Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton

Ole-Christoffer Granmo
Department of ICT, University of Agder, Aust-Agder, Norway

Received 18 November 2008
Revised 28 April 2009
Accepted 7 May 2009

Abstract

Purpose – 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. The purpose of this paper is to report research into a completely new family of solution schemes for the TABB problem: the Bayesian learning automaton (BLA) family.

Design/methodology/approach – Although computationally intractable in many cases, Bayesian methods provide a standard for optimal decision making. BLA avoids the problem of computational intractability by not explicitly performing the Bayesian computations. Rather, it is based upon merely counting rewards/penalties, combined with random sampling from a pair of twin Beta distributions. This is intuitively appealing since the Bayesian conjugate prior for a binomial parameter is the Beta distribution.

Findings – BLA is to be proven instantaneously self-correcting, and it converges to only pulling the optimal arm with probability as close to unity as desired. Extensive experiments demonstrate that the BLA does not rely on external learning speed/accuracy control. It also outperforms established non-Bayesian top performers for the TABB problem. Finally, the BLA provides superior performance in a distributed application, namely, the Goore game (GG).

Originality/value – The value of this paper is threefold. First of all, the reported BLA takes advantage of the Bayesian perspective for tackling TABBs, yet avoids the computational complexity inherent in Bayesian approaches. Second, the improved performance offered by the BLA opens up for increased accuracy in a number of TABB-related applications, such as the GG. Third, the reported results form the basis for a new avenue of research – even for cases when the reward/penalty distribution is not Bernoulli distributed. Indeed, the paper advocates the use of a Bayesian methodology, used in conjunction with the corresponding appropriate conjugate prior.

Keywords Learning processes, Programming and algorithm theory, Automata theory, Stochastic processes

Paper type Research paper

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

The author thanks Chancellor's Professor B. John Oommen from Carleton University, Canada, for introducing him to the field of LA, and for his valuable feedback and insight, both as a friend and as a colleague. The author is also grateful to a Professor M.A.L. Thathachar, from the Indian Institute of Science in Bangalore, India, for his deep and insightful comments on an earlier version of this paper. Preliminary versions of some of the results of this paper were presented at SGAI'08 and ICMLA'08 in December 2008. This unabridged journal paper is expanded with proofs for the theoretical results, additional experiments, as well as a number of salient extensions, and enhancements.

International Journal of Intelligent Computing and Cybernetics
Vol. 3 No. 2, 2010
pp. 207-234
© Emerald Group Publishing Limited
1756-378X
DOI 10.1108/17563781011049179

problem captures the essence of this conflict, and has thus occupied researchers for over 50 years (Wyatt, 1997). This paper introduces a new family of techniques for solving the classical two-armed Bernoulli bandit (TABB) problem, and reports theoretical and empirical results that demonstrate its advantages over established top performers from two distinct research areas, namely, learning automata (LA) and Bandit playing algorithms (see Thathachar and Sastry (2004), Narendra and Thathachar (1989) and Vermorel and Mohri (2005), respectively, for an overview of current algorithms).

1.1 The TABB problem

The 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.

With the above in mind, we intend to evaluate an agent's arm selection strategy in terms of the so-called regret, and in terms of the probability of selecting the optimal arm[2]. The regret measure is non-trivial, and so we provide further clarification here.

Definition 1.1. (Regret). In brief, the regret can be seen 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. To exemplify, assume that a reward yields a value (utility) of 1, and that a penalty is associated with the value 0. This implies that the expected utility of pulling arm i is r_i. Thus, if the optimal arm is Arm 1, the regret after N plays would become:

$$r_1 N - \sum_{n=1}^{N} \hat{r}_n, \qquad (1)$$

with $\hat{r}_n$ being the expected reward at arm pull n, given the agent's arm selection strategy.

In other words, as will be clear in the following, we consider the case where rewards are undiscounted, as discussed in Auer *et al.* (2002).

Consider the scenario in which an inferior arm appears to be superior by virtue of estimates which are inaccurate. This can occur, for example, when the superior arm is sampled less and when, in any specific sequence of arm pulls, the inferior arm yields a higher percentage of rewards. Although, asymptotically this is an event which occurs with probability zero, we would prefer that when this does occur in a finite number of arm pulls, the scheme is capable of emerging from such an erroneous convergence towards the correct one. Since this is a salient characteristic of the Bayesian learning automaton (BLA), to present our paper in the right perspective, we now introduce the concept of a strategy being instantaneously self-correcting. Informally speaking, an algorithm possessing this property is able to recover after incorrect reward probability estimates.

Definition 1.2. (Instantaneously self-correcting). A bandit playing algorithm is said to be instantaneously self-correcting if whenever it incorrectly infers the reward estimate for an arm to be too low/high, the probability of choosing the alternate arm at the next time instant is higher/lower than at the current time instant, independent of the estimate of the reward probability of the latter arm.

We would like to emphasize that this is a strong property. An algorithm that possesses this property is not merely asymptotically superior. Rather, at every time instant the algorithm works towards the superior solution even if it currently has an inaccurate idea of the correct solution due to the randomness of the estimation process.

In the last decades, several computationally efficient algorithms for tackling the TABB problem have emerged. From a theoretical perspective, LA are known for their ϵ-optimality. However, from the field of bandit playing algorithms, confidence interval-based algorithms are known for logarithmically growing regret.

1.2 Applications

Solution schemes for bandit problems have formed the basis for dealing with a number of applications. For instance, UCB-tuned (Auer *et al.*, 2002) is used for move exploration in MoGo, a top-level computer-go program on 9×9 go boards (Gelly and Wang, 2006). Further, the so-called UCB1 scheme has formed the basis for guiding Monte-Carlo planning, thus improving planning efficiency significantly in several domains (Kocsis and Szepesvari, 2006).

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 (Granmo *et al.*, 2007). LA have also been applied for solving NP-Hard satisfiability problem problems (Granmo and Bouhmala, 2007). Also, in Misra *et al.* (2007), LA have been used to optimize throughput in Multiprotocol Label Switching traffic engineering. Note that regret minimizing algorithms also have found applications in network routing (Blum *et al.*, 2006).

1.3 Contributions of this paper

Although computationally intractable in many cases, Bayesian methods provide a standard for optimal decision making. Furthermore, they can be used to analyze heuristic methods that do not explicitly manipulate probabilities (Mitchell, 1997). In spite of the fact that this was derived as early as 1933 in the so-called Thompson (1993) sampling principle (as we shall explain later), its application to the TABB is relatively unreported.

The main contribution of this paper can thus be stated as being a novel algorithm for solving the TABB problem. The algorithm is Bayesian in nature, thus providing the benefits of Bayesian methods. Yet, it avoids the problem of computational intractability by only relying on counting rewards/penalties, combined with random sampling from a pair of twin Beta distributions.

We shall refer to this algorithm as the BLA because it can be modelled as a stochastic automaton, with each automaton state being associated with unique arm selection probabilities, and with state transitions being determined by the arm selected and the resulting feedback from the TABB. Still, as we will presently clarify, the BLA possesses an infinite as well as discrete state space, and thus, strictly speaking, it is quite distinct from both the variable structure and the fixed structure LA families (Thathachar and Sastry, 2004).

In all brevity, the contributions of the paper can be listed as follows:

(1) The BLA is a TABB algorithm that takes advantage of the Bayesian perspective in a computationally efficient manner by only relying on random sampling and mere counting of rewards/penalties, akin to the Thompson (1933) sampling principle, rather than intricate a posteriori computations.

(2) We report the first analytical results for the BLA, by proving that it is instantaneously self-correcting, and that it converges to only pulling the optimal arm with a probability as close to unity as desired.

(3) We also provide a solid empirical evaluation demonstrating that, in contrast to most LA, the properties of the BLA do not depend on its external learning speed/accuracy control. It also outperforms recently-proposed confidence interval-based algorithms in a broad range of TABB problems.

(4) Furthermore, we demonstrate that the BLA provides superior performance in a real-world application, namely, the Goore game (GG), applicable for quality of service (QoS) control in sensor networks.

(5) The final significant aspect is that we can now devise solutions to the two-armed bandit problem even for cases when the reward/penalty distribution is not Bernoulli distributed. Indeed, we advocate the use of a Bayesian methodology, used in conjunction with the corresponding appropriate conjugate prior (Duda *et al.*, 2000), as also suggested by the Thompson (1933) sampling principle.

We thus believe that the BLA can replace state-of-the-art TABB algorithms in a number of applications, leading to improved performance, and that it forms the basis for a new avenue of research.

1.4 Paper organization

The paper is organized 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 BLA. In contrast to the latter reviewed schemes, the BLA is inherently Bayesian in nature, yet relies simply on counting and random sampling. Section 4 provides two formal results, demonstrating that BLA is instantaneously self-correcting and that it converges to only pulling the optimal arm with probability as close to unity as desired. In Section 5, we provide extensive experimental results that demonstrate that, in contrast to the linear reward-inaction ($L_{R\text{-}I}$) and pursuit schemes (P-schemes), the performance of the BLA does not rely on its external learning speed/accuracy control. The BLA also outperforms UCB-tuned – which is one of the acclaimed and established top performers from the field of bandit playing algorithms[3]. Accordingly, based on the above perspective, it is our belief that the BLA represents a new avenue of research, and so, 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 wide range of research fields. From a machine learning point of view, Sutton *et al.* placed emphasis on computationally efficient solution techniques that are suitable for reinforcement learning. While there are algorithms for computing the Bayes optimal way to balance exploration and exploitation, these are computationally intractable for the general case (Sutton and Barto, 1998), mainly because of the huge state space associated with typical bandit problems.

From a broader point of view, one can distinguish two distinct fields that address bandit-like problems, namely, the field of LA and the field of bandit playing algorithms. A myriad of approaches have been proposed within these two fields, and we refer the reader to Narendra and Thathachar (1989), Thathachar and Sastry (2004) and Vermorel and Mohri (2005) for an overview and comparison of these families of schemes. Although these fields are quite related, research spanning them both are surprisingly sparse. In this paper, however, we will include established top performers from both of them. These are reviewed here in somewhat detail in order to cast light on the distinguishing properties of BLA, both from an LA perspective and from the perspective of bandit playing algorithms.

2.1 LA – the $\mathrm{L}_{\text{R-I}}$ *and P-schemes*

LA have been used to model biological systems (Tsetlin, 1973; Lakshmivarahan, 1981; Najim and Poznyak, 1994; Narendra and Thathachar, 1989; Obaidat *et al.*, 2002; Poznyak and Najim, 1997; Thathachar and Sastry, 2004) 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. For the sake of conceptual simplicity, note that, in this subsection, we assume that we are dealing with a bandit associated with two arms.

More notable approaches include the family of linear updating schemes, with the $L_{R\text{-}I}$ automaton being designed for stationary environments (Narendra and Thathachar, 1989). In short, the $L_{R\text{-}I}$ maintains an arm selection probability vector $\bar{p} = [p_1, p_2]$, with $\mathrm{p}_2 = 1 - p_1$. The question of which arm is to be pulled is decided randomly by sampling from $\bar{p}$. Initially, $\bar{p}$ is uniform. 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:

$$p'_1 = p_1 + (1-a) \times (1-p_1) \quad \text{if pulling Arm 1 results in a reward}$$

$$p'_1 = a \times p_1 \quad \text{if pulling Arm 2 results in a reward}$$

$$p'_1 = p_1 \quad \text{if pulling Arm 1 or Arm 2 results in a penalty.}$$

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

A distinguishing feature of the $L_{R\text{-}I}$ scheme, and indeed the of best LA within the field, is its ϵ-optimality (Narendra and Thathachar, 1989): 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.

A 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 that are received to update $\bar{p}$ directly, they are rather used to update the ML estimates. The ML estimates, in turn, are used to decide which arm selection probability p_i is to be increased. In brief, a P-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 the $L_{R\text{-}I}$, in which the reward from an inferior arm can cause unsuitable probability updates, in the P-scheme, these rewards will not influence the learning progress in the short-term, except by modifying the estimate of the reward vector. This, of course, assumes that the ranking of the ML estimates are correct, which is what it will be if each action is chosen a "sufficiently large number of times." Accordingly, a P-scheme consistently outperforms the $L_{R\text{-}I}$ in terms of its rate of convergence.

Discretized and continuous variants of the P-scheme has been proposed (Agache and Oommen, 2002; Lanctôt, 1989; Lanctôt and Oommen, 1992; Oommen and Agache, 2001; Oommen and Lanctôt, 1990; Sastry, 1985), with slightly superior performances. But, in general, any P-scheme can be seen to be representative of this entire family.

2.2 The ϵ-greedy and ϵ_{n}-greedy policies

The ϵ-greedy rule is a well-known strategy for the bandit problem (Sutton and Barto, 1998). 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 parameter ϵ. 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 Auer *et al.* (2002). The purpose is to gradually shift focus from exploration to exploitation. The latter work focuses on algorithms that minimizes the so-called regret. As defined earlier, the 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 asymptotically provides a logarithmically increasing regret. Indeed, it has been proved that a scheme yielding a logarithmically increasing regret is the best possible strategy (Auer *et al.*, 2002).

2.3 Confidence interval-based algorithms

A promising avenue for solving the TABB involve the methods which consider the estimation of confidence intervals, wherein the scheme estimates a confidence interval for the reward probability of each arm, and an "optimistic reward probability estimate" is identified for each arm. The arm with the most optimistic reward probability estimate is then greedily selected (Vermorel and Mohri, 2005; Kaelbling, 1993).

In Auer *et al.* (2002), the authors analyzed several confidence interval-based algorithms. 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:

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

where μ_i and σ_i^2 are the sample mean and variance of the rewards that have been obtained from arm i, n is the total number of arm pulls, 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 the corresponding 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 application of a Bayesian philosophy for searching in such probabilistic spaces has a long and well-documented path through the mathematical literature, probably pioneered by Thompson (1933) even as early as 1933[4]. Thompson used this principle in probabilistic matching algorithms by introducing the so-called Thompson sampling principle, in which an action is chosen a fraction of the time that corresponds to the probability that the action is optimal[5].

The TABB has also been extensively analyzed from a Bayesian perspective. For instance, in Bhulai and Koole (2000) the TABB is modelled as a partially observable Markov decision process. The authors of the paper demonstrated that the difference in rewards between the actions of stopping learning and acquiring full information goes to zero as the number of arm pulls increases indefinitely.

The authors of Wyatt (1997) have also proposed the use of a Bayesian philosophy in their probability matching algorithms. By using conjugate priors, they have resorted to a Bayesian analysis to obtain a closed form expression for the probability that each arm is optimal given the prior observed rewards/penalties. Informally, the method proposes a policy which consists of calculating the probability of each arm being optimal, and then randomly selecting the arm to be pulled next using these probabilities. Unfortunately, for the case of two arms in which the rewards Bernoulli-distributed, the computation time becomes unbounded, and it increases with the number of arm pulls. Furthermore, it turns out that for the multi-armed case, the resulting integrations have no analytical solution.

More recently, however, Wang *et al.* (2005) combined so-called sparse sampling with Bayesian exploration, where the entire process was specified within a finite time horizon. By achieving this, they were able to search the space based on a sparse lookahead tree which was produced based on the Thompson sampling principle. By doing this, they were able to take advantage of the Bayesian posterior of the belief state, and avoid a complete enumeration of the possible action branches. We conclude this brief subsection by citing the work of Dimitrakakis (2006). In Dimitrakakis (2006), the author derived optimal decision thresholds for the multi-armed bandit problem, for both the infinite horizon discounted reward case and for the finite horizon undiscounted case. Based on these thresholds, he then went on to propose practical algorithms, which can perhaps be perceived to be enhancements of the Thompson sampling principle or of the family of probability matching algorithms because they succeed in linking uncertainty and the need for exploration explicitly to the reward horizon.

3. The Bayesian learning automaton

Bayesian reasoning is a probabilistic approach to inference which is of significant importance in machine learning because it allows the 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 (Mitchell, 1997).

We present here a scheme for solving the TABB problem that inherently builds upon the Bayesian reasoning framework. We refer to the scheme as the BLA since it can be modelled as a state machine with each state associated with its unique arm selection probability, as in the case of LA.

A unique feature of the BLA is its computational simplicity, achieved by relying implicitly on Bayesian reasoning principles. In essence, at the heart of the BLA is the Beta distribution, whose shape is determined by two positive parameters, usually denoted by α and β, which leads to the following probability density function (PDF):

$$f(x;\alpha,\beta)=\frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du},\quad x\in[0,1] \tag{3}$$

whose corresponding cumulative distribution function is:

$$F(x;\alpha,\beta)=\frac{\int_0^x t^{\alpha-1}(1-t)^{\beta-1}dt}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du},\quad x\in[0,1]. \tag{4}$$

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. Second, a novel feature of BLA is that it uses the Beta distribution as the basis for a randomized arm selection mechanism.

3.1 Algorithm

As shown in Figure 1, the state of the BLA is simply the parameters of two Beta PDFs, each of which is associated with an arm of the TABB at hand. The BLA interacts with

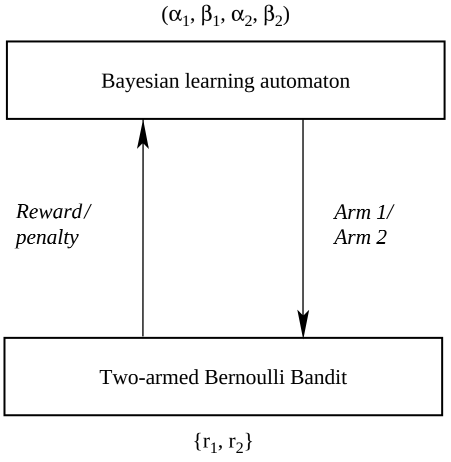

$(\alpha_1, \beta_1, \alpha_2, \beta_2)$
Bayesian learning automaton
Reward/ penalty
Arm 1/ Arm 2
Two-armed Bernoulli Bandit
$\{r_1, r_2\}$

Figure 1. The BLA interacting with a TABB

the TABB, by either pulling Arm 1 or Arm 2. The TABB responds by emitting either a reward or a penalty, which allows the BLA to update its state, resulting in a new arm being pulled.

The following algorithm contains the essence of how BLA learns which arm is the optimal one, i.e. by pulling arms and processing the corresponding responses obtained from the TABB.

BLA algorithm:

```
(1) Initialization: $\alpha_1^1 = \beta_1^1 = \alpha_2^1 = \beta_2^1 = 1$.
(2) For $N = 1,2,\ldots,$ do:
    • Draw a value $x_1$ randomly from Beta distribution $f(x_1; \alpha_1^N, \beta_1^N)$ with parameters $\alpha_1^N, \beta_1^N$.
    • Draw a value $x_2$ randomly from Beta distribution $f(x_2, \alpha_2^N; \beta_2^N)$ with parameters $\alpha_2^N, \beta_2^N$.
    • If $x_1 > x_2$ then pull Arm 1 else pull Arm 2. Denote pulled arm: Arm $i$.
    • Receive either reward or penalty as a result of pulling Arm $i$:
        – Upon reward: $\alpha_i^{N+1} = \alpha_i^N + 1; \beta_i^{N+1} = \beta_i^N$; and $\alpha_j^{N+1} = \alpha_j^N, \beta_j^{N+1} = \beta_j^N$ for $j \neq i$.
        – Upon penalty: $\alpha_i^{N+1} = \alpha_i^N; b_i^{N+1} = b_i^N + 1$; and $\alpha_j^{N+1} = \alpha_j^N, \beta_j^{N+1} = \beta_j^N$ for $j \neq i$.
End algorithm.
```

As seen from the above BLA algorithm, N is a discrete time index and the parameters $\phi^N = (\alpha_1^N, \beta_1^N, \alpha_2^N, \beta_2^N)$ form an infinite discrete four-dimensional state space, which we denote by Φ. The BLA navigates within Φ by iteratively adding unity to either α_1^N, β_1^N, α_2^N, or β_2^N.

Since the state space of the BLA is both discrete and infinite, strictly speaking, it is quite different from both the variable structure and the fixed structure LA families (Thathachar and Sastry, 2004), traditionally referred to as LA. However, in all brevity, the novel aspects of the BLA (when compared to machines which are traditionally referred to as being LA, and to the existing solutions to bandit problems) are listed below:

(1) In traditional LA, the action chosen (i.e. arm pulled) is based on the so-called action probability vector. The BLA does not maintain such a vector, but chooses the arm based on the distribution of the components of the estimate vector.

(2) The second difference is that we have not chosen the arm based on the a posteriori distribution of the estimate. Rather, it has been chosen based on the magnitude of a random sample drawn from the a posteriori distribution, and thus it is more appropriate to state that the arm is chosen based on the order of statistics of instances of these variables[6].

(3) The third significant aspect is that we now can consider the design of pursuit LA in which the estimate used is not of the ML family, but on a Bayesian updating scheme. As far as we know, such a mechanism is also unreported in the literature.

(4) The final significant aspect is that we can now devise solutions to the two-armed bandit problem even for cases when the reward/penalty distribution is not Bernoulli distributed. Indeed, we advocate the use of a Bayesian methodology with the appropriate conjugate prior (Duda *et al.*, 2000).

In the interest of notational simplicity, let Arm 1 be the arm under investigation. Then, for any parameter configuration $\phi^N \in \Phi$ we can state, using a generic notation[7], that the probability of selecting Arm 1 is equal to the probability $P(X_1^N > X_2^N | \phi^N)$ – the probability that a randomly drawn value $x_1 \in X_1^N$ is greater than the other randomly drawn value $x_2 \in X_2^N$ at time step N, where the associated stochastic variables X_1^N, X_2^N are Beta distributed, with parameters $\alpha_1^N, \beta_1^N, \alpha_2^N, \beta_2^N$, respectively. In the following, we will let $p_1^{\phi^N}$ denote this latter probability.

The probability $p_1^{\phi^N}$ can also be interpreted as the probability that Arm 1 is the optimal one, given the observations α_1^N, β_1^N, and α_2^N, β_2^N. This means that the BLA will gradually shift its arm selection focus towards the arm which most likely is the optimal one, as more observations are received.

Finally, observe that the BLA does not rely on any external parameters that must be configured to optimize its performance for specific problem instances[8]. This is in contrast to the traditional families of LA algorithms, where a "learning speed/accuracy" parameter is inherent in ϵ-optimal schemes.

3.2 Example run

In order to provide an intuitive understanding of the workings of the BLA, the following example run demonstrates its behaviour as it interacts with one particular TABB. The reward probability r_1 of Arm 1 is 0.9, while the reward probability r_2 of Arm 2 is 0.6, and hence the BLA should reveal that Arm 1 is the superior one with as few arm pulls as possible:

Initialization. The BLA parameters are initialized so that both the Beta PDFs of Arms 1 and 2 becomes uniform (i.e. initially, we have no knowledge about the reward probabilities r_1 and r_2 of the TABB at hand): $\alpha_1^1 = \beta_1^1 = \alpha_2^1 = \beta_2^1 = 1$. The initial probability of selecting Arm 1, $p_1^{\phi^1}$, is thus 0.5 (calculated as shown in Section 4).

Step 1. The BLA randomly draws a value x_1 from $f(x_1; \alpha_1^1 = 1, \beta_1^N)$ and a value x_2 from $f(x_2; \alpha_2^1 = 1, \beta_2^N = 1)$. It turns out that x_2 is greater than x_1, leading to the selection of Arm 2. Pulling Arm 2 triggers a penalty and the Beta PDF of Arm 2 is updated accordingly. The resulting Beta PDFs for both Arms 1 and 2 are shown to the left in Figure 2. As seen, the PDF of Arm 1 is still uniform because no new information has been obtained for that arm. For Arm 2, on the other hand, a lower reward probability is now more likely. Accordingly, the probability of selecting Arm 1 increases to $p_1^{\phi^2} = 2/3$.

Step 2. Again, the BLA randomly draws two values x_1 and x_2, but this time from the Beta distributions obtained in Step 1, found in the left plot of Figure 2. As before, the result is that x_2 is greater than x_1, and Arm 2 is selected again. Arm 2 emits a reward, and the Beta PDFs are updated as seen in the right plot of Figure 2. The probability of selecting Arm 1 has now become $p_1^{\phi^3} = 0.5$ again.

Step 3. Finally, a value x_1 is drawn from the PDF of Arm 1 that is greater than the value drawn for Arm 2. At this juncture, Arm 1 is pulled, and the result is a reward. The updated Beta PDFs are shown in the leftmost plot of Figure 3, and the probability of selecting Arm 1 based on these PDFs becomes $p_1^{\phi^4} = 0.7$.

...

Step 10. In the following seven steps, the Arm 1 was pulled persistently, and every pull resulted in a reward. This leaves us with the PDFs shown in the rightmost plot of Figure 3. At this stage, the probability that Arm 1 is selected (based on comparing samples from the PDFs) is $p_1^{\phi^{11}} \approx 0.95$. In other words, already after ten steps in this example run, the BLA is focusing in on Arm 1.

...

Step 100. After 100 steps, the BLA has selected Arm 1 95 times, resulting in 86 rewards and nine penalties. Similarly, Arm 2 has only been pulled five times, producing three rewards and two penalties. The probability of selecting Arm 1 based on the resulting PDFs, shown to the right of Figure 4, becomes $p_1^{\phi^{101}} \approx 0.98$.

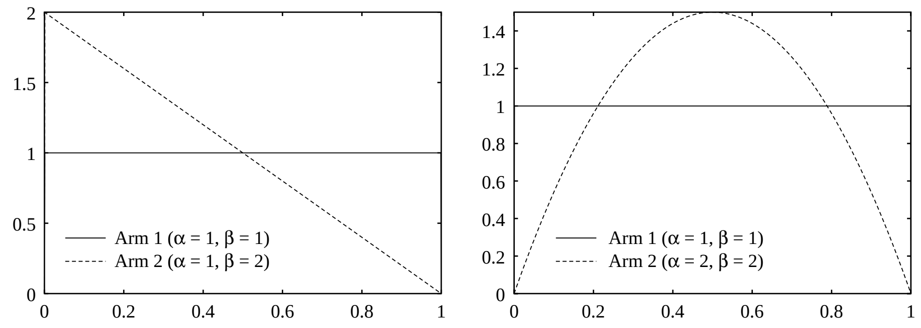


Figure 2. The Beta PDFs associated with Arms 1 and 2 after one arm pull (left) and two arm pulls (right)

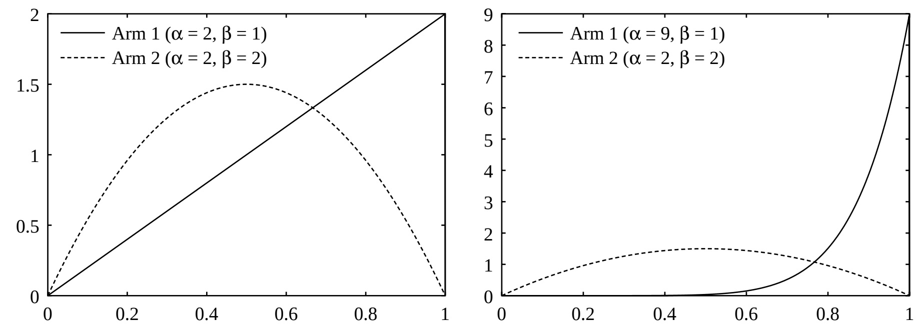


Figure 3. The Beta PDFs associated with Arms 1 and 2 after three arm pulls (left) and ten arm pulls (right)

...

Step 500. Finally, the rightmost plot of Figure 4 shows the state after 500 steps. At this point, the probability that the BLA pulls Arm 1 is approaching unity with $p_1^{\phi^{501}} \approx 0.999$.

From the above example run, it should now be clear how BLA is able to operate, simply based on counting rewards/penalties and on random sampling from a pair of twin Beta distributions.

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. In brief, we establish a collection of new results for the case when two Beta distributions are applied in conjunction, as a randomized arm selection mechanism. Note that we intend to analyze the BLA from an LA perspective. Analyzing the BLA from a bandit playing algorithm point of view is further work.

A well-known and pertinent feature of significant LA is their absolute expediency – the expected probability of selecting the best arm increases monotonically with each additional arm pull. Furthermore, such LA converge 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 (Narendra and Thathachar, 1989). With this perspective as a backdrop, we now proceed to report similar results for the BLA.

4.1 Arm selection probability

First of all, however, note that for many LA schemes, such as the $L_{R\text{-}I}$ scheme, the arm selection probabilities at any given time step N can be expressed as a linear function of the arm selection probabilities of the previous time step, thus facilitating Markov chain-based convergence analysis (Narendra and Thathachar, 1989). In contrast, the corresponding arm selection probabilities of the BLA are related to the state $\phi^N = (\alpha_1^N, \beta_1^N, \alpha_2^N, \beta_2^N)$ at time step N, in terms of a complex nonlinear function, which we state here without further ado. The probability $p_1^{\phi^N}$ of selecting Arm 1, given the state $\phi^N = (\alpha_1^N, \beta_1^N, \alpha_2^N, \beta_2^N)$, is:

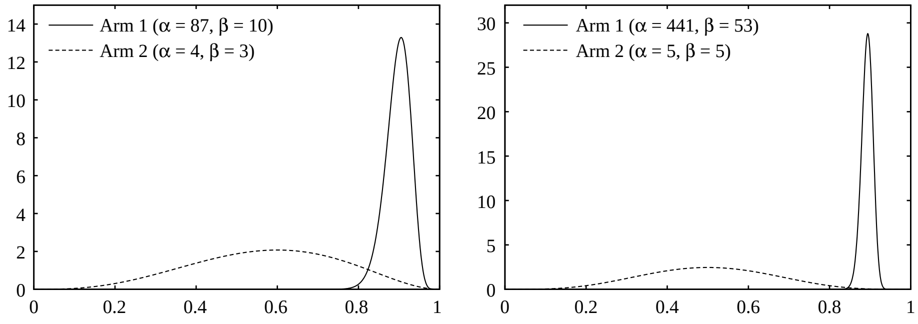


Figure 4.
The Beta PDFs associated with Arms 1 and 2 after 100 arm pulls (left) and 500 arm pulls (right)

$$p_1^{\phi^N} = \frac{(\alpha_1^N + \beta_1^N - 1)!(\alpha_2^N + \beta_2^N - 1)!}{(\alpha_1^N - 1)!(\beta_1^N - 1)!(\alpha_1^N + \beta_1^N + \alpha_2^N + \beta_2^N - 2)!} \times \sum_{j=\alpha_2^N}^{\alpha_2^N+\beta_2^N-1} \frac{(j + \alpha_1^N - 1)!(\beta_1^N + \alpha_2^N + \beta_2^N - j - 2)!}{j!(\alpha_2^N + \beta_2^N - 1 - j)!}. \tag{5}$$

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 Wyatt (1997), the above expression is not computed explicitly in the BLA algorithm.

Combined with an infinite state space $\phi^N \in \Phi$ of high dimensionality, the BLA does not lend itself towards traditional Markov chain-based analysis, as presented in the LA literature. This complication has led us to investigate an alternate strategy for analyzing its convergence properties, which, in turn, allows us to avoid explicitly dealing with the above complexity.

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. Let N_1 denote the number of times Arm 1 has been pulled, and N_2 the number of times Arm 2 has been pulled, i.e. $N_1 + N_2 = N$. It is well known that as the number of arm pulls, N_i, tends to infinity for a given arm i, the expected value, $E[X_i^N] = (\alpha_i^N/(\alpha_i^N + \beta_i^N))$, 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 the 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 $(\alpha_1^N/(\alpha_1^N + \beta_1^N))$, associated with Arm 1 (the optimal arm), fluctuates to a value that is less than the estimate $(\alpha_2^N/(\alpha_2^N + \beta_2^N))$ associated with Arm 2, one could risk that the probability of selecting Arm 1 would tend to 0 before $(\alpha_1^N/(\alpha_1^N + \beta_1^N))$ is given the opportunity to converge to the true value r_1.

Thus, in the latter scenario, with $(\alpha_2^N/(\alpha_2^N + \beta_2^N))$ tending to r_2 and $(\alpha_1^N/(\alpha_1^N + \beta_1^N))$ 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: the BLA is instantaneously self-correcting

First of all, we shall show that the BLA is instantaneously self-correcting in the sense that we may expect the BLA to increase the probability of selecting Arm 1 after one more arm pull if the estimate of r_1 has become too low, i.e. below r_2. Consequently, the probability of receiving more observations for more accurately estimating r_1 also increases. This result can be expressed formally as follows:

Theorem 4.1. Assume that the expected value of the Beta distribution associated with Arm 2, $(\alpha_2^N/(\alpha_2^N + \beta_2^N))$, is approaching r_2 because BLA has wrongly settled with Arm 2 as the preferred choice. In other words, it is obtaining a progressively better Bayesian estimate for r_2 with variance approaching zero, $E\{(X_2^N - r_2)^2\} \rightarrow 0$. Furthermore, let $p_1^{r_2}[\alpha_1^N, \beta_1^N]$ denote the probability of selecting Arm 1 at some time step N in this latter setting, with r_2 given and with (α_1^N, β_1^N) being the parameters of the Beta distribution associated with Arm 1. Then the following is true:

$$E\{p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] | p_1^{r_2}[\alpha_1^N, \beta_1^N]\} > p_1^{r_2}[\alpha_1^N, \beta_1^N] \Longleftrightarrow \frac{\alpha_1^N}{\alpha_1^N + \beta_1^N} < r_1. \tag{6}$$

Significance of the result. The implication of the result is the following. Given the arm selection probability $p_1^{r_2}[\alpha_1^N, \beta_1^N]$ at time step N, the expected value of $p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}]$ at time step $N+1$ will be greater than $p_1^{r_2}[\alpha_1^N, \beta_1^N]$ if and only if the current reward probability estimate r_1 for Arm 1, $(\alpha_1^N/(\alpha_1^N + \beta_1^N))$, is less than r_1. Thus, if the estimate of r_1 is too low, i.e. below r_2, we may expect the BLA to increase the probability of selecting Arm 1 after a single arm pull, and in this sense it is instantaneously self-correcting.

Proof. The proof consists of demonstrating the following inequality for $p_1^{r_2}[\alpha_1^N, \beta_1^N]$:

$$E\{p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] | p_1^{r_2}[\alpha_1^N, \beta_1^N]\} > p_1^{r_2}[\alpha_1^N, \beta_1^N], \tag{7}$$

with α_1^N and β_1^N being constants.

First of all, we reformulate $E\{p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] | p_1^{r_2}[\alpha_1^N, \beta_1^N]\}$ in terms of $p_1^{r_2}[\alpha_1^N + 1, \beta_1^N]$ and $p_1^{r_2}[\alpha_1^N, \beta_1^N + 1]$. At any time step N, either Arm 1 or Arm 2 is pulled. Arm 1 is pulled with probability $p_1^{r_2}[\alpha_1^N, \beta_1^N]$ while Arm 2 is pulled with probability $1 - p_1^{r_2}[\alpha_1^N, \beta_1^N]$. Pulling Arm 1 either produces a reward or a penalty. A reward is obtained with probability r_1 and a penalty is obtained with probability $1 - r_1$. If Arm 2 is pulled, on the other hand, we have that $p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] = p_1^{r_2}[\alpha_1^N, \beta_1^N]$, simply because r_2 is given and because no feedback is obtained for Arm 1, i.e. $\alpha_1^{N+1} = \alpha_1^N$ and $\beta_1^{N+1} = \beta_1^N$. Thus:

$$\begin{aligned} &E\{p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] | p_1^{r_2}[\alpha_1^N, \beta_1^N]\} \Longleftrightarrow \\ &\left(p_1^{r_2}[\alpha_1^N + 1, \beta_1^N] r_1 + p_1^{r_2}[\alpha_1^N, \beta_1^N + 1](1 - r_1)\right). \\ &p_1^{r_2}[\alpha_1^N, \beta_1^N] + p_1^{r_2}[\alpha_1^N, \beta_1^N]\left(1 - p_1^{r_2}[\alpha_1^N, \beta_1^N]\right) \end{aligned} \tag{8}$$

By performing the above indicated substitution, and then dividing the left and right side of the inequality by $p_1^{r_2}[\alpha_1^N, \beta_1^N]$ we get:

$$\begin{aligned} &E\{p_1^{r_2}[\alpha_1^{N+1}, \beta_1^{N+1}] | p_1^{r_2}[\alpha_1^N, \beta_1^N]\} > p_1^{r_2}[\alpha_1^N, \beta_1^N] \Longleftrightarrow \\ &\left(p_1^{r_2}[\alpha_1^N + 1, \beta_1^N] r_1 + p_1^{r_2}[\alpha_1^N, \beta_1^N + 1](1 - r_1)\right) + \end{aligned} \tag{9}$$

$$\begin{aligned} &\left(1 - p_1^{r_2}[\alpha_1^N, \beta_1^N]\right) > 1 \Longleftrightarrow \\ &p_1^{r_2}[\alpha_1^N + 1, \beta_1^N] r_1 + p_1^{r_2}[\alpha_1^N, \beta_1^N + 1](1 - r_1) > p_1^{r_2}[\alpha_1^N, \beta_1^N]. \end{aligned} \tag{10}$$

By virtue of the BLA algorithm, we then have that $p_1[\alpha_1^N, \beta_1^N | r_2]$ can be expressed in terms of a cumulative Beta distribution:

$$p_1\left[\alpha_1^N, \beta_1^N | r_2\right] = P\left(X_1^N > r_2 | \alpha_1^N, \beta_1^N\right) \tag{11}$$

$$= 1 - I_{r_2}\left(\alpha_1^N, \beta_1^N\right) \tag{12}$$

where $I_x(\alpha,\beta)$ is the regularized incomplete Beta function, which also happens to be the cumulative Beta distribution $F(x;\alpha,\beta)$ with parameters α,β:

$$I_x(\alpha, \beta) = \sum_{j=\alpha}^{\alpha+\beta-1} \frac{(\alpha+\beta-1)!}{j!(\alpha+\beta-1-j)!} x^j (1-x)^{\alpha+\beta-1-j}. \tag{13}$$

Accordingly, by replacing $p_1[\alpha_1^N, \beta_1^N | r_2]$ with $1 - I_{r_2}(\alpha_1^N, \beta_1^N)$ in equation (11) we get:

$$\begin{aligned} &p_1^{r_2}\left[\alpha_1^N + 1, \beta_1^N\right] r_1 + p_1^{r_2}\left[\alpha_1^N, \beta_1^N + 1\right](1 - r_1) > p_1^{r_2}\left[\alpha_1^N, \beta_1^N\right] \Longleftrightarrow \\ &\left(1 - I_{r_2}\left(\alpha_1^N + 1, \beta_1^N\right)\right) r_1 + \end{aligned} \tag{14}$$

$$\begin{aligned} &\left(1 - I_{r_2}\left(\alpha_1^N, \beta_1^N + 1\right)\right)(1 - r_1) > 1 - I_{r_2}\left(\alpha_1^N, \beta_1^N\right) \Longleftrightarrow \\ &I_{r_2}\left(\alpha_1^N, \beta_1^N\right) - r_1 I_{r_2}\left(\alpha_1^N + 1, \beta_1^N\right) - (1 - r_1) I_{r_2}\left(\alpha_1^N, \beta_1^N + 1\right) > 0. \end{aligned} \tag{15}$$

It turns out that $I_x(\alpha + 1,\beta)$ and $I_x(\alpha + 1,\beta)$ can be expressed in terms of $I_x(\alpha,\beta)$:

$$I_x(\alpha + 1, \beta) = I_x(\alpha, \beta) - \frac{(\alpha+\beta-1)!}{\alpha!(\beta-1)!}(1-x)^\beta x^\alpha \tag{16}$$

$$I_x(\alpha, \beta + 1) = I_x(\alpha, \beta) + \frac{(\alpha+\beta-1)!}{(\alpha-1)!\beta!}(1-x)^\beta x^\alpha. \tag{17}$$

Hence, again by means of substitution, as indicated above, we get:

$$\begin{aligned} &I_{r_2}\left(\alpha_1^N, \beta_1^N\right) - r_1 I_{r_2}\left(\alpha_1^N + 1, \beta_1^N\right) - (1 - r_1) I_{r_2}\left(\alpha_1^N, \beta_1^N + 1\right) > 0 \Longleftrightarrow \\ &I_{r_2}\left(\alpha_1^N, \beta_1^N\right) - \left(r_1\left(I_{r_2}\left(\alpha_1^N, \beta_1^N\right) - \frac{(\alpha_1^N + \beta_1^N - 1)!}{\alpha_1^N!(\beta_1^N - 1)!}(1 - r_2)^{\beta_1^N} r_2^{\alpha_1^N}\right) + \right. \\ &\left.(1 - r_1)\left(I_{r_2}\left(\alpha_1^N, \beta_1^N\right) + \frac{(\alpha_1^N + \beta_1^N - 1)!}{(\alpha_1^N - 1)!\beta_1^N!}(1 - r_2)^{\beta_1^N} r_2^{\alpha_1^N}\right)\right) > 0 \end{aligned} \tag{18}$$

which, finally, can be simplified as follows:

$$I_{r_2}\left(\alpha_1^N, \beta_1^N\right) - \left(r_1\left(I_{r_2}\left(\alpha_1^N, \beta_1^N\right)\right) - \frac{(\alpha_1^N + \beta_1^N - 1)!}{\alpha_1^N!(\beta_1^N - 1)!}(1 - r_2)^{\beta_1^N} r_2^{\alpha_1^N}\right) + \tag{19}$$

$$(1 - r_1)\left(I_{r_2}\left(\alpha_1^N, \beta_1^N\right) + \frac{(\alpha_1^N + \beta_1^N - 1)!}{(\alpha_1^N - 1)!\beta_1^N!}(1 - r_2)^{\beta_1^N} r_2^{\alpha_1^N}\right) > 0 \Longleftrightarrow \frac{\alpha_1^N}{\alpha_1^N + \beta_1^N} < r_1. \tag{20}$$

□

Remark. The reader should observe that the above property is true independent of the value that r_2 takes. Furthermore, due to the symmetry of the situation, by switching the identity of the arms and by inverting the direction of the inequality, we also can claim the following corollary.

Corollary 4.2. If at any time instant the estimate of the reward probability of Arm 2 becomes greater than r_2, the expected arm selection probability of pulling Arm 2 at the next time instant will be lower than its current arm selection probability!

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 $E[(X_2^N - r_2)^2]$ tending to 0. After all, for the $L_{R\text{-}I}$ and pursuit LA schemes, such arbitrarily long reward/penalty sequences occur with an arbitrarily small probability. We now consider the question of whether such sequences can occur for the BLA.

Since all of the quantities involved in equation (5) are positive, one can infer that the probability of selecting Arm 1 never becomes 0. This applies, of course, to Arm 2 also, and thus, at any finite time instant, the arm selection probabilities will never become unity or zero, respectively. As suggested in Wyatt (1997), the worst possible scenario is a sequence of rewards/penalties which only yields penalties for Arm 1 (the optimal arm) and only rewards for Arm 2 (the inferior arm), which occurs with the following probability:

$$p_1^{\phi^N} = \frac{(\beta_1^N)!(\alpha_2^N)!.}{(\beta_1^N + \alpha_2^N)!}. \tag{21}$$

Thus, based on Theorem 4.1, we conjecture that both of the arms will be selected an arbitrarily large number of times as N approaches infinity. This leads us to the following result.

Theorem 4.3. As the arms are pulled an arbitrarily large number of times, the BLA converges to persistently selecting the optimal arm with probability as close to unity as desired:

$$\lim_{N\to\infty} p_1^{\phi^N} \to 1 \quad \textit{with} \quad r_1 > r_2. \tag{22}$$

Proof. By virtue of the Bayesian convergence of the parameters of the Binomial distribution, as the number of arm pulls, N_i, tends to infinity for a given Arm i, the expected value, $E[X_i^N] = (\alpha_i^N/(\alpha_i^N + \beta_i^N))$, of the associated Beta distribution converges to the true reward probability r_i. Furthermore, its variance tends to zero. Thus, the correct arm will gradually and eventually reveal itself, and the probability of pulling Arm 1 by drawing a value $x_1 > x_1$ (where x_1 is drawn from a Beta distribution with parameters (α_1^N, β_1^N), and x_2 is drawn from a Beta distribution with parameters (α_2^N, β_2^N)), tends to unity. □

5. Experiments

In this section, we evaluate the BLA by comparing it with UCB-tuned, the best performing algorithm from Auer *et al.* (2002), as well as the $L_{R\text{-}I}$ and P-schemes, which can be seen as established top performers in the LA field. Based on our comparison with these "reference" algorithms, it should be quite straightforward to also relate the BLA performance results to the performance of other similar algorithms.

For the sake of fairness, we base our comparison on the experimental setup for the TABB found in Auer *et al.* (2002). Furthermore, we have chosen the decentralized GG, used for QoS control in sensor networks, as a basis for evaluating the performance of the different algorithms in a more complex application.

5.1 Application I: two-armed Bernoulli bandits

Although we have conducted numerous experiments using various reward distributions, we report, for the sake of brevity, results for the following five reward/penalty distributions since they are representative for our findings. 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 difference between the arms, we increase the difficulty of the TABB problem. *Distribution 2* ($r_1 = 0.9, r_2 = 0.8$) fulfills this purpose in Auer *et al.* (2002), however, in order to stress the schemes further, we also apply *Distribution 3* ($r_1 = 0.9, r_2 = 0.85$) and *Distribution 4* ($r_1 = 0.9, r_2 = 0.89$). The challenge of *Distribution 5* ($r_1 = 0.55, r_2 = 0.45$) is its high variance combined with the small difference between the two arms.

For these distributions, an ensemble of 1,000 independent replications with different random number streams was performed to minimize the variance of the reported results. In each replication, the BLA, UCB-tuned, the $L_{R\text{-}I}$, and the P-scheme conducted 100,000 arm pulls.

Note that real-world instantiations of the bandit problem, such as resource allocation in web polling (Granmo *et al.*, 2007), may exhibit any reward probability in the interval [0,1]. Hence, a solution scheme designed to tackle bandit problems in general, should perform well across the complete space of reward probabilities.

Figure 5 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 the number of arm pulls.

Because of the logarithmically scaled x-axis, it is clear from the plots that both the BLA and UCB-tuned attain a logarithmically growing regret. Moreover, the performance of the BLA is significantly better than that of UCB-tuned. Surprisingly, both of the LA schemes converge to a constant value of regret. This can be explained by their ϵ-optimality and the relatively small learning speed parameter used ($a = 0.01$). In brief, the LA did not converge to only selecting the optimal arm in all of the 1,000 replications.

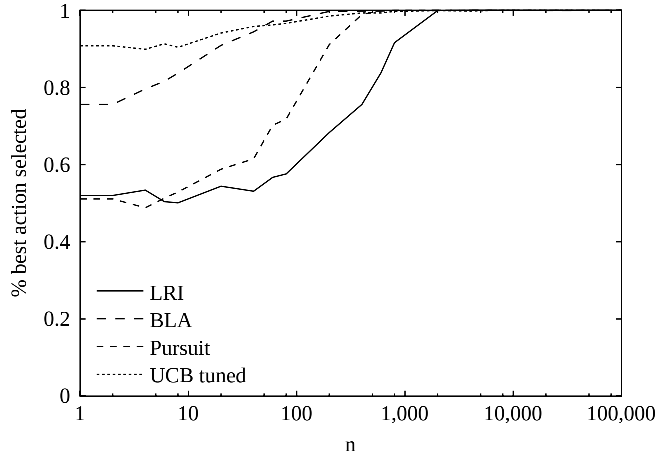
Figure 6 contains the comparison on Distribution 2. As shown, the LA still achieve a constant value of the regret, and again, the BLA outperforms UCB-tuned.

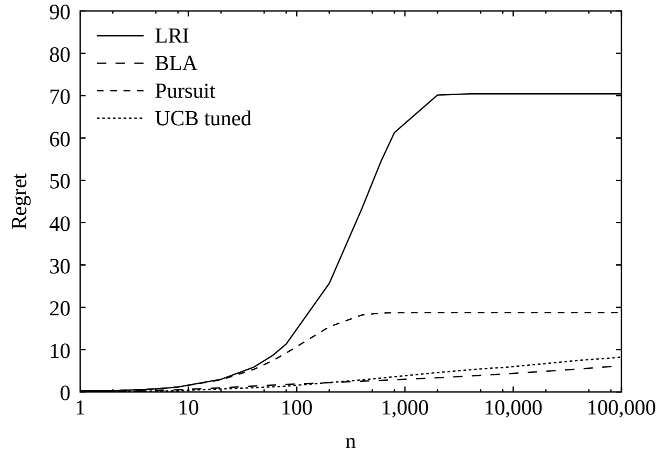
For Distribution 3, however, it turns out that the set learning accuracy of the LA is too low to always converge to only selecting the optimal arm. In some of the replications, the LA also converges to selecting the inferior arm only, and as seen in Figure 7, this leads to a linearly growing regret. As also seen, the BLA continues to provide significantly better performance than UCB-tuned.

The latter property of LA becomes even more apparent with even more similar arms, as seen in Figure 8.

In spite of this, we observe that the performance of both the BLA and UCB-tuned are surprisingly unaffected by the increased difficulty. Indeed, the performance advantage of the BLA becomes even more apparent in terms of the regret.

Finally, we observe that the high variance of Distribution 5 reduces the performance gap between the BLA and UCB-tuned, as seen in Figure 9, leaving UCB-tuned with slightly lower regret compared to the BLA.

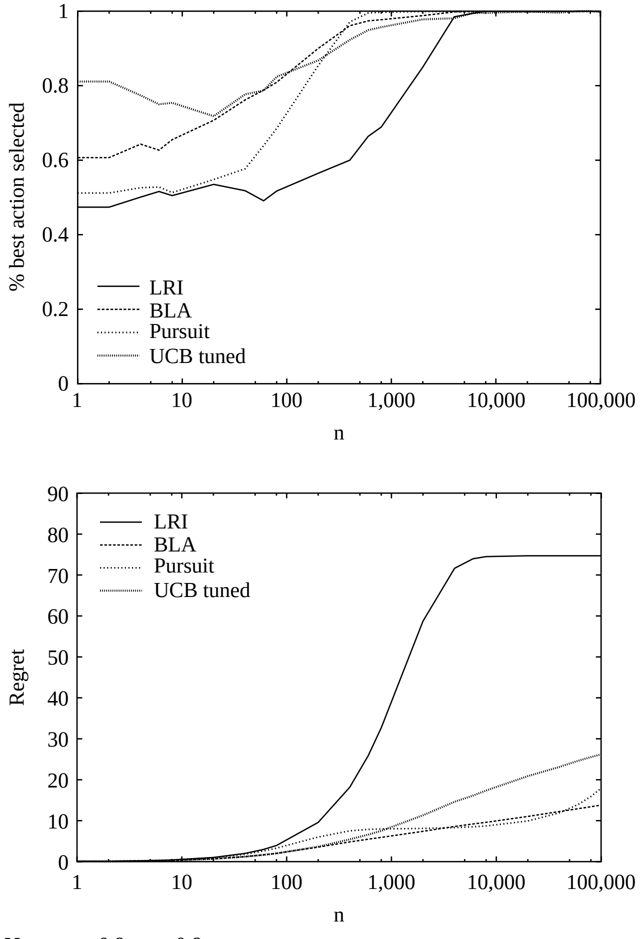




Note: $r_1 = 0.9$; $r_2 = 0.6$

Figure 5.
Probability of selecting the best arm (top) and the regret (bottom) for Distribution 1

From the above results, it is clear that the performance of UCB-tuned and the BLA vary with the reward probabilities. In order to systematically explore this variation, Figure 10 plots the mean regret after 100,000 arm pulls. In the figure, the x-axis spans the space of reward probabilities $r_1 \in [0,1]$ that can be associated with Arm 1, with r_2 being set to $r_1 - 0.1$. Note that, for r_2 we chose a value close to r_1 in the interest of stressing the UCB-tuned and BLA schemes. As seen from the plot, which includes the accompanying 99 percent confidence intervals, the BLA is clearly superior in situations where the reward probabilities are relatively low or high. Furthermore, the BLA and UCB-tuned produce statistically similar performance with reward probabilities close to 0.5. Also notice that the performance of UCB-tuned is persistently "unstable" in certain regions of the reward probability space, producing a high variance compared to the mean regret. This confirms the observations from Audibert *et al.* (2007) where the high variance of UCB-tuned was first reported.

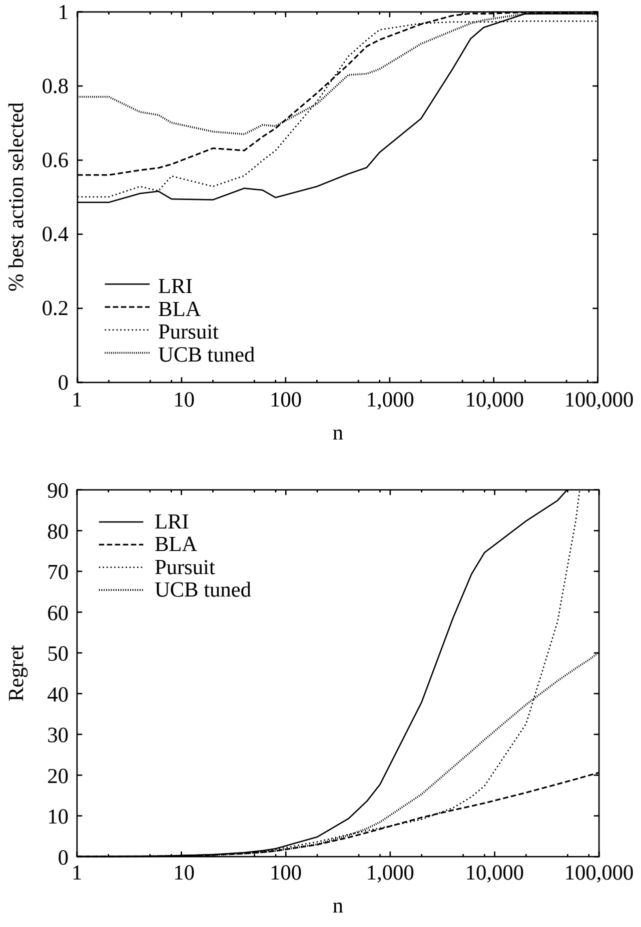


Note: $r_1 = 0.9$; $r_2 = 0.8$

Figure 6.
Probability of selecting the best arm (top) and the regret (bottom) for Distribution 2

Remark 1. Note that the LA can achieve constant regret in all of the latter experiments too, by increasing the learning accuracy. However, this significantly reduces the learning speed, which for the present setting is already worse than that of the BLA and UCB-tuned.

Remark 2. One possible explanation of the superiority of the BLA is the following. In brief, the BLA is the only one of the above schemes that maintains a Bayesian posterior distribution of the possible reward probabilities of the arms. Thus, when the BLA makes a decision, it is based on the best possible knowledge about the arm reward probabilities, given the rewards and penalties that have been received thus far. UCB-tuned, on the other hand, only updates a running average and its variance, after each arm pull. It is true that this average will be normally distributed after a large number of arm pulls, but for the initial, and perhaps most important part of the learning, the latter will not be the case. Accordingly, in this sense, UCB-tuned makes its decision based on inaccurate information

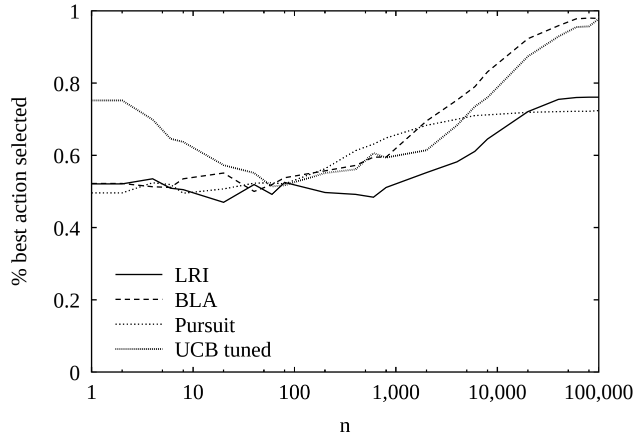


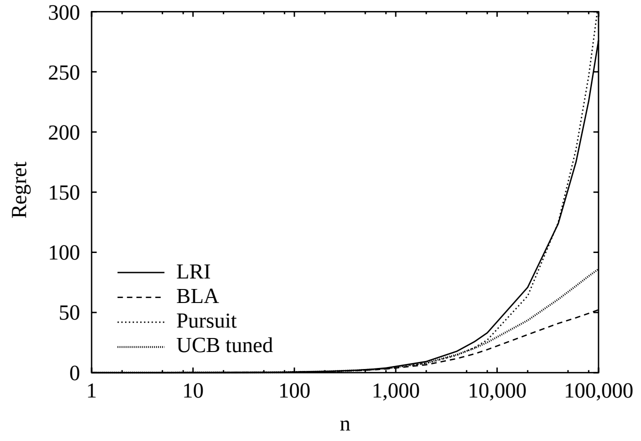
Note: $r_1 = 0.9$; $r_2 = 0.85$

Figure 7.
Probability of selecting the best arm (top) and the regret (bottom) for Distribution 3

about the reward probabilities of the arms. It is thus to be expected that BLA will outperform UCB-tuned even though both are parametric.

Remark 3. When it comes to computational performance, it is clear that all of the tested algorithms spend computational resources quite modestly. However, the BLA is slightly more complex that UCB-tuned, which, in turn, is more complex than the P-scheme. Computationally, the least complex of these is the L_{RI} scheme. Measuring the number of arm pulls that can be processed per second using a straightforward Python implementation of the schemes, running on a modern laptop[9], exposes this difference. The L_{RI} scheme processes about 70,000 arm pulls per second, while the P-scheme is able to process 40,000 arm pulls. The performance of UCB-tuned is 20,000 arm pulls each second, and finally, the BLA is able to process about 10,000. This slight difference in computational performance is, however, mainly relevant when executing simulations; in real world applications, it is typically the environment that the





Note: $r_1 = 0.9$; $r_2 = 0.89$

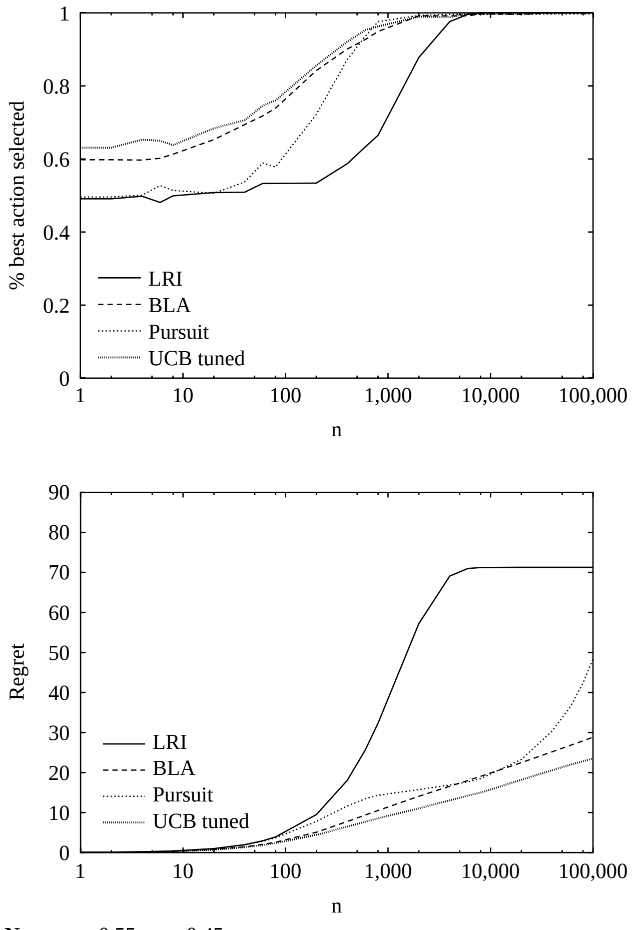
Figure 8. Probability of selecting the best arm (top) and the regret (bottom) for Distribution 4

algorithm operates in that dictates the arm pull rate. And from this perspective, the computational cost of all of the algorithms is quite negligible, in practice.

5.2 Application II: the GG

The GG. One of the most fascinating games studied in the field of artificial games is the GG. We describe it using the following informal formulation given in Narendra and Thathachar (1989):

> Imagine a large room containing N cubicles and a raised platform. One person (voter) sits in each cubicle and a Referee stands on the platform. The Referee conducts a series of voting rounds as follows. On each round the voters vote "Yes" or "No" (the issue is unimportant) simultaneously and independently (they do not see each other) and the Referee counts the fraction, λ, of "Yes" votes. The Referee has a uni-modal performance criterion $G(\lambda)$, which is optimized when the fraction of "Yes" votes is exactly λ^*. The current voting round ends with the Referee awarding a dollar with probability $G(\lambda)$ and assessing a dollar with probability



Note: $r_1 = 0.55$; $r_2 = 0.45$

Figure 9.
Probability of selecting the best arm (top) and the regret (bottom) for Distribution 5

$1 - G(\lambda)$ to every voter independently. On the basis of their individual gains and losses, the voters then decide, again independently, how to cast their votes on the next round.

The game has many interesting and fascinating features which render it both non-trivial and intriguing. These are listed below:

(1) The game is a non-trivial non-zero-sum game.

(2) Unlike the games traditionally studied in the AI literature (like chess, checkers, lights-out, etc.) the game is essentially a distributed game.

(3) The players of the game are ignorant of all of the parameters of the game. All they know is that they have to make a choice, for which they are either rewarded or penalized. They have no clue as to how many other players there are, how they are playing, or even of how/why they are rewarded/penalized.

(4) The stochastic function used to reward or penalize the players can be completely arbitrary, as long as it is uni-modal.

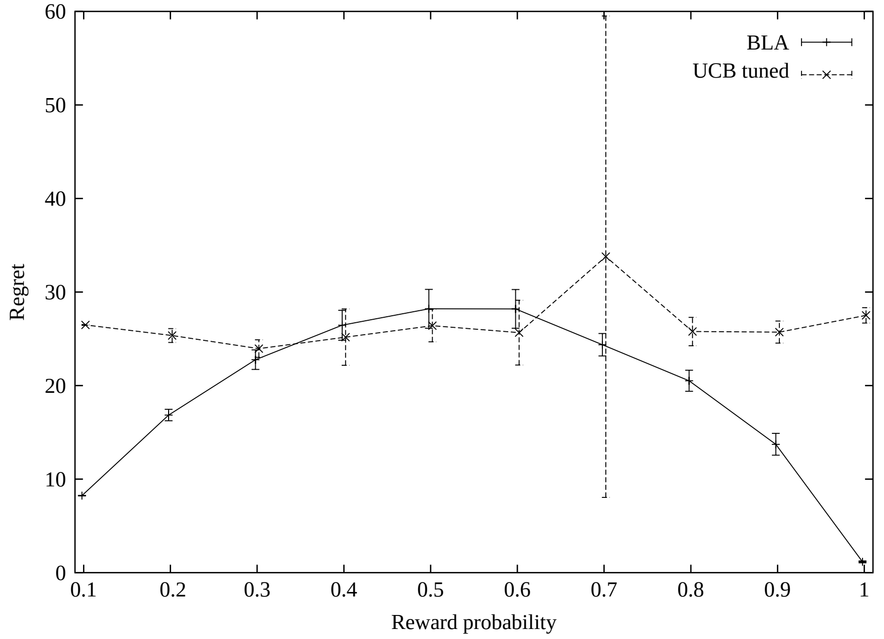


Figure 10.
The regret of the BLA and UCB-tuned after 100,000 arm pulls as a function of the reward probability

The literature concerning the GG is sparse. It was initially studied in the general learning domain, and, as far as we know, was for a long time merely considered as an interesting pathological game. Recently, however, the GG has found important applications within two main areas, namely, QoS support in wireless sensor networks (Chen and Varshney, 2004) and within cooperative mobile robotics as summarized in Cao *et al.* (1997).

QoS control in sensor networks. The GG has found applications within the field of sensor networks, as explained briefly here. Consider a base station that collects data from a sensor network. The sensors of the network are battery driven and have been dropped from the air, leaving some of them non-functioning. The functioning sensors can either be switched on or off, and since they are battery-driven, it is expedient that they should be turned off whenever possible. The base station, on the other hand, has been set to maintain a certain resolution (i.e. QoS), and therefore requires that Q sensors are switched on. Unfortunately, it does not know the number of functioning sensors, and it is only able to contact them by means of a broadcast, leaving it unable to address them individually. This leaves us with the following challenge: how can the base station turn on exactly Q sensors, only by means of its limited broadcast capability?

Iyer and Kleinrock (2003) proposed a scheme where the base station provided broadcasted QoS feedback to the sensors of the network. Using this model, the above problem was solved by modeling it as a GG (Tung and Kleinrock, 1996). From the GG perspective, a sensor is seen as a voter that chooses between transmitting data or remaining idle in order to preserve energy[10]. Thus, in essence, each sensor takes the role of a GG player that either votes “on” or “off,” and acts accordingly. The base station, on the other hand, is seen as the GG referee with a uni-modal performance function $G(\cdot)$ whose maximum is found at Q normalized by the total number of sensors available. The "trick" is to let the base station:

- count the number of sensors that have turned on; and
- use the broadcast mechanism to distribute, among the sensors, the corresponding reward based on the probability obtained from $G(\cdot)$.

The application of the GG solution to the field of sensor networks is thus both straightforward and obvious.

Other possible cooperative robotics applications include controlling a moving platform and guarding a specified perimeter (Cao *et al.*, 1997). In all of these cases, the solution to the problem in question would essentially utilize the solution to the GG in a plug-and-play manner.

Experimental setup. We have chosen the decentralized *GG*, used for QoS control in sensor networks, as a basis for evaluating the performance of the different algorithms in a more complex application. The GG that we used for testing consisted of ten players, with each player being replaced by the bandit playing algorithm being tested. We use the following reward function:

$$G(\lambda) = 0.2 + 0.8e^{-0.02(10\times\lambda-9)^2}. \tag{23}$$

Note that we have chosen the optimal value of λ to be near the maximum value of λ, i.e. 0.9, in order to stress the algorithms. Even voting completely at random (i.e. uniformly), would perform expedient if the optimal value was set to be near 0.5.

As seen in topmost plot in Figure 11, the team of ten BLAs not only learns quicker than the other schemes, but also attains a lower average absolute error. The team of UCB-tuned players outperform the pursuit and L_{RI} schemes in this setting too. However, its performance is more unstable than that of the other schemes, fluctuating with a greater magnitude. As seen from the regret of each team of players, it is clear that BLA excels in the GG as well.

From the above results, we conclude that the BLA is the superior choice both for the TABB problem and for the *GG* application, outperforming established top performers from both the field of LA, as well as from the field of bandit playing algorithms.

6. Conclusion and further work

In this paper, we presented the BLA for tackling the classical TABB. In contrast to previous LA and regret-minimizing approaches, the 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, the BLA is instantaneously self-correcting and converges to only pulling the optimal arm with a probability as close to unity as desired.

Extensive experimental results demonstrated that, unlike the $L_{R\text{-}I}$ and P-schemes, the BLA does not rely on an external learning speed/accuracy control. The BLA also outperformed UCB-tuned, achieving a logarithmically growing regret.

Accordingly, from the above perspective, it is our belief that the BLA represents a new promising avenue of research, opening up for an array of research problems.

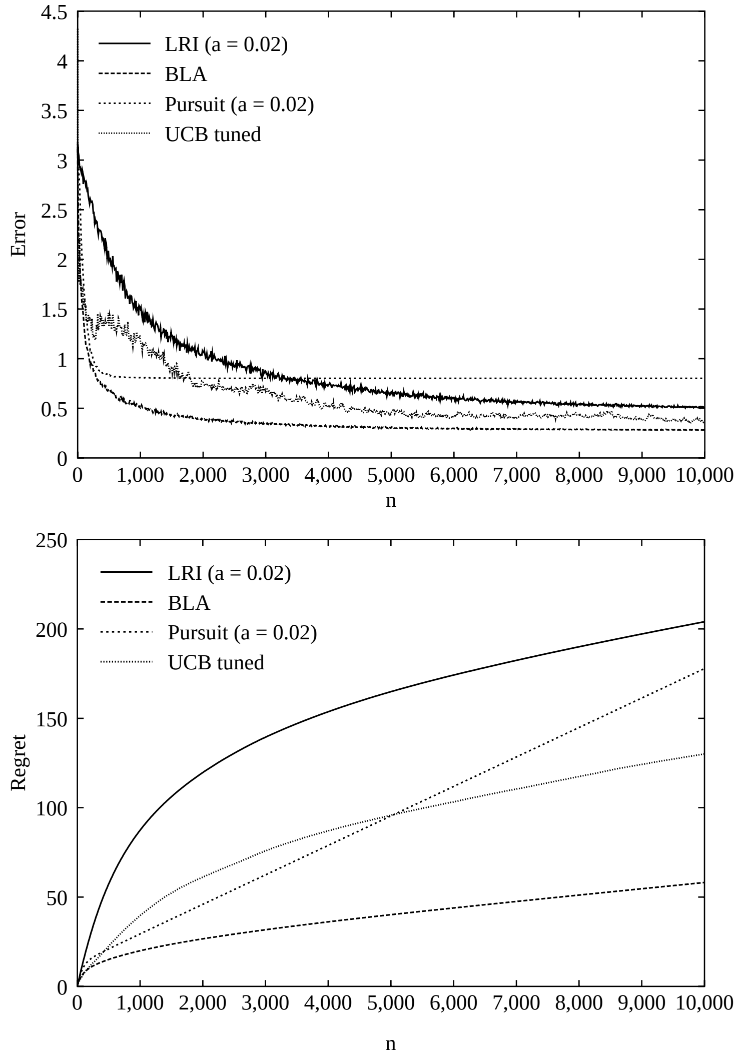


Figure 11.
The average absolute error (top) and the regret (bottom) for each iteration of a GG consisting of ten players

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. Second, incorporating other reward distributions, such as Gaussian and multinomial distributions, into our scheme is of interest. Third, 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. Finally, systems of BLA, such as those involved in the GG, can be studied from a game theoretic point of view, where multiple BLAs interact forming the basis for multi-agent systems.

Notes

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.
2. Using the regret as a performance measure is typical in the literature on bandit playing algorithms. As opposed to this, using the arm selection probability is typical in the LA literature. We will use both of these metrics for the sake of comprehensiveness.
3. A comparison of bandit playing algorithms can be found in Vermorel and Mohri (2005), with the UCB-tuned distinguishing itself in Auer *et al.* (2002).
4. The author is extremely grateful to Dr Dimitrakakis for informing him about this publication. Thompson's result seems to have been buried in the vast body of literature, and the citations to it in the TABB literature is almost non-existent.
5. It should be mentioned that one of the central equations that we derive (i.e. equation (5)) was actually derived by Thompson as early as 1933 in the context of the so-called Thompson sampling principle!
6. To the best of our knowledge, the concept of having automata choose actions based on the order of statistics of instances of estimate distributions, has been unreported in the literature.
7. By this, we mean that P is not a fixed function. Rather, it denotes the probability function for a random variable, given as an argument to P.
8. This, of course, does not mean that the BLA algorithm does not contain configurable parts, only that the basic BLA algorithm perform well in all tested environments, without performing any fine tuning of the algorithm.
9. Intel Core Due 2.00 GHz CPU.
10. Neither Iyer and Kleinrock (2003) and Tung and Kleinrock (1996) or we consider the computational power required to run the LA on these sensors. But our belief is that this is marginal because it essentially involves maintaining a single probability and performing at most two multiplications per broadcast.

References

Agache, M. and Oommen, B.J. (2002), "Generalized pursuit learning schemes: new families of continuous and discretized learning automata", *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 32 No. 6, pp. 738-49.

Audibert, J.-Y., Munos, R. and Szepesvarri, C. (2007), "Tuning bandit algorithms in stochastic environments", *Proceedings of the 18th International Conference of Algorithmic Learning Theory, Sendai, Japan*, Springer, Berlin, pp. 150-65.

Auer, P., Cesa-Bianchi, N. and Fischer, P. (2002), "Finite-time analysis of the multiarmed bandit problem", *Machine Learning*, Vol. 47, pp. 235-56.

Bhulai, S. and Koole, G. (2000), "On the value of learning for Bernoulli bandits with unknown parameters", *IEEE Transactions on Automatic Control*, Vol. 45 No. 11, pp. 2135-40.

Blum, A., Even-Dar, E. and Ligett, K. (2006), "Routing without regret: on convergence to Nash equilibria of regret-minimizing algorithms in routing games", *Proceedings of the Twenty-Fifth Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2006)*, ACM, New York, NY, pp. 45-52.

Cao, Y.U., Fukunaga, A.S. and Kahng, A. (1997), "Cooperative mobile robotics: antecedents and directions", *Autonomous Robots*, Vol. 4 No. 1, pp. 7-27.

Chen, D. and Varshney, P.K. (2004), "QoS support in wireless sensor networks: a survey", paper presented at The 2004 International Conference on Wireless Networks (ICWN 20 04), Las Vegas, NV.

Dimitrakakis, C. (2006), "Nearly optimal exploration-exploitation decision thresholds", *Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 2006), Athens, Greece*, Lecture Notes in Computer Science, Springer, Berlin, pp. 850859.

Duda, R., Hart, P. and Stork, D. (2000), *Pattern Classification*, 2nd ed., Wiley, New York, NY.

Gelly, S. and Wang, Y. (2006), "Exploration exploitation in go: UCT for Monte-Carlo go", *Proceedings of NIPS-2006*, NIPS.

Granmo, O.-C. and Bouhmala, N. (2007), "Solving the satisfiability problem using finite learning automata", *International Journal of Computer Science and Applications*, Vol. 4 No. 3, pp. 15-29.

Granmo, O.-C., Oommen, B.J., Myrer, S.A. and Olsen, M.G. (2007), "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*, Vol. 37 No. 1, pp. 166-75.

Iyer, R. and Kleinrock, L. (2003), "QoS control for sensor networks", *IEEE International Conference on Communications*, Vol. 1, pp. 517-21.

Kaelbling, L.P. (1993), *Learning in Embedded Systems*, PhD thesis, Stanford University, Stanford, CA.

Kocsis, L. and Szepesvari, C. (2006), "Bandit based Monte-Carlo planning", *Proceedings of the 17th European Conference on Machine Learning (ECML 2006), Springer, Berlin*, pp. 282-93.

Lakshmivarahan, S. (1981), *Learning Algorithms Theory and Applications*, Springer, Berlin.

Lanctôt, J.K. (1989), "Discrete estimator algorithms: a mathematical model of computer learning", Master's thesis, Department of Mathematics and Statistics, Carleton University, Ottawa.

Lanctôt, J.K. and Oommen, B.J. (1992), "Discretized estimator learning automata", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-22 No. 6, pp. 1473-83.

Misra, S., Oommen, B.J. and Granmo, O.-C. (2007), "Routing bandwidth guaranteed paths in MPLS traffic engineering: a multiple race track learning approach", *IEEE Transactions on Computers*, Vol. 56 No. 7, pp. 959-76.

Mitchell, T.M. (1997), *Machine Learning*, McGraw-Hill, New York, NY.

Najim, K. and Poznyak, A.S. (1994), *Learning Automata: Theory and Applications*, Pergamon Press, Oxford.

Narendra, K.S. and Thathachar, M.A.L. (1989), *Learning Automata: An Introduction*, Prentice-Hall, Englewood Cliffs, NJ.

Obaidat, M.S., Papadimitriou, G.I. and Pomportsis, A.S. (2002), "Learning automata: theory, paradigms and applications", *IEEE Transactions on Systems Man and Cybernetics*, Vol. SMC-32, pp. 706-9.

Oommen, B.J. and Agache, M. (2001), "Continuous and discretized pursuit learning schemes: various algorithms and their comparison", *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 31, pp. 277-87.

Oommen, B.J. and Lanctôt, J.K. (1990), "Discretized pursuit learning automata", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-20 No. 4, pp. 931-8.

Poznyak, A.S. and Najim, K. (1997), *Learning Automata and Stochastic Optimization*, Springer, Berlin.

Sastry, P.S. (1985), *Systems of Learning Automata: Estimator Algorithms Applications*, PhD thesis, Dept of Electrical Engineering, Indian Institute of Science, Bangalore, June.

Sutton, R.S. and Barto, A.G. (1998), *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA.

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

Thompson, W.R. (1933), "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples", *Biometrika*, Vol. 25, pp. 285-94.

Tsetlin, M.L. (1973), *Automaton Theory and Modeling of Biological Systems*, Academic Press, New York, NY.

Tung, B. and Kleinrock, L. (1996), "Using finite state automata to produce self-optimization and self-control", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 7 No. 4, pp. 47-61.

Vermorel, J. and Mohri, M. (2005), "Multi-armed bandit algorithms and empirical evaluation", *Proceedings of the 16th European Conference on Machine Learning (ECML 2005), Porto, Portugal*, Springer, Heidelberg, pp. 437-48.

Wang, T., Lizotte, D., Bowling, M. and Scuurmans, D. (2005), "Bayesian sparse sampling for on-line reward optimization", *Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany*, pp. 956-63.

Wyatt, J. (1997), "Exploration and inference in learning from reinforcement", PhD thesis, University of Edinburgh, Edinburgh.

About the author

Ole-Christoffer Granmo was born in Porsgrunn, Norway on August 26, 1974. He obtained his MSc in 1999 and the PhD degree in 2004, both from the University of Oslo, Norway. He is currently an Associate Professor in the Department of ICT, University of Agder, Norway. His research interests include intelligent systems, stochastic modelling and inference, machine learning, pattern recognition, LA, distributed computing, and surveillance and monitoring. He is the author of more than 30 refereed journal and conference publications. Ole-Christoffer Granmo can be contacted at: ole.granmo@uia.no

This article has been cited by:

1. Richard W. Morris, Amir Dezfouli, Kristi R. Griffiths, Bernard W. Balleine. 2014. Action-value comparisons in the dorsolateral prefrontal cortex control choice between goal-directed actions. *Nature Communications* **5**. . [CrossRef]
2. Pedro A Ortega, Daniel A Braun. 2014. Generalized Thompson sampling for sequential decision-making and causal inference. *Complex Adaptive Systems Modeling* **2**:1, 2. [CrossRef]
3. Xuan Zhang, Ole-Christoffer Granmo, B. John Oommen. 2013. On incorporating the paradigms of discretization and Bayesian estimation to create a new family of pursuit learning automata. *Applied Intelligence* **39**:4, 782-792. [CrossRef]
4. Ole-Christoffer Granmo, Sondre Glimsdal. 2013. Accelerated Bayesian learning for decentralized two-armed bandit based decision making with applications to the Goore Game. *Applied Intelligence* **38**:4, 479-488. [CrossRef]