DATA SHEET

Department of Computer Science and Engineering

1. **Name : Subhojyoti Mukherjee**

2. **Roll No : CS15S300**

3. **Registered for : MS**

4. **Specialization : Computer Science and Engineering**

5. **Category : Regular**

6. **Guide(s) : Balaraman Ravindran(CSE), Nandan Sudarsanam(DoMS)**

7. **Date of Joining : 6.1.2015**

8. **Date of Registration : 6.1.2015**

9. **Area of Research : Reinforcement Learning, Online Learning, Bandits**

Details of Course Work

Sl. No.	**Course No.**	**Course Title**	**Semester/ Year**	**Core/ Elective**	**Credit**	**Grade**
1	ID6020	Introduction to Research	01	PPF	2	P
2	ID6021	Introduction to Research	01	PPF	2	P
3	CH5440	Multivariate Data Analysis for Process Modeling	01	Elective	3	C
4	MS6320	Fundamentals of Experimentation in Management	01	Core	2	A
5	CS5011	Introduction to Machine Learning	02	Core	4	B
6	MS6031	Data Analysis for Research	02	Core	2	A
7	CS6370	Natural Language Processing	02	Elective	4	A
8	CS6015	Linear Algebra and Random Processes	02	Elective	4	C
9	CS6770	Reinforcement Learning	03	Core	4	A

Seminar Dates: N/A

Signature of the Scholar

Signature of the Advisor
(Dr. Balaraman Ravindran)

Progress Report

1 Introduction

In today's world a large number of problems in science and engineering, robotics and game playing, resource management, financial portfolio management, medical treatment design, ad placement, website optimization and packet routing can be modeled as sequential decision-making under uncertainty. Many of these real-world interesting sequential decision-making problems can be formulated as reinforcement learning (RL) problems ([BT96], [SB98]). In an RL problem, an agent interacts with a dynamic, stochastic, and unknown environment, with the goal of finding an action-selection strategy or policy that optimizes some long-term performance measure. Every time when the agent interacts with the environment it receives a signal/reward from the environment based on which it modifies its policy. The agent learns to optimize the choice of actions over several time steps which is learned from the sequences of data that it receives from the environment. This is the crux of online sequential learning. This is in contrast to supervised learning methods that deal with labeled data which are independently and identically distributed (i.i.d.) samples from the domain and train some classifier on the entire training dataset to learn the pattern of this distribution to predict future samples (test dataset) with the assumption that it is sampled from the same domain, whereas the RL agent learns from the samples that are collected from the trajectories generated by its sequential interaction with the system. For an RL agent the trajectory consists of a series of sequential interactions whereby it transitions from one state to another following some dynamics intrinsic to the environment while collecting the reward till some stopping condition is reached. This is known as an episode. For a single-step interaction, i.e., when the episode terminates after a single transition, the problem is captured by the multi-armed bandit (MAB) model. Our work will focus on this idea of MAB model.

2 Motivation

The MAB model fits very well in various real-world scenarios that can be modeled as sequential decision-making problems. Some of which are mentioned as follows:-

1. *Online Shop Domain ([GMP$^+$15]):* In the online shop domain, a retailer aims to maximize profit by sequentially suggesting products to online shopping customers. In this scenario, at

every timestep, the retailer displays an item to a customer from a pool of items which has the highest probability of being selected by the customer. The episode ends when the customer selects or does not select a product (which will be considered as a loss to the retailer) and the process is again repeated till a pre-specified number of times with the retailer gathering valuable information regarding the customer from this behaviour and modifying its policy to display the next item.

2. *Medical Treatment Design ([Tho33]):* Here at every timestep, the agent chooses to administer one out of several treatments sequentially on a patient. Here, the episode ends when the patient responds well or does not respond well to the treatment whereby the agent modifies its policy for the next suggestion.

3. *Financial Portfolio Management:* In financial portfolio management MAB model can be used. Here, the agent is faced with the choice of selecting the most profitable stock option out of several stock options. The simplest strategy where we can employ a bandit model is this; at the start of every trading session the agent suggests a stock to purchase worth Re 1, while at the closing of the trading session it sells off the stock to witness its value after a day's trading. The profit recorded is treated as the reward revealed by the environment and the agent modifies its policy for the next day.

The thresholding bandit problem is a special case of combinatorial MAB problem where the learner has to suggest the best set of arms above a real valued threshold. This has several relevant industrial applications. The variants of TopM problem (identifying the best M arms from K given arms) can be readily used in the thresholding problem.

1. *Product Selection:* A company wants to introduce a new product in market and there is a clear separation of the test phase from the commercialization phase. In this case the company tries to minimize the loss it might incur in the commercialization phase by testing as much as possible in the test phase. So from the several variants of the product that are in the test phase the learning agent must suggest the product variant(s) that are above a particular threshold τ at the end of the test phase that have the highest probability of minimizing loss in the commercialization phase. A similar problem has been discussed for single best product variant identification without threshold in [BMS11].

2. *Mobile Phone Channel Allocation:* Another similar problem as above concerns channel allocation for mobile phone communications ([AMS09]). Here there is a clear separation be-

tween the allocation phase and communication phase whereby in the allocation phase a learning algorithm has to explore as many channels as possible to suggest the best possible set of channel(s) that are above a particular threshold τ. The threshold depends on the subscription level of the customer. With higher subscription the customer is allowed better channel(s) with the τ set high. Each evaluation of a channel is noisy and the learning algorithm must come up with the best possible suggestion within a very small number of attempts.

3. *Anomaly Detection and Classification:* Thresholding bandit can also be used for anomaly detection and classification where we define a cutoff level τ and for any samples above this cutoff gets classified as an anomaly. For further reading we point the reader to section 3 of [LGC16].

In all the above examples the MAB model performs well mainly because all of them suffer from *exploration-exploitation dilemma*. This is characterized by action-selection choice faced by the agent where it must decide whether to stay with the action yielding highest reward till now or to explore newer actions which might be more profitable in the long run. MAB's are suited for such scenarios because

1. They are easy to implement.
2. The switch between exploration and exploitation is more well defined theoretically.
3. They perform well empirically.

3 Problem Definition

We formally define the MAB model in this section. We will mainly focus on the stochastic MAB model in our work. In this setting, the learning algorithm is provided with a set of decisions (or arms) with reward distributions unknown to the algorithm. These decisions can be the items to show the customers, or medical treatment, or the stock options. The learning proceeds in an iterative fashion, where in each timestep, the algorithm chooses an arm and receives a stochastic reward that is drawn from a stationary distribution specific to the arm selected. There are two types of goal we will be focusing our work on:-

1. Maximizing cumulative reward over the entire time horizon
2. Minimizing probability of error in suggesting the best arm(s) at any timestep

Given the goal of maximizing the cumulative reward, the learning algorithm faces the exploration-exploitation dilemma, i.e., in each round should the algorithm select the arm which has the highest observed mean reward so far (*exploitation*), or should the algorithm choose a new arm to gain more knowledge of the true mean reward of the arms and thereby avert a sub-optimal greedy decision (*exploration*).

Formally, let r_i, $i = 1, \dots, K$ denote the mean rewards of the K arms and $r^* = \max_i r_i$ the optimal mean reward. The objective in the stochastic bandit problem is to maximize cumulative reward by minimizing the cumulative regret, which is defined as follows:

$$R_T = r^* T - \sum_{i \in A} r_i N_i(T),$$

where T is the number of rounds, $N_i(T) = \sum_{m=1}^{T} I(I_m = i)$ is the number of times the algorithm chose arm i up to round T. The expected regret of an algorithm after T rounds can be written as

$$\mathbb{E}[R_T] = \sum_{i=1}^{K} \mathbb{E}[N_i(T)] \Delta_i,$$

where $\Delta_i = r^* - r_i$ denotes the gap between the means of the optimal arm and of the i-th arm.

In the pure exploration thresholding bandit setup the goal is different than minimizing the cumulative regret. Here the learning algorithm is provided with a threshold τ and it has to output all such arms i whose mean of reward distribution r_i is above τ after T rounds. This is a specific instance of combinatorial pure exploration where the learning algorithm can explore as much as possible given a fixed horizon T and not be concerned with the usual exploration-exploitation dilemma. Let A be the set of all arms. Formally we can define a set $S_\tau = \{i \in A : r_i \geq \tau\}$ and the complementary set $S_\tau^C = \{i \in A : r_i < \tau\}$. Also we define $\hat{S}_\tau = \hat{S}_\tau(T) \subset A$ and its complementary set $\hat{S}_\tau^C$ as the recommendation of the learning algorithm after T rounds. Given such sets exists, the performance of the learning agent is measured by how much accuracy it can discriminate between S_τ and S_τ^C after time horizon T. The loss $\mathcal{L}$ is defined as:-

$$\mathcal{L}(T) = I\big(\{S_\tau \cap \hat{S}_\tau^C \neq \emptyset\} \cup \{\hat{S}_\tau \cap S_\tau^C \neq \emptyset\}\big)$$

The goal of the learning agent is to minimize $\mathcal{L}(T)$. So, the expected loss after T rounds is

$$\mathbb{E}[\mathcal{L}(T)] = \mathbb{P}\big(\{S_\tau \cap \hat{S}_\tau^C \neq \emptyset\} \cup \{\hat{S}_\tau \cap S_\tau^C \neq \emptyset\}\big)$$

which we can say is the probability of making mistake, that is whether the learning agent at the end of round T rejects arms from S_τ or accepts arms from S_τ^C in its final recommendation. Also, we are looking at an anytime algorithm, so the knowledge of T may not be known to the learner.

4 Literature Review

An early work involving a bandit setup is [Tho33], where the author deals the problem of choosing between two treatments to administer on patients who come in sequentially. Following the seminal work of [Rob52], bandit algorithms have been extensively studied in a variety of applications. From a theoretical standpoint, an asymptotic lower bound for the regret was established in [LR85]. In particular, it was shown that for any consistent allocation strategy, we have $\liminf_{T\to\infty} \frac{\mathbb{E}[R_T]}{\log T} \geq \sum_{\{i:r_i<r^*\}} \dfrac{(r^*-r_i)}{D(p_i\|p^*)}$, where $D(p_i\|p^*)$ is the Kullback-Leibler divergence between the reward densities p_i and p^*, corresponding to arms with mean r_i and r^*, respectively.

There have been several algorithms with strong regret guarantees. The foremost among them is UCB1 by [ACBF02], which has a regret upper bound of $O\left(\dfrac{K\log T}{\Delta}\right)$, where $\Delta = \min_{i:\Delta_i>0}\Delta_i$. This result is asymptotically order-optimal for the class of distributions considered. However, the worst case gap independent regret bound of UCB1 can be as bad as $O\left(\sqrt{TK\log T}\right)$. In [AB09], the authors propose the MOSS algorithm and establish that the worst case regret of MOSS is $O\left(\sqrt{TK}\right)$ which improves upon UCB1 by a factor of order $\sqrt{\log T}$. However, the gap-dependent regret of MOSS is $O\left(\dfrac{K^2\log\left(T\Delta^2/K\right)}{\Delta}\right)$ and in certain regimes, this can be worse than even UCB1 (see [AB09],[Lat15]). The UCB-Improved algorithm, proposed in [AO10], is a round-based algorithm[1] variant of UCB1 that has a gap-dependent regret bound of $O\left(\dfrac{K\log T\Delta^2}{\Delta}\right)$, which is better than that of UCB1. On the other hand, the worst case regret of UCB-Improved is $O\left(\sqrt{TK\log K}\right)$.

In the pure exploration setup, a significant amount of research has been done on finding the best arm(s) from a set of arms. The pure exploration setup has been explored in mainly two settings:-

1. Fixed Budget setting: In this setting the learning algorithm has to suggest the best arm(s) within a fixed number of attempts that is given as an input. The objective here is to maximize the probability of returning the best arm(s). One of the foremost papers to deal with single best arm identification is [AMS09] where the authors come up with the algorithm UCBE and Successive Reject(SR) with simple regret guarantees. The relationship between cumulative regret and simple regret is proved in [BMS11] where the authors prove that minimizing the simple regret necessarily results in maximizing the cumulative regret. In the combinatorial fixed budget setup [GGLB11] come up with Gap-E and Gap-EV algorithm which suggests the best m (given as input) arms at the end of the budget with high probability. Similarly,

[1]An algorithm is *round-based* if it pulls all the arms equal number of times in each round and then proceeds to eliminate one or more arms that it identifies to be sub-optimal.

[BWV13] comes up with the algorithm Successive Accept Reject(SAR) which is an extension of the SR algorithm. SAR is a round based algorithm whereby at the end of round an arm is either accepted or rejected based on certain conditions till the required top m arms are suggested at the end of the budget with high probability.

2. Fixed Confidence setting: In this setting the the learning algorithm has to suggest the best arm(s) with a fixed (given as input) confidence with as less number of attempts as possible. The single best arm identification has been handled in [EDMM06] where they come up with an algorithm called Successive Elimination (SE) which comes up with an arm that is ϵ close to the optimal arm. In the combinatorial setup recently [KTAS12] have suggested the LUCB algorithm which on termination returns m arms which are atleast ϵ close to the true top m arms with $1 - \delta$ probability.

Apart from these two settings some unified approach has also been suggested in [GGL12] which proposes the algorithms UGapEb and UGapEc which can work in both the above two settings. A similar combinatorial setup was also explored in [CLK^{+}14] where the authors come up with more similarities and dissimilarities between these two settings in a more general setup. In their work, the learning algorithm, called Combinatorial Successive Accept Reject (CSAR) is similar to SAR with a more general setup. The thresholding bandit problem is a specific instance of the pure exploration setup of [CLK^{+}14]. In the latest work in [LGC16] the algorithm Anytime Parameter-Free Thresholding (APT) algorithm comes up with a better anytime guarantee than CSAR for the thresholding bandit problem.

5 Objective and Scope

The main objectives of our work.

1. UCB-Improved has several shortfalls. It conducts too much early exploration and the arm elimination is very conservative. Our proposed algorithm tries to remedy this. We also explore the idea of employing clustering techniques in MAB which, till now has only been studied in the contextual bandit setup (see [LCLS10], [BJM12], [CBGZ13] , [GLZ14], [NL14]). We propose an algorithm that minimizes cumulative regret in the multi-armed stochastic bandit setup using clustering and improved exploration and performs at par with the best algorithms currently available. We also show that there is a distinct advantage of using clustering in action elimination type of algorithms in MABs.

2. We also propose an algorithm that minimizes probability of error in a combinatorial stochastic bandit setup whereby the learning algorithm, provided with a threshold τ, proposes the best set of arms such that $r_i \geq \tau$ at every timestep using action elimination method. This problem is similar to the pure exploration setup with a major difference being that the algorithm has to suggest as many arms whose r_i is above τ which may be more than one arm. Our proposed methodology is in contrast with APT which uses an UCB1 type of strategy to suggest the best arms above the threshold whereas we employ an UCB-Improved type of strategy with better exploration parameters.

3. In both the above scenarios we intend to explore both the theoretical guarantees and empirical performance and verify against the best algorithms available in the field.

6 Proposed Methodology

We propose a variant of UCB algorithm, henceforth referred to as ClusUCB, that incorporates clustering and an improved exploration scheme. ClusUCB is a round-based algorithm that starts with a partition of the arms into small clusters, each having same number of arms. The clustering is done at the start with a pre-specified number of clusters. Each round of ClusUCB involves both (individual) arm elimination as well as cluster elimination.

The clustering of arms provides two benefits. First, it creates a context where UCB-Improved like algorithm can be run in parallel on smaller sets of arms with limited exploration, which could lead to fewer pulls of sub-optimal arms with the help of more aggressive elimination of sub optimal arms. Second, the cluster elimination leads to whole sets of sub-optimal arms being simultaneously eliminated when they are found to yield poor results. These two simultaneous criteria for arm elimination can be seen as borrowing the strengths of UCB-Improved as well as other popular round based approaches.

The motivation for our work stems from the remark in Section 2.4.3 of [BCBL12], where the authors conjecture that one should be able to obtain a bandit algorithm with a gap-dependent regret bound that is better than MOSS ([AB09]) and UCB-Improved ([AO10]), in particular, with a regret bound of the order $O\left(\frac{K\log(\frac{T}{H})}{\Delta}\right)$, where $H = \sum_{i:\Delta_i>0}\frac{1}{\Delta_i^2}$. While ClusUCB does not achieve the conjectured regret bound, the theoretical analysis establishes that the gap-dependent regret of ClusUCB is always better than that of UCB-Improved and better than that of MOSS when $\sqrt{\frac{e}{T}} \leq \Delta \leq 1$ (see Table 1). Moreover, the gap-independent bound of ClusUCB is of the

Table 1: Gap-dependent regret bounds for different bandit algorithms

Algorithm	Upper bound
UCB1	$O\left(\frac{K\log T}{\Delta}\right)$
UCB-Improved	$O\left(\frac{K\log(T\Delta^2)}{\Delta}\right)$
MOSS	$O\left(\frac{K^2\log\left(T\Delta^2/K\right)}{\Delta}\right)$
ClusUCB	$O\left(\frac{K\log\left(\frac{T\Delta^2}{\sqrt{\log(K)}}\right)}{\Delta}\right)$

same order as UCB-Improved, i.e., $O\left(\sqrt{KT\log K}\right)$. However, ClusUCB is not able to match the gap-independent bound of $O(\sqrt{KT})$ for MOSS.

The algorithm. As mentioned in a recent work [LT16], UCB-Improved has two shortcomings:
(i) A significant number of pulls are spent in early exploration, since each round m of UCB-Improved involves pulling every arm an identical $n_m = \left\lceil \frac{2\log(T\epsilon_m^2)}{\epsilon_m^2} \right\rceil$ number of times. The quantity ϵ_m is initialized to 1 and halved after every round.
(ii) In UCB-Improved, arms are eliminated conservatively, i.e, only after $\epsilon_m < \frac{\Delta_i}{2}$, the sub-optimal arm i is discarded with high probability. This is disadvantageous when K is large and the gaps are identical ($r_1 = r_2 = \cdots = r_{K-1} < r^*$) and small.
To reduce early exploration, the number of times n_m each arm is pulled per round in ClusUCB is lower than that of UCB-Improved and also that of Median-Elimination, which used $n_m = \frac{4}{\epsilon^2}\log\left(\frac{3}{\delta}\right)$, where ϵ, δ are confidence parameters. To handle the second problem mentioned above, ClusUCB partitions the larger problem into several small sub-problems using clustering and then performs local exploration aggressively to eliminate sub-optimal arms within each clusters with high probability.

As described in the pseudocode in Algorithm 1, ClusUCB begins with a initial clustering of arms that is performed by random uniform allocation. The set of clusters S thus obtained satisfies $|S| = p$, with individual clusters having a size that is bounded above by $\ell = \left\lceil \frac{K}{p} \right\rceil$. Each round of ClusUCB involves both individual arm as well as cluster elimination conditions. These elimination conditions are inspired by UCB-Improved. Notice that, unlike UCB-Improved, there is no longer

Algorithm 1 ClusUCB

```
**Input:** Number of clusters $p$, time horizon $T$, exploration parameters $\rho_a$, $\rho_s$ and $\psi$.
**Initialization:** Set $B_0 := A$, $S_0 = S$ and $\epsilon_0 := 1$.
Create a partition $S_0$ of the arms at random into $p$ clusters of size up to $\ell = \left\lceil \frac{K}{p} \right\rceil$ each.
**for** $m = 0, 1, .. \left\lfloor \frac{1}{2}\log_2 \frac{7T}{K} \right\rfloor$ **do**
    Pull each arm in $B_m$ so that the total number of times it has been pulled is $n_m = \left\lceil \frac{2\log(\psi T \epsilon_m^2)}{\epsilon_m} \right\rceil$.
    ***Arm Elimination***
        For each cluster $s_k \in S_m$, delete arm $i \in s_k$ from $B_m$ if
        $$\hat{r}_i + \sqrt{\frac{\rho_a \log(\psi T \epsilon_m^2)}{2n_m}} < \max_{j \in s_k} \left\{ \hat{r}_j - \sqrt{\frac{\rho_a \log(\psi T \epsilon_m^2)}{2n_m}} \right\}$$
    ***Cluster Elimination***
        Delete cluster $s_k \in S_m$ and remove all arms $i \in s_k$ from $B_m$ if
        $$\max_{i \in s_k} \left\{ \hat{r}_i + \sqrt{\frac{\rho_s \log(\psi T \epsilon_m^2)}{2n_m}} \right\} < \max_{j \in B_m} \left\{ \hat{r}_j - \sqrt{\frac{\rho_s \log(\psi T \epsilon_m^2)}{2n_m}} \right\}.$$
    Set $\epsilon_{m+1} := \frac{\epsilon_m}{2}$
    Set $B_{m+1} := B_m$
    Stop if $|B_m| = 1$ and pull $i \in B_m$ till $T$ is reached.
**end for**
```

a single point of reference based on which we are eliminating arms. Instead we now have as many reference points to eliminate arms as number of clusters formed.

The exploration regulatory factor ψ governing the arm and cluster elimination conditions in ClusUCB is more aggressive than that in UCB-Improved. With appropriate choices of ψ, ρ_a and ρ_s, we can achieve aggressive elimination even when the gaps Δ_i are small and K is large.

In [LT16], the authors recommend incorporating a factor of d_i inside the log-term of the UCB values, i.e., $\max\{\hat{r}_i + \sqrt{\frac{d_i \log T\epsilon_m^2}{2n_m}}\}$. The authors there examine the following choices for d_i: $\frac{T}{z_i}$, $\frac{\sqrt{T}}{z_i}$ and $\frac{\log T}{z_i}$, where z_i is the number of times an arm i has been sampled. Unlike [LT16], we employ cluster as well as arm elimination and establish from a theoretical analysis that the choice $\psi = \frac{T}{196\log(K)}$ helps in achieving a better gap-dependent regret upper bound for ClusUCB as compared to UCB-Improved and MOSS.

We also approach the threshold bandit setup with a similar type of algorithm. The algorithm Augmented UCB is presented below:-

Algorithm: In algorithm 2, hence referred to as AugUCB, we have two exploration parameters, ρ_μ and ρ_v which are the arm elimination parameters. ψ_m is the exploration regulatory factor. The main approach is based on UCB-Improved with modifications suited for the thresholding bandit problem. The active set B_0 is initialized with all the arms from $\mathcal{A}$. We divide the entire budget T into rounds/phases as like UCB-Improved, CCB, SAR and CSAR. At every timestep AugUCB checks for arm elimination conditions and update parameters after finishing a round. As suggested by [LT16] to make AugUCB an anytime algorithm and to overcome too much early exploration, we no longer pull all the arms equal number of times in each round but pull the arm that minimizes, $\min_{i\in B_m}\left\{|\hat{r}_i - \tau| - 2\sqrt{\frac{\rho_v\psi_m\hat{V}_i\log(T\epsilon_m)}{4n_i} + \frac{\rho_v\psi_m\log(T\epsilon_m)}{4n_i}}\right\}$ in the active set B_m. This condition makes it possible to pull the arms closer to the threshold τ and with suitable choice of ρ_μ and ρ_v we can fine tune the exploration. Also because of the said condition, like [LT16] we also claim that AugUCB is an anytime algorithm. The choice of exploration factor $\psi_m = \frac{T\epsilon_m}{(\log(\frac{3}{16}K\log K))^2}$ comes directly from [AB10] and [BMS11] which states that in pure exploration setup, the exploring factor must be linear in T to give us an exponentially small probability of error rather than logarithmic in T which is suited for minimizing cumulative regret.

7 Work Done

1. On four synthetic setups with small gaps, we observe empirically that ClusUCB outperforms UCB-Improved ([AO10]) and MOSS ([AB09]) as well as other popular stochastic bandit algorithms such as DMED ([HT10]), UCB-V ([AMS09]), Median Elimination ([EDMM06]), Thompson Sampling ([AG11]) and KL-UCB ([GC11]). The results can be seen in Appendix 10 section. We also prove strong theoretical guarantees for Clustered UCB, and the manuscript is currently under review in AISTATS 2017.

2. We have started work on AugUCB algorithm and are currently trying to prove it's simple regret bounds. Empirically the algorithm has performed well and the result is shown in Appendix 10.

8 Contribution

From a theoretical viewpoint, we conclude that the gap-dependent regret bound of ClusUCB is lower than that of MOSS and UCB-Improved. From the numerical experiments on settings with small gaps between optimal and sub-optimal mean rewards, we observed that ClusUCB outperforms several popular bandit algorithms. While we exhibited better regret bounds for ClusUCB, it would be interesting future research to improve the theoretical analysis of ClusUCB to achieve the gap-independent regret bound of MOSS and possibly also the gap-dependent bound conjectured in Section 2.4.3 of [BCB12].

9 Schedule

In this section we present the schedule for the next 6 months. It is shown in Table 2.

Table 2: Schedule for the next six months

Month	**Work to be done**
January-February	Based on the initial reviews from AISTATS, 2017 we have made further modifications to the ClusUCB algorithm and came up with Efficient ClusUCB algorithm. We will submit to ICML, 2017.
January-February	We will also want to come up with a short version of our AugUCB algorithm for the thresholding bandit problem for IJCAI, 2017.
March-June	Based on the two sets of reviews we wish to work further on these two setups. I also wish to work on restless bandits where these two algorithms can be applied (see [GM11]).

Algorithm 2 AugmentedUCB

```
Input: Time horizon $T$, exploration parameters $\rho_\mu$, $\rho_v$ and threshold $\tau$.
Initialization: Set $B_0 := \mathcal{A}$, $M = \left\lfloor \frac{1}{2}\log_2 \frac{T}{e} \right\rfloor$, $m := 0$, $\epsilon_0 := 1$, $\psi_0 = \frac{T\epsilon_0}{8K\log K}$, $\ell_0 = \left\lceil \frac{2\psi\log(T\epsilon_0)}{\epsilon_0} \right\rceil$ and $N_0 = K\ell_0$.
Pull each arm once

for $t = K+1, .., T$ do
    Pull arm $i \in \arg\min_{j\in B_m} \left\{ |\hat{r}_j - \tau| - 2s_j \right\}$
    $t := t + 1$
    Arm Elimination
        For each arm $i \in B_m$, remove arm $i$ from $B_m$ if
            $\hat{r}_i + c_i < \tau - c_i$ or $\hat{r}_i - c_i > \tau + c_i$
            where $c_i = \sqrt{\frac{\rho_\mu \psi_m \log(T\epsilon_m)}{2n_i}}$
    Arm Elimination by Mean and Variance Estimation
        For each arm $i \in B_m$, remove arm $i$ from $B_m$ if
            $\hat{r}_i + s_i < \tau - s_i$ or $\hat{r}_i - s_i > \tau + s_i$
            where $s_i = \sqrt{\frac{\rho_v \psi_m \hat{V}_i \log(T\epsilon_m)}{4n_i} + \frac{\rho_v \psi_m \log(T\epsilon_m)}{4n_i}}$
    if $t \geq N_m$ and $m \leq M$ then
        Reset Parameters
            $\epsilon_{m+1} := \frac{\epsilon_m}{2}$
            $B_{m+1} := B_m$
            $\psi_{m+1} = \frac{T\epsilon_{m+1}}{(\log(\frac{3}{16}\log K))^2}$
            $\ell_{m+1} := \left\lceil \frac{2\psi_{m+1}\log(T\epsilon_{m+1})}{\epsilon_{m+1}} \right\rceil$
            $N_{m+1} := t + |B_{m+1}|\ell_{m+1}$
            $m := m + 1$
    end if
end for
Output $\hat{S}_\tau = \{i : \hat{r}_i \geq \tau\}$.
```

References

[AB09] Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.

[AB10] Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *COLT-23th Conference on Learning Theory-2010*, pages 13–p, 2010.

[ACBF02] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[AG11] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. *arXiv preprint arXiv:1111.1797*, 2011.

[AMS09] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009.

[AO10] Peter Auer and Ronald Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.

[BCB12] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and non-stochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*, 2012.

[BCBL12] Sébastien Bubeck, Nicolo Cesa-Bianchi, and Gábor Lugosi. Bandits with heavy tail. *arXiv preprint arXiv:1209.1727*, 2012.

[BJM12] Loc Bui, Ramesh Johari, and Shie Mannor. Clustered bandits. *arXiv preprint arXiv:1206.4169*, 2012.

[BMS11] Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in finitely-armed and continuous-armed bandits. *Theoretical Computer Science*, 412(19):1832–1852, 2011.

[BT96] Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming (optimization and neural computation series, 3). *Athena Scientific*, 7:15–23, 1996.

[BWV13] Sébastien Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identifications in multi-armed bandits. In *ICML (1)*, pages 258–265, 2013.

[CBGZ13] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. A gang of bandits. In *Advances in Neural Information Processing Systems*, pages 737–745, 2013.

[CGK12] Olivier Cappe, Aurelien Garivier, and Emilie Kaufmann. pymabandits, 2012. `http://mloss.org/software/view/415/`.

[CLK$^+$14] Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, pages 379–387, 2014.

[EDMM06] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *The Journal of Machine Learning Research*, 7:1079–1105, 2006.

[GC11] Aurélien Garivier and Olivier Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. *arXiv preprint arXiv:1102.2490*, 2011.

[GGL12] Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems*, pages 3212–3220, 2012.

[GGLB11] Victor Gabillon, Mohammad Ghavamzadeh, Alessandro Lazaric, and Sébastien Bubeck. Multi-bandit best arm identification. In *Advances in Neural Information Processing Systems*, pages 2222–2230, 2011.

[GLZ14] Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *ICML*, pages 757–765, 2014.

[GM11] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory*, pages 174–188. Springer, 2011.

[GMP$^+$15] Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. *Bayesian reinforcement learning: a survey*. World Scientific, 2015.

[HT10] Junya Honda and Akimichi Takemura. An asymptotically optimal bandit algorithm for bounded support models. In *COLT*, pages 67–79. Citeseer, 2010.

[KTAS12] Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 655–662, 2012.

[Lat15] Tor Lattimore. Optimally confident ucb: Improved regret for finite-armed bandits. *arXiv preprint arXiv:1507.07880*, 2015.

[LCLS10] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[LGC16] Andrea Locatelli, Maurilio Gutzeit, and Alexandra Carpentier. An optimal algorithm for the thresholding bandit problem. *arXiv preprint arXiv:1605.08671*, 2016.

[LR85] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

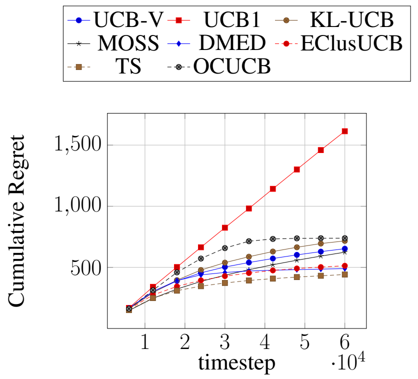
[LT16] Yun-Ching Liu and Yoshimasa Tsuruoka. Modification of improved upper confidence bounds for regulating exploration in monte-carlo tree search. *Theoretical Computer Science*, 2016.

[NL14] Trong T Nguyen and Hady W Lauw. Dynamic clustering of contextual multi-armed bandits. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1959–1962. ACM, 2014.

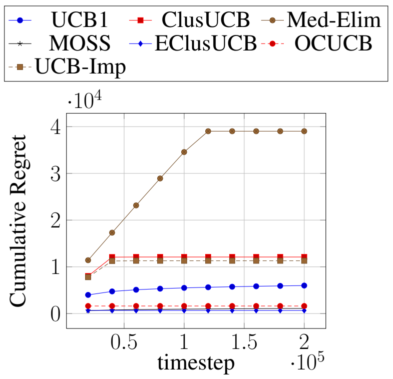
[Rob52] Herbert Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1952.

[SB98] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

[Tho33] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933.



(a) Experiment 1: 20 Bernoulli-distributed arms with $r_{i_{i\neq *}} = 0.07$ and $r^* = 0.1$.



(b) Experiment 2: 100 Gaussian-distributed arms with $r_{i_{i\neq *:1-33}} = 0.1$, $r_{i_{i\neq *:34-99}} = 0.6$ and $r^*_{i=100} = 0.9$.

Figure 1: Cumulative regret for various bandit algorithms on two stochastic K-armed bandit environments.

10 Appendix

In this appendix, we show the different experiments carried on various test-beds. Appendix A contains experiments on ClusUCB while Appendix B contains experiments on AugUCB.

Appendix A

For the purpose of performance comparison using cumulative regret as the metric, we implement the following algorithms: KL-UCB[GC11], DMED[HT10], MOSS[AB09], UCB1[ACBF02], UCB-Improved[AO10], Median Elimination[EDMM06], Thompson Sampling(TS)[AG11] and UCB-V[AMS09][2]. The parameters of ClusUCB/EClusUCB algorithm for all the experiments are set as follows: $\psi = \frac{T}{196 \log K}$, $\rho_s = 0.5$, $\rho_a = 0.5$ and $p = \lceil \frac{K}{\log K} \rceil$.

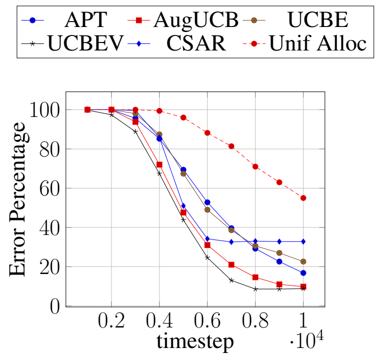
The first experiment is conducted over a testbed of 20 arms for the test-cases involving Bernoulli reward distribution with expected rewards of the arms $r_{i_{i\neq *}} = 0.07$ and $r^* = 0.1$. These type of cases are frequently encountered in web-advertising domain. The horizon T is set to 60000. The regret is averaged over 100 independent runs and is shown in Figure 1(a). EClusUCB, MOSS, UCB1, UCB-V, KL-UCB, TS and DMED are run in this experimental setup and we observe that EClusUCB performs better than all the aforementioned algorithms except TS. Because of the small gaps and short horizon T, we do not implement UCB-Improved and Median Elimination on this test-case.

[2]The implementation for KL-UCB and DMED were taken from [CGK12]

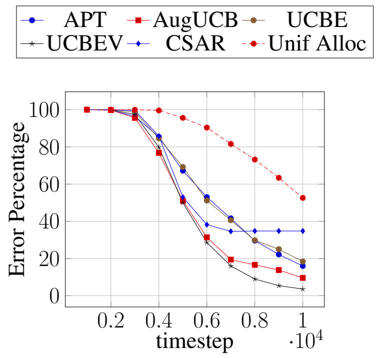
The second experiment is conducted over a testbed of 100 arms involving Gaussian reward distribution with expected rewards of the arms $r_{i_{i\neq *:1-33}} = 0.1$, $r_{i_{i\neq *:34-99}} = 0.6$ and $r^*_{i=100} = 0.9$ with variance set at $\sigma_i^2 = 0.3, \forall i \in A$. The horizon T is set for a large duration of 2×10^5 and the regret is averaged over 100 independent runs and is shown in Figure 1(b). In this case, in addition to EClusUCB, we also show the performance of ClusUCB algorithm. From the results in Figure 1(b), we observe that EClusUCB outperforms ClusUCB as well as MOSS, UCB1, UCB-Improved and Median-Elimination($\epsilon = 0.1, \delta = 0.1$). But ClusUCB and UCB-Improved behaves almost similaryly in this environment. Also the performance of UCB-Improved is poor in comparison to other algorithms, which is probably because of pulls wasted in initial exploration whereas EClusUCB with the choice of ψ, ρ_a and ρ_s performs much better.

Appendix B

In this section we compare the empirical performance of AugUCB against APT, Uniform Allocation, CSAR, UCBE and UCBEV algorithm. The threshold τ is set at 0.5 for all experiments. Each algorithm is run independently 500 times for 10000 timesteps and the output set of arms suggested by the algorithms at every timestep is recorded. The output is considered erroneous if the correct set of arms is not $i = \{6, 7, 8, 9, 10\}$ (true for all the experiments). The error percentage over 500 runs is plotted against 10000 timesteps. For the uniform allocation algorithm, for each $t = 1, 2, .., T$ the arms are sampled uniformly. For UCBE algorithm ([AMS09]) which was built for single best arm identification, we modify it according to [LGC16] to suit the goal of finding arms above the threshold τ. So the exploration parameter a in UCBE is set to $a = \frac{T-K}{H_1}$. Again, for UCBEV, following [GGLB11], we modify it such that the exploration parameter $a = \frac{T-2K}{H_1^\sigma}$. Then for each timestep $t = 1, 2, .., T$ we pull the arm that minimizes $\{|\hat{r}_i - \tau| - \sqrt{\frac{a}{n_i}}\}$, where n_i is the number of times the arm i is pulled till $t-1$ timestep and a is set as mentioned above for UCBE and UCBEV respectively. Also, APT is run with $\epsilon = 0.05$, which denotes the precision with which the algorithm suggests the best set of arms and we modify CSAR as per [LGC16] such that it behaves as a Successive Reject algorithm whereby it rejects the arm farthest from τ after each phase. For AugUCB we take $\rho_\mu = \frac{1}{8}$ and $\rho_v = \frac{1}{3}$ as in Theorem **??**. The first experiment is conducted on a testbed of 100 arms involving Gaussian reward distribution with expected rewards of the arms $r_{1:4} = 0.2 + (0 : 3) * 0.05$, $r_5 = 0.45$, $r_6 = 0.55$, $r_{7:10} = 0.65 + (0 : 3) * 0.05$ and $r_{11:100}$=0.4. The means of first 10 arms are set as arithmetic progression. Variance is set as $\sigma^2_{1:5} = 0.5$ and $\sigma^2_{6:10} = 0.6$. Then $\sigma^2_{11:100}$ is chosen uniform randomly between $0.38 - 0.42$. The means in the



(a) Experiment 1: Experiment with Arithmetic Progression



(b) Experiment 2: Experiment with Geometric Progression

Figure 2: Experiments with thresholding bandit

testbed are chosen in such a way that any algorithm has to spend a significant amount of budget to explore all the arms and variance is chosen in such a way that the arms above τ have high variance whereas arms below τ have lower variance. The result is shown in Figure 2(a). In this experiment we see that UCBEV which has access to the problem complexity and is a variance-aware algorithm beats all other algorithm including UCBE which has access to the problem complexity but does not take into account the variance of the arms. AugUCB with the said parameters outperforms UCBE, APT and the other non variance-aware algorithms that we have considered.

The second experiment is conducted on a testbed of 100 arms with the means of first 10 arms set as Geometric Progression. The testbed involves Gaussian reward distribution with expected rewards of the arms as $r_{1:4} = 0.4 - (0.2)^{1:4}$, $r_5 = 0.45$, $r_6 = 0.55$ and $r_{7:10} = 0.6 + (0.2)^{5-(1:4)}$. The variances of all the arms and $r_{11:100}$ are set in the same way as in experiment 1. AugUCB, APT, CSAR, Uniform Allocation, UCBE and UCBEV with the same settings as experiment 1 are run on this testbed. The result is shown in Figure 2(b). Here, again we see that AugUCB beats APT, UCBE and all the non-variance aware algorithms with only UCBEV beating AugUCB.