case of no change, the regret \( R_{change} \leq \mathcal{O} \left( \Delta_{max} \sqrt{T} \right) + K_2. \)

Using the above development, we can bound the regret of TS-GE as follows:
\[ R(T) = R_{ETC}(T_{ETC}) + \sum_{i=1}^{N_1} R_i(T_i) \leq \mathcal{O} \left( K \log T \right) \]
\[ + \left( N_1 - N_C \right) R_{no change} + N_C \mathbb{P}_{change} \]
\[ \leq \mathcal{O} \left( K \log T + \sqrt{T \max \{N_C (1 + \log K), T^2 \}} \right) \tag{18} \]

**References**
[1] C. Shi and C. Shen, "Multi-player multi-armed bandits with collision-dependent reward distributions," IEEE Trans. Signal Process., vol. 69, pp. 4385–4402, 2021.
