CS-E4870 Report: Multiarmed Bandit Based Automation of a Marketing Problem

Muhammad Abdullah Khan
muhammad.a.khan@aalto.fi

Supervisor: Neda Barzegar Marvasti

Abstract

This report firstly introduces the idea of Multiarmed Bandit (MAB), a sub-category under reinforcement learning methods. Subsequently, classical MAB problems are briefly discussed. Following this, potential solutions are introduced to overcome the classical problems. These preferred methods are then evaluated by carefully conducting multiple experiments. Experimental results are consequently employed to set up a feasibility study of the methods. Experimental study shows that the Epsilon-Greedy (EG) algorithm needs more time in making informed decisions and also has the tendency to get stuck on the second best reward bandit. In comparison Thompson Sampling (TS) provides the best rewards due to the Bayesian theorem and wins out on competitive algorithms such as the Upper Confidence Bound (UCB) and Stochastic Gradient Ascent (SGA) who also provide relatively good rewards. Hence, the study concludes that for reinforcement learning techniques, a good rate of learning from events in addition to sufficient prior information will yield the maximum reward.

KEYWORDS: MAB, EG, UCB, SGA, TS, PPC, SEO, SEM;

1 Introduction

MAB problems can be categorized as sequential resource allocation tasks, where one or more resources must be chosen wisely and efficiently allocated among competing actions. This must be typically performed in such a fashion so as to maximize the overall expected gain [1]. The main dilemma in these particular problems is to either naturally choose between possible paths that yield instantly the maximum gain currently (*exploitation*) or sacrifice current gain over better future gains (*exploration*). Since strategies for these problems adequately represent a subsection of reinforcement learning methods, the ultimate objective is to achieve the most appropriate balance between exploration and exploitation, consequently maximizing the overall rewards. MAB problems arise in many scientific and technological fields, like manufacturing, website optimization, the effectiveness of medicinal drugs and communication networks, etc.

2 Classical MAB Problems and their Solutions

The first section provides a brief overview of classical MAB problem. The final section suggests five solutions, namely the EG approach with two variants, i.e., one with normal initial estimates and the other with optimistic estimates, UCB, SGA and lastly TS. These methods are presented to address classical MAB problems.

2.1 Classical MAB Problems

The classical MAB problem typically consists of multiple actions or bandits and one resource allocator (also known as a learner). At each instance of time, only an individual action can be allocated a resource; all other actions remain completely frozen. Considering this restriction, the state of each action is also updated only when it is allocated a resource [2]. In the classical multi-armed bandit problem, each of the actions has an expected or mean reward given that one of them is allocated a resource. The *regret* of the allocation can be defined as the difference between the expected cumulative reward a actions can yield and the maximum reward it yields if it is always allocated the resource. The reward gained by each resource allocation to a particular action can be defined as follows:

$$q_*(a) = \mathbb{E}[R_t \mid A_t = a] \tag{1}$$

In the equation above, the action to which a resource is allocated on time step t is denoted by A_t. The corresponding reward equals R_t. The value each allocation a is

denoted by $q_*(a)$ which is the expected reward given that a is selected [3].

Since the value of each allocation is not known beforehand, we assume certain estimates. Hence, for each allocation a, the estimated value of it on time step t is $Q_t(a)$. Ideally, $Q_t(a)$ must be as close to $q_*(a)$ to yield maximum benefit [3].

2.2 Strategies for Classical MAB Problems

This section introduces five strategies to overcome classical MAB problems. Before delving into the intricacies of these strategies, it is critical to interpret the data that will be manipulated to ultimately solve the problem.

For the strategies to be discussed, the *estimated action-value* will represent the real expected rewards over the course of the rounds. For each action a and the elapsed rounds t, the *estimated action-value* will be defined as follows:

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}} \tag{2}$$

From this equation, the *estimated action-value* is equivalent to the sum of all the rewards r_t for action $A_t = a$ prior to the elapsed rounds t over the number of times a was chosen prior to t. Here the reward is bounded as $R_i \in [0,1]$ [5].

According to the Chernoff-Hoeffding bound theorem [6], the difference between the *real mean* and the *estimated action-value* will be equivalent to:

$$|Q_t(a) - \mu| \leq c\sqrt{\frac{\ln(t)}{N_t(a)}} \tag{3}$$

Here, t denotes the rounds that have elapsed and $N_t(a)$ is the number of times that action a has been chosen.

Epsilon-Greedy (EG) Approach

The EG algorithm explores a random action with some probability ε. The algorithm procedure is as follows:

Algorithm 1 Epsilon-Greedy (EG)

for actions a = 1 to k

$Q_1(a) \leftarrow 0$

$N_1(a) \leftarrow 0$

for n = 1 to $total\ rounds$

$$A \leftarrow \begin{cases} argmax_a Q_n(a) & \text{with probability } 1-\varepsilon \\ random\,action & \text{with probability } \varepsilon \end{cases}$$

$R_n \leftarrow bandit(A)$

$N_{n+1}(A) \leftarrow N_n(A) + 1$

$Q_{n+1}(A) \leftarrow Q_n(A) + \frac{1}{N_{n+1}(A)}[R_n - Q_n(A)]$

The goal of this algorithm is to *explore* to get the action with the best action-value estimate and then *exploit* it till such time that another action manages to give a better estimate. At each elapsed round, the reward R_n) is calculated and subsequently the reward estimate $Q_n(a)$ is updated for the chosen action [5].

A variant of the EG algorithm with optimistic initial estimates $Q_n(a)$ ensures that more exploration takes place in the initial phase of the learning process and the agents tend to exploit the best option after a considerable number of time steps. This ensures that the EG algorithm does not get stuck on the second-most optimal action and leads to higher rewards [3].

Upper Confidence Bound (UCB)

The mechanics of the UCB algorithm are to gain the highest reward. It does so by choosing a action which yields the largest *empirical reward* estimate up to a certain point in time and adds a term inversely proportional to the number of times that particular action has been allocated with a resource. The term represents the difference between the real and the estimated action-value as discussed earlier as can be also considered as the *uncertainty* of choosing a certain action. [7]. At each resource allocation, the UCB algorithm assigns each action a after t elapsed rounds:

$$UCB_t(a) = Q_t(a) + c\sqrt{\frac{\ln(t)}{N_t(a)}} \tag{4}$$

The procedure of the UCB algorithm is as follows:

Algorithm 2 Upper Confidence Bound (UCB)

for actions a = 1 to k

$Q_1(a) \leftarrow 0$

$N_1(a) \leftarrow 0$

for i = 1 to k

$R_2 \leftarrow bandit(i)$

$N_2(i) \leftarrow N_1(i) + 1$

$Q_2(i) \leftarrow Q_1(i) + \frac{1}{N_2(i)}[R_2 - Q_1(i)]$

$Q_2(i) \leftarrow Q_1(i) + \frac{1}{N_2(i)}[R_2 - Q_1(i)]$

$UCB_2(i) \leftarrow Q_2(i) + c\sqrt{\frac{\ln(2)}{N_2(i)}}$

for n = k to $total\ rounds$

$A \leftarrow argmax_a UCB_n(a)$

$R_n \leftarrow bandit(A)$

$N_{n+1}(A) \leftarrow N_n(A) + 1$

$Q_{n+1}(A) \leftarrow Q_n(A) + \frac{1}{N_{n+1}(A)}[R_n - Q_n(A)]$

$UCB_{n+1}(A) \leftarrow Q_{n+1}(A) + c\sqrt{\frac{\ln(n)}{N_{n+1}(A)}}$

From the algorithmic steps provided above, it can be clearly seen that each of the actions $a = 1tok$ are chosen once at the start as they are considered *maximizing actions* when $N_n(a) = 0$. The constant variable $c > 0$ controls the degree of exploration. [7]. In this algorithm, the reward R_n and estimated action-values for each action $Q_n(a)$ at round n, are updated in the same fashion as in EG. The UCB value $UCB_n(A)$ is calculated for the chosen action A by adding an *uncertainty*, i.e., $UCB_n(A) = Q_n(A) + c\sqrt{\frac{\ln(n-1)}{N_n(A)}}$ to the action-value estimate until n elapsed rounds which decreases with each choice of that particular action. The highest UCB is chosen iteratively, thus ensuring the right balance of exploration and exploitation.

Stochastic Gradient Ascent (SGA)

The SGA algorithm aims not only for the highest *empirical reward* estimate, but also keeps the previous selections of each action in contention to calculate numerical *preference* for all the actions. These *preferences* are initially equivalent, i.e., $H_1(a) = 0$ for all actions $a = 1tok$ [3]. Subsequently, at each time step, the probability of *preference* for the chosen and non-chosen actions are calculated. At each resource allocation, the SGA algorithm selects the action with the highest probability of preference, i.e., $\pi_n(a)$:

$$Pr\{A_t = a\} = \frac{e^{H_t(a)}}{\sum_{b=1,k} e^{H_t(b)}} \tag{5}$$

The procedure of the SGA algorithm is as follows:

Algorithm 3 Stochastic Gradient Ascent (SGA)

for actions a = 1 to k

$H_1(a) \leftarrow 0$

$\pi_1(a) \leftarrow 0$

$TR \leftarrow 0$

for n = 1 to $total\ rounds$

$A \leftarrow random\ action\ choice\ with\ \pi_n(a)$

$R_n \leftarrow bandit(A)$

$N_{n+1}(A) \leftarrow N_n(A) + 1$

$TR \leftarrow TR + R_n$

$\hat{R}_n \leftarrow \frac{TR}{n}$

$H_{n+1}(A) \leftarrow H_n(A) + \alpha(R_n - \hat{R}_n)(1 - \pi_n(A))$

$H_{n+1}(a) \leftarrow H_n(a) - \alpha(R_n - \hat{R}_n)\pi_n(a)$ for actions $a \neq A$

$\pi_{n+1}(a) \leftarrow \frac{e^{H_{n+1}(a) - max(H_{n+1}(a))}}{\sum_{b=1,k} e^{H_{n+1}(b) - max(H_{n+1}(b))}}$ for all actions a

From the algorithmic steps provided above, it can be observed that the reward at each elapsed round R_n is generated in the same way as for EG and UCB. The value $\pi_n(a)$ represents the probability of preference of each action a at time-step n. Furthermore, the value $H_n(a)$ represents the *learning preference* for each action at n elapsed rounds. The *learning preference* of the chosen action A is updated as follows at n elapsed rounds: $H_{n+1}(A) = H_n(A) + \alpha(R_n - \hat{R}_n)(1 - \pi_n(A))$ where $\hat{R}_n$ is the average reward gained by action A till n elapsed rounds and α denotes a constant time-step for each action. Similarly, the *learning preference* for the non-chosen actions $a \neq A$ is updated as follows: $H_{n+1}(a) = H_n(a) - \alpha(R_n - \hat{R}_n)\pi_n(a)$. The probability preference is then updated as follows based on the new learning preferences: $\pi_{n+1}(a) = \frac{e^{H_{n+1}(a) - max(H_{n+1}(a))}}{\sum_{b=1,k} e^{H_{n+1}(b) - max(H_{n+1}(b))}}$ for all actions a and consequently, the action with the highest probability is chosen, thus ensuring proper exploration in addition to significant exploitation [3].

Thompson Sampling (TS)

The TS algorithm considers a *prior* belief for obtaining the maximum reward out of the list of available actions. Before proceeding to the details of this algorithm, it is necessary to highlight Bayesian Learning. Keeping the problem simple, let us assume a Bernoulli distribution where the random variable can take values of 1 with probability μ and 0 with probability $1 - \mu$. Based on this, the prior distribution can

be defined as follows [8]:

$$p(x) = Pr[\mu = x] \tag{6}$$

Using Bayes theorem, the *posterior* distribution can be determined after observation of a few rounds/trials as follows:

$$Pr[\mu = x \mid D] \propto Pr[D \mid \mu = x] * p(x) \tag{7}$$

In the equation, $Pr[D \mid \mu = x]$ represents the probability of generating data D from the Bernoulli distribution with parameter x, also known as the *likelihood* [8, 9]. Considering a Beta prior, the posterior can be defined as a Beta distribution in terms of α and β as follows:

$$Pr[\mu = x \mid D] \propto Beta_{\alpha+r,\beta+1-r}(\theta) \tag{8}$$

In the equation above, r is the reward bounded as $r \in \{0,1\}$. The Beta posterior can then be written as $Beta(S_n+1, F_n+1)$, where S_n and F_n correspond to the number of 1s and 0s (rewards and regret accumulated) observed in the n elapsed rounds respectively. The *estimated mean* $\hat{\mu}$ and *variance* σ^2 for the distribution can be defined as follows:

$$\hat{\mu} = \frac{S_n+1}{n+2}, \;\; \sigma^2 = \frac{\hat{\mu}(1-\hat{\mu})}{n+2} \tag{9}$$

The *estimated mean* $\hat{\mu}$ converges to the mean of the Bernoulli distribution μ and the *variance* σ^2 also decreases as the number of rounds n increases, therefore ensuring more accuracy in predicting the reward with each round [8].

Using all this information, the procedure of the TS algorithm is as follows:

Algorithm 4 Thompson Sampling (TS) (Bernoulli)

for actions a = 1 to k

$S_1(a) \leftarrow 0$

$F_1(a) \leftarrow 0$

$\theta_1(a) \leftarrow 0$

for n = 1 to $total\ rounds$

$A \leftarrow argmax_a \theta_n(a)$

$R_n \leftarrow bandit(A)$

if $R_n == 1$

$S_{n+1}(A) \leftarrow S_n(A) + 1$

else

$F_{n+1}(A) \leftarrow F_n(A) + 1$

$\theta_n(A) \sim Beta(S_{n+1}(A), F_{n+1}(A))$

Here, A is the specific action chosen for resource allocation. $S_{n+1}(A)$ and $F_{n+1}(A)$ are the occurrences of 1s and 0s for all resource allocations to the action A after n elapsed rounds respectively. Furthermore $\theta_n(A)$ is the posterior calculated based on the previous observations for the chosen action A and with each round, the maximum value of $\theta_n(a)$ for all actions a to ensure an informed decision in terms of exploring the best action and then exploiting it as much as possible. [8].

3 Experimental Setup, Results and Analysis

This section is concerned mainly with the set up of the experimental phase of testing the MAB techniques mentioned in the previous sections. It comprises of two sub-sections. The first one introduces the experimental environment including but not limited to the problem statement and variables of interest. The second and final sub-section highlights the results obtained from the iterative tests, in addition to the reasons between the performance of each algorithm obtained from the results in terms of the *average reward* and the *optimal action choice percentage* from all the choices made.

3.1 Experimental Setup

For the experimental setup, the reinforcement learning model has been followed for testing five algorithms, i.e., the EG basic variant, EG variant with initial optimistic estimates, UCB, SGA and TS algorithms. The bandit problem in this scenario is in the form of choosing the website which has the highest likelihood of obtaining

a click. The motivation for this problem is the competition between Pay-Per-Click (PPC) websites which use various techniques such as designing, user experience, Search Engine Optimization (SEO) and Search Engine Marketing (SEM) to increase the likelihood of clicks to get the maximum reward.

In this scenario, 5 websites with different probabilities of receiving a click (unknown before the experiment) have been defined which are as follows:

Website #	Reward Probability (%)
1	32
2	24.5
3	8.7
4	64.5
5	78.8

Table 1. This table depicts the probabilities of receiving a click for the websites.

For preserving simplicity, reward in the experiments has been represented by a Binomial distribution for highlighting the success or failure in receiving a click for each website. An *environment* holds all the data such as the total number of *trials* for each experiment, the competing *actions*, which in this case are the websites and their total *rewards*.

The *agent* which in this scenario is the employed algorithm, learns from prior data supplied by the *environment* to make choices between the websites in order to yield the highest reward through a mix and match of *exploration* and *exploitation*.

For the experiment, random samples are taken from a Binomial distribution and the resulting reward is either 0 or 1 depending upon the random sample compared to the real reward probabilities. The total number of *trials* per experiment is 10000 and the total of experiments conducted for each *agent* is 100. The *average reward* and percentage of the *optimal choices made* are plotted at the end of the experimental phase to aid the comparative analysis of the different techniques.

3.2 Results and Analysis

During the experiments, the choices of websites made by the different MAB techniques were documented and the techniques were then compared through two metrics, i.e., the *optimal action percentage* and the *average reward* accumulated by each of them over the course of the 100 experiments. The *efficiency* and *effectiveness* of each of the techniques was measured by the rate of growth and the quantitative values of these two metrics respectively.

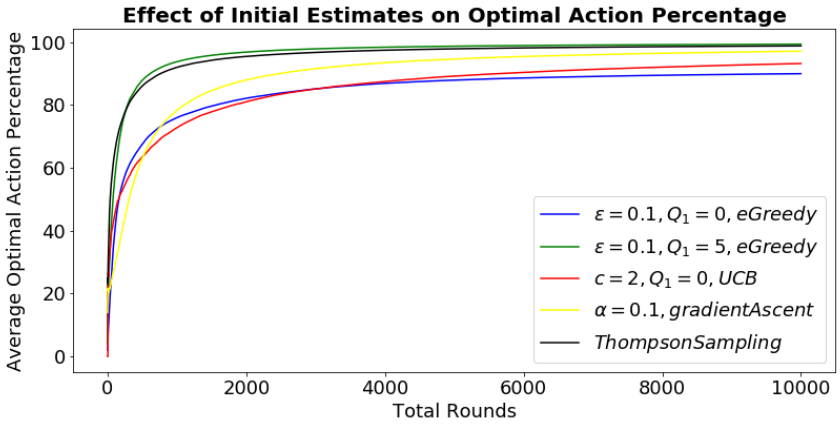

Figure 1. This figure provides total percentage of the times the optimal action was chosen over the 100 experiments.

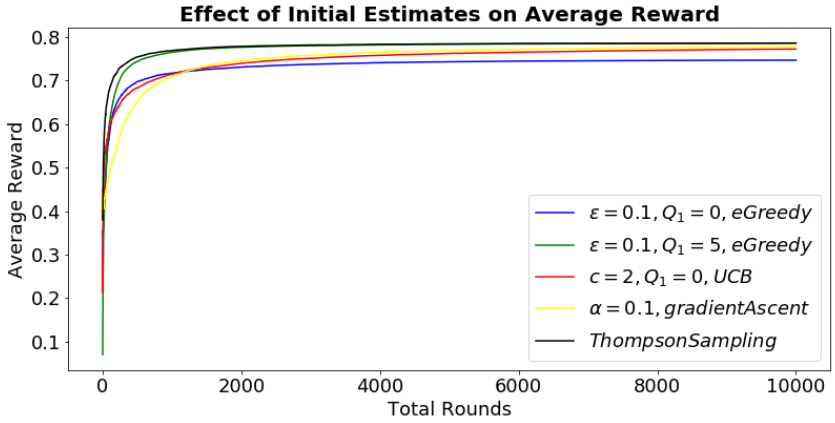

Figure 2. This figure provides the overall average rewards accumulated by the algorithms over the 100 experiments.

As apparent from the graph the EG variant with the optimistic initial estimates and the TS algorithms perform relatively well as compared to the rest of the algorithms. The EG variant with normal initial estimates performs well initially as compared to the UCB and SGA algorithms but gradually accumulates *regret* due to the fact that constant exploitation after insufficient exploration leads to a sub-optimal action/web-site being chosen most of the time.

The UCB algorithm after choosing all the actions exactly once also considers the *uncertainty* in the reward estimates for each action. As the time steps and the count of each chosen action progresses, the uncertainty in the reward estimates tends to decreases in thus, it is trivial to choose the action with the highest reward estimate consistently through the experiments. This is the reason why UCB starts slowly but improves significantly in performance once the uncertainty in the reward estimates has decreased.

The SGA algorithm considers the action choices solely on basis of the preference

probability of each action in a given point in time. At first, all the actions have an equal probability of preference but after each selection with every time-step, the chosen action receive a boost in preference based on its accumulated and mean reward. Similarly, the unchosen actions receive a decrease in their preference. This procedure continues until the action ensuring the best average reward and the highest probability of preference is exploited most of the time. Hence the SGA algorithm performs better than the UCB algorithm.

The EG variant with the optimistic initial estimates performs much better than the three algorithms described earlier as high initial estimates ensure the fact that the agent continues to explore as much as possible to verify the optimal choice of action instead of exploiting without sufficient knowledge and building up regret as in the case of the earlier EG variant. This eventually ensures a proper balance of explorations and exploitation hence a better overall reward.

Lastly, TS is the most superior of all the techniques mentioned and its performance completely endorses this statement. A Bayesian approach continuously builds upon the prior knowledge of the gained rewards at each time step as well as the total regret accumulated. This eventually leads to the agent making highly informed choices based on the previous history and hence the optimal choice is selected most of the time resulting in a higher average reward.

4 Conclusion

From the multiple experiments conducted on the MAB problem involving PPC websites, we have deduced that classical MAB techniques such as the EG algorithm with optimistic initial estimates and TS provide exceptional results. As highlighted by the experimental results, both techniques manage to mostly exploit the website with the highest probability. Furthermore, both techniques also auto-optimize themselves through a good balance of exploration and exploitation of the available choices of websites without sufficient prior knowledge. A head-to-head comparison highlights the fact that TS always exploits the best website, unlike the EG algorithm which sometimes fails to recognize the best website and continues exploiting the website with the second highest probability. Since the context of the experiment is a very confined one, i.e., not involving rewards in a Gaussian distribution in addition to the number of actions being a handful, it is early to comment on which MAB technique would be best-suited for a given scenario. But concerning the problem statement provided, TS clearly performs relatively better than the rest of the techniques due to its superior learning capability.

References

[1] Multi-armed Bandit. https://en.wikipedia.org/wiki/Multi-armed_bandit.

[2] Aditya Mahajan and Demosthenis Teneketzis. Multi-armed bandit problems. In *Foundations and Applications of Sensor Management*, pages 121–151. Springer, 2008.

[3] Richard S Sutton, Andrew G Barto, Francis Bach, et al. *Reinforcement learning: An introduction*. 1998.

[4] Li Zhou. A survey on contextual multi-armed bandits. *arXiv preprint arXiv:1508.03326*, 2015.

[5] Stochastic Multiarmed Bandit (IID model). https://agrawal.wikischolars.columbia.edu/file/view/Lecture%202.pdf/573474319/Lecture%202.pdf.

[6] Chernoff Bound. https://en.wikipedia.org/wiki/Chernoff_bound.

[7] UCB Algorithm, Worst-Case Regret Bound. https://agrawal.wikischolars.columbia.edu/file/view/Lecture%203%20part%201.pdf/573454731/Lecture%203%20part%201.pdf.

[8] Thompson Sampling. https://agrawal.wikischolars.columbia.edu/file/view/Lecture%204.pdf/574197747/Lecture%204.pdf.

[9] Bayes' Theorem. https://en.wikipedia.org/wiki/Bayes%27_theorem.