Time-Constrained Restless Multi-Armed Bandits with Applications to City Service Scheduling

Yi Mao
The Ohio State University
Columbus, United States
mao.496@osu.edu

Andrew Perrault
The Ohio State University
Columbus, United States
perrault.17@osu.edu

ABSTRACT

Municipalities maintain critical infrastructure through inspections, both proactive and in response to complaints. For example, the Chicago Department of Public Health (CDPH) periodically inspects 7000 food establishments to maintain the safety of food bought, sold, or prepared for public consumption. Restless multi-armed bandit (RMABs) appear to be a useful tool for optimizing the scheduling of inspections, as the schedule aims to keep as many establishments in the "passing" state subject to an action limit per time period. However, a key challenge arises: satisfying timing and frequency constraints. Municipal agencies often provide an inspection window to each establishment (e.g., a two-week period where an inspection will occur) and guarantee about the minimum frequency of inspection (e.g., once per year). We develop an extension to Whittle index-based systems for RMABs that can guarantee both action window constraints and minimum frequencies. Briefly, we take a Whittle index-based view, enforcing window constraints by integrating the window structure into individual MDPs, and frequency constraints through a higher-level scheduling algorithm that aims to maximize the Whittle index. We demonstrate the performance and scalability of our methods in experiments using both synthetic and real data (with 7000 establishments inspected per year). Not only does our approach enforce constraints more effectively than naive methods, it also achieves higher rewards, up to 20%.

KEYWORDS

ACM Reference Format:
Yi Mao and Andrew Perrault. 2024. Time-Constrained Restless Multi-Armed Bandits with Applications to City Service Scheduling. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 8 pages.

1 INTRODUCTION

Restless multi-armed bandits (RMABs) [24] describe a sequential decision problem where an agent aims to manage a large population of Markov decision processes (MDPs) that are independent except for a shared action budget at each timestep. These arise in a variety of settings—as a motivating example for this work, we consider food establishment inspections carried out by a city. The city aims to keep as many establishments in a "good" state as possible (e.g., that they would pass an inspection if they were inspected) subject to a limit of k on the number of inspections they can perform per time period. Each establishment is represented by an MDP with different transition dynamics, i.e., propensities to be in a passing state without inspection.

We argue that RMABs are a valuable framework to study for two reasons. First, they are a practical way to introduce a sequential component to many real-world, large-scale, optimization problems that are naturally sequential without adding much complexity. In many tasks, if data is employed at all, it is used to estimate the probability that a process is currently in a bad state and prioritize acting on those processes (e.g., [11] in the food inspection case). Such a heuristic can have poor alignment with the true objective because it does not consider the effect of acting on the process (i.e., the ability of an action to return it to a good state) nor the propensity of a process to recover on its own.

Second, RMABs can often be solved approximately optimally in a computationally efficient manner through the use of the Whittle index heuristic [24]. Despite their combinatorially large action space size and exponential state space, under a technical condition known as indexability, they can be decomposed into independent optimization problems via dualizing the budget constraint. The resulting heuristic offers asymptotically optimal rewards.

As a result, RMABs have attracted wide interest over the past several decades in a large variety of resource allocation tasks, including wireless networking [13], machine maintenance [1, 7], and planning health interventions [3, 15, 17].

In practical applications of RMABs, it is common to place to constraints on the timing and frequency of arm pulls. For example, the Chicago Department of Public Health (CDPH) provides establishments with an inspection window: they state a particular time period during which the routine inspection will occur. This window makes the inspection less disruptive to the establishment. A similar constraint is used by a field study of applying bandits in the child health [17] in the public health information setting, where each beneficiary receives at most one call each fixed number of weeks. In addition, CDPH guarantees at least one inspection per year, per establishment, providing a baseline level of service.

In this paper, we develop methods for integrating action constraints into RMABs. Our contributions are as follows:

- **Window constraints.** For window constraints (at most one action during a prescribed time window), we show that they can be written into the structure of the MDP by introducing new timing states and describe the cost of doing so. Because we propose a structural approach, it ensures that window constraints are never violated, and state-of-the-art RMAB techniques can be used to solve the resulting instances.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand.

- **Frequency constraints and lookahead.** Frequency constraints present challenges because the Whittle index heuristic is greedy and does not look into the future. In addition, window constraints can require lookahead to prevent conflicts. We introduce an integer programming-based planner that aims to maximize the sum of Whittle indices subject to constraints. We find this integer program to be highly scalable and show that the constraint matrix is totally unimodular in some cases.
- **Empirical evaluation.** It is important to quantitatively evaluate the impact of introducing explicit constraint modeling in RMABs, as this introduces additional complexity. We evaluate reward, constraint violations, and computation time in both real and synthetic data. In our tested cases, we find that ad hoc methods for handling constraints do not perform much better than non-sequential baselines. However, RMABs with explicitly modeled constraints can improve the objective value by around 20%.

Collectively, our approach is a key advance for constrained RMABs and integrating lookahead planning into RMABs. We are the first to integrate lookahead (which is often intractable), and we show that it produces substantial benefits when the goal is to achieve high rewards and satisfy constraints simultaneously.

2 RELATED WORK

Food safety inspections. In 2015, CDPH leveraged historical food inspection data and trained a supervised learning model to predict the probability that an inspection would uncover a critical violation [11]. [12] independently analyzed the impact of prediction-driven scheduling. However, such models only consider one-shot predictions for critical violations and do not include the sequential aspect of scheduling. Fairness is of substantial interest in the provision of municipal services [22]. We consider fairness outside the scope of this paper, but a potentially interesting direction for future work.

Restless multi-armed bandits (RMABs). RMABs are PSPACE-hard in the worst case, but [24] showed that a subclass of them, so-called indexable RMABs, admit an efficient asymptotically optimal solution. Particularly relevant classes of indexable RMABs are those that extend the machine maintenance problem families [8], and scheduling problems for sensors [23], wireless transmission [10], and health interventions [16]. These RMABs are structured so the state of each process declines if it is not acted on, and differ in the details of action effect and what information is observed with or without an action. This work aims to develop techniques to integrate action constraints into RMABs of these types.

RMABs with Constraints. Several RMAB models have included constraints. In a project applying RMABs to assist maternal and child health via phone calls, a "sleeping period" for arms was enforced after they were pulled by the Whittle index heuristic [17] (see 4.2). It appears to have been enforced in an ad hoc manner, by blocking pulls that would have violated the constraint. [25] deployed RMABs in a deadline scheduling setting and integrated the deadline constraints by adding dummy arms. Fairness is another setting where constraints can arise. [9] introduced the ProbFair policy, ensuring a strictly positive lower bound on the probability of being pulled at each time step while still satisfying the budget constraints. To the best of our knowledge, we are the first to consider action window and frequency constraints.

3 PRELIMINARIES

An RMAB consists of N binary action MDPs (*arms*). We define the ith two-action MDP [20] as a tuple $(\mathcal{S}_i, \mathcal{A}, P_i, R_i, s_i^{(0)}, \gamma)$. The discount factor γ and action space $\mathcal{A} = \{0, 1\}$ are fixed across all MDPs. When the action 1 (resp. 0) is taken on an arm at time t, we refer to that arm as *active* (resp. *passive*). The rest are arm specific: $\mathcal{S}_i$ is the state space, $P_i : \mathcal{S}_i \times \mathcal{A} \rightarrow \Delta\mathcal{S}_i$ is the transition function, $s_i^{(0)}$ is the start state, and $R_i : \mathcal{S}_i \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function.. Because there are only two actions, the transition function P_i can be decomposed into an an active transition $P_i^{(1)} : \mathcal{S}_i \rightarrow \Delta\mathcal{S}_i$ and a passive transition $P_i^{(0)} : \mathcal{S}_i \rightarrow \Delta\mathcal{S}_i$.

A RMAB consists of N binary action MDPs and a per timestep budget constraint k. At each round t, the agent has a budget k, where $k \ll N$, meaning at most k arms can be "pulled", i.e., have their action set to 1. The MDP which is pulled transits actively and otherwise transits passively. Upon transitions, the rewards from all MDPs are collected and accumulated over time. The goal is to find an optimal policy $\pi^\star$ to maximize our total rewards—formally,

$$\pi^* = \arg\max_{\pi} J = \arg \max_{\pi: \sum_i \pi_i(S_t) \leq k} \sum_i \sum_t \gamma^t R_i(s_{i,t}, \pi(S_t)), \quad (1)$$

where $\pi_i(S_t) \in \mathcal{A}$ is the action selected by π for arm i, $S_t \in \mathcal{S}_1 \times \ldots \times \mathcal{S}_N$ is the joint state of all arms at time t, and $s_{i,t} \in \mathcal{S}_i$ is the state of arm i at time t,

3.1 Whittle Indices

General RMABs have an exponentially large state space and a combinatorially large action space. The Whittle index method provides tractability for some classes of RMABs [24]. It works by computing a "benefit of acting" for each arm, called the *Whittle index*. The *Whittle index heuristic* then acts on the k arms with highest Whittle indices.

To calculate the Whittle index for each arm, we search over "subsidies" for the passive action m. Formally the subsidy m modifies the reward function R_i into $R_i^{(m)}$:

$$R_i^{(m)}(s_i, 0) = R_i(s_i) + m; R_i^{(m)}(s_i, 1) = R_i(s_i). \quad (2)$$

The goal is to identify the smallest subsidy m such, for the current state $s_{i,t}$, the long-term reward for the passive and active actions are the same. To define this formally, we first define the Q function for arm i under subsidy m:

$$Q_i^{(m)}(s_i, a) = R_i^{(m)}(s_i, a) + \gamma \max_{a' \in \mathcal{A}} \sum_{s'_i \in \mathcal{S}_\rangle} P_i(s'_i | s_i, a) Q_i^{(m)}(s'_i, a'). \quad (3)$$

Definition The Whittle index for state $s_{i,t}$ is the smallest m which makes it equally optimal to take the active and passive actions:

$$w(s_{i,t}) = \inf_m \left\{ m : Q_i^{(m)}(s_{i,t}, a = 0) \geq Q_i^{(m)}(s_{i,t}, a = 1) \right\}. \quad (4)$$

For the Whittle index heuristic to have asymptotic optimality guarantees, each arm must satisfy a technical condition called *indexability* [24]. Intuitively, indexability says that, as m increases, the optimal action can only switch to passive and cannot switch back to active. Let $W_i^{(m)}$ be the set of states for which $Q_i^{(m)}(s_{i,t}, a = 0) \geq Q_i^{(m)}(s_{i,t}, a = 1)$, i.e., the passive action has an equal or higher return than the active action.

Definition (Indexability). An arm is said to be indexable if $W_i^{(m)}$ is non-decreasing in m, i.e., for any $m_1, m_2 \in \mathbb{R}$ such that $m_1 \leq m_2$, we have $W_i^{(m_1)} \subseteq W_i^{(m_2)}$. An RMAB is indexable if every arm is indexable.

3.2 Weighted b-Matching

The lookahead planning algorithm we develop will be reducible to variants of the weighted b-matching problem [21]. A weighted b-matching instance is described by an undirected graph $G = (V, E)$, an edge weight vector $w : E \to \mathbb{R}$, and a non-negative b vector $b : V \to \mathbb{N}_+$. The objective in a maximum weight b-matching is to find a set of edges x with maximum weight, subject to the constraint that only $b(v)$ edges that are adjacent to node v can be selected. Formally,

$$\max_x w^T x, \quad \text{s.t.} \sum_u x_{u,v} \leq b(v), \forall v \in V \tag{5}$$

Weighted b-matchings can be solved in polynomial time, e.g., in $O(|V|^2 \max_v b(v))$ [19].

A more challenging weighted b-matching variant is weighted bipartite b-matching [4]. In this variant, graph nodes are partitioned into a right set U and a left set V, and there are no edges within each partition. Nodes in the left (resp., right) set have maximum matching cardinality L^+ (resp., R^+) and minimum cardinality L^- (resp., R^-). Under these constraints, finding a maximum weight b-matching is NP-hard.

4 PROBLEM FORMULATION

We study two types of action constraints that arise in the motivating food establishment inspection problem. We begin by defining a sample RMAB with domain-motivated constraints (Sec. 4.1). Window constraints specify an action window where the arm is allowed to be acted on (Sec. 4.2). Frequency constraints specify a minimum number of actions each arm must receive over a period of time (Sec. 4.4).

4.1 Motivating Inspection RMAB

Motivated by the food establishment setting, we define a model RMAB with action constraints. This RMAB can be viewed as a collapsing bandit [16] or a resetting bandit [14], and both have indexability guarantees. Each establishment has an unobserved binary state that is either 1 (i.e., inspection passing) or 0 (i.e., inspection failing). When we act on the establishment, we assume that it is restored to the passing state and define the reward function to be 1 for each time period the establishment is in the passing state and 0 otherwise. We think of time periods as months—each establishment needs to be inspected once a year and will have a two-month period where this inspection can occur.

As the true states are not directly observable, each arm is a partially observed Markov decision process (POMDP) [2]. We can rewrite the POMDP as a fully observed belief-state MDP, allowing for direct representation as an RMAB.

For the underlying MDP, we assume passive transitions $P_i^{(0)}$ and active transitions $P_i^{(1)}$ as follows:

$$P_i^{(1)} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}, \quad P_i^{(0)} = \begin{pmatrix} P_i^{(00)} & P_i^{(01)} \\ P_i^{(10)} & P_i^{(11)} \end{pmatrix}$$

Each establishment has its own passive transition probabilities and all share the same action impacts—actions always restore the establishment to the passing state in the next timestep.

Converting this POMDP to a belief-state MDP yields a set of belief states that are reachable from the passing state $b_1 = [0, 1]$ (as a column vector), i.e., $(P_i^{(0)})^t b_1$, where t is any non-negative integer. In practice, the number of states needed to model belief dynamics precisely enough is dependent on the rate of MDP mixing. A faster mixing MDP will reach its stationary state faster and require fewer states—once we are sufficiently close to the stationary state, we can have the state transition to itself. The resulting belief-state MDP has a chain structure as shown in Fig 1 and resets to the head of the chain when the active action is taken.

Collapsing bandits generalize this setting by allowing $P_i^{(1)}$ to vary per arm, resulting in a two-chain structure. In general, our methods will also apply to this setting with minor modifications.

4.2 Action Windows and MDP Encoding

We use action windows as an exemplar for the family constraints where the constraint can be directly encoded into the RMAB structure, i.e., a vanilla RMAB with an action window constraint can be rewritten as a vanilla RMAB with a different arm structure. This is in some sense the ideal way to add constraints—we can apply whatever existing state-of-the-art algorithm directly.

To add action windows to the MDP structure, we add two pieces of information to the states (in addition to the belief state $b \in [0, 1]$), and modify the transitions to remove the impact of actions outside the window.

- t: the current timestep. In our motivating example, we can use t mod 12, as the inspection window for each establishment is at the same time each year. Alternatively, if the windows are not periodic, this can be replaced with a pair of counters, with one indicating the remaining time in the current window and the other the time until the next window.
- m: A counter for the number of actions remaining in the action window. When the process enters the action window, this is set to the total number of active actions allowed during the window (in our motivating example, this is 1). Each active action decrements the counter by 1. If the counter is zero, the active action is still available, but it has the same transitions as the passive action.

As shown in Figure 1, such an encoding increases the number of states in the MDP. The number of states is increased by a factor of $O(LM)$, where L is the number of counter values required to track when the window is active, and M is the total number of actions allowed during the window. For the motivating RMAB, this

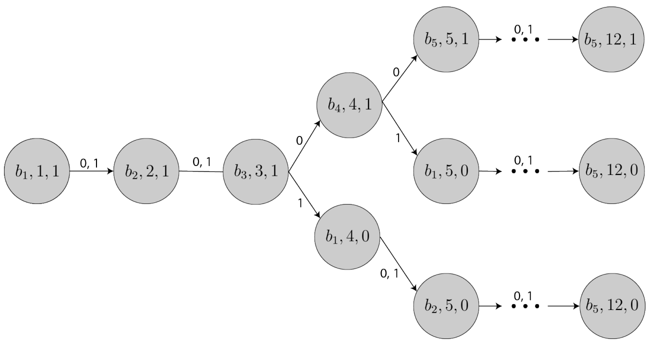

Figure 1: A example portion of the MDP after encoding the action window constraint. Suppose we have 5 belief states ($b_1, ..., b_5$), an action window at months 3 and 4, and 12 months between action windows. 0 is the passive action, 1 is the active action. After $(b_5, 12, 0)$ is reached, a new chain begins at $(b_5, 1, 0)$ (not shown).

increase is by a factor of 14. But such encoding is also applicable to more complex situations: An arm has multiple action windows and multiple inspections allowed per window. We provide a precise description of the new MDP in Alg. 1. All new transitions are deterministic (probability = 1).

To enforce that there are η timesteps of no action (sleep) after each action, as in [17], requires a factor of η more states. We add a counter to the state which records the number of timesteps until the next pull is allowed. When the counter is positive, the effect of an action is the same as the effect of no action.

Observation 1. *The Whittle index of arms that are not eligible to be pulled is zero.*

Arms outside the action window (or during mandatory sleep) have no advantage for the active over the passive action. Thus, their Whittle index will be zero. In practice, this means the Whittle heuristic will never select these arms, as long as there are some arms with positive action effects. If they are selected anyway, the agent can discard these actions to no ill effect.

Adding action windows to the MDP encoding will cause the Whittle index to increase when the end of a window is reached. This makes it more likely that an arm will be pulled before its window expires. Nevertheless, we begin to encounter the limits of the greedy Whittle index heuristic. For example, if we have several arms that have action windows that end at the same timestep, we may miss some action opportunities without planning ahead. In the next section, we develop a method for planning with lookahead, which will allow us to enforce frequency constraints as well as optimize timing of actions subject to window constraints.

Indexability. We are not aware of an existing class of indexable RMABs that includes the action window MDPs with counters that we define in this section. We empirically check for indexability through tracking the set of passive states as the subsidy changes and find no violations.

Algorithm 1 Encoding an Action Window in the Motivating RMAB

Input: $P^0, P^1, S = \{b_1, b_2, ..., b_j\}, T = \{t_1, t_2, ..., t_l\}$
Parameter: Number of steps between action windows L
Output: $P^{(0)}_{new}, P^{(1)}_{new}$

Initialization: $S_{new}, P^{(0)}_{new}, P^{(1)}_{new} = \emptyset$
for $i = 1, 2, ..., J$ **do**
  for $t = 1, 2, ..., L$ **do**
    add $[b_i, t, 0]$ into S_{new}
    if $t \in T$ **then**
      add $[b_i, t, 1]$ into S_{new}
    end if
  end for
end for
for all $s = [b, t, m] \in S_{new}$ **do**
  if $t = \|L\|$ **then**
    add $[b, t, m] \rightarrow [P^{(0)}(b), (t+1)\%L, 0]$ to $P^{(0)}_{new}, P^{(1)}_{new}$
  end if
  if $t \notin T$ **then**
    add $[b, t, m] \rightarrow [P^{(0)}(b), (t+1)\%L, m]$ to $P^{(0)}_{new}, P^{(1)}_{new}$
  else
    if $m = 0$ **then**
      add $[b, t, m] \rightarrow [P^{(0)}(b), (t+1)\%L, m]$ to $P^{(0)}_{new}, P^{(1)}_{new}$
    else
      add $[b, t, m] \rightarrow [P^{(1)}(b), (t+1)\%L, m-1]$ to P^{1}_{new}
      add $[b, t, m] \rightarrow [P^{(0)}(b), (t+1)\%L, m]$ to $P^{(0)}_{new}$
    end if
  end if
end for
return solution

4.3 Reforming MDP Transitions

As we can see from the encoding, one state is split into more than 10 new states, so transitions are also reformed. This reforming algorithm is intuitive: the belief state transits to the next based on the original MDP, time increases by one and the information whether inspected passes to the next.

4.4 Frequency Constraints and Lookahead

It is possible to enforce maximum action limits via editing the individual MDPs, but it is not possible to enforce minimums this way. In the motivating RMAB, we want to enforce the constraint that each establishment is inspected exactly once or multiple times per year since in the food inspection task, the authority has responsibilities to inspect every food establishment and never skip one. To enforce this kind of frequency constraint, we will replace the Whittle index heuristic with a sequential planning component that aims to maximize the sum of indices of pulled arms over a lookahead window, not just in the next time step.

We begin with the case where each arm needs to be pulled exactly one time over the lookahead window (and later relax this). In the motivating RMAB, this window will be one year. Formally, we let $a_{i,t}$ be whether arm i is pulled at time t and $w_{i,t}$ be the Whittle index of arm i at time t. These Whittle indices can come from an RMAB with any encoded constraints, such as those in the previous section. We seek to maximize $\sum_{i=1}^{N}\sum_{t=1}^{T} a_{i,t} w_{i,t}$, subject to the following constraints:

(1) $\sum_{i=1}^{N} a_{i,t} \leq k$: only k arms can be pulled in each timestep.
(2) $\sum_{t=1}^{T} a_{i,t} \leq 1$: each arm needs to be pulled at most once during the lookahead period. This is needed to make defining $w_{i,t}$ simple—otherwise $w_{i,t}$ depends on the time of the last pull.
(3) $a_{i,t} = \begin{cases} 1 \text{ or } 0 & \text{if t in action window} \\ 0 & \text{otherwise} \end{cases}$ This constraint forces a out of the action window to be 0, which satisfies one of our problem setting: arms can only be pulled during their action windows.
(4) Additional desired frequency constraints, e.g., each arm must be pulled at least once during certain timesteps.

Theorem 1. *Maximizing the sum of Whittle indices without additional frequency constraints OR with the constraint that each arm must be pulled exactly once during the lookahead window can be reduced to a weighted b-matching.*

Proof. The proof converts each timestep and each arm to nodes in the matching graph with different b-values. We formulate the weighted b-matching instance as follows. For each arm $i \in [N]$, create a node i. For each timestep t in the lookahead period, create a node t. For each arm-timestep pair (i, t) where an action can occur (i.e., no timing constraints are violated), create an edge of weight $w_{i,t}$ between i and t. Set the $b(t) = k$ for all t and $b(i) = 1$ for all nodes i. We claim that the maximum weight b-matching can be converted to an optimal lookahead schedule by taking each arm-timestep (i, t) pair that is included in the maximum weight b-matching and pulling the arm i at timestep t. Constraints 1 and 2 are satisfied by the definition of weighted b-matching. Constraint 3 is satisfied because edge (i, t) exists only if t is in i's action window. Thus, the optimal solution to the weighted b-matching must be the optimal solution to the lookahead problem.

To account for the additional frequency constraint that each arm must receive at least one pull in the lookahead window, if possible, a large constant can be added to all Whittle indices. The constant will cause each arm to be pulled once, if possible, because it is much larger than the increase in objective value that can be achieved by shifting the pull time for any individual arm. □

The proof implies that this form of lookahead can be optimized in strongly polynomial time. We remark that it is common in many applications for each arm to be pulled one or zero times over the next several timesteps.

In practice, it is convenient to solve this lookahead problem as an integer program (IP). We can do so with NL binary variables $a_{i,t}$ (where L is the length of the action window) and $T + N$ constraints. Because the polynomial tractability of weighted b-matching arises from total unimodularity [21], the IP can be solved very quickly via its LP relaxation. However, polynomial tractability is lost when sufficiently complex minimum and maximum number of pulls are added as additional constraints as the problem becomes equivalent to weighted bipartite b-matching [4].

The IP can be extended to more complex cases, e.g., where there are multiple pulls for each arm in the lookahead window. To modify the Whittle index for a time t' based on whether an arm pull at t happened or not, we can add constraints of the form:

$$w_{i,t'} \leq M(1 - a_{i,t}) + a_{i,t} w'_{i,t'} \tag{6}$$

where $w'_{i,t'}$ is the Whittle index at t' for arm i if it was pulled at time t. Note that Whittle indices will always decrease when a pull happens under our assumptions that an action improves the state of an arm. Thus, we can add LN additional constraints to allow for an additional pull during the lookahead period.

5 EXPERIMENTAL STUDY

We study the impact of different planning policies on reward, constraint satisfaction and computation time, both in synthetic (Sec. 5.2) and real data from CDPH (Sec. 5.3) domains. We describe the compared policies in Sec. 5.1.

5.1 Planning Policies

We compare the policies introduced by this paper with naive policies and baselines from the literature. The two policies introduced by this paper are:

- **Time-Constrained RMAB (TCB)** is a Whittle index heuristic policy with action windows encoded into the MDP as described in Sec. 4.2.
- **Time-Constrained RMAB with IP Lookahead (IP)** is the MDP constraint encoding of Sec. 4.2, using the IP-based lookahead of Sec. 4.4 rather than the Whittle index heuristic. W
- **Time-Constrained RMAB with IP Lookahead Equality (IPE)** is the MDP constraint encoding of Sec. 4.2, using the IP-based lookahead of Sec. 4.4 rather than the Whittle index heuristic and enforces that each arm is pulled exactly the desired number of times in its action windows. The main difference with IP policy is that IPE would reject the input if it is impossible to satisfy all arms' frequency constraints.

We use Gurobi 10.0.3 to solve the IP.

We include the following baseline and naive policies:

- **Random Policy (RP)**: k arms are selected randomly from the arms that are currently eligible to be pulled (i.e., in their action window). We use this as a lower bound on the reward achievable by any policy.
- **Risk-First Policy (RFP)** A common strategy in food establishment inspections is to target those who are most likely to fail inspections first [11, 12]. We simulate this by having the agent pull the k arms with smallest $P_i^{(0)}[1, 1]$ arms which are now in their action windows. Intuitively, these are the arms that are most likely to transition into the failing state.
- **Whittle-Index Policy (WIP)** is the Whittle index heuristic with the modification that the top k arms that are within their action windows are selected, an ad hoc modification that guarantees that window constraints are satisfied.

The same method is used to compute Whittle indices for all policies that require them. We simultaneously compute Whittle indices for all states of each arm using binary search over subsidies

		Number of Arms				
Policy	Budget	10	50	100	1000	5000
RFP	9%	/	17.79%	18.07%	20.27%	19.70%
	15%	27.38%	19.92%	19.77%	22.44%	21.81%
	20%	23.41%	19.49%	18.54%	20.84%	20.17%
	25%	26.26%	17.62%	16.21%	18.69%	17.91%
	30%	20.00%	14.88%	14.48%	16.20%	15.74%
TCB	9%	/	19.43%	**19.89%**	**22.13%**	**21.38%**
	15%	28.22%	21.41%	20.28%	22.86%	**22.28%**
	20%	**23.62%**	19.45%	18.63%	21.18%	20.54%
	25%	**26.54%**	17.67%	16.24%	**18.78%**	**18.02%**
	30%	19.94%	**15.01%**	**14.65%**	**16.29%**	**15.83%**
WIP	9%	/	18.98%	19.57%	21.93%	21.10%
	15%	26.91%	20.71%	19.34%	22.51%	21.84%
	20%	23.37%	19.23%	18.18%	20.65%	20.00%
	25%	26.28%	17.34%	16.00%	18.47%	17.74%
	30%	20.02%	14.80%	14.43%	16.12%	15.66%
IP	9%	/	**19.59%**	19.36%	21.66%	20.85%
	15%	**29.12%**	**21.85%**	**20.61%**	**23.40%**	22.66%
	20%	23.22%	**19.68%**	**18.95%**	**21.25%**	**20.61%**
	25%	26.13%	17.78%	**16.28%**	18.62%	17.86%
	30%	**20.04%**	14.98%	14.49%	16.13%	15.70%
IPE	9%	/	/	/	/	/
	15%	/	/	/	/	/
	20%	23.22%	**19.68%**	**18.95%**	**21.25%**	**20.61%**
	25%	26.13%	17.76%	**16.28%**	18.62%	17.86%
	30%	**20.04%**	14.90%	14.47%	16.13%	15.69%

Table 1: Total rewards improvement achieved for each in policy in the synthetic data domain as the budget and number of arms are varied. Results shown as percentage improvement compared with RP under same RMABs and budgets. TCB, IP or IPE performs best in all settings.

		Number of Arms				
Policy	Budget	10	50	100	1000	5000
RP	9%	/	0%	0%	0%	0%
	15%	0%	0%	0%	0%	0%
	20%	0%	0%	0%	0%	0%
	25%	0%	0%	0%	0%	0%
	30%	0%	2%	1%	0%	0%
RFP	9%	/	22%	30%	35%	33%
	15%	30%	52%	65%	67%	67%
	20%	90%	86%	86%	86%	85%
	25%	90%	98%	98%	99%	99%
	30%	**100%**	**100%**	**100%**	**100%**	**100%**
TCB	9%	/	22%	28%	31%	29%
	15%	**40%**	64%	62%	65%	66%
	20%	90%	84%	84%	89%	90%
	25%	90%	92%	95%	99%	98%
	30%	**100%**	**100%**	100%	100%	100%
WIP	9%	/	18%	25%	31%	29%
	15%	30%	56%	60%	64%	63%
	20%	80%	80%	81%	80%	81%
	25%	80%	88%	91%	91%	92%
	30%	90%	96%	98%	96%	96%
IP	9%	/	**32%**	**43%**	**42%**	**42%**
	15%	**40%**	**68%**	**75%**	**75%**	**75%**
	20%	**100%**	**100%**	**100%**	**100%**	**100%**
	25%	**100%**	**100%**	**100%**	**100%**	**100%**
	30%	**100%**	**100%**	**100%**	**100%**	**100%**
IPE	9%	/	/	/	/	/
	15%	/	/	/	/	/
	20%	**100%**	**100%**	**100%**	**100%**	**100%**
	25%	**100%**	**100%**	**100%**	**100%**	**100%**
	30%	**100%**	**100%**	**100%**	**100%**	**100%**

Table 2: Percentage of arms which their all constraints(action window and frequency) are satisfied. Under the same budget, TCB+IP satisfies at most 15% more arms compared with RFP. And also IP is the only policy which can satisfy all arms under 20% budget.

with the tolerance 10^{-6}. All experiments are run on a single core of AMD 3960X (4.5GHz).

5.2 Synthetic Domain

We begin with experiments using synthetic instances.

5.2.1 Data Preparation and Setup. In the synthetic domain, we generate $P_i^{(0)}[0,0]$ by sampling from Beta($\alpha = 5, \beta = 1$) and $P_i^{(0)}[1,0]$ by sampling from Beta($\alpha = 1, \beta = 5$). Each simulation is run for 60 timesteps. Each arm has 2 action windows, one window is two consecutive months, randomly selected, which then occurs every 12 timesteps. One pull is allowed in each window. We vary the number of arms in $[10, 50, 100, 1000, 5000]$ and the budget in $[9\%, 15\%, 20\%, 25\%, 30\%]$ per round. We start from 9% because at least we need a budget of at least 8.33% of all arms to satisfy all arms' constraints.

5.2.2 Results. The total reward accrued for each policy is presented in Table 1 as percentage improvements relative to the reward achieved by RP. The benefit of explicitly modeling constraints is clearly seen—TCB, IP, or IPE achieves the highest reward improvements in all settings. Intuitively, RFP and WIP act by identifying arms that are at risk of being in the bad state. The result is, when one of these risky arms enters its action window, it will always be pulled immediately. This is disruptive to the objective of pulls as many impactful arms as possible—it may cause other pulls to be wasted as arms exited their action window.

The IP lookahead is particularly effective in increasing coverage and enforcing coverage constraints. Table 2 compares the coverage (i.e., the percent of arms pulled in a year) for all policies. IP substantially increases coverage, even when there is no explicit coverage constraint, by avoiding conflicts between arms. IPE adds a frequency constraint requiring that each arm is pulled exactly desired times each year. Once the budget is large enough to allow for full coverage, it is achieved in every instance. Furthermore, reward is not decreased as a result of adding this constraint.

Despite a budget of as little as 8.33% providing enough pulls in principle, at least 20% is needed in order to satisfy the coverage constraint due to the constraint structure. Larger instances require

Policy	Time (sec)
RFP	163.52±0.39
WIP	172.02±1.29
TCB	2599.60±17.82
IP (Whittle computation)	2598.86±10.68
IP (Optimization)	119.51±1.19

Table 3: The average running time in seconds for a 5000-arm RMAB over 12 steps. For IP, we separate time spent on Whittle index computation and lookahead optimization. The largest computational costs come from the larger MDP that must be solved when the constraint encoding is introduced.

Policy	RP	RFP	TCB	WIP	IP	IPE
Reward	309555	317034	**318609**	317038	318601	318601
Cover	0%	89%	91%	89%	**100%**	**100%**
Months	47.12	48.03	**48.53**	48.01	48.51	48.51

Table 4: Rewards and coverage on 6750 establishments from Chicago, with a budget of 10% per timestep. Similarly to the synthetic data case, we see much TCB and IP perform best and IP/IPE is able to cover 100%, 10% higher than others. Months mean the number of average months establishment stay in passing state (out of 60) in the real data experiments.

slightly less than smaller ones due to the law of large numbers, but the impact of this is more limited than we expected.

5.2.3 Algorithm Computational Costs. Whittle index computation for one state is around **0.005 s**. Table 3 compares running times for the tested policies. Our policies consume around an order of magnitude more computational time than benchmarks due primarily to the need to compute Whittle indices for MDPs with more states. Theoretically faster algorithms for Whittle index computation exist and could decrease this difference. We experimented with the method of [6], but found that existing code fails on some test instances.

RAM consumption is low for all policies. IP consumes around 700MB RAM when running on a 5000-arm instance, TCB consumes about 500MB, and WIP consumes less than 200MB.

5.3 Food Establishment Inspection Domain

Using inspection data from the Chicago Data Portal [5], we implement a realistic RMAB setting.

5.3.1 Data and Setup. From 2010, CDPH has published every food establishment inspection result on the Chicago Data Portal [5]. The Food Inspection Dataset is a tabular dataset with 17 attributes for each establishment including license number, address, etc. The inspection results are shown in the "Violations" column: 0 means no violations and pass, 1 means violations appear and 2 means pass with conditions. In the experiment, both 0 and 2 are merged into a single good state and 1 is the bad state.

To create a realistic instance, we must infer the transition probabilities from the inspection trajectories for each arm. We limit ourselves to the 6750 establishments with at least 10 inspection records and use these inspection results to infer the probabilities in the transition matrix. We seek to compute the maximum likelihood transition matrix P_i for each establishment by minimizing the negative log-likelihood:

$$\sum_{t=0}^{T-1} \log P_i^{(a_{i,t})}[s_{i,t}, s_{i,t+1}], \tag{7}$$

where $P_i^{(a_{i,t})}[s_{i,t}, s_{i,t+1}]$ is the entry of the transition matrix corresponding to the transition between the states $s_{i,t}$ and $s_{i,t+1}$ that is observed, under the observed action $a_{i,t}$. Due to the partially observable nature of the problem, we only receive observations from establishments when they are inspected. Instead, we minimize the difference between the belief induced by the transition matrix and the observed state for each pair of consecutive inspections:

$$\sum_j (s_{i,t(i,j)} [P_i^{(0)}]^{t(i,j+1)-t(i,j)} - s_{i,t(i,j+1)})^2, \tag{8}$$

where j indexes consecutive inspections, $t(i,j)$ is the timestep of the jth inspection of arm i. We minimize this difference using Nelder-Mead [18].

In these experiments, we have 6750 arms, 60 timesteps, and a budget of 10% per timestep. We use the same random action windows of two consecutive steps per year as in the synthetic data.

5.3.2 Results. We compare the rewards achieved by all policies in Table. 4 relative to the reward achieved by RP. The outcome is similar as in the synthetic data case—we find TCB, IP and IPE have larger impacts on reward than RFP and WIP. IP and IPE can achieve 100% coverage, over 10% higher than others.

We can interpret the reward values in terms of the expected number of months each establishment stays in the inspection-passing state over the 60 timesteps in the last row of Table 4. We see that IP and TCB keep establishments in the passing state for about 1.4 additional months on average over the five years, 0.5 months longer than WIP or RFP. This supports our contention that explicitly modeling window constraints can have a substantial impact on practical RMAB settings.

The impact of all policies is less in the real data case than in the synthetic because arms are more likely to stay in the passing state without intervention. Some of this difference is likely due to modeling—because we limit the data to establishments with at least 10 inspections, we suspect that the modeled establishments, which have survived for longer, have higher pass rates than the overall population.

6 CONCLUSIONS

We present an RMAB-based method to solve scheduling problems with frequency and timing restrictions. To the best of our knowledge, ours is the first RMAB study to optimize scheduling problems under such constraints. Both synthetic data results and those using real food inspection data from CDPH suggest that our method for explicitly modeling constraints is critical for RMABs to have an impact in this setting. We hope our work paves the way for applying RMABs to other critical infrastructure maintenance and public service problems under constraints.

REFERENCES

[1] Abderrahmane Abbou and Viliam Makis. 2019. Group maintenance: A restless bandits approach. *INFORMS Journal on Computing* 31, 4 (2019), 719–731.

[2] Mauricio Araya, Olivier Buffet, Vincent Thomas, and Françcois Charpillet. 2010. A POMDP Extension with Belief-dependent Rewards. In *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (Eds.), Vol. 23. Curran Associates, Inc.

[3] Biswarup Bhattacharya. 2018. Restless bandits visiting villages: A preliminary study on distributing public health services. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*. 1–8.

[4] Cheng Chen, Lan Zheng, Venkatesh Srinivasan, Alex Thomo, Kui Wu, and Anthony Sukow. 2016. Conflict-Aware Weighted Bipartite B-Matching and Its Application to E-Commerce. *IEEE Transactions on Knowledge and Data Engineering* 28, 6 (2016), 1475–1488.

[5] Chicago Data Portal. 2023. Food Inspections. https://data.cityofchicago.org/Health-Human-Services/Food-Inspections/4ijn-s7e5.

[6] Nicolas Gast, Bruno Gaujal, and Kimang Khun. 2022. Computing whittle (and gittins) index in subcubic time. *Mathematical Methods of Operations Research* (2022).

[7] Pedro Cesar Lopes Gerum, Ayca Altay, and Melike Baykal-Gürsoy. 2019. Data-driven predictive maintenance scheduling policies for railways. *Transportation Research Part C: Emerging Technologies* 107 (2019), 137–154.

[8] Kevin D Glazebrook, Diego Ruiz-Hernandez, and Christopher Kirkbride. 2006. Some indexable families of restless bandit problems. *Advances in Applied Probability* 38, 3 (2006), 643–672.

[9] Christine Herlihy, Aviva Prins, Aravind Srinivasan, and John P. Dickerson. 2023. Planning to Fairly Allocate: Probabilistic Fairness in the Restless Bandit Setting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Long Beach, CA, USA) *(KDD '23)*. Association for Computing Machinery, New York, NY, USA, 732–740.

[10] Yu-Pin Hsu. 2018. Age of Information: Whittle Index for Scheduling Stochastic Arrivals. In *2018 IEEE International Symposium on Information Theory (ISIT)*. 2634–2638.

[11] Tom Schenk Jr; Gene Leynes, Aakash Solanki, Stephen Collins, Gavin Smart, Ben Albright, and David Crippin. 2015. Forecasting restaurants with critical violations in Chicago. https://github.com/Chicago/food-inspections-evaluation/blob/master/REPORTS/forecasting-restaurants-with-critical-violations-in-Chicago.Rmd.

[12] Vinesh Kannan, Matthew A. Shapiro, and Mustafa Bilgic. 2019. Hindsight Analysis of the Chicago Food Inspection Forecasting Model. In *Proceedings of AAAI FSS-19: Artificial Intelligence in Government and Public Sector*.

[13] Kesav Kaza, Varun Mehta, Rahul Meshram, and S. N. Merchant. 2018. Restless bandits with cumulative feedback: Applications in wireless networks. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. 1–6.

[14] Ali Al Khansa, Raphael Visoz, Yezekael Hayel, and Samson Lasaulce. 2021. Resource allocation for multi-source multi-relay wireless networks: A multi-armed bandit approach. In *Ubiquitous Networking: 7th International Symposium, UNet 2021, Virtual Event, May 19–22, 2021, Revised Selected Papers 7*. Springer, 62–75.

[15] Elliot Lee, Mariel S Lavieri, and Michael Volk. 2019. Optimal screening for hepatocellular carcinoma: A restless bandit model. *Manufacturing & Service Operations Management* 21, 1 (2019), 198–212.

[16] Aditya Mate, Jackson A. Killian, Haifeng Xu, Andrew Perrault, and Milind Tambe. 2020. Collapsing Bandits and Their Application to Public Health Interventions. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*.

[17] Aditya Mate, Lovish Madaan, Aparna Taneja, Neha Madhiwalla, Shresth Verma, Gargi Singh, Aparna Hegde, Pradeep Varakantham, and Milind Tambe. 2022. Field study in deploying restless multi-armed bandits: Assisting non-profits in improving maternal and child health. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 12017–12025.

[18] John A Nelder and Roger Mead. 1965. A simplex method for function minimization. *The computer journal* 7, 4 (1965), 308–313.

[19] Williamk Pulleyblank. 1973. *Facets of I-matching polyhedra.* Ph.D. Dissertation.

[20] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (1st ed.). John Wiley & Sons, Inc., USA.

[21] Alexander Schrijver et al. 2003. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. Springer.

[22] Shubham Singh, Bhuvni Shah, Chris Kanich, and Ian A Kash. 2022. Fair decision-making for food inspections. In *Equity and Access in Algorithms, Mechanisms, and Optimization*. 1–11.

[23] Bejjipuram Sombabu, Aditya Mate, D. Manjunath, and Sharayu Moharir. 2020. Whittle Index for AoI-Aware Scheduling. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. 630–633.

[24] P. Whittle. 1988. Restless bandits: activity allocation in a changing world. *Journal of Applied Probability* 25, A (1988), 287–298.

[25] Zhe Yu, Yunjian Xu, and Lang Tong. 2018. Deadline scheduling as restless bandits. *IEEE Trans. Automat. Control* 63, 8 (2018), 2343–2358.