A Finite Memory Automaton for Static and Dynamic Two-Armed Bernoulli Bandit Problems

Ariel Rao
Carnegie Mellon University
arielrao@cmu.edu

Abstract

Existing approaches to the multi-armed bandit (MAB) primarily rely on perfect recall of past actions to generate estimates for arm payoff probabilities; it is further assumed that the decision maker knows whether arm payoff probabilities can change. To capture the computational limitations many decision making systems face, we explore performance under bounded resources in the form of imperfect recall of past information. We present a finite memory automaton (FMA) designed to solve static and dynamic MAB problems. The FMA demonstrates that an agent can learn a low regret strategy without knowing whether arm payoff probabilities are static or dynamic and without having perfect recall of past actions. Roughly speaking, the automaton works by maintaining a relative ranking of arms based on payoff probabilities rather than estimating precise payoff probabilities.

1 Introduction

In the classic multi-armed bandit (MAB) problem, an agent is faced with a repeated decision of selecting an arm to pull from a set of arms, each with an unknown payoff structure. The agent can only observe the outcome of an individual arm pull and will receive a payoff for that outcome. The agent's goal is to determine, over time, a good arm selection strategy. A strategy can be evaluated by various metrics such as the rate of learning for the strategy, convergence to an optimal strategy, or cumulative payoff [2].

The MAB problem highlights the exploration versus exploitation dilemma found in reinforcement learning. In a good arm selection strategy, the agent will allocate sufficient resources for exploring the various arms to get an estimate of which arm is the best, before exploiting that arm. Having a reasonable exploration period protects the agent from prematurely committing to the wrong arm based on incorrect beliefs; learning quickly allows the agent to play the best arm for a larger proportion of the time.

Since its first formulation, the bandit problem has seen many permutations to reflect specific applications. To capture the computational limitations many decision making systems face, we explore performance under bounded resources in the form of imperfect recall of past information. In this paper, we present a finite memory automaton (FMA) designed to solve MAB problems in both static scenarios where the arm payoff probabilities remain constant and dynamic scenarios where arm payoff probabilities can change. In these environments, the FMA can match high performing perfect recall algorithms, even in decision problems with well over ten thousand stages.

2 The Multi-Armed Bandit Problem

The MAB problem is inspired by the situation a gambler is faced with when selecting amongst a set of slot machines to play. Each slot machine has an associated probability distribution for rewards that is unknown to the gambler. The gambler can only pull one slot machine arm at a time, but can choose from any of the slot machines. The gambler receives a reward from each arm pull and can use that result to inform future slot machine selections. Overall, the gambler wishes to maximize the rewards received for the duration of game play [2].

As a generalization of the gambler's situation, MAB problems model an agent who must reason about a set of arms, each with an unknown payoff structure. For simplicity, we restrict our analysis of the FMA's performance to the class of two-armed Bernoulli bandit (TABB) problems. In a TABB problem, there are exactly two bandit arms for the agent to select between. The payoff probabilities for each arm follow Bernoulli distributions, which give a payoff of 1 with a probability p and a payoff of 0 with a probability $1 - p$. We represent the decision problem as a tuple $D = (\theta_1, \theta_2)$ where θ_1 and θ_2 represent the probability distributions associated with playing arm 1 and arm 2 respectively. We call each decision point where the agent must select an arm to play a *stage*; the decision problem is built from a sequence of such stages and we call the full sequence a *horizon*. The length of a horizon may be characterized in a number of ways:

- **known infinite horizon:** the agent knows a priori that they will be playing an infinite number of stages.
- **known finite horizon:** the agent knows a priori the length of the full horizon (with length $l_h \in \mathbb{N}$).
- **unknown (finite) horizon:** the horizon is finite (with probability 1), but the agent is uncertain of the exact length. Each stage is considered to have a probability η of being the final stage.

In TABB problems, the true payoff probabilities associated with each arm can remain constant or change over time. The agent may or may not know the behavior of payoff probabilities in the decision problem. It is useful to distinguish three potential behaviors of payoff probabilities over time:

- **static TABB:** the arm payoff probabilities are fixed at the start and do not change over time.

- **dynamic TABB with small incremental changes:** the arm payoff probabilities are dynamic; all changes to the payoff probabilities are randomized, small incremental changes and typically occur after every stage. This class of TABB captures environments with constantly drifting payoff probabilities. An example process that follows this behavior is Brownian motion.
- **dynamic TABB with unbounded changes:** the arm payoff probabilities are dynamic; changes to the payoff probabilities can be of random, unbounded magnitude and typically occur with low frequency. We use this class of TABB to reason about environments where payoff probabilities are subject to punctuated and infrequent changes.

We index dynamic decision problems with a time variable t, as in D_t, to represent the arm payoff probabilities at stage t. The output of playing any arm is a signal $s \in \{0, 1\}$ where we interpret $s = 0$ as is a poor payoff and $s = 1$ as is a good payoff.

A strategy dictates how the agent selects an arm to play at each stage. Evaluation of a strategy may take a number of factors into consideration. For instance, convergence of an arm selection policy to the optimal arm selection strategy is a property often expected in infinite horizon MAB problems. In contrast, cumulative payoff may be more appropriate in finite horizon MAB problems. Other metrics such as regret compare the arm selection strategy employed against an optimal strategy at every stage played. In this case, we usually define the optimal strategy as one where the arm with the highest payoff probability is always selected.

3 Applications

Static MAB [1] [3] [4] and dynamic MAB with small incremental changes in payoff probabilities [5] have been studied in previous literature. We present an example of a static MAB problem from web design evaluation as well as an example of a dynamic MAB problem with small incremental changes from distribution logistics. We also motivate dynamic MAB with unbounded changes by providing an example in forward proxy server selection. This class of dynamic MAB captures a fundamentally different type of change to the decision problem which is not afforded by dynamic MAB with small incremental changes. These three applications highlight static and dynamic properties of MAB in isolation but we also suggest how a commbination of properties can be used to model more realistic decision problems.

3.1 Web Design Evaluation

Web browsing behavior has been shown to vary based on the structure in which content is presented; web designers are interested in comparing user behavior in response to different web page designs. For instance, an e-commerce company would like to design a search results page optimal for encouraging viewers to click on the products. The evaluation metric here is click-through-rate, which represents the proportion of viewers who select on a product link to the total number of viewers who come to the page; a higher click-through-rate reflects a better page design.

In this setting, each distinct web page design in question is associated with an unknown click-through-rate. As an MAB problem, a web page design is a bandit arm where the click-through-rate is the associated payoff probability. When a user is presented with a web page, their click-through behavior serves as an indicator for the design quality of that page. Navigating to the page and clicking on an item is taken to be a good payoff while navigating to and away from the page without clicking on any items is taken to be a poor payoff. Thus, we can easily model the decision problem of identifying the web page design with the best relative click-through-rate as a static MAB problem.

3.2 Distribution Logistics

Distribution logistics is concerned with the flow of goods from a vendor to the customer. Imagine a vendor with a single product that has a fixed cost of production and a fixed sale price, but a dynamic cost of shipment. Take the marginal profit the vendor receives to be the difference between the revenue from the sale and the costs of production and shipment. A vendor's goal is to maximize marginal profit.

A shipment can take one of a set of fixed paths to reach the final destination; the dynamic cost of shipment is determined by the totality of shipping costs, which considers the transportation method, route distance, shipping time, and similar factors. Further, the effect of individual factors on the final shipping expense can fluctuate during different times of the day or different months of the year. For example, any path that includes driving on a highway may take more time during rush hour and be less desirable. In contrast, all transportation methods would be costly during a winter snow storm, but paved highways are safer to drive on and thus more desirable. Overall, which shipping route is optimal can change over time, as optimality is contingent on conditions that change over time. While the potential shipping costs can vary greatly from one set of conditions to another, it is not likely that the totality of conditions change drastically between shipments. Instead, individual contributing factors may change and result in small shifts in the the overall expense. Here, we see that a dynamic MAB problem with small incremental changes in arm payoff probabilities captures the task of path selection where each potential path is an arm and the marginal profit derived from the shipping expense is the payoff.

3.3 Forward Proxy Selection

A proxy server facilitates requests between a user and a target server. A forward proxy serves as an intermediary between any number of users and external resources such as the Internet. Typically, the speed of service is inversely proportional to the number of current users. For a given request, a user can select from a set of forward proxies, where the number of other users for that proxy is unknown. As a MAB problem, we can represent each proxy as an arm with the speed of service equivalent to expected payoff.

Typically, the traffic for a given proxy is fairly stable. However, it is conceivable for the load of a proxy to undergo changes unannounced to users. For instance, a server S may go down; the traffic previously directed through S will be distributed across remaining proxies with no guarantee of uniform distribution. Similarly, a server may be upgraded and be able to handle a larger load without a drop in speed. A third situation is where a different user may select one of the proxies to run a large batch job, tying up the selected proxy for some duration of time. None of these cases are likely frequent occurrences, but in the event of any, there could be a change in performance of unbounded magnitude to one or more proxies. Further, users are not necessarily aware of these external factors

and are given no prompt to abandon their previous beliefs about the proxies, except from the observed performance. This type of behavior is captured nicely in our presentation of an unbounded dynamic MAB problem.

3.4 A Few Remarks

Web design evaluation, distribution logistics, and forward proxy selection are just three examples of the applications for multi-armed bandit problems, selected to showcase different features of MAB problems. Specifically, arms as characterized by these domains tend to predominantly exhibit one type of arm behavior. However, it is more conceivable that they actually exhibit all three types of arm behavior, to different extents. For instance, we can easily see how a larger change in the environment can effect distribution logistics; if a water main breaks and results in the closing of a road, the agent is in an environment where both classes of dynamic MAB apply. Similarly, for web design evaluation, the demographics of the target user group may change over time and thus the optimal page design required to reach the new demographic will slowly change in response. From these situations, we can see that allowing a decision problem to take on more than one form allows for more representative models. Conversely, an algorithm which can perform well in multiple environments without additional mechanisms or tuning would be able to reason about more complex and realistic decision problems.

4 Related Work

A number of approaches have been explored for solving bandit problems under computational constraints.

One of the most simple strategies is ϵ-greedy which uniformly balances exploration and exploitation. With a probability of ϵ, a random arm is selected in exploration across the set of arms. With a probability of $1 - \epsilon$, the arm currently believed to have the highest payoff probability is selected in exploitation. The beliefs of about an arm's payoff probability is contingent on the signals received from that arm in the previous stages; the arm currently believed to have the highest payoff probability is the one which historically has given the highest average payoff. In its normal form, ϵ-greedy does not necessarily converge to the optimal strategy. Variations of this strategy exist such as ϵ-decreasing and ϵ-first to try and remedy this limitation [8].

Another class of strategies is based on determining a confidence interval around estimates of arm payoff probability, such as with an upper confidence bound (UCB) [1]. The agent maintains beliefs about the upper bound of the payoff probability for every arm, selecting the arm with highest upper bound at each round.

Similarly, Bayesian approaches track beliefs about the true payoff probabilities. One such approach is the Bayesian Learning Automaton (BLA) [3] [4] which maintains only a beta distribution for each arm to represent the agent's beliefs about the payoff probability of that arm. The α and β parameters for each beta distribution are the number of previously received signals, good and poor signals respectively, from the corresponding arm; these parameters are updated after each stage. Sampling from each of the distributions is used to determine an the agent's action at a stage while simultaneously balancing exploration and exploitation. The BLA was designed for the static MAB problem and converges to the optimal strategy with probability 1. Variations of the BLA have been developed to cater to the class of dynamic problems where payoff probabilities can undergo small incremental changes [5].

Across these approaches, there is a general interest in determining the true payoff probabilities of the arms; ϵ-greedy strategies sample all arms to get a rough estimate of each arm, UCB estimates a range where the true payoff probabilities are likely to lie within, and BLA uses a beta distribution to estimate true payoff probabilities. However, the MAB is not fundamentally an estimation problem; the MAB is a ranking problem. An agent is best rewarded when playing an arm with the highest expected payoff probability. Naturally, one way to identify the arm with the highest expected payoff probability is to form an estimate of the payoff probabilities of all arms and select the maximal arm. However, the agent's payoff is not contingent on the accuracy of these estimates; identifying the best arm is the extent to which an agent is interested in estimating the true payoff probabilities of all arms. The FMA bypasses rigorous payoff probability estimation and directly produces a relative ranking of the arms. The FMA explicitly considers how different the payoff distribution amongst the arms are, rather than just what the estimated payoff probabilities are, to identify the arm with the highest expected payoff probability.

Further, most existing approaches to MAB problems rely on the full history of received signals for perfect Bayesian updating of beliefs about the arm payoff probabilities. There have been a number of techniques using finite automata [9] and Turing machines [6] [7] to represent a computationally limited agent for other problems within game and decision theory. In these systems, *memory* is defined by the number of states in the automata and represents the automata's computational power. The finite automata has been shown to perform well on a variant of the one-armed bandit, where an agent must decide whether an arm is "good" or "bad" given a sequence of signals.

The FMA borrows constructs for a bounded decision making structure from automata theory while leveraging a payoff probability tracking mechanism similar to that presented in the unbounded BLA. However, by implicitly encoding beliefs about the relative ranking of the bandit arms within a finite state space, rather than a payoff probability estimate for each arm like the BLA, the FMA is essentially restricted to a finite set of possible beliefs. This constraint prevents the FMA from updating belief on all previous signals and captures the notion of a computationally limited agent. Even still, the FMA's ability to maintain rankings rather than point-estimates makes it a flexible enough framework to perform well in static and dynamic cases alike, without any modifications.

5 The Algorithm

We model a decision maker as a stochastic finite state automaton where each state encodes discrete beliefs about the payoff probabilities of each arm in the decision problem. A transition between states is determined by the previous arm selected and its corresponding output signal.

The following subsections explain our decision maker automaton as defined for the TABB problem. We present the framework for the FMA in Section 5.1 and the full algorithm in Section 5.2. Motivations for further constraints on various components of the FMA are given in Sections 5.1 - 5.1, as one way to specify a high performing FMA.

5.1 Decision Maker

To limit the automaton to a finite state space, fix the constant $m \in \mathbb{N}$ to denote the size of the automaton. m is interpreted to be the granularity of beliefs a decision maker can have about any given arm. For simplicity, we take m to be uniform across all arms.

Take a state to be a tuple $\omega = (r_1, r_2)$ which encodes the decision maker's beliefs about the ranks of the two arms. Each r_i is constrained by $1 \leq r_i \leq m$ and represents the current rank of arm i. We interpret $r_i > r_j$ as the decision maker believing that arm i has a higher payoff probability than arm j.

A decision maker for the TABB problem is a tuple $DM = (\Omega, \omega_0, a, t)$ defined by the following conditions:

- C1 $\Omega = \{1, ..., m\}^2$ is the state space;
- C2 $\omega_0 \in \Omega$ is the starting state;
- C3 $a : \omega \to \{1, 2\}$ where $a(\omega)$ specifies which arm the decision maker should play when at state ω;
- C4 $t : \omega \times \{1, 2\} \times \{0, 1\} \to \omega$ where $t(\omega, i, d)$ determines the decision makers' new state as updated by the judgement of the last signal from arm i.

Ranking

In our DM, the ranking structure is bound by the granularity constant m. At each state, every arm in the decision problem has an associated rank which represents the decision maker's beliefs about the payoff probability of that arm. The DM does not track all previous outcomes (possibly infinitely many) and is only allowed a coarse estimate of an arm's payoff probability as encoded by the state space. As we will see, arms with a good payoff probability will tend to maintain higher rankings while arms with a poor payoff probability will tend to have lower rankings.

When $m = 1$, the decision maker can have only a constant belief about the payoff probability of any arm and will do exactly as well as random selection. In the subsequent analysis, we consider only $m \geq 2$.

State Space

The size of a decision maker's state space is uniquely determined by the parameter $m \in \mathbb{N}$, which restricts the granularity of rankings the decision maker can have. For the TABB problem with two arms, the size of the state space is given by:

$$\mid \Omega \mid = m^2$$

When we want a decision maker to have more refined beliefs, we have a larger state space. Taking a larger state space to be analogous to a more computationally expensive problem, we see that our framework directly relates computational power to the required precision in decision making.

Starting State

In the general case, we take the decision maker's starting state to be the tuple

$$\omega_0 = (n, n) \in \Omega, \qquad 1 \leq n \leq m$$

Specifically, the decision maker assigns the same ranking to each arm in the decision problem.

However, a decision maker's starting state can also capture information about prior beliefs. If the decision maker believes a priori that one arm has a higher payoff probability than another, these prior beliefs can be captured in the rankings.

Action

The action function considers the DM's beliefs as determined by the current state and is used to encode a decision maker's tendencies for exploration and exploitation. For each $\omega \in \Omega$, the action function a assigns a probability measure for selecting the next arm to play.

In the early stages, we would like the DM to explore both arms. A nice balance of exploitation and exploration can be achieved when a DM's tendency to explore is inversely proportional to the DM's confidence in the arms' relative rankings. For instance, take two arms with current ranks r_1 and r_2. Without loss of generality, suppose $r_1 \geq r_2$. Intuitively, the larger the magnitude of $r_1 - r_2$, the more confident the DM is that $\theta_1 \geq \theta_2$. We loosely use the terms *strong* and *weak* here to characterize the magnitude of differences between ranks, where strong beliefs capture large differences in relative arm payoff probabilities and weak beliefs capture small differences in relative arm payoff probabilities. Naturally, we want the DM to play the arm strongly believed to have a better payoff probability at least as often as when that arm is weakly believed to have a better payoff probablity. In general, we capture the confidence in rankings with

$$\mid r_1 - r_2 \mid$$

Thus, we formalize exploitation as a weighted combination of the 2 components: the magnitude of the arm ranks and the difference in ranks.

$$e_\omega = c_1 \Big(\frac{\sum_{i=1}^{2} |r_i - \frac{m}{2}|}{2} \Big) + c_2 \Big(\mid r_1 - r_2 \mid \Big)$$

The DM is highly likely to explore both arms early on but will exploit the current best arm with increasing probability as the DM becomes more confident in the ranking assigned to each arm as well as in the relative ranking overall.

The notation i_h is introduced to denote the arm with a higher rank at the current state; i_l denotes the arm with a lower rank. $\mathcal{U}(a, b)$ is used to denote sampling from a uniform distribution with an interval from a to b. The constants α and C_a outline rate at which exploration and exploitation are traded off. We define the action function:

$$a(\omega) = \begin{cases} i_h & r_h = m \\ i_h & \mathcal{U}(0, {e_\omega}^{\alpha + C_a}) \geq {e_\omega}^{\alpha} \text{ for } \alpha, C_a \geq 1 \\ i_l & o.w. \end{cases}$$

When both arms have the same rank, the DM has no preference between them and may employ some predefined tie-breaking metric to select the next arm. An example is a DM who prefers arms with a lower index value. However, since the DM cannot distinguish these arms in terms of expected payoff probability, we will consider DMs that randomize selection in the event of a tie.

A special case for consideration is when an arm has the highest ranking, m. It is advantageous to eliminate exploration in these states and play only that highest ranking arm. Generally, exploration is in place to prevent the DM from getting stuck playing a sub-optimal arm. When we eliminate exploration for this class of states, we do not introduce any problematic behavior. If the currently highest ranked arm does not have perfect payoff, the DM will eventually transition out of the state that assigns that arm the highest ranking and return to a state where exploration behaves as usual. However, if the highest ranked arm has perfect payoff, our DM is able to take full advantage and exploit that arm in all future stages.

In the general case, the DM selects arm i_h with a probability

$$\frac{{e_\omega}^\alpha}{{e_\omega}^{\alpha+C_a}}$$

With the constraints $\alpha, C_a \geq 1$, we ensure that the DM exploits arm i_h with a much higher probability when e_w is high.

Transition

Once the DM has played a particular arm and received the resulting signal, the DM must update its beliefs. To make an update, the DM would transition into a state where the ranks capture the updated beliefs. A judgement function j is first used to determine how the ranks change in response to a certain signal; an arm can either increase rank up to m or decrease rank down to 1. At any given stage, we require that the magnitude of a change in rank be at most 1.

Recall that $s = 0$ is a poor payoff and $s = 1$ is a good payoff. Thus, for the TABB, we take the judgement function:

$$j(s) = \begin{cases} +1 & s = 1 \\ -1 & s = 0 \end{cases}$$

Given current beliefs about all arms and the interpretation of a new signal from j, the DM must decide whether or not to update on this information. In general, the transition decision is reduced to either making the transition on the new information or ignoring the new information and preserving current beliefs. If a transition occurs, only the rank of the arm last played can change. Since the DM has a finite state space and can only retain a sliding window of past signals, the DM becomes more sensitive to more recent signals. Thus, we develop the notion of *inertia* to mitigate this recency bias.

Traditionally, inertia characterizes the resistance to change. To capture a resistance to belief updates in our DM, we assign each undirected edge in the state space a score that tracks the frequency of traversal along that edge. The frequently traversed edges, with high edge scores, are considered to have low inertia while the edges that have hardly been traversed will have low edge scores and high inertia. However, the persistence for a single traversal is finite. Over time, edges that have been traversed with high frequency in the recent stages will have high edge scores while edges that have been traversed the same number of times but in much earlier stages will have relatively lower edge scores. We now formulate transition inertia contingent on edge scores. Specifically, a DM is more likely to make a transition along an edge with a high score than along an edge with a low score. With transition inertia, the DM becomes partially constrained within a cluster of states that broadly capture estimates about arm rankings. In other words, the DM's performance is bound less to the strict assignment of ranks to arms and, as a result, becomes less sensitive to specific sub-sequences of signals.

We refer to the inertia of moving between states ω_j and ω_k as $inertia_{j,k}$ as represented by the edge score between these states. Since edges are undirected, we take $inertia_{j,k}$ to be equivalent to $inertia_{k,j}$.

Let $\omega^{i,d}$ be the state where the r_i at $\omega^{i,d}$ is equivalent to $r_i + d$ at ω. The constants β and C_t outline rate at which inertia impacts transition probabilities.

Formally, we define the transition function:

$$t(\omega, i, d) = \begin{cases} \omega & (r_i = 1, d = -1) \text{ or } (r_i = m, d = +1) \\ \omega^{i,d} & \mathcal{U}(0, {inertia_{\omega,\omega^{i,d}}}^{\beta+C_t}) \geq {inertia_{\omega,\omega^{i,d}}}^{\beta} \\ & \text{for } \beta, C_t \geq 0 \\ \omega & o.w. \end{cases}$$

Since we assume independence amongst arms and have isolated the judgement of signals, there is usually exactly one other state that the DM can transition to, $\omega^{i,d}$. There are two exceptions captured by the first case in the transition function. Suppose i is played.

Exception 1: $j = -1$ and $r_i = 1$

Exception 2: $j = +1$ and $r_i = m$

In both situations, the DM already holds the strongest possible beliefs about arm i and will remain in the current state.

In the general case, the DM selects arm i_h with a probability

$$\frac{{inertia_{\omega,\omega^{i,d}}}^{\beta}}{{inertia_{\omega,\omega^{i,d}}}^{\beta+C_t}}$$

With the constraints $\beta, C_t \geq 1$, we ensure that the DM transition with a much higher probability when edge scores are high (and inertia is low). If the DM falls into the second case and makes a transition, the score along that edge is incremented by C_I. To enforce the finite persistence of traversals, all edges are subject to a probabilistic decrement at the end of each stage. Take the rate of refresh to be C_U and the magnitude of a decrement to be C_D. Effectively, edges that have been traversed recently will maintain a moderate score while edges traversed in the past will only be decremented.

5.2 Algorithm

Algorithm 1 Finite Memory Automaton

procedure FMA($(\theta_1, ..., \theta_n), (\Omega, \omega_0, a, t)$) ▷ a decision problem and decision maker respectively
  $currentState \leftarrow \omega_0$
  $continue \leftarrow$ *True*
  while $continue$ **do**
    $armToPull \leftarrow$ *a(currentState)* ▷ the arm to pull next
    $s \leftarrow pull(armToPull)$ ▷ the output signal
    $d \leftarrow$ *j(s)* ▷ the transition judgement
    $currentState \leftarrow t(currentState, armToPull, d)$ ▷ the new state after updating
    $updateInertia()$
    if *end*$() < \eta$ **then** ▷ a probability η of ending at the current stage
      $continue \leftarrow False.$

6 An Example

The following section walks through an example with a TABB problem to demonstrate how the different components of the FMA interact.

6.1 Problem Setup

Define a TABB problem with payoff distributions 0.3 and 0.7. We can characterize this decision problem as $D = (0.3, 0.7)$. For this example, assume we have an infinite horizon.

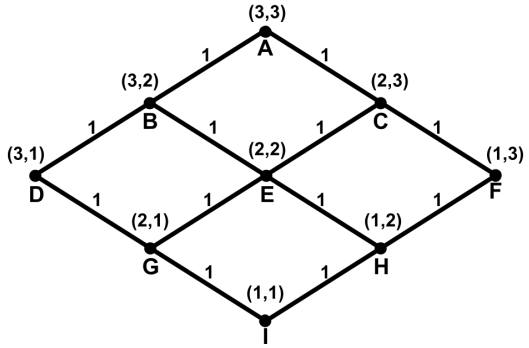

Figure 1: Ω given a two arms and $m = 3$; edges initialized to 1

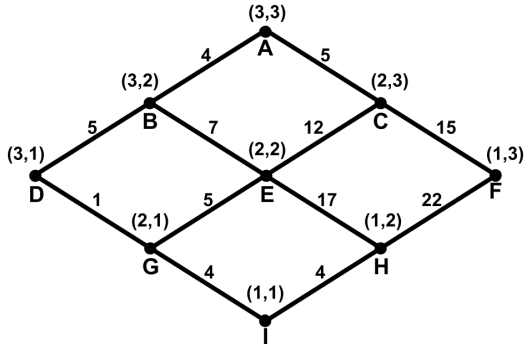

Figure 2: Ω before stage 100.

Take a decision maker that can distinguish between 3 ranks ($m = 3$). A state space diagram is given in Figure 1 for the full state space Ω. A node represents a state with distinct rankings. A letter has been assigned to label each state for this example. An edge between two nodes indicates that a valid transition can be made between these two states. The edge weights are displayed along the edges and have been initialized to 1.

Let the decision maker be unbiased. We derive the starting state ω_0 with

$$\omega_0 = (\lfloor \tfrac{m+1}{2} \rfloor, \lfloor \tfrac{m+1}{2} \rfloor) = (2, 2),$$

which is state E.

For a, define the constants $\alpha = 1$ and $C_a = 2$. For exploitation e_ω, take $c_1 = c_2 = 0.5$. For t, define the constants $\beta = 2$ and $C_t = 2$. For each transition, take the edge score parameters to be $C_I = 1$, $C_D = 1$, $C_U = 0.1$. We will use the following definitions for a, j, and t.

$$a(\omega) = \begin{cases} i_h & r_h = m \\ i_h & \mathcal{U}(0, {e_\omega}^3) \geq e_\omega \\ i_l & o.w. \end{cases} \qquad j(s) = \begin{cases} +1 & s = 1 \\ -1 & s = 0 \end{cases}$$

$$t(\omega, i, d) = \begin{cases} \omega & (r_i = 1, d = -1) \text{ or } (r_i = m, d = +1) \\ \omega^{i,d} & \mathcal{U}(0, inertia_{\omega,\omega^{i,d}}{}^4) \geq inertia_{\omega,\omega^{i,d}}{}^2 \\ \omega & o.w. \end{cases}$$

We have now specified the decision maker $DM = (\Omega, E, a, t)$

6.2 Stage 1

We begin by selecting an arm to play with $a(E)$. The exploitation e_E is

$$e_\omega = 0.5\Big(\tfrac{\sum_{i=1}^{2} |r_i - \frac{m}{2}|}{2}\Big) + 0.5\Big(\mid r_1 - r_2 \mid\Big) = 0$$

which falls into the third case of a. Since the ranks of arm 1 and 2 are the same, a random arm is selected. Suppose arm 1 is selected and pulling arm 1 returns a signal -1. We determine the direction of change d with $j(-1)$, getting $d = -1$. Thus, if any transition occurs, the new state would be $H = (1, 2)$. We have $inertia_{E,H} = 1$. Suppose $\mathcal{U}(0, 1) = 0$, falling into the third case in t. The decision maker does not transition and remains in state E. Since no transitions were made, there are no changes to edge scores. All edge scores are at the minimum value 1 so no regrowth is applicable at this stage.

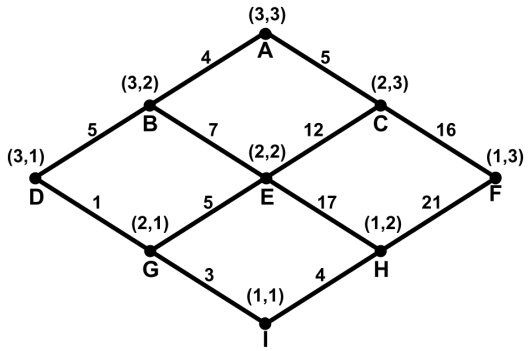

Figure 3: Ω after stage 100.

6.3 Stage 100

Suppose that, after 99 stages, the current state is now $C = (2, 3)$. Figure 2 displays the edge scores at the start of this stage.

We select another arm to play with $a(C)$, where we have exploitation $e_C = 1$. Suppose $\mathcal{U}(0, 1) = 0$ and we fall into the fourth case in a. Instead of exploiting arm 2, we now recompute the next arm to pull excluding arm 2. Since arm 1 is the only remaining arm, we fall into the second case of a and pull arm 1. Suppose we receive the signal -1, giving $d = -1$. If any transition occurs, the new state would be $F = (1, 3)$. We have $inertia_{C,F} = 15$. Suppose $U(0, 15^4) = 3654$, falling into the second case in t. The decision maker does transition and is now in state F. Having made this transition, we increase the score along this edge by c_I to $inertia_{C,F} = 16$.

After the transition, the edge scores must be recalibrated. Each edge has in independent probability $C_U = 0.1$ of being decremented by $C_D = 1$. Suppose the edge between H and F and the edge between G and I are subject to regrowth. We are now in state F with edge scores as specified in Figure 3.

7 Experiments

The following section contains results from static TABB problems and two classes of dynamic TABB problems. To convey that the FMA framework performs well in multiple TABB environments, each DM is run with the same settings, varying only m. We further compare the performance of the FMA in each environment to other algorithms specialized for that environment.

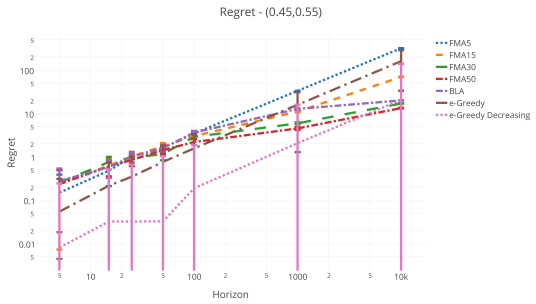

Figure 4: Plot of the estimate of regret for $D = (0.45, 0.55)$ comparing the FMA with various memory capacities to other algorithms on a log scale.

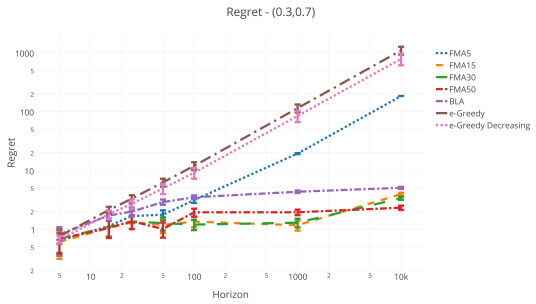

Figure 5: Plot of the estimate of regret for $D = (0.3, 0.7)$ comparing the FMA with various memory capacities to other algorithms on a log scale.

To see how performance varies over horizon length, we take η to be 0 for the infinite horizon. Cumulative payoff is sampled at various horizons, representing the payoffs an agent would have received if the decision problem had ended at that stage.

7.1 Evaluation Metric

We evaluate a strategy with a measure of regret, comparing the actual payoff of playing that strategy to an optimal benchmark. In our evaluation, we take the optimal strategy to be the one where the highest paying arm is played at each stage for the full length of the horizon. At each stage, there is also a discount factor η which corresponds to the likelihood of arriving at this stage. Thus regret is defined:

$$r = \sum_{t=1}^{h} (1-\eta)^h \big(max(D_t) - s_t\big)$$

We use this metric across both static and dynamic TABB problems. Since the payoffs of the arms are stochastic, each decision problem is simulated over 500 trials with the reported values averaged across trials.

7.2 Static World

In a static TABB problem, we begin by fixing the payoff probablities of the two arms. We then evaluate the performance of the FMA for different values of m. Performance of the BLA is used as a baseline, as the BLA outperforms many of the other techniques in the general case. ϵ-greedy with $\epsilon = 0.2$ and ϵ-greedy decreasing with $\epsilon = 0.2$ and $\delta = 0.001$ are also reported.

Experiments are run over decision problems with varying degrees of difficulty, where a difficult decision problem requires selecting amongst arms with similar payoffs. Figure 4 and Figure 5 plot regret sampled from horizons of length 5 to 10,000 for the decision problems $(0.45, 0.55)$ and $(0.3, 0.7)$ respectively.

From these results, we can see that there is a large variance in performance in a very short horizon. This behavior is reasonable as there is insufficient information for having well formed beliefs about the ranking of the arms. Further, since so few signals are received in such a short horizon, there is no statistically significant improvement to be had with more granular beliefs (i.e. higher m). However, as the length of the horizon increases, the ability to make more granular distinctions in beliefs about ranking make a statistically significant improvement. We see that lower m is required for the FMA to match the BLA's performance when faced with an easier decision problem.

Another nice property of the FMA is stability in results. Evident in the decision problem $(0.45, 0.55)$, the good average performance of algorithms such as ϵ-greedy masks the variance in regret. The FMA, like the BLA, has much lower variance across the 500 trials. Note that there is an upper bound on how well the FMA can perform, as values for exploration are fixed. Even still, the FMA performs well by quickly reaching states with an optimal ranking and minimal exploration.

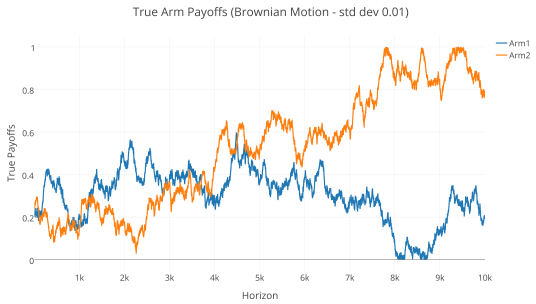

Figure 6: Example of change in true payoff probabilities of two arms following Brownian motion with a standard deviation of 0.1.

7.3 Dynamic World - Incremental Changes

One class of non-stationary TABB is presented where bandit arms follow Brownian motion [5]. A modification to the BLA incorporates dynamic Thompson sampling which allows the beta distributions to respond to observed changes in payoff probabilities. Effectively, the Thompson BLA (TBLA) enforces a weak upper bound on how certain the agent can be when estimating payoff probabilities and consequently allows the TBLA to better estimate the payoff probabilities as they shift. When testing the FMA, we consider Brownian motion with a cutoff boundary at $[0, 1]$. Take a standard deviation of σ and let the magnitude of change in payoff probability be given by $\delta \sim N(0, \sigma^2)$. The true payoff probability of θ_i at D_t is given by:

$$(\theta_i)_t = \begin{cases} (\theta_i)_{t-1} + \delta & 0 \leq (\theta_i)_{t-1} + \delta \leq 1 \\ 0 & (\theta_i)_{t-1} + \delta < 0 \\ 1 & 1 < (\theta_i)_{t-1} + \delta \end{cases}$$

Figure 6 shows one such sequence of changes for a decision problem with $\sigma = 0.1$ and a cutoff boundary.

The FMA, along with the BLA, dynamic Thompson BLA, and ϵ-greedy are evaluated against different values of σ for a horizon of 5000. Figure 7 shows the results of each algorithm run on the same sequence of arms; the maximum regret for the sequence is plotted as a worst case baseline.

For small values of σ, we see similar results to the static MAB problem. For larger values of σ, we see that FMA

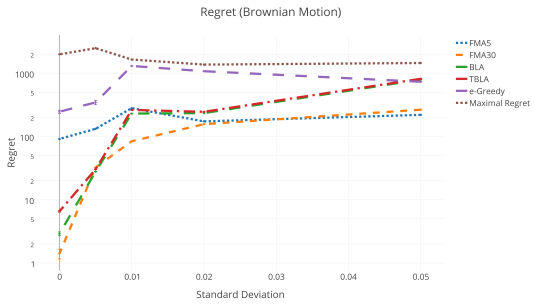

Figure 7: Compare the estimate of regret for the FMA with various memory capacities to other algorithms in decision problems with different standard deviations.

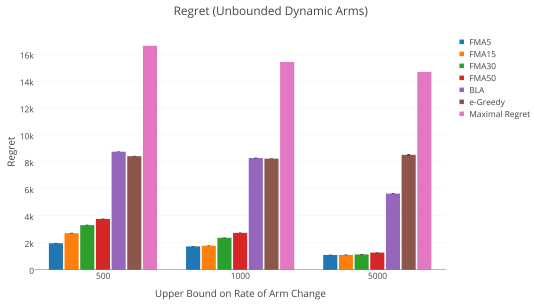

Figure 8: Compare the estimate of regret for the FMA with various memory capacities to other algorithms in decision problems where an arm's payoff probability stochastically changes for $b_h = 500, 1000, 5000$ in a horizon of 50000.

with $m = 5$ perform as well as FMA with $m = 30$. As the true payoff probabilities of the arms are changing quickly, the relative ranks of the arms are likely to be changing quickly as well. Thus, the FMA is not given much time to make use of the additional ranking power and relies on smaller differences in ranking.

7.4 Dynamic World - Unbounded Changes

A second, less explored interpretation of non-stationary arms is where the payoff probabilities of one or more arms can change at any given time, bounded only to $[0, 1]$. For each of the arms, we take an upper bound b_h where the payoff of an arm changes at least once every b_h stages in the horizon. We use this bound to analyze how the frequency of change impacts performance. Neither ϵ-greedy nor BLA strategies are equipped for this class of dynamic MAB, but they are used as a baseline for comparison. These algorithms are evaluated against different values of b_h for a horizon of 50000. The maximal regret for the sequence of arms is reported as well.

As the FMA maintains only a relative ranking of arms, it is quick to respond to changes in arm payoff. Since the FMA cannot recall the infinite history of signals, it is actually better at prioritizing recently received signals. In fact, unlike in the previous two TABB environments, we see here that higher m does not necessarily result in lower regret; FMA with lower m, a lower ability to recall previous signals, perform much better when payoff probabilities change with higher frequency. In contrast, traditional techniques like ϵ-greedy and BLA which work to estimate arm payoff probabilities with the full history of signals have no mechanism for recognizing when there is a sudden change of payoff. Thus, it becomes more difficult for ϵ-greedy and BLA to re-calibrate estimates. We do see that when $b_h = 5000$, where frequency of arm change is low, the BLA performs relatively better as it has sufficient time to adjust its estimates. Still, the BLA is not designed for this type of problem.

8 Discussion

From our experimental results across different TABB conditions, we make the following observations about the FMA framework.

8.1 Relative Ranking

The FMA encodes in a finite state space information about the relative payoff probabilities for the arms. Unlike high performing algorithms such as the BLA, which focus on the precision of arm payoff probability estimation, the FMA is concerned with the accuracy of the ranking of arm payoff probability. From the static TABB problem experimental results, we see that such a ranking is sufficient for learning a strategy with low regret for horizons of at least ten thousand stages. We find that this ranking can be built without relying on perfect recall of past signals. When we further consider dynamic TABB settings, we see that a ranking is more suitable for recognizing and responding to changes in arm payoff probability. In contrast, algorithms which need to adjust probability estimates require a longer sequence of stages to recognize shifts. This difference is most notable in the dynamic world with unbounded changes.

8.2 Fast Learning

By sampling measures of regret along the horizon, we can examine how well each algorithm can learn a strategy as time goes by. Considering performance from the start of the decision problem to horizons with 10,000 stages, we see that an FMA with small state space (low m) can consistently identify a strategy with low regret very quickly. For decision problems with a horizon of fewer than 50 stages, it is more difficult to discern statistically significant differences in performance for any TABB setting. However, by horizons of 100, it is evident that the FMA achieves the strategy with the lowest measured regret; the FMA remains the algorithm which can learn the lowest regret strategy well past a horizon of 10,000 stages. These results further serve as a counterpoint to algorithms evaluated for convergence behavior. This learning behavior is especially valuable in applied systems where there is not an expectation of infinite data points and fast learning of a high performance strategy has a real impact.

8.3 Flexible Framework

We have evaluated the FMA, under three different TABB problems, with a single fixed setting to show that the FMA can perform well in each environment without requiring specialized tuning or additional mechanisms. An assumption that most TABB problem algorithms make is that the agent knows what type of behavior is expected of the bandit arms, such as whether arm payoff probabilities are static or dynamic. Further, it is assumed that the arms exhibits exactly one of these behaviors. Thus, different approaches to TABB problems have typically been developed to cater to a specific type of arm payoff probability behavior and are usually evaluated in isolation. For instance, ϵ-decreasing and the BLA are appropriate in static environments while ϵ-greedy and the TBLA perform better in dynamic environments. However, an agent may not necessarily know this information a priori, so this assumption is

not always appropriate. An even more compelling reason to relax this assumption is that many applications of bandit arms exhibit all three types of behavior at various stages in the decision problem; it may not be clear to the typical agent which type of behavior an arm is currently exhibiting so an agent should perform well in all environments.

9 Conclusions and Future Work

The paper's contributions are as follows:

- We show that full arm payoff probability estimation is not necessary for learning a low regret strategy by presenting an algorithm which produces a relative ranking of arms.
- We demonstrate that a perfect recall of the history of signals is not required to perform well in classic TABB problems for static arms or dynamic arms with small incremental changes.
- We identify and motivate a new class of dynamic MAB problems to model domains where arms can have large but infrequent changes in payoff probability.
- Based on these considerations, we design and evaluate the FMA framework which can match the performance of existing techniques that rely on more information.
- We weaken the assumption that a decision making agent must know whether arms are static or dynamic by showing that the FMA, without special tuning or additional modifications, can match the performance of algorithms specializing in each of the three settings.

In future work, we plan to extend the analysis of the FMA to MAB with more arms and other payoff structures. With more arms in the decision problem, especially when considering dynamic environments, the likelihood of changes to the optimal strategy increases with the addition of new arms; we would expect the leverage given by the use of a ranking system over a payoff estimation system to be amplified. When working with payoff structures beyond the Bernoulli distribution, we would be able to explore more powerful judgement functions. However, the extension of the FMA for the broader class of multi-armed bandit problems is non-trivial and decisions must be made regarding the behavior of components such as the action function. More precisely, work needs to be done to determine how the FMA should explore when the current best arm is not exploited. This decision is not obvious and a comparison of the existing strategies ϵ-greedy and the BLA already show two distinct options; ϵ-greedy employs uniform exploration across all arms while the BLA explores each arm with a likelihood corresponding to how high the expected payoff probability is associated with that arm. Further, our work focuses on a bounded decision maker which can perform well in a complex environment with static or dynamic arms, so any extension to the FMA must be made with that direction in mind.

An additional point of consideration is more complex signal processing. When working with TABB, we restrict possible signals to the set $\{0, 1\}$. However, many applications call for the ability to reason about a larger set of signals. In these settings, the conversion of signals into updated beliefs must be re-evaluated. This direction has not been explored as extensively for MAB problems but proper handling of a larger signal set would allow agents to reason about much more complex situations.

Previous work with finite automata [8] also demonstrate the ability to capture human decision making phenomenon such as confirmation/first-impression biases and belief polarization. We suggest that an agent using coarse relative rankings produces a better model of human decision making than an agent who uses point-probability estimation. For example, it seems plausible that the average gambler seems to rely more on an internalized ranking of slot machines rewards than explicit payoff tracking and perfect Bayesian calculations. Investigating to what extent the FMA produces a descriptively adequate model of human decision making is another natural direction for future work.

10 Acknowledgements

I would like to thank my advisor Professor Adam Bjorndahl for his guidance. I am also grateful to the Game Theory reading group for their insights on bounded rationality and bandit problems.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235256, 2002.

[2] D. A. Berry, B. Fristedt. Bandit Problems: Sequential Allocation of Experiments. London: Chapman and Hall, 1985.

[3] O. C. Granmo. A Bayesian Learning Automaton for Solving Two-Armed Bernoulli Bandit Problems. *2008 Seventh International Conference on Machine Learning and Applications*, 2008.

[4] O. C. Granmo. Solving Twoarmed Bernoulli Bandit Problems Using a Bayesian Learning Automaton. *International Journal of Intelligent Computing and Cybernetics Int Jnl of Intel Comp & Cyber*, 3.2:207-234, 2010.

[5] N. Gupta, O. C. Granmo, and A. Agrawala. Thompson Sampling for Dynamic Multi-armed Bandits. *2011 10th International Conference on Machine Learning and Applications and Workshops*, 2011.

[6] J. Y. Halpern and R. Pass. Algorithmic Rationality: Game Theory with Costly Computation. *Journal of Economic Theory*, 156:246-268, 2015.

[7] J. Y. Halpern, R. Pass, and L. Seeman. Decision Theory with Resource-Bounded Agents. *Topics in Cognitive Science Top Cogn Sci*, 6.2:245-257, 2014.

[8] C. Mayo-Wilson, K. Zollman, and D. Danks. Wisdom of Crowds versus Groupthink: Learning in Groups and in Isolation. *International Journal of Game Theory Int J Game Theory* , 42.3:695-723, 2012.

[9] Wilson, Andrea. Bounded Memory and Biases in Information Processing. *Econometrica*, 82.6: 2257-2294, 2014