2008 Seventh International Conference on Machine Learning and Applications

A Bayesian Learning Automaton for Solving Two-Armed Bernoulli Bandit Problems

Ole-Christoffer Granmo
Department of ICT
University of Agder
Grimstad, Norway
ole.granmo@uia.no

Abstract

The two-armed Bernoulli bandit (TABB) problem is a classical optimization problem where an agent sequentially pulls one of two arms attached to a gambling machine, with each pull resulting either in a reward *or a* penalty. *The reward probabilities of each arm are unknown, and thus one must balance between exploiting existing knowledge about the arms, and obtaining new information.*

In the last decades, several computationally efficient algorithms for tackling this problem have emerged, with Learning Automata *(LA) being known for their ϵ-optimality, and* confidence interval based *for logarithmically growing regret. Applications include treatment selection in clinical trials, route selection in adaptive routing, and plan exploration in games like Go. The TABB has also been extensively studied from a Bayesian perspective, however, in general, such analysis leads to computationally inefficient solution policies.*

This paper introduces the Bayesian Learning Automaton (BLA). The BLA is inherently Bayesian in nature, yet relies simply on counting rewards/penalties and on random sampling from a pair of twin beta distributions. Furthermore, we report that BLA is self-correcting *and converges to only pulling the optimal arm with probability* 1. *Extensive experiments demonstrate that, in contrast to most LA, BLA does not rely on external learning speed/accuracy control. It also outperforms recently proposed confidence interval based algorithms. We thus believe that BLA opens up for improved performance in a number of applications, and that it forms the basis for a new avenue of research.*

1 Introduction

The conflict between exploration and exploitation is a well-known problem in reinforcement learning, and other areas of artificial intelligence. The two-armed bandit problem captures the essence of this conflict, and has thus occupied researchers for over forty years [21]. This paper introduces a new family of techniques for solving the classical two-armed Bernoulli bandit problem, and reports theoretical and empirical results that demonstrate its advantages over recent solution approaches such as UCB-Tuned [1], but also established schemes like the L_{R-I} and Pursuit Learning Automata [14].

1.1 The Two-Armed Bernoulli Bandit Problem

The two-armed Bernoulli bandit (TABB) problem is a classical optimization problem that explores the trade off between exploitation and exploration in reinforcement learning. The problem consists of an agent that sequentially pulls one of two arms attached to a gambling machine, with each pull resulting either in a *reward* or a *penalty*[1]. The sequence of rewards/penalties obtained from each arm i forms a Bernoulli process with *unknown* reward probability r_i and penalty probability $1 - r_i$. This leaves the agent with the following dilemma: Should the arm that so far seems to provide the highest chance of reward be pulled once more, or should the inferior arm be pulled in order to learn more about *its* reward probability? Sticking prematurely with the arm that is presently considered to be the best one, may lead to not discovering which arm is truly optimal. On the other hand, lingering with the inferior arm unnecessarily, postpones the harvest that can be obtained from the optimal arm.

In the last decades, several computationally efficient algorithms for tackling this problem have emerged, with *Learning Automata* (LA) being known for their ϵ-optimality, and *confidence interval based* for logarithmi-

[1] A penalty may also be seen as the absence of a reward. However, we choose to use the term *penalty* as is customary in the LA literature.

978-0-7695-3495-4/08 $25.00 © 2008 IEEE
DOI 10.1109/ICMLA.2008.67

cally growing regret.

1.2 Applications

Solution schemes for bandit problems have formed the basis for tackling a number of applications. For instance, UCB-Tuned [1] is used for move exploration in MoGo, a top-level Computer-Go program on 9×9 Go boards [5]. Furthermore, the so-called UCB1 scheme has formed the basis for guiding Monte-Carlo planning, improving planning efficiency significantly in several domains [10].

The applications of LA are many – the following more recent. LA have been used to allocate polling resources optimally in web monitoring, and for allocating limited sampling resources in binomial estimation problems [8]. LA have also been applied for solving NP-complete SAT problems [7]. Furthermore, in [2], LA optimize throughput in MPLS traffic engineering [2]. Note that regret minimizing algorithms also have found applications in network routing [4].

1.3 Contributions and Paper Organization

The contributions of this paper can be summarized as follows. In Section 2 we briefly review a selection of main TABB solution approaches, including LA and confidence interval based schemes. Then, in Section 3 we present the Bayesian Learning Automaton (BLA). In contrast to the latter reviewed schemes, the BLA is inherently Bayesian in nature, yet relies simply on counting and random sampling. Thus, to the best of our knowledge, BLA is the first TABB algorithm that takes advantage of the Bayesian perspective in a computationally efficient manner. Another contribution of this paper is the theoretical results found in Section 4, submitting that BLA is *self-correcting* and converges to only pulling the optimal arm with probability 1. In Section 5 we provide extensive experimental results that demonstrate that, in contrast to the L_{R-I} and Pursuit schemes, BLA does not rely on external learning speed/accuracy control. The BLA also outperforms UCB-Tuned in all but one tested environment. Accordingly, in the above perspective, it is our belief that the BLA represents a new avenue of research, and in Section 6 we list open BLA related research problems, in addition to providing concluding remarks.

2 Related Work

The TABB problem has been studied in a disparate range of research fields. From a machine learning point of view, Sutton et al. [17] put an emphasis on computationally efficient solution techniques that are suitable for reinforcement learning. A selection of main approaches that also have had a significant impact when it comes to applications are *briefly* reviewed here.

2.1 Learning Automata (LA) — The L_{R-I} and Pursuit Schemes

LA have been used to model biological systems [11, 13–16, 18, 19] and have attracted considerable interest in the last decade because they can learn the optimal action when operating in (or interacting with) unknown stochastic environments. Furthermore, they combine rapid and accurate convergence with low computational complexity.

More notable approaches include the family of linear updating schemes, with the Linear Reward-Inaction (L_{R-I}) automaton being designed for stationary environments [14]. In short, L_{R-I} maintains an arm probability selection vector $\bar{p} = [p_1, p_2]$, with $p_2 = 1 - p_1$. Which arm to be pulled is decided randomly by sampling from $\bar{p}$. Initially, $\bar{p}$ is uniform and each arm is selected with equal probability. The following linear updating rules summarize how rewards and penalties affect $\bar{p}$ with p_1' and $1 - p_1'$ being the resulting updated arm selection probabilities:

$$\begin{aligned} p_1' &= p_1 + (1-a) \times (1 - p_1) \\ &\quad \text{if} \quad \text{pulling arm 1 results in a reward} \\ p_1' &= a \times p_1 \\ &\quad \text{if} \quad \text{pulling arm 2 results in a reward} \\ p_1' &= p_1 \\ &\quad \text{if} \quad \text{pulling arm 1 or arm 2 results in a penalty.} \end{aligned}$$

Above the parameter a ($0 \ll a < 1$) governs learning speed. As seen, after an arm i has been pulled, the associated probability p_i is increased using the linear updating rule upon receiving a reward, with p_{3-i} being decreased correspondingly. Note that $\bar{p}$ is left unchanged upon a penalty.

A distinguishing feature of L_{R-I}, and indeed the field of LA as a whole, is its ϵ-optimality [14]: *By a suitable choice of some parameter of the LA, the expected reward probability obtained from each arm pull can be made arbitrarily close to the optimal reward probability, as the number of arm pulls tends to infinity.*

The *pursuit scheme* (P-scheme) makes the updating of $\bar{p}$ more goal-directed in the sense that it maintains maximum likelihood (ML) estimates $(\hat{r_1}, \hat{r_2})$ of the reward probabilities (r_1, r_2) associated with each arm. Instead of using the rewards/penalties received to update $\bar{p}$ directly, the rewards/penalties are instead used to update the ML estimates. The ML estimates, in turn, are used to decide which arm selection probability p_i to increase. In brief, the Pursuit scheme increases the arm selection probability p_i associated with the currently largest ML estimate $\hat{r}_i$, instead of the arm actually producing the reward. Thus, unlike L_{R-I}, when

the inferior arm produces rewards in the Pursuit scheme, these rewards will not influence learning progress (assuming that the ranking of the ML estimates are correct). Accordingly, the pursuit scheme usually outperforms L_{R-I} when it comes to rate of convergence.

Variants of the Pursuit scheme has been proposed [11, 13–16, 18, 19], with slightly improved performance, however, the pursuit scheme can be seen as representative for these additional approaches.

2.2 The ϵ-Greedy and ϵ_n-Greedy Policies

The ϵ-greedy rule is a well-known strategy for the bandit problem [17]. In short, the arm with the presently highest average reward is pulled with probability $1-\epsilon$, while a randomly chosen arm is pulled with probability ϵ. In other words, the balancing of exploration and exploitation is controlled by the ϵ-parameter. Note that the ϵ-greedy strategy persistently explores the available arms with constant effort, which clearly is sub-optimal for the TABB problem (unless the reward probabilities are changing with time).

As a remedy for the above problem, ϵ can be slowly decreased, leading to the ϵ_n-greedy strategy described in [1]. The purpose is to gradually shift focus from exploration to exploitation. The latter work focuses on algorithms that minimizes so-called *regret*. Regret can be defined as *the difference between the sum of rewards expected after n successive arm pulls, and what would have been obtained by only pulling the optimal arm*. In other words, the regret is the expected loss caused by the fact that a strategy does not always select the optimal arm. It turns out that the ϵ_n-greedy strategy provides a *logarithmically* increasing regret asymptotically. Indeed, it has been proved that logarithmically increasing regret is the best possible [1].

2.3 Confidence Interval Based Algorithms

A promising line of thought is the interval estimation methods, where a confidence interval for the reward probability of each arm is estimated, and an "optimistic reward probability estimate" is identified for each arm. The arm with the most optimistic reward probability estimate is then greedily selected [9, 20].

In [1], several confidence interval based algorithms are analysed. These algorithms also provide logarithmically increasing regret, with *UCB-Tuned* – a variant of the well-known UCB1 algorithm — outperforming both *UCB1*, *UCB2*, as well as the ϵ_n-greedy strategy. In brief, in UCB-Tuned, the following optimistic estimates are used for each arm i:

$$\mu_i + \sqrt{\frac{\ln n}{n_i}\min\{1/4, \sigma_i^2 + \sqrt{\frac{2\ln n}{n_i}}\}} \qquad (1)$$

with μ_i and σ_i^2 being the sample mean and variance of the rewards that have been obtained from arm i, n is the number of arms pulled in total, and n_i is the number of times arm i has been pulled. Thus, the quantity added to the sample average of a specific arm i is steadily reduced as the arm is pulled, and uncertainty about the reward probability is reduced. As a result, by always selecting the arm with the highest optimistic reward estimate, UCB-Tuned gradually shifts from exploration to exploitation.

2.4 Bayesian Approaches

The TABB has also been extensively analysed from a Bayesian perspective. For instance, in [3] the TABB is modelled as a partially observable Markov decision processes, and it is shown that the difference in rewards between stopping learning and acquiring full information goes to zero as the number of arm pulls grows large.

Another related example is the probability matching algorithms proposed in [21]. Bayesian analysis is used to obtain a closed form expression for the probability that each arm is optimal given the rewards/penalties observed so far. The policy consists of always pulling the arm which has the greatest probability of being optimal. Unfortunately, computation time is unbounded, rising with the number of arm pulls [21]. Accordingly, the approach is of theoretical interest, but has limited applicability in practice.

3 The Bayesian Learning Automaton (BLA)

Bayesian reasoning is a probabilistic approach to inference which is of significant importance in machine learning because it allows quantitative weighting of evidence supporting alternative hypotheses, with the purpose of allowing optimal decisions to be made. Furthermore, it provides a framework for analyzing learning algorithms [12].

We here present a scheme for solving the TABB problem that inherently builds upon the Bayesian reasoning framework. We coin the scheme *Bayesian Learning Automaton* (BLA) since it can be modelled as a state machine with each state associated with unique arm selection probabilities, in an LA manner.

A unique feature of the BLA is its computational simplicity, achieved by relying *implicitly* on Bayesian reasoning principles. In essence, at the heart of BLA we find the *beta distribution*. Its shape is determined by two positive parameters, usually denoted by α and β, producing the following probability density function:

$$f(x;\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du}\,, \quad x \in [0,1] \qquad (2)$$

and the corresponding cumulative distribution function:

$$F(x;\alpha,\beta) = \frac{\int_0^x t^{a-1}\,(1-t)^{b-1}\,dt}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du}\,, \quad x \in [0,1]. \quad (3)$$

Essentially, BLA uses the beta distribution for two purposes. First of all, the beta distribution is used to provide a *Bayesian estimate* of the reward probabilities associated with each of the available bandit arms. Secondly, a novel feature of BLA is that it uses the beta distribution as the basis for a *randomized arm selection mechanism*. The following algorithm contains the essence of the BLA approach.

Bayesian Learning Automaton Algorithm.

- **Initialization:** $\alpha_1 := \beta_1 := \alpha_2 := \beta_2 := 1$.
- **Loop:**
  1. Draw a value x_1 randomly from beta distribution $f(x_1, \alpha_1, \beta_1)$ with parameters α_1, β_1.
  2. Draw a value x_2 randomly from beta distribution $f(x_2, \alpha_2, \beta_2)$ with parameters α_2, β_2.
  3. **If** $x_1 > x_2$ **then** pull *Arm* 1 **else** pull *Arm* 2. Denote pulled arm: *Arm* i.
  4. Receive either *Reward* or *Penalty* as a result of pulling *Arm* i.
  5. Increase α_i with 1 upon *Reward* ($\alpha_i := \alpha_i + 1$), and increase β_i with 1 upon *Penalty* ($\beta_i := \beta_i + 1$).
  6. **Goto** 1.

As seen from the above BLA algorithm, the parameters $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ form a infinite discrete four dimensional state space Φ, within which the BLA navigates by iteratively increasing either α_1, β_1, α_2, or β_2 with 1.

Also note that for any parameter configuration $(\alpha_1, \beta_1, \alpha_2, \beta_2) \in \Phi$ we have that the probability of selecting *Arm* 1, $P(Arm\ 1)$, is equal to $P^{\Phi}(X_1 > X_2)$ — the probability that a randomly drawn value $x_1 \in X_i$ is greater than a randomly drawn value $x_2 \in X_2$, when the associated stochastic variables X_1 and X_2 are beta distributed, with parameters α_1, β_1 and α_2, β_2, respectively.

The probability $P^{\Phi}(X_1 > X_2)$ can also be interpreted as the probability that *Arm* 1 is the optimal one, given the observations α_1, β_1 and α_2, β_2. This means that BLA will gradually shift its arm selection focus towards the arm which most likely is the optimal one, as observations are received.

Finally, observe that BLA does not rely on any external parameters that must be configured to optimize performance for specific problem instances. This is in contrast to the traditional Learning Automata family of algorithms, where a "learning speed/accuracy" parameter is inherent in ϵ-optimal schemes.

4 Theoretical Results

In this section we present our main results for the BLA. Note that these results are distinct from previous results concerning the beta distribution used in Bayesian parameter estimation. Instead, we establish a collection of new results for the case when two beta distributions are applied in conjunction, as a randomized arm selection mechanism.

A well-known and pertinent feature of significant LA is their *absolute expediency* — the expected probability of selecting the best arm is increasing monotonically with each additional arm pull. Furthermore, such LA converges to only selecting the optimal arm with probability arbitrarily close to unity, by a suitable choice of some parameter, which is referred to as ϵ-optimality [14]. With this historical perspective in mind, we will in the following report similar, although stronger, results for the BLA.

4.1 Arm Selection Probability

First of all, however, note that for many LA schemes, such as the L_{R-I} scheme, the arm selection probabilities of any given time step can be expressed as a linear function of the arm selection probabilities of the previous time step, thus facilitating Markov chain based convergence analysis [14]. In contrast, the arm selection probabilities of BLA is related to the state $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ in terms of a complex nonlinear function, which we state here without further ado. The probability p_1^{Φ} of selecting arm 1, given the state $\Phi = (\alpha_1, \beta_1, \alpha_2, \beta_2)$, is:

$$p_1^{\Phi} = \frac{(\alpha_1+\beta_1-1)!(\alpha_2+\beta_2-1)!}{(\alpha_1-1)!(\beta_1-1)!(\alpha_1+\beta_1+\alpha_2+\beta_2-2)!} \times \sum_{j=\alpha_2}^{\alpha_2+\beta_2-1} \frac{(j+\alpha_1-1)!(\beta_1+\alpha_2+\beta_2-j-2)!}{j!(\alpha_2+\beta_2-1-j)!}$$

Notice that the computation time of the above expression is unbounded, rising with the number of arm pulls, making it unsuitable for practical application. Also note that unlike the probability matching algorithms found in [21], the above expression is not computed explicitly in the BLA algorithm.

Combined with an infinite state space Φ of high dimensionality, BLA does not lend itself towards traditional Markov chain based analysis, as presented in the LA literature. This complication have led us to investigate an alternative strategy for analysing the convergence properties of BLA, which allows us to avoid dealing with the complexity implied above explicitly.

4.2 TABB and Bayesian Estimation

Recall that the BLA uses the beta distribution to provide a Bayesian estimate of the reward probabilities associated

with each of the available bandit arms. It is well known that as the number of arm pulls, N_i, tends to infinity for a given arm i, the expected value, $E[X_i] = \frac{\alpha_i}{\alpha_i+\beta_i}$, of the associated beta distribution, converges to the true reward probability r_i. Furthermore, its variance tends to zero. In other words, if both arms are pulled extensively, the optimal arm reveals itself eventually.

However, for the TABB problem the key is to avoid pulling the inferior arm unnecessarily. With this latter aim in mind, it is easy to see that the following scenario may potentially foil BLA in its attempt at identifying the optimal arm (which for the sake of notational simplicity is assumed to be arm 1 from this point): *If the reward probability estimate $\frac{\alpha_1}{\alpha_1+\beta_1}$, associated with arm 1 (the optimal arm), fluctuates to a value that is less than the estimate $\frac{\alpha_2}{\alpha_2+\beta_2}$ associated with arm 2, one could risk that the probability of selecting arm 1 would tend to 0 before $\frac{\alpha_1}{\alpha_1+\beta_1}$ is given the opportunity to converge to the true value r_1.*

Thus, in the latter scenario, with $\frac{\alpha_2}{\alpha_2+\beta_2}$ tending to r_2 and $\frac{\alpha_1}{\alpha_1+\beta_1}$ being less than r_2, it is apparent that arm 2 will be selected more often than arm 1. In the worst potential case, as we will discuss presently, the system could converge to only selecting arm 2, thus failing to identify the optimal arm.

4.3 Result I: BLA Is Self-Correcting

First of all, it turns out that the BLA is *self-correcting* in the sense that the more the estimate $\frac{\alpha_i}{\alpha_i+\beta_i}$ falls below the true value r_i, the more the probability p_1^Φ of selecting arm i increases (of course, as long as p_1^Φ has not already reached zero). Consequently, the probability of receiving more observations for estimating r_i also increases. This pertinent result can be expressed formally as follows:

Theorem 1. *Let $p_1^{\Phi^*}$ denote the probability of selecting arm 1 after an additional pull of that arm, with $\Phi = (\alpha_1, \beta_1, \alpha_2, \beta_2)$ being the state before the arm pull, and Φ^* being the state afterwards, i.e., either $(\alpha_1+1, \beta_1, \alpha_2, \beta_2)$ or $(\alpha_1, \beta_1+1, \alpha_2, \beta_2)$. Then, if the former reward probability estimate $\frac{\alpha_1}{\alpha_1+\beta_1}$ is less than r_1 with $\frac{\alpha_2}{\alpha_2+\beta_2}$ approaching r_2 ($E\left[(X_2 - r_2)^2\right] \to 0$), the following is true:*

$$E[p_1^{\Phi^*}|p_1^\Phi] > p_1^\Phi. \tag{4}$$

That is, the expected value $E[p_1^{\Phi^*}|p_i^\Phi]$ of the arm selection probability after an additional arm pull is greater than the present arm selection probability p_1^Φ (assuming that $p_1^\Phi > 0$). Thus, if the estimate of r_1 is too low, we may expect the BLA to increase the probability of selecting arm 1 after one more arm pull, in this sense being *self-correcting*.

Proof. The proof is quite involved and is found in [6]. It is omitted here in the interest of brevity. □

4.4 Result II: BLA Converges to Only Pulling the Optimal Arm

At first sight, it is conceivable that certain sequences of rewards/penalties can make the BLA converge to only selecting the inferior arm (i.e., arm 2), resulting in the variance $E\left[(X_2 - r_2)^2\right]$ tending to 0. After all, for the L_{R-I} and Pursuit LA schemes, such reward/penalty sequences occur with non-zero probability. However, the following results imply that such sequences cannot occur with BLA.

Lemma 1. *The probability of selecting arm 1 after N_1 selections, given $E\left[(X_2 - r_2)^2\right] \to 0$ and α_1, is:*

$$P(X_1 > X_2|N_1, \alpha_1, r_2) = \tag{5}$$

$$\sum_{j=0}^{\alpha_1-1} \binom{N_1+1}{j} r_2^j (1-r_2)^{N_1+1-j}. \tag{6}$$

In other words, the arm selection probability p_1^Φ can be reformulated in terms of a *cumulative binomial distribution*, $P(Y \leq \alpha_1 - 1)$, when $E\left[(X_2 - r_2)^2\right] \to 0$, with Y being a generic binomially distributed random variable. *This relates the arm selection probability p_1^Φ of arm 1 directly to the chances of obtaining no more than $\alpha_1 - 1$ rewards from arm 2 in $N_1 + 1$ arm pulls!*

Proof. The proof can be found in [6]. □

Theorem 2. *The BLA converges to persistently selecting the optimal arm with probability 1:*

$$\lim_{N\to\infty} p_1^\Phi = 1 \quad \textit{with} \quad r_1 > r_2. \tag{7}$$

Proof. This proof is also quite involved and can be found in [6]. It is omitted here in the interest of brevity. □

In contrast, note that the L_{R-I} LA converges to only selecting the optimal arm with probability *arbitrarily* close to 1, thus never reaching 1.

5 Experiments

In this section we evaluate the BLA by comparing it with UCB-Tuned, the best performing algorithm from [1], as well as the L_{R-I} and Pursuit schemes from the LA field.

For the sake of fairness, we base our comparison on the experimental setup for the TABB found in [1]. Although several experiments were conducted using various reward distributions, we report, for the sake of brevity, only results for the following four reward/penalty distributions. The first distribution, **Distribution 1** ($r_1 = 0.9$, $r_2 = 0.6$), forms the most simple environment, with low variance and a large difference between the two arms. By gradually reducing the arm difference, we increase the difficulty of the TABB

problem, and it is of interest to observe how the different schemes react as the challenge increases. **Distribution 2** ($r_1 = 0.9$**,** $r_2 = 0.8$) fulfills this purpose in [1], however, in order to stress the schemes further, we also apply **Distribution 3** ($r_1 = 0.9$**,** $r_2 = 0.89$). The challenge of **Distribution 4** ($r_1 = 0.55$**,** $r_2 = 0.45$) is its high variance combined with the small difference between the two arms.

For these distributions, an ensemble of 1000 independent replications with different random number streams was performed to minimize the variance of the reported results. In each replication, BLA, UCB-Tuned, L_{R-I}, and the Pursuit scheme conducted $100\ 000$ arm pulls.

Figure 1 and 2 contains the comparison on distribution 1. The former plot shows the probability of choosing the optimal arm as each scheme explores the two arms, with n being the number of arm pulls performed. The latter plot shows the accumulation of regret with number of arm pulls.

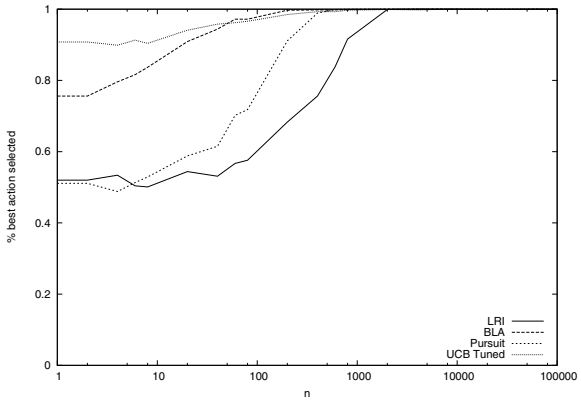

Figure 1. Probability of selecting best arm for distribution 1.

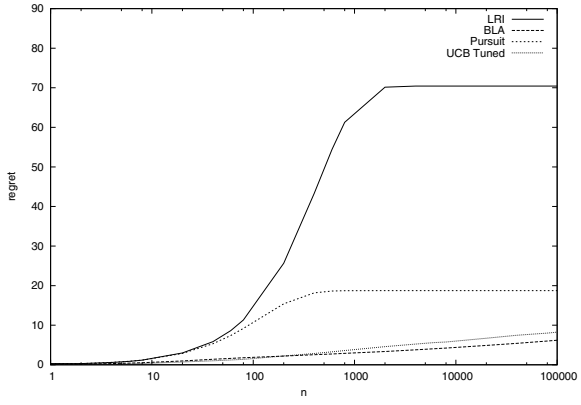

Figure 2. Regret for distribution 1

Because of the logarithmically scaled x-axis, it is clear from the plots that both BLA and UCB-Tuned attain a logarithmically growing regret. Moreover, the performance of BLA is significantly better than that of UCB-Tuned. Surprisingly, both of the LA schemes converge to constant regret. This can be explained by their ϵ-optimality and the relatively small learning speed parameter used ($a = 0.01$). In brief, the LA converged to only selecting the optimal arm in all of the 1000 replications.

Figure 3 and 4 contains the comparison on distribution 2. As shown, the LA still achieve constant regret, and again, the BLA outperforms UCB-Tuned.

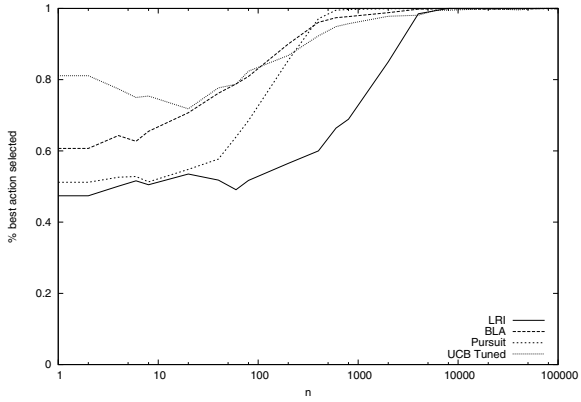

Figure 3. Probability of selecting best arm for distribution 2.

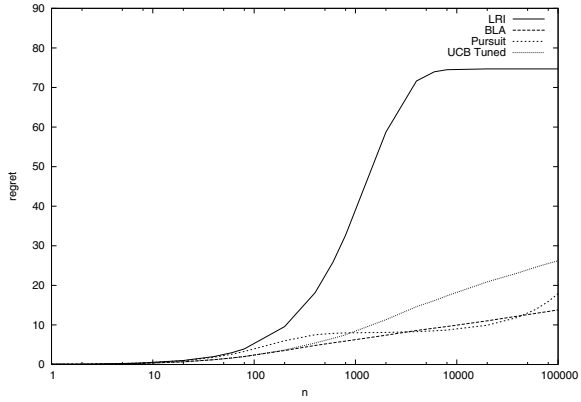

Figure 4. Regret for distribution 2.

For Distribution 3, however, it turns out that the set learning accuracy of the LA is too low to always converge to only selecting the optimal arm. In some of the replications, the LA also converges to selecting the inferior arm only, and as seen in Figure 5 and 6, this leads to linearly growing regret.

As also seen, the BLA continues to provide significantly better performance than UCB-Tuned.

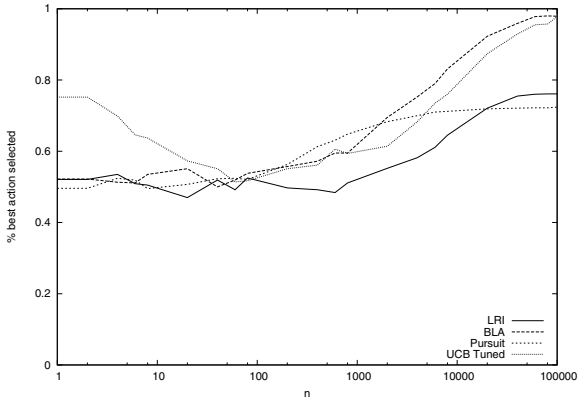

Figure 5. Probability of selecting best arm for distribution 3.

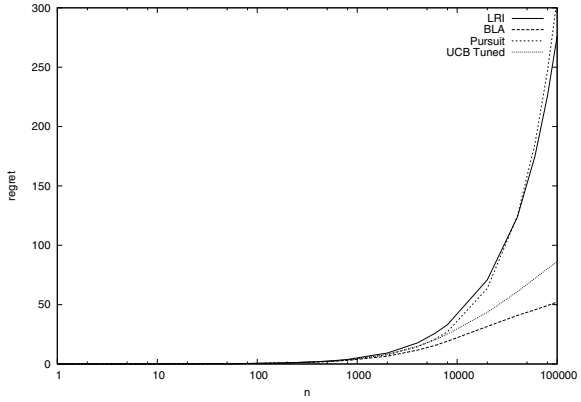

Figure 6. Regret for distribution 3.

Finally, we observe that the high variance of Distribution 4 reduces the performance gap between BLA and UCB-Tuned, as seen in Figures 7 and 8, leaving UCB-Tuned with slightly lower regret compared to BLA.

Note that the LA can achieve constant regret in all of the latter experiments too, by increasing learning accuracy. However, this significantly reduces learning speed, which for the present setting already is worse than that of BLA and UCB-Tuned.

6 Conclusion and Further Work

In this paper we presented the Bayesian Learning Automaton (BLA) for tackling the classical two-armed

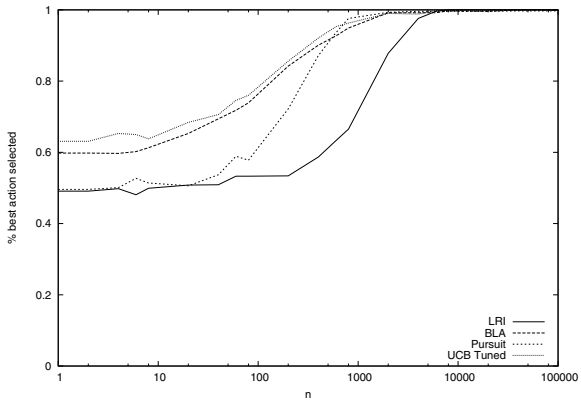

Figure 7. Probability of selecting best arm for distribution 4.

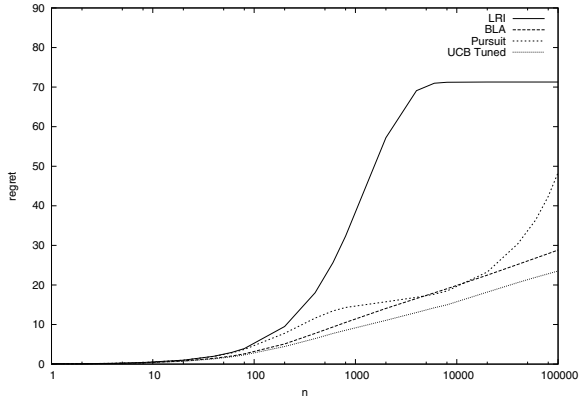

Figure 8. Regret for distribution 4.

Bernoulli bandit (TABB). In contrast to previous LA and regret minimizing approaches, BLA is inherently Bayesian in nature. Still, it relies simply on counting of rewards/penalties and random sampling from a pair of twin beta distributions. Furthermore, BLA is *self-correcting* and converges to only pulling the optimal arm with probability 1. Thus, to the best of our knowledge, BLA is the first TABB algorithm that takes advantage of Bayesian estimation in a computationally efficient manner.

Extensive experimental results demonstrated that, unlike L_{R-I} and Pursuit schemes, BLA does not rely on external learning speed/accuracy control. The BLA also outperformed UCB-Tuned under all but one tested reward distribution, achieving logarithmically growing regret.

Accordingly, in the above perspective, it is our belief that the BLA represents a new avenue of research, opening up for an array of research problems. First of all, the BLA can quite straightforwardly be extended to handle the multi-

armed bandit problem. Indeed, preliminary experimental results indicate that the observed qualities carry over from the TABB problem case. Secondly, incorporating other reward distributions, such as Gaussian and multinomial distributions, into our scheme is of interest. Thirdly, we believe that our scheme can be modified to tackle bandit problems that are non-stationary, i.e., where the reward probabilities are changing with time. It is furthermore of interest to determine whether BLA allows logarithmically growing regret in general — it is not known whether UCB-Tuned possesses this property [1]. Finally, systems of BLA can be studied from a game theory point of view, where multiple BLAs interact forming the basis for multi-agent systems.

Acknowledgement

I want to thank Chancellor's Professor B. John Oommen from Carleton University, Canada, for introducing me to the field of Learning Automata and for his valuable feedback and insight, both as a friend and as a colleague.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235–256, 2002.

[2] S. M. B. J. Oommen and O.-C. Granmo. Routing Bandwidth Guaranteed Paths in MPLS Traffic Engineering: A Multiple Race Track Learning Approach. *IEEE Transactions on Computers*, 56(7):959–976, 2007.

[3] S. Bhulai and G. Koole. On the Value of Learning for Bernoulli Bandits with Unknown Parameters. *IEEE Transactions on Automatic Control*, 45(11):2135–2140, 2000.

[4] A. Blum, E. Even-Dar, and K. Ligett. Routing Without Regret: On Convergence to Nash Equilibria of Regret-Minimizing Algorithms in Routing Games. In *Proceedings of the Twenty-Fifth Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2006)*, pages 45–52. ACM, 2006.

[5] S. Gelly and Y. Wang. Exploration exploitation in Go: UCT for Monte-Carlo Go. In *Proceedings of NIPS-2006*. NIPS, 2006.

[6] O.-C. Granmo. Solving Two-Armed Bandit Problems Using a Bayesian Learning Automaton, 2008. Unabridged version of this paper. Submitted for publication.

[7] O.-C. Granmo and N. Bouhmala. Solving the Satisfiability Problem Using Finite Learning Automata. *International Journal of Computer Science and Applications,*, 4(3):15–29, 2007.

[8] O.-C. Granmo, B. J. Oommen, S. A. Myrer, and M. G. Olsen. Learning Automata-based Solutions to the Nonlinear Fractional Knapsack Problem with Applications to Optimal Resource Allocation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(1):166–175, 2007.

[9] L. P. Kaelbling. *Learning in Embedded Systems*. PhD thesis, Stanford University, 1993.

[10] L. Kocsis and C. Szepesvari. Bandit Based Monte-Carlo Planning. In *Proceedings of the 17th European Conference on Machine Learning (ECML 2006)*, pages 282–293. Springer, 2006.

[11] S. Lakshmivarahan. *Learning Algorithms Theory and Applications*. Springer-Verlag, 1981.

[12] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[13] K. Najim and A. S. Poznyak. *Learning Automata: Theory and Applications*. Pergamon Press, Oxford, 1994.

[14] K. S. Narendra and M. A. L. Thathachar. *Learning Automata: An Introduction*. Prentice Hall, 1989.

[15] M. S. Obaidat, G. I. Papadimitriou, and A. S. Pomportsis. Learning automata: Theory, paradigms and applications. *IEEE Transactions on Systems Man and Cybernetics*, SMC-32:706–709, 2002.

[16] A. S. Poznyak and K. Najim. *Learning Automata and Stochastic Optimization*. Springer-Verlag, Berlin, 1997.

[17] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[18] M. A. L. Thathachar and P. S. Sastry. *Networks of Learning Automata: Techniques for Online Stochastic Optimization*. Kluwer Academic Publishers, 2004.

[19] M. L. Tsetlin. *Automaton Theory and Modeling of Biological Systems*. Academic Press, 1973.

[20] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. In *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, pages 437–448. Springer, 2005.

[21] J. Wyatt. *Exploration and Inference in Learning from Reinforcement*. PhD thesis, University of Edinburgh, 1997.