Survey of Different Variants of Thompson Sampling

IE 613 Project Report

by
Vinay Chourasiya (173190011)
Karan Patel (173190012)
Manoj Kumar (16i190003)

Under the guidance of
Prof. M.K. Hanawal

Inter Disciplinary Programme
in
Industrial Engineering and Operations Research
Indian Institute of Technology Bombay
5th May, 2018

Contents

1 Introduction

1.1 What is MAB problem?

The Multi-Armed Bandit (MAB) is a balanced framework in machine learning and optimization in which there is a finite set of actions, each of which has a reward associated with them from some stochastic process, and a player selects actions to optimize his/her reward in long-term performance. The MAB framework also gives a abstract fundamental decision problem – whether to exploit or explore in the face of uncertainty.

The multi-armed bandit problem can also be defined as an agent that attempts to acquire new knowledge (called "exploration") and optimize his/her decisions based on existing knowledge (called "exploitation") simultaneously. The agent attempts to balance these competing tasks in order to maximize his total value over the considered time horizon.

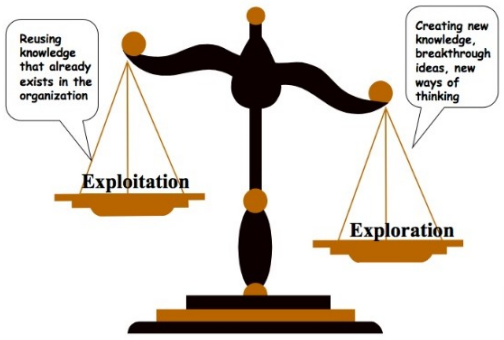

Figure 1: Meaning of exploitation and exploration

Or it may be understood as let you are faced with many slot machines (also called multi-armed bandits). Each bandit has unknown probability distribution of prize or reward. Some bandits may be generous but others may not. Also you don't know probability and amount of rewards associated with the arms. Only by choosing one bandit at a time, we can know reward which corresponds to selected arm. Figure on next page shows slot machines with different distributions of rewards labelled as D1, D2 ans so on.

Figure 2: Slot machines with their respective labelled distribution of reward

1.2 Applications of the Bandit Model

- In dynamic routing efforts, for minimizing delays in a network.
- In clinical trials, where investigation of the effects of different experimental treatments are carried out while minimizing patient losses.
- In designing financial portfolio. Here while allocating assets one aims at maximizing the expected return and minimizing the risk.

- In internet display advertising, companies have a suite of potential ads they can display to visitors, but the company is not sure which ad strategy to follow to maximize sales.
- In Ecology: animals have a finite amount of energy to expend, and certain behaviors has uncertain rewards. How does the animal maximize its fitness?

1.3 History of Thompson Sampling

In artificial intelligence, Thompson sampling is helpful for choosing actions that addresses the exploration-exploitation difficulty in the multi-armed bandit problem. It consists in choosing the action that maximizes the expected reward.

It was initially described by William R. Thompson (named after him) in 1933 [1] for allocating experimental effort in two- armed bandit problems arising in clinical trials but ignored by community of artificial intelligence. It was also rediscovered many times in framework of reinforcement learning [2] [3]. The first application to Markov decision processes was in 2000. A related approach (Bayesian control rule) was published in 2010 and in the same year it was also shown that this algorithm is instantaneously self-correcting. In 2012 [4] Optimistic Bayesian Sampling was introduced in which the probability of playing an action increases with the uncertainty in the estimate of the action value. Nowadays, Thompson Sampling found wide use in online learning problems like in online advertising [5]; a Double Thompson Sampling is proposed for dueling bandits [6] (which can be understood as modified MAB).

In real life, we come across the many decision problems which requires learning about the outcomes of different choices, either with the collaboration of environment, or interacting with the virtual world. Here comes the notion of multi-objective multi-armed bandit which is explained briefly in its section.

There are many algorithms available for the stochastic bandit problem, some popular ones include Upper Confidence Bound (UCB) family of algorithms (viz. UCB, UCB-V, KL-UCB etc.) which have good theoretical guarantees but if we compare Thompson sampling algorithm with all of them it gives minimum regret and achieves sub-linear bound quickly as clear from the following picture:

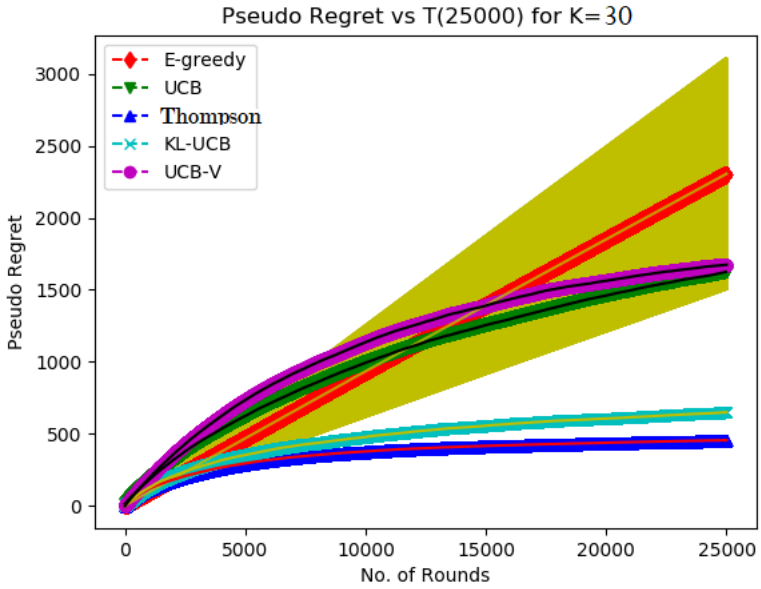

Figure 3: Comparison of ϵ - greedy, some UCB family algoriths and Thompson sampling algorithm for 30 arms and 25000 rounds

2 Thompson Sampling for the Bernoulli Bandit Setting

Bernoulli Bandit: Suppose there are K arm, and when any arm among them is played it yields either a success or failure. Here success and failure corresponds to giving out reward 1 and 0 respectively. Let any arm $\in \{1, 2, ..., K\}$ produces a reward of one with probability $\mu_k \in [0, 1]$ and a reward of zero with probability 1-μ_k. The success probabilities or mean rewards corresponding to different arms $\mu = (\mu_1, \mu_2..., \mu_k)$ may be different and are unknown, but fixed over time. Hence it can be learned by experimentation.

Let [**?**] in the first round, an arm k_1 is pulled and a reward $r_1 \in \{0, 1\}$ is generated with success probability $\mathbb{P}(r_1 = 1|k_1, \mu) = \mu_{k_1}$. After observing r_1, the agent will play another arm k_2, observes reward r_2, and this process continues. Also let player begin with an independent prior belief over each μ_k. in Bernoulli setting case we consider these priors as beta-distributed with parameters $S = (S_1, S_2, ..., S_k)$ and $F = (F_1, F_2, ..., F_k)$. Where S is the number of success and F is Number of failure. Beta distribution is convenient because of it is an conjugate distribution (whih means that the posterior distributions and the prior probability distribution are in the same probability distribution family). Hence each arm's posterior distribution will also be beat with updated parameters.

The update rule for selected arm in round t^{th} round: $(S_k, F_k) = \begin{cases} (S_k, F_k + 1), & \text{if } r_t = 0 \\ (S_k + 1, F_k), & \text{if } r_t = 1 \end{cases}$

Following is the algorithm.

After certain number of rounds distribution will converge to its true mean. The regret of an algorithm can calculated as the difference between the mean reward of optimal arm and reward of the arm selected by the algorithm. Mathematically,

$$Regret_t(\mu) = max_k(\mu_k - \mu_{k_t})$$

Algorithm 1 Thomson Sampling with Bernoulli setting

Input: No. of arms (K), $S_k(1) = 0 \text{and} F_k(1) = 0$.
for t= 1,2,...**do**:
  for k = 1, 2,...,K **do**:
    Sample $\theta_k(t) \sim$ Beta($S_k + 1$,$F_k + 1$)
  end for
  Sample and select arm:
  $k_t \leftarrow argmax_k\ \theta_i(t)$
  Select arm k_t and observe reward r_t
  Update posterior distribution:
  If $r_t = 1$,
    $S_{k_t} = S_{k_t} + 1$
  else
    $F_{k_t} = F_{k_t} + 1$
end for

3 Thompson sampling for General setting

Beyond Bernoulli Bandit, Thompson sampling can be applied successfully to many online decision problems. Now, we explain the more general setting.

Let there is a finite set of arms K and the player pulls arms $k_1, k_2, \ldots$. This set of arms may be finite or infinite. After pulling arm k_t in round t the player observe an outcome y_t which is randomly generated by system according to some conditional probability measure. The player enjoys a reward of $\tilde{r}_t = r(y_t)$, where r is a known function also let $\tilde{r}_t$ is such that its value lies in interval [0,1]. Initially the player is uncertain about reward and represents his uncertainty by using a prior distribution. Now Bernoulli trial is performed with success probability equal to observed reward ($\tilde{r}_t$) and output is observed in terms of 0 or 1, and posterior distribution is updated accordingly (same as in case of Bernoulli setting). The algorithm is as follows:

Algorithm 2 Thomson Sampling with General setting

Input: No. of arms (K), $S_k(1) = 0 \text{and} F_k(1) = 0$.
for t= 1,2,...**do**:
  for k = 1, 2,...,K **do**:
    Sample $\theta_k(t) \sim$ Beta($S_k + 1$,$F_k + 1$) distribution.
  end for
  Sample and select arm:
  $k_t \leftarrow argmax_k\ \theta_i(t)$
  Select arm k_t and observe reward $\tilde{r}_t$.
  Now by taking success probability as $\tilde{r}_t$ and get output r_t
  by using Bernoulli distribution.
  Update posterior distribution:
  If $r_t = 1$,
    $S_{k_t} = S_{k_t} + 1$
  else
    $F_{k_t} = F_{k_t} + 1$
end for

4 Contextual Multi-Arm Bandits

One of the popular variant of Multi-arm bandits (MAB) is contextual MAB.In contextual MAB we have side information [7] related to each arm this is we called context of arm.
Contextual MAB has wide range of application like news personalization system, on website (Yahoo!, bing browser) [8].The news personalization system can relate with MAB contextual setting:

- All the different news article consider as arm.
- The inputs or feature of both user and news article consider as context related to particular arm.
- Here we want to maximize the CTR[1] by using the feature vector of each arm.

Contextual Multi Arm Bandits

The Extension of contextual MAB come from the side information of arm,the side information is related to reward generation of arm. for example in news article recommendation, side information from past activity of user may be the user profile (age,demographic condition,gender) and the article feature like types (sports, politics,entertainment,fashion related), both user and article feature consider as a side information or context, using these side information we can recommend the news article for the particular user,this same as the selecting the arm (article) which has highest payoff, here payoff means high CTR for article by the user.

In contextual MAB, the feature vector at current round t revealed and we also have the information of previous all round that we called history. [9] In history we have the arm that is selected $a(\tau)$, rewards $r_{a(\tau)}(\tau)$, and feature vector corresponding to all arms $b_i(\tau)$ where $i = 1, 2, ..K$ and $\tau = 1..t-1$.so, using the feature vector of round t and the history upto $t-1$, we are interested to predict the arm for the current round to increase the reward or alternately decrease the regret.

Contextual MAB can be seen as general stochastic MAB with context vector $b_i(t)$ is one all the time (specially in case of linear payoff $\mu b_i(t)$).so,the algorithm like UCB and Thompson can be extend for the setting of contextual MAB.there is rich literature on contextual MAB [8–10], many algorithm has proposed Contextual MAB setting. some well known algorithms are as follows:

(1.) **LinUCB :** [11] is the extend version of UCB for specially contextual case with linear Realizability assumption.In *LinUCB*, assume that the reward generated corresponding to the $b_i(t)$ feature vector is distributed with mean $b_i(t)\mu^*$ where μ^* is fix parameter, that underlying parameter algorithm learn.

[1] Click Through Rate

LinUCB algorithm choose arm at time t is $a_t^* = \arg\max_i (b_i(t)\mu^*)$ where $i = 1, 2, ..K$ and regret for T round define as

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} r_{t,a_t^*} - \sum_{t=1}^{T} r_{t,a_t}\right]$$
$$= \sum_{t=1}^{T} b_{a_t^*}(t)\mu^* - \sum_{t=1}^{T} b_{a_t}(t)\mu^*$$

where $r_{t,i}$ reward by the best arm i generate from unknown distribution with mean $b_i(t)\mu^*$ at round t. and the a_t is arm selected by the algorithm at round t **Regret Bound of *LinUCB* [12] :**

$$R_T \leq O\left(d\frac{\sqrt{Tln(1+T)}}{\delta})\right)$$

(2.) **LinREL** [10], setting of *LinREL* is same as *LinUCB*, it also assume the linearity assumption of payoff arm (*i.e.* reward distributed with mean $b_i(t)\mu^*$ and this distribution is unknown.) *LinUCB* use l_2 *norm* but in case of *LinREL*, do regularization by setting $D_t^T D_t$, where D matrix is the collection of feature vector of arms that are selected in round $t-1$, so, the dimension of D_t is $(t-1, K)$.For the algorithm of *LinREL* refer **[10]**

4.1 Thompson algorithm for Contextual MAB :

Our main focus on, how Thompson algorithm suitably work for contextual MAB setting.Remarkable work done by *Goyal and Agrawal* [9] on the Thompson algorithm,they analyze the Thompson Sampling for Contextual Bandits with Linear Payoffs..

Problem setting for contextual MAB we discussed earlier, similar setting consider by *Goyal and Agrawal* [9], There are N arm, at time t context vector associated with arm i is $b_i(t) \in R^d$. for all arm context vector $b_i(t)$ revealed , using context vector $b_i(t)$, and History of all previous round $1, 2..t-1$, player has to decide which arm choose for round t.Reward generate through unknown distribution with mean $b_i(t)^T\mu$ for each arm i and the assumption over reward generation is $r_i(t) \in \left[b_i(t)^T\mu - R, b_i(t)^T\mu + R\right]$. this assumption is called R-sub-Gaussian assumption.
Given $b_i(t)$ at round t, the reward of arm i generate from an unknown distribution with mean $b_i(t)^T\mu$, where $\mu \in R^d$ is parameter, that is learn by the algorithm.let $a^*(t)$ denote optimal arm and $a(t)$ is the arm choose by the algorithm at time t,

$$a^*(t) = \arg\max_i (b_i(t)\mu)$$

and define $\triangle_i(t)$ as ,

$$\triangle_i(t) = b_{a^*(t)}(t)\mu - b_i(t)\mu$$

so, the regret at time t is $\hat{r}(t) = \triangle_{a(t)}(t)$, and total regret define as

$$R(T) = \sum_{t=1}^{T} \hat{r}(t)$$

.

We know that Thompson sampling is a Bayesian algorithm, it assume some prior over unknown reward distribution. for contextual MAB setting Gaussian prior most suitable. [9].

For given context vector $b_i(t)$ at round t and parameter μ,*Goyal and Agrawal* propose prior for μ in Thompson sampling is $N(b_i(t)^T\hat{\mu}, v^2)$, here $v = R\sqrt{\frac{24}{\epsilon}dln(\frac{1}{\delta})}$ with $\epsilon \in (0,1)$. Let

$$B(t) = I_d + \sum_{\tau=1}^{t-1} b_{a(\tau)}(\tau)b_{a(\tau)}(\tau)^T$$

$$\hat{\mu}(t) = B(t)^{-1}\left(\sum_{\tau=1}^{t-1} b_{a(\tau)}(\tau)b_{a(\tau)}(\tau)^T\right)$$

In Thompson sampling algorithm we sample $\tilde{\mu}(t)$ from distribution $N(b_i(t)^T\hat{\mu}, v^2)$ and play arm $a(\tau)$ s.t.

$$a(\tau) = \arg\max_i (b_i(t)^T\tilde{\mu(t)})$$

Algorithm 3 Thomson Sampling for Contextual MAB with linear payoff

1: **Set:** $B = I_d$, $\hat{\mu} = 0_d$,$f = 0_d$
2: **for** t= 1,2,...**do**:
3: Sample $\tilde{\mu}$ from distribution $N(\hat{\mu}, v^2B^{-1})$
4: Play arm $a(t) := \arg\max_i(b_i(t)\tilde{\mu})$ and observe reward r_t
5: Update $B = B + b_{a(t)}(t)b_{a(t)}(t)^T,\ f = f + b_{a(t)}(t)r_t,\ \hat{\mu} = B^{-1}f$
6: **end for**

Goyal and Agrawal [9] showed that for stochastic contextual MAB with linear payoff function, with probability $1-\delta$, the total regret R(T) for Thompson sampling (3) is bounded by $O\left(\frac{d^2}{\epsilon}\sqrt{T^{1+\epsilon}(ln(Td)ln(\frac{1}{\delta}))}\right)$, for any $0 < \epsilon < 1$ and $0 < \delta < 1$.

Generalized Thompson Sampling for Contextual MAB for Contextual Bandits proposed by the *Lihong Li* [13].this is the extension of work of *Goyal and Agrawal*. Based on the connection between the Thompson sampling and exponential update, he propose the family algorithm called Generalized Thompson algorithm in the expert-learning framework.

Let N be the set of arms and player observe context $x_t \in \chi$,and x_t chosen by the adversary.and reward r_t of arm $a \in N$ generate from the mean $\mu(x_t, a_t)$, now suppose learner has the access of expert to set of experts $E \in \{E_1, ...E_n\}$, and each expert make prediction about the average reward $\mu(x, a)$, let f_i be the prediction associated with E_i, so, the arm-selection for context x done based on the $E_i(x) = \max_{a\in N} f_i(x, a)$. Regret define as ,

$$R(T) = \max_{i\in N}\sum_{t=1}^{T}\mu(x_t, E_i(x_t)) - \mathbf{E}\left[\sum_{t=1}^{T}\mu(x_t, a_t)\right]$$

under the realizability assumption there must exist E^* that predict correctly average reward. *Lihong Li* [13] define **prior** $P = (p_1, p_2, ...p_n) \in R_+^n$ where $\|P\|_1 = 1$, prior p_i can be seen as prior probability that expert E_i predict maximum average reward. The algorithm start with first posterior as $w_1 = (w_{1,1}, w_{2,1}, ..., w_{n,1})$, where $w_{i,1} = p_i$. at every step t, algorithm sample an expert from w_t and follow the expert prediction.and the update weights as

$$w_{i,t+1} \to w_{i,t}exp\left(-l(f_i(x_t, a_t), r_t)\right)$$

where $l(f, r)$ is the negative log-likelihood.

Algorithm 4 Generalized Thomson Sampling

1: **Input:** $\eta > 0$, $\gamma > 0$ $(E_1, ..., E_N)$, and prior **p**
2: Initialize posterior: w_1 from p; W_1 from $\parallel w_1 \parallel_1 = 1$
3: **for** t= 1,2,...**do**:
4: Receive context $x_t \in X$
5: Select arm a_t according to the mixture probabilities: for each a.

$$\Pr(a) = (1-\gamma)\sum_{i=1}^{N}\frac{w_{i,t}\mathbb{1}(E_i(x_t)=a)}{W_t} + \frac{\gamma}{K}$$

6: Observe reward r_t and update weights:

$$\forall i : w_{i,t+1} \leftarrow w_{i,t}.exp(-\eta.l(f_i(x_t,a_t),r_t)); W_{t+1} \leftarrow \parallel w_{t+1} \parallel_1 = \sum_i w_{i,t+1}$$

7: **end for**

5 Multi-Play Multi Arm Bandits

In Multi-play or Combinatorial MAB (MP-MAB) problem player play several arms (say L arm) from given number of arms K in each round. general MAB problem where player play only one arm, can be seen as special case of the MP-MAB where $L = 1$(only one arm pull).
MP-MAB has various application like **Online advertising on website** [11], website has set of webpages and set of users that visit the website and advertiser wants to place a advertisement on selected webpages (due the budget constraint), each user visit the certain number of pages and visited page has click-through probabilities that unknown to advertiser, so, advertiser try to learn to click-trough probabilities by repeatedly selecting the set of webpages. and finally choose best L webpages that has the highest click through probabilities. given problem can be model as the MP-MAB where set of webpages on website called all arms (K), and number of webpages selected by the advertiser called set of arm that we have pull each time (L), and goal of the problem, player want to learn top L arms that has highest mean.

5.1 Thompson Sampling for MP-MAB

we already discuss about general MAB or single play MAB where player play only one arm.for standard MAB problem we have seen that Thompson sampling (TS) optimal algorithm compare to UCB family algorithm and ϵ-greedy algorithm.Empirically TS achieves regret comparable to the lower bound of *Lai and Robbins* [14].
Komiyama et al. [12] discussed first time about the multi-play Thompson algorithm for general and Bernoulli reward distribution.

Let K arms, each arm has reward distribution that is unknown to player and player selects $L < K$ arms in each round and observe the rewards associated with L arms. Let $N_i(t)$ be the number of draws of arm i upto time t.
one can think given setting is same as general MAB setting so, it easy to formulate as general MAB setting and regret analysis is only depend only on the number of sub-optimal arm pull by the algorithm,but for MP-MAB problem regret analysis has a different structure.*Komiyama et al.* [12] presented a example in his study that show that, only calculate the expected number of time sub-optimal arm pull by the algorithm not sufficient for the calculation of optimal regret.
Example: [12] Let K=4,L=2 and $\mu_1 = .10$, $\mu_2 = .09$, $\mu_3 = .08$ and $\mu_4 = .07$. 1st and 2nd are optimal and 3rd and 4th are sub-optimal arms.

Rounds	Game-1	Game-2
t=1	I(1)={1,2} r(1)=0	I(1)={1,3} r(1)=0.01
t=2	I(1)={3,4} r(1)=.04	I(1)={1,4} r(1)=0.02
TotalRegret	R(2)=0.04	R(2)=0.03

In the given example in both game number of sub-optimal arm selected is 2 but regret is different for both the game, this is clearly show the, the consider number of the sub-optimal arm pull by the algorithm not lead to the optimal regret.

Anantharam et al. [15] first time prove that for any strongly consistent algorithm and sub-optimal arm i, the number of arm i draws is lower-bounded as

$$E[N_i(T+1)] \geq \left(\frac{1-o(1)}{d(\mu_i, \mu_L)}\right) logT$$

And regret for T round is,

$$E[R(T)] \geq \sum_{i \in [K]\backslash[L]} \left(\frac{1-o(1)\triangle_{i,L}}{d(\mu_i, \mu_L)}\right) logT$$

where, μ_L is the mean of arm L^{th} top arm among the K arms.and $\triangle_{i,L}$ is the difference mean (μ_i) of arm $i \in [K]\backslash[L]$ from the L^{th} mean (μ_L).

Algorithm 5 Multiple-play Thomson Sampling (MP-TS) for binary rewards

Input: Number of arms (K), number of selection (L)
for i= 1,2,...K**do**:
    $A_i, B_i = 1, 1$
end for
$t \leftarrow 1$
for t **= 1,2,...,T do**
    for i**= 1,2,...,K do**
        $\theta_i(t) \sim Beta(A_i, B_i)$
    end for
    I_t = top-L arms ranked by $\theta_i(t)$.
    for $i \in I_t$ **do**
        if $X_i(t) = 1$ **then**
            $A_i \leftarrow A_i + 1$
        else
            $B_i \leftarrow B_i + 1$
        end if
    end for
end for

Optimal Regret Bound : for any small $\epsilon_1 > 0$ and $\epsilon_2 > 0$ the regret of MP-TS is upper bounded as

$$E[R(T)] \leq \sum_{i \in [K] \setminus [L]} \left(\frac{1 - o(1) \triangle_{i,L}}{d(\mu_i, \mu_L)} \right) logT + O((logT)^{\frac{2}{3}})$$

Experiment Analysis of MP-TS : [12] we compare MP-TS with *CUCB* algorithm [16] and found performance of MP-TS is batter than the *CUCB*
we consider 20-armed bandits,the simulations include 20 Bernoulli arms with $\mu_1 = 0.15$, $\mu_2 = 0.12$, $\mu_3 = 0.10$, $\mu_i = 0.05$ for $i \in \{4, 5..., 12\}$, $\mu_i = 0.03$ for $i \in \{13, 14, ..., 20\}$, and L = 3.

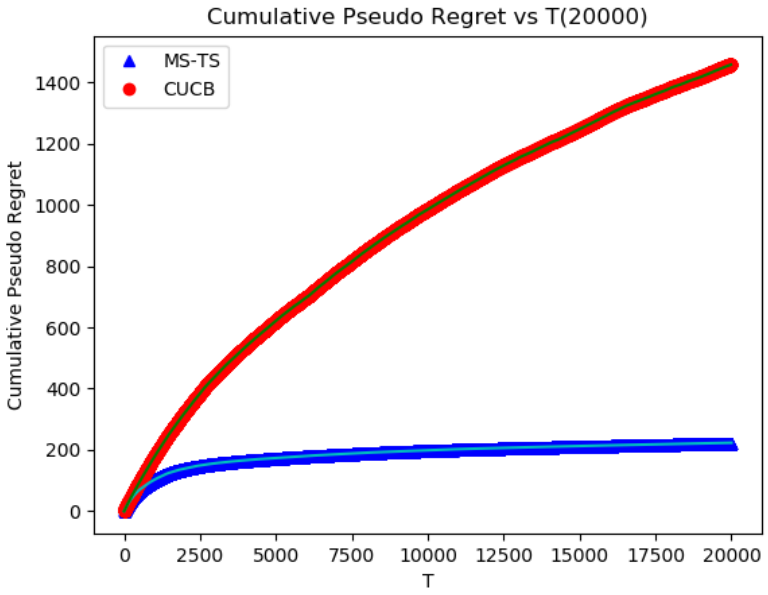

Figure 4: Comparison of *CUCB* and *MP-TS*

6 Budgeted Multi-Armed Bandits

One of the new variant of MAB is Budgeted MAB (BMAB).In budgeted MAB,reward and cost associated with each arm. In each round player observe reward as well as the random cost associated with pulled arm (both reward and the cost distributed with some unknown parameter). Whenever the player pull the arm he observe the cost and reward of the selected arm, he has fixed amount of budget(B) to pull the arm. Player can't pull the arm if cost incur in the all round is to exceeded budget. Budgeted MAB can be viewed as a stochastic version of the knapsack problem in which the value and weight of the items are both stochastic [17].

The player's goal is to find the best arm with given the budget constraint, because of variable-cost setting,we need to explore not only the reward of an arm but also its cost.The application of BMAB is many scenario like on-spot instance bidding of cloud computing, This new setting models many Internet applications (e.g., ad exchange, sponsored search, and cloud computing), using BMAB we also model problem consists in determining the optimal sensor to interrogate in a wireless sensor network scenario(when we retrieve information from a sensor, we gain information about the monitored process and, at the same time, we spend budget in terms of energetic costs. Also the problem of a service provider trying to balance the costs of the employed resources and the revenues gained by the provided services fits the BMAB model). [18].

In budgeted MAB, we consider a slot machine with K arms,At round t, a player pulls an arm $i \in [K]$, receives a random reward $r_i(t)$, and pays a random cost $c_i(t)$ until he runs out of his budget B. Both the reward $r_i(t)$ and the cost $c_i(t)$ are supported on [01].A few algorithms have been proposed to solve the budgeted MAB problem.ϵ-first algorithm was proposed which first spends ϵB budget on pure explorations, and then keeps pulling the arm with the maximum empirical reward-to-cost ratio, in UCB family there is algorithm name as UCB-BV1 algorithm [19], In UCB-BV1 algorithm define $D_{i,t}$ for each arm $i \in K$,

$$D_{i,t} = \frac{\bar{r_{i,t}}}{\bar{c_{i,t}}} + \frac{\left(1+\frac{1}{\lambda}\right)\sqrt{\frac{ln(t-1)}{n_{i,t}}}}{\lambda - \sqrt{\frac{ln(t-1)}{n_{i,t}}}}$$

and pull arm a_t,

$$a_t = \arg\max_i D_{i,t}$$

where, $\bar{c_{i,t}}$ is average cost of arm i,$\bar{r_{i,t}}$ is average reward of arm i and $n_{i,t}$ is the times that arm i has been pulled before step t. *Ding et al.* [19] showed that the bound for UCB-BV1 is $O(ln(B))$.

6.1 Thompson algorithm for Budgeted MAB

As we discussed many MAB setting in which Thompson sampling achieve optimal policy, so this obvious to formulate a algorithm based on Thompson sampling that is achieve optimal policy for budgeted MAB.The Thompson sampling algorithm *BTS* for budgeted MAB proposed by *Xia-Li et al.* [17],they assume reward and cost are supported by $[0, 1]$, and propose algorithm for bernoulli distribution as well as general distribution.

Let expected reward and cost of arm i is μ_i^r and μ_i^c respectively, also we denote arm 1 as optimal arm ($\arg\max_{i\in[K]} \frac{\mu_i^r}{\mu_i^c} = 1$). define pseudo-regret [17] .

$$\text{Regret} = R^* - E\left[\sum_{t=1}^{T_B} r_t\right]$$

where R^* is is the expected reward of the optimal policy,r_t is the reward received by an algorithm at round t, T_B is the stopping time of the algorithm.

The BTS algorithm [17] is shown in (reference here), In the algorithm, $S_i^r(t)$ denotes the times that the player receives reward 1 from arm i before (excluding) round t and $S_i^c(t)$ denotes the times that the player pays cost 1 for pulling arm i before (excluding) round t.

Xia-Li et al. [17] *prove that for* $\forall\gamma \in (0, 1)$*, for both Bernoulli bandits and general bandits, the regret of the BTS algorithm can be upper bounded as below.*

$$\text{Regret} \leq \sum_{i=2}^{K}\left\{\frac{2\log(B)}{\gamma^2\mu_i^c\Delta_i}\left(\frac{\mu_1^r}{\mu_1^c}+1\right)^2 + \Phi_i(\gamma)\right\} + O\left(\frac{K}{\gamma^2}\right)$$

where, we consider the arm 1 is optimal one. and $\Delta_i = \frac{\mu_1^r}{\mu_1^c} - \frac{\mu_i^r}{\mu_i^c} \qquad \forall i \geq 2.$

7 Multi-objective multi armed Bandit

Unlike to the above, multi-armed bandit with single objective, there are many problem which have more than one objective attached to the single action or arm. As each objective of a single

Algorithm 6 Budgeted Thomson Sampling (BTS)

For each arm $i \in [K]$, set $S_i^r(1) \leftarrow 0$, $F_i^r(1) \leftarrow 0$, $S_i^c \leftarrow 0$ and $F_i^c \leftarrow 0$;
Set $B_1 \leftarrow B; t \leftarrow 1$;
while $B_t > 0$ **do**
    For each arm $i \in [K]$,
    sample $\theta_i^r(t) \sim Beta(S_i^r(t) + 1, F_i^r(t) + 1$, and
    sample $\theta_i^c(t) \sim Beta(S_i^c(t) + 1, F_i^c(t) + 1$;
    Pull arm $I_t = argmax_{i \in [K]} \frac{\theta_i^r(t)}{\theta_i^c(t)}$; receive reward r_t;
    pay cost c_t; update $B_{t+1} \leftarrow B_t - c_t$;
    For Bernoulli bandits, $\tilde{r} \leftarrow r_t$, $\tilde{c} \leftarrow c_t$,
    for general bandit, sample $\tilde{r}$ from $B(r_t)$ and sample $\tilde{c}$ from $B(c_t)$;
    $S_{I_t}^r(t+1) \leftarrow S_{I_t}^r(t) + \tilde{r}$, $F_{I_t}^r(t+1) \leftarrow F_{I_t}^r(t) + 1 - \tilde{r}$;
    $S_{I_t}^c(t+1) \leftarrow S_{I_t}^c(t) + \tilde{c}$, $F_{I_t}^c(t+1) \leftarrow F_{I_t}^c(t) + 1 - \tilde{c}$;
    $\forall j \neq I_t, S_j^r(t+1) \leftarrow S_j^r(t), F_j^r(t+1) \leftarrow F_j^r(t)$,
    $S_j^c(t+1) \leftarrow S_j^c(t), F_j^c(t+1) \leftarrow F_j^c(t)$;
    Set $t \leftarrow t + 1$.
end while

action or arm conflicting with others, thus its difficult to calculate the outcome. For example, an agent learning the best strategy to redistribute the ambulances from a set of choice, may want to minimize the response time, while minimising the fuel cost and the stress on the drivers. To tackle these type of problems, we need some different setting from the single objective multi-armed bandit setting. In this report we will discuss the multi objective multi armed bandit settings(MOMABs). Till now, researcher focus on the setting of MOMABs in which no preference has been taken into consideration [20]. Next examples showing that how the user interaction in multi-objective learning problem is beneficial, the exploration of preventive strategies for epidemics using computationally expensive simulations under objective like minimising morbidity, the number of people infected, the costs of the preventive strategy, and future household robots that will need to learn the preferences of their users with respect to the outcomes like performance, speed, energy usage of different ways to perform a household task [21]. In scalar MAB, main goal is to maximize the expected cumulative reward, $E[\sum_{t=1}^{T} \mu_{a(t)}]$, where T is the time horizon, $a(t)$ is the arm pulled at time t. In starting, the agent does not know the distribution of reward, and get the information of distribution at each step when a arm $a(t)$, pulled. In scalar MAB setting, we studied two main algorithms which are, UCB and Thompson sampling in which rewards are distributed as Bernoulli. In UCB, we keeps estimates of the mean of an arm, $x_{a(t)}$ and uses upper confidence bounds for exploration. At every round we pulled arm $a(t)$ with highest values of mean plus exploration term. Exploration term is defined in such a way that it decrease with the number of pulls of an arm n_a, increase and increases slowly with the total number of pulls of arms, t. In Thompson sampling, algorithm maintains posterior distributions for the parameters of the reward distributions for each arm $a \in A$ and pulls the arm based on samples from the posterior. Like UCB, this also has distributions of rewards as Bernoulli and this has only one parameter, i.e., μ_a. The prior for these μ_a are Beta distribution. The posterior distribution can be computed by counting the number of success and number of failures and generate the samples according to this posterior distribution.

In Multi-Objective problems, there are n objectives, all are desirable. Hence, as rewards are randomly samples, i.e., we called the rewards to stochastic rewards $r(t)$, and for each choice of μ_a, rewards defined as the vector. As it may possible that there is more than one arm which is optimal according to the different preferences of user regrading the different objectives. Here,

utility function $u(\mu, w)$ is defined by such preferences that is weighted by vector w and returns the scaler value of μ. Here, agent does not know the vector w in beforehand. Similar to the single multi arm bandit, rewards in MOMABs are vector of d independent Bernoulli distributed. Whole Multi-Objective reinforcement learning research assumes that utility function is unknown to the agent throughout the learning phase and user can only access it in selection phase. This setting is known as Off-line Multi-Objective reinforcement learning. As we can see, in this setting learning phase and selective phase are different which take more time to return the outcomes. Therefore, on-line Multi-Objective reinforcement learning is better as it will take the user preferences in learning phase itself. In this we will discuss the two algorithms which are extension of UCB and Thompson sampling of single objective multi arm bandit setting. These two algorithms are also consider the user preferences and ask to user for the comparison between two arms which gives the optimal regret under different objectives.

We studied following algorithms for multi-objective multi armed bandit setting on the idea of classical MAB. To discuss the algorithm first we need to some other important things such as utility function u.

Definition 1 *A linear utility function is a weighted sum of the values in each objective, μ, of an alternative, i.e.,*

$$u(\mu, w) = w.\mu, \tag{1}$$

where w is a vector of non-negative weights that sum to 1, and express the preferences of the user w.r.t. each objective positively to the utility.

In the off-line MORL decision support scenario where the utility is an unknown linear utility function, solution is convex hull (CH) such as
$\text{CH}(\mathbb{A})=\{a \in \mathbb{A} \mid \exists w \forall (a^{'} \in \mathbb{A}) : w.\mu_a \geq w.\mu_{a^{'}}\}$.
In on-line MORL decision support scenario, we want to minimize user regret, i.e., the amount of utility that is lost due to playing suboptimal arms. We defined the value of optimal arm as

$$\mu^* = \text{argmax}_a w^*.\mu_a,$$

, where w^* are the ground truth weights of a linear utility function. Similar to the single objective MAB, we also define the regret of pulling arm, a in multi objective such as

$$\Delta_a = \text{argmax}_a w^*.\mu_a.$$

Definition 2 *The expected total user regret of pulling a sequence of arms for each time step between $t = 1$ and a time horizon T in a MOMAB is*

$$\mathbb{E}\left[w^*.(\sum_{t=1}^{T}(\mu^* - \mu_{a(t)}))\right] = \sum_a (w^*.\Delta_a)\mathbb{E}\left[n_a(T)\right]$$

where $n_a(T)$ is the number of times arm a is pulled until time step T.

Here, we assume that we can interact with the user once before (or after) pulling an arm, in the form of pairwise comparison [22], [23] and [24]. In contrast to [24], we present the user with expected reward vectors, rather than single arm pulls. We ask the users to compare two vectors, x and y, and observe whether the user prefers x to y, denoted by $x \succ y$. At each time

t, we have access to a data set, C, of j of such preference pairs, where $j \leq t$, is the number of comparisons performed until t:

$$C = \{(x_i \succ y_j)\}_{i=1}^{j}. \tag{2}$$

As we assume that $u(\mu, w)$ is a linear utility function, and in the form of Equation 2, we can estimate w^* using logistic regression.
Now come to the first algorithm, which is known as *utility*-MAP UCB, where MAP is the maximum a posterior estimates of w using logistics regression.

Algorithm 7 Utility MAP-UCB

Input: A parameter prior on the distribution of w.
C $\rightarrow \emptyset$, // previous comparisons
$\bar{x}_a \rightarrow$ initialise with single pull, r_a, for each a
$n_a \rightarrow 1$ for each a
for $t =\mid \mathbb{A} \mid, ..., T$ **do**
  $\bar{w} \rightarrow \mathbb{P}_{simplex}(MAP(w \mid C))$
  $\bar{a}^* \rightarrow \text{argmax}_a \bar{w}.\bar{x}_a$
  $a(t) \rightarrow \text{argmax}_a(\bar{w}.\bar{x}_a + c(\bar{w}, \bar{x}_a, n_a, t))$
  $r(t) \rightarrow playa(t) and observereward$
  $\bar{x}_{a(t)} \rightarrow \frac{n_{a(t)}\bar{x}_a + r(t)}{n_{a(t)}+1}$
  $n_{a(t)} + +$
  if $\bar{a}^* \neq a(t)$ **then**
    perform user comparison for $\bar{x}_{\bar{a}_*}$ ant $\bar{x}_{a(t)}$
    and add result $((\bar{x}_{\bar{a}^*} \succ \bar{x}_{a(t)}))$ or $((\bar{x}_{a(t)} \succ \bar{x}_{\bar{a}^*}))$
    to C

In Algorithm 7, we noted the exploration term $c(\bar{w}, \bar{x}_a, n_a, t))$ which can be defined as

$$c(\bar{w}, \bar{x}_a, n_a, t)) = c_1(\bar{w}.\bar{x}_a, n_a, t), \quad \text{or} \quad c(\bar{w}, \bar{x}_a, n_a, t)) = c_{ch}(\bar{w}.\bar{x}_a, n_a, t)$$

where

$$c_1(\bar{x}_a, n_a, t) = \sqrt{\frac{2lnt}{n_a}} \tag{3}$$

defined for original UCB 1 bound or upper confidence bound derived from Chernoff's bound[7]:

$$c_{ch}(\bar{w}.\bar{x}_a, n_a, t) = \sqrt{\frac{2\bar{x}_a ln\sqrt{t}}{n_a}} + \frac{2ln\sqrt{t}}{n_a} \tag{4}$$

In both the algorithms 7 and 8, reward distribution is taken as the vector of independent Bernoulli distributions and prior distribution for preference vector w is taken as Gaussian distribution. In Iterative Thompson Sampling algorithm 8 two independent samples both from the posteriors for the parameters of the reward distributions of each arm (θ_1^t and θ_2^t), and from the posterior for the parameters of the utility function (η_1^t and η_2^t).
Now,we want to move the discussion of Thompson sampling to the dueling bandit problem. The dueling bandit Problem is a different form of classic multi-armed bandit setting problems, which takes feedback in the form of pairwise comparisons. It is useful in many systems such as information retrieval, where preferences of users are easily available and stable. There are two different dueling bandit settings, One of them is Condorcet dueling bandit setting other is Copeland dueling bandit. In Condorcet dueling bandit setting, there exist an arm called

Algorithm 8 Interactive Thompson Sampling

Input: Parameter priors on reward distributions, and on w distribution.
C $\rightarrow \emptyset$; // previous comparisons
D $\rightarrow \emptyset$; // observed reward data
for $t = 1, ..., T$ **do**
  $\eta_1^t, \eta_2^t \rightarrow$ draw 2 samples from $P(\eta^t|C)$
  $\theta_1^t, \theta_2^t \rightarrow$ draw 2 samples from $P(\theta^t|D)$
  $a_1(t) \rightarrow \text{argmax}_a \mathbb{E}_{P(r,w|a,\theta_1^t,\eta_1^t)}[w.r]$
  $a_2(t) \rightarrow \text{argmax}_a \mathbb{E}_{P(r,w|a,\theta_2^t,\eta_2^t)}[w.r]$
  r(t)$\rightarrow$ play $a_1(t)$ and observe reward append $(r(t), a_1(t))$ to D
  if $a_1(t) \neq a_2(t)$ **then**
    $\tilde{\mu}_{1,a_1(t)} \rightarrow \mathbb{E}_{P(r|a_1(t),\theta_1^t)}[r]$
    $\tilde{\mu}_{2,a_2(t)} \rightarrow \mathbb{E}_{P(r|a_2(t),\theta_2^t)}[r]$
    perform user comparison for $\tilde{\mu}_{1,a_1(t)}$ and $\tilde{\mu}_{2,a_2(t)}$
    and add result $((\tilde{\mu}_{1,a_1(t)} \succ \tilde{\mu}_{2,a_2(t)}))$ or $((\tilde{\mu}_{2,a_2(t)} \succ \tilde{\mu}_{1,a_1(t)}))$ to C

Condorcet winner, that beats all other arms whereas, in Copeland dueling bandit setting, there is an arm (or arms) called Copeland winner(s) that beat mostly all other arms.
There exist many algorithms which are generalized versions of classical MAB. One of the alternative and effective solution is Thompson Sampling in classical MAB. Which works on the principle of choosing the optimal arm that maximizes reward obtained, where belief is randomly drawn. As we have seen that Thompson Sampling achieves the lower regret then other algorithms. This thing motivate to think about whether and how Thompson sampling can be applied in dueling bandit setting.
The standard Thompson Sampling, is challenging to apply directly to the dueling bandits, due to the the reason that not all pairwise comparisons provides the complete statistical information about the system. Thus, Thompson Sampling needs some adjustments in such a way that allows the comparison of the winners against themselves, and avoid the comparisons of non-winners against themselves. In this report we will discuss the double Thompson sampling for dueling bandit.
Thompson sampling uses the confidence bounds to through the likely non winners arms. Thompson Sampling avoid trapping in suboptimal comparisons by eliminating the likely non winner arms with the help of confidence bound. The Double Thompson Sampling (D-TS) ha two important features. One of them is that double sampling structure of D-TS is same as dueling bandits. Two independent rounds of sampling provides the opportunity to select the same arm in both the round and thus to compare with winners against themselves [25].
D-TS has both practical and theoretical advantages in compare to prior studies on dueling bandit. The simple framework of D-TS applies to general Copeland dueling bandits. This double sampling structure enables to obtain theoretical bounds for the regret of D-TS. The theoretical analysis in dueling is more difficult due to the selection of arms involves more factors. To tackle this issue, D-TS draw two independent samples. D-TS not address the tie case, when there is multiple winners. To address these tie cases, we modify the D-TS as D-TS$^+$, which involves the tie breaking for multiple winners. Even D-TS$^+$ gives the same regret bound as D-TS, but performs better than D-TS.

8 Model of D-TS for dueling bandit

We consider a dueling bandit problem with K arms, where $K \geq 2$, denoted by $A = \{1, 2, ..., K\}$. At every time $t > 0$, user get a pair of arm $(a_t^{(1)}, a_t^{(2)})$ and a noisy comparison outcome w_t is obtained in such a way, that if user prefer $a_t^{(1)}$ over $a_t^{(2)}$ the $w_t = 1$, else $w_t = 2$. It is also assumed that user preference are stationary over time and distribution of noisy comparison outcomes is described by the preference matrix $P = [p_{ij}]_{K\times K}$, where $p_{ij} = \mathbb{P}\{i \succ j\}, i, j = 1, 2, ..K$, $p_{ij} + p_{ji} = 1 \& p_{ii} = \frac{1}{2}$. We say that arm i beats arm j if $p_{ij} > \frac{1}{2}$ [25].
In general Copeland dueling bandits, Copeland winner is defined as the arm which maximize the number of the other arms it beats [26], [27]. The Copeland score is defined as $\sum_{j\neq i} \mathbb{1}(p_{ij} > \frac{1}{2})$ and the normalized Copeland score is defined as $\zeta_i = \frac{1}{K-1} \sum_{j\neq i} \mathbb{1}(p_{ij} > \frac{1}{2})$, where $\mathbb{1}(.)$ is the indicator function. Let $\zeta_i^* = \max_{1\leq i\leq K} \zeta_i$. We can define the Copeland winner(s) in term of ζ^* as $\mathcal{C}^* = \{i : 1 \leq i \leq K, \zeta_i = \zeta^*\}$. We can say that Condorcet winner is a special case of Copeland winner with $\mathcal{C}^* = 1$.
Let us define, a filtration $\mathcal{H}_{t-1}$ are the history observed by dueling bandit algorithm before time t, i.e., $\mathcal{H}_{t-1} = \{a^{(1)_\tau}, a_\tau^{(2)}, w_\tau, \tau = 1, 2..., t-1\}$. Then the performance of a dueling bandit algorithm algorithm Γ is measured by its expected cumulative regret, which is defined as

$$\mathcal{R}_\Gamma(T) = \zeta^* T - \frac{1}{2} \sum_{t=1}^{T} \mathbb{E}[\zeta_{a^{(1)_t}} + \zeta_{a^{(2)_t}}] \tag{5}$$

The Objective of dueling bandit algorithm Γ is to minimize $\mathcal{R}_\Gamma(T)$.

8.1 Double Thompson Sampling

Algorithm 9 D-TS for Copeland Dueling Bandits

Init : B$\leftarrow 0_{K\times K}$; // B_{ij} is the number of time slots that the user prefer arm i over j.
for $t = 1$ **to** T **do**
// Phase 1: Choose the first candidate $a^{(1)}$
$U := [u_{ij}], L := [l_{ij}]$, where $u_{ij} = \frac{B_{ij}}{B_{ij}+B+ji} + \sqrt{\frac{\alpha log t}{B_{ij}+B_{ji}}}$ if $i \neq j$
and $u_{ij} = l_{ij} = \frac{1}{2} \ \ \forall i \ \ ; // \frac{x}{0} := 1$for any x $\hat{\zeta}_i \leftarrow \frac{1}{K-1} \sum_{j\neq i} \mathbb{1}(u_{ij} > \frac{1}{2})$; // upper bound of the normalized Copeland score.
$\mathcal{C} \leftarrow \{i : \hat{\zeta}_i = \max_j \hat{\zeta}_j\}$;
for $i, j = 1, ..K$ with $i < j$ **do**
Sample $\theta_{ij}^{(1)} \sim Beta(B_{ij} + 1, B_{ji} + 1)$;
$\theta_{ij}^{(1)} \leftarrow 1 - \theta_{ij}^{(1)}$;
$a^{(1)} \leftarrow \text{argmax}_i \in \mathcal{C} \sum_{j\neq i} \mathbb{1}(\theta_{ij}^{(1)} > \frac{1}{2})$; // Choosing from $\mathcal{C}$ to eliminate likely non winner arms; Ties are broken randomly.
// Phase 2: Choose the second candidate $a^{(2)}$
Sample $\theta_{ia^{(1)}}^{(2)} \sim Beta(B_{ia^{(1)}} + 1, B_{a^{(1)}i} + 1)$ $\forall i \neq a^{(2)}$, and let $\theta_{a^{(1)}a^{(2)}}^{(2)} = \frac{1}{2}$;
$a^{(2)} \leftarrow \text{argmax}_{i:l_{ia^{(1)}} \leq \frac{1}{2}} \theta_{ia^{(1)}}^{(2)}$; // choosing only from uncertain pairs.
// Compare and update
Compare pair $(a^{(1)}, a^{(2)})$ and observe the result w;
Update B: $B_{a^{(1)}a^{(2)}} \leftarrow B_{a^{(1)}a^{(2)}} + 1$ if $w = 1$, or $B_{a^{(2)}a^{(1)}} \leftarrow B_{a^{(2)}a^{(1)}} + 1$ if $w = 2$;

The basic idea of Double Thompson sampling is that it selects both the candidates by Thompson sampling. For each pair $(i, j), i \neq j$, we assume beta distribution for preference

matrix p_{ij} as prior distribution. The posterior distributions are updated according to the comparison result $B_{ij}(t-1)$, where $B_{ij}(t-1)$ is the number of time slots when arm i beats arm j before t. D-TS selects the two candidates by sampling form posterior distributions. The following theorem talks about the regret bound for D-TS algorithm.
Before this we will talk about some important things as given below:-
Gap to $\frac{1}{2}$: In dueling bandit, preference probability of optimal arm is considered as $\frac{1}{2}$. So Δ_{ij} is called the gap between p_{ij} and $\frac{1}{2}$, mathematically can be defined as $\Delta_{ij} = \|p_{ij} - \frac{1}{2}\|$.
Number of Comparisons: In D-TS algorithm, pair of arms (i, j) can be compared as $(a^{(1)}, a^{(2)}) = (i, j)$ or $(a^{(1),a^{(2)}}) = (j, i)$. For these two different ways, two counters are defined as follows: $N_{ij}^{(1)}(t) = \sum_{\tau=1}^{t} \mathbb{1}(a_{tau}^{(1)} = i, a_{tau}^{(2)} = j)$ and $N_{ij}^{(2)}(t) = \sum_{\tau=1}^{t} \mathbb{1}(a_{tau}^{(1)} = j, a_{tau}^{(2)} = i)$. Then the total number of comparisons is $N_{ij}(t) = N_{ij}^{(1)}(t) + N_{ij}^{(2)}(t)$ for $i \neq j$, and $N_{ii}(t) = N_{ii}^{(1)}(t) = N_{ii}^{(2)}(t)$ for $i = j$. Following are the assumption to obtain the theoretical regret bound for D-TS:-

Assumption 1 *The preference probability $p_{ij} \neq \frac{1}{2}$ for any $i \neq j$ [25].*

Preposition 1 *[25]When applying D-TS with $\alpha > 0.5$ in a Copeland dueling bandit with a preference matrix $P = [p_{ij}]_{K \times K}$ satisfying Assumption 1, its regret is bounded as:*

$$\mathcal{R}_{D-TS}(T) \leq \sum_{i \neq j : p_{ij} < \frac{1}{2}} \left[\frac{4\alpha log T}{\Delta_{ij}^2} + (1+\epsilon) \frac{log T}{D(p_{ij} \parallel \frac{1}{2})} \right] + O(\frac{K^2}{\epsilon^2})$$

where $\epsilon > 0$ is an arbitrary constant, and $D(p \parallel q) = plog\frac{p}{q} + (1-p)log\frac{1-p}{1-q}$ is the KL divergence.

We are going into the proof of above preposition. Let us refine the regret bound for D-TS which reduce its scaling factor with respect to the number of arm K. For each $i \notin \mathcal{C}^*$, let $(\sigma_i(1), \sigma_i(2), ..., \sigma_i(K))$ be a permutation of $(1, 2, ..., K)$, such that $p_{\sigma_{i(1)},i} \geq p_{\sigma_{i(2)},i} \geq ... \geq p_{\sigma_{i(K)},i}$. For a Copeland winner $i^* \in \mathcal{C}$, we will define $L_C = \sum_{j=1}^{K} \mathbb{1}(p_{ji^*} > \frac{1}{2})$ be the number of arms that beats arm i^*. To, refine the regret, we introduce an additional no-tie assumption as given below:

Assumption 2 *[25]For each arm $i \notin \mathcal{C}^*, p_{\sigma_i(L_C+1),i} > p_{\sigma_{i(j),i}}$, for all $j > L_C + 1$.*

The following theorem define the refined regret bound for D-TS :

Theorem 8.1 *[25]When applying D-TS with $\alpha > 0.5$ in a Copeland dueling bandit with a preference matrix $P = [p_{ij}]_{K \times K}$ satisfying Assumption 1 and 2, its regret is bounded as:*

$$\mathcal{R}_{D-TS}(T) \leq \sum_{i^* \in \mathcal{C}^*} \left[\sum_{j : p_{ji} > \frac{1}{2}} \frac{4\alpha log\ T}{\Delta_{ji}^2} + \sum_{j : p_{ji} < \frac{1}{2}} (1+\epsilon) \frac{log\ T}{D(p_{ji} \| \frac{1}{2})} \right] + \sum_{i \notin \mathcal{C}^*} \sum_{j=1}^{L_C+1} \frac{4\alpha\ log\ T}{\Delta^2_{\sigma_{i(j)},i}}$$

$$+\beta(1+\epsilon)^2 \sum_{i \notin \mathcal{C}^*} \sum_{j=L_C+2}^{K} \frac{log\ log\ T}{D(p_{\sigma_{i(j)},i} \| p_{\sigma_{i(L_C+1)},i})} + O(K^3) + O(\frac{K^2}{\epsilon^2}),$$

where $\beta > 2$ and $\epsilon > 0$ are constants, and $D(. \parallel .)$ is the KL-divergence.

In theorem 8.1, the first term corresponds to the regret when the first candidate $a_t^{(1)}$ is a winner, and is $O(K \mid \mathcal{C}^* \mid \log T)$. The second term corresponds to the comparisons between a non-winner arm and its first $L_C + 1$ superiors, which is bounded by $O(K(L_C + 1) \log T)$. The remaining terms correspond to the comparisons between a non-winner arm and the remaining arms, and it bounded by $O(K^2 \log \log T)$.
As we have seen that Algorithm 9 breaks the tie case randomly. To break the tie case carefully,

we need to modify the Algorithm as discussed below. D-TS tends to explore all potential winners by randomly breaking the ties. This may be useful in some application such as restaurant recommendation, where users may not want to stick to a single winner. However, because of this, the regret of D-TS scales with the number of winners, i.e., $\mid \mathcal{C}^* \mid$. To reduce the regret, we can break the ties according to estimated regret.// The normalized score for each arm i can be estimated as $\tilde{\zeta^*}(t) = \frac{1}{K}\sum_{j\neq i}\mathbb{1}(\theta_{ij}^{(1)}(t) > \frac{1}{2})$. Then, $\tilde{\zeta^*}(t) = max_i\tilde{\zeta_i}(t)$ is the maximum Copeland score and $\tilde{r_{ij}}(t) = \tilde{\zeta^*}(t) - \frac{1}{2}\left[\tilde{\zeta_i}(t) + \tilde{\zeta_j}(t)\right]$ is the loss of comparing arm i and arm j. For $p_{ij} \neq \frac{1}{2}$, we need about $\Theta(\frac{logT}{D(p_{ij}\|\frac{1}{2})})$ time slots to distinguish it from $\frac{1}{2}$[5]. Thus, when choosing i as the first candidate, the regret of comparing it with all other arms can be estimated by $\tilde{R}_i^{(1)}(t) = \sum_{j:\theta_{ij}^{(1)}(t)\neq\frac{1}{2}} \tilde{r}_{ij}(t)/D(\theta_{ij}^{(1)}(t) \parallel \frac{1}{2})$. The following D-TS$^+$ algorithm breaks the ties to minimize $\tilde{R}_i^{(1)}(t)$.

D-TS$^+$: Implement the same operation as D-TS, expect for the selection of the first candidate is replaced by the following two steps:

$$\mathbb{A}^{(1)} \rightarrow \{i \in \mathcal{C} : \zeta_i = max_{i\in\mathcal{C}} \sum_{j\notin i} \mathbb{1}(\theta_{ij}^{(1)} > \frac{1}{2})\};$$

$$a^{(1)} \rightarrow argmin_{i\in\mathbb{A}^{(1)}} \tilde{R}_i^{(1)};$$

D-TS$^+$ only changes the tie-breaking criterion in selecting the first candidate. Thus, the regret bound of D-TS directly applies to D-TS$^+$.

Corollary 1 *[25]The regret of D-TS$^+$, satisfies theorem 8.1 under Assumption 1 and 2.*

Above corollary (1), provides an upper bound for the regret of D-TS$^+$. However, in practise D-TS$^+$ performs better the D-TS in the case of multiple winners. This algorithm reduce the regret from $O(K \mid \mathcal{C}^* \mid \log T)$ to $O(K \log T)$.

As a result, we had studied two algorithm for on-line multi-objective reinforcement learning based on Bayesian learning in which the agent can interact with its user. Both the algorithms 7 and 8 build upon state of art learning and Bayesian machine learning to learn about the environment and about the utility function of user. In both algorithm pose the pairwise comparison queries to the user. Algorithm 7 uses exploration bonuses, and focus on the preferences when the best mean estimate for the current MAP estimate of the weight vector, and the best estimated mean plus exploration bonus for the same estimate, recommend different arms. Iterative Thompson sampling focus on the preferences by pulling two sets of samples from both the posteriors of the means of the arms and the posterior for w, and querying the user when those two sets of samples have a different best arm. We discussed the Thompson sampling for dueling bandits and studies about the D-TS algorithm and its modified version D-TS$^+$ for general Copeland dueling bandits, including Condorcet dueling bandits as special case. We had seen the different perspectives of D-TS and D-TS$^+$ for the theoretical and practical use. The regret of D-TS and D-TS$^+$ is bounded by $O(K^2 \log T)$ in general Copeland dueling bandits, and can be refined to $O(K \log T + K^2 \log\log T)$ in Condorcet dueling bandits and most practical Copeland dueling bandits.

Bibliography

[1] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[2] Jeremy Wyatt. Exploration and inference in learning from reinforcement. 1998.

[3] Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML*, pages 943–950, 2000.

[4] Benedict C May, Nathan Korda, Anthony Lee, and David S Leslie. Optimistic bayesian sampling in contextual-bandit problems. *Journal of Machine Learning Research*, 13(Jun):2069–2106, 2012.

[5] Ole-Christoffer Granmo and Sondre Glimsdal. Accelerated bayesian learning for decentralized two-armed bandit based decision making with applications to the goore game. *Applied intelligence*, 38(4):479–488, 2013.

[6] Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. In *Advances in Neural Information Processing Systems*, pages 649–657, 2016.

[7] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[8] 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.

[9] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pages 127–135, 2013.

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

[11] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214, 2011.

[12] Junpei Komiyama, Junya Honda, and Hiroshi Nakagawa. Optimal regret analysis of thompson sampling in stochastic multi-armed bandit problem with multiple plays. *arXiv preprint arXiv:1506.00779*, 2015.

[13] Lihong Li. Generalized thompson sampling for contextual bandits. *arXiv preprint arXiv:1310.7163*, 2013.

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

[15] Venkatachalam Anantharam, Pravin Varaiya, and Jean Walrand. Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-part i: Iid rewards. *IEEE Transactions on Automatic Control*, 32(11):968–976, 1987.

[16] Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International Conference on Machine Learning*, pages 151–159, 2013.

[17] Yingce Xia, Haifang Li, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Thompson sampling for budgeted multi-armed bandits. In *IJCAI*, pages 3960–3966, 2015.

[18] Francesco Trovò, Stefano Paladino, Marcello Restelli, and Nicola Gatti. Budgeted multi-armed bandit in continuous action space. In *ECAI*, pages 560–568, 2016.

[19] Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-armed bandit with budget constraint and variable costs. In *AAAI*, 2013.

[20] Peter Auer, Chao-Kai Chiang, Ronald Ortner, and Madalina Drugan. Pareto front identification from stochastic bandit feedback. In *Artificial Intelligence and Statistics*, pages 939–947, 2016.

[21] Diederik M Roijers, Luisa M Zintgraf, and Ann Nowé. Interactive thompson sampling for multi-objective multi-armed bandits. In *International Conference on Algorithmic DecisionTheory*, pages 18–34. Springer, 2017.

[22] Brochu Eric, Nando D Freitas, and Abhijeet Ghosh. Active preference learning with discrete choice data. In *Advances in neural information processing systems*, pages 409–416, 2008.

[23] Gerald Tesauro. Connectionist learning of expert preferences by comparison training. In *Advances in neural information processing systems*, pages 99–106, 1989.

[24] Masrour Zoghi, Shimon Whiteson, Remi Munos, and Maarten Rijke. Relative upper confidence bound for the k-armed dueling bandit problem. In *International Conference on Machine Learning*, pages 10–18, 2014.

[25] Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. In *Advances in Neural Information Processing Systems*, pages 649–657, 2016.

[26] Masrour Zoghi, Zohar S Karnin, Shimon Whiteson, and Maarten De Rijke. Copeland dueling bandits. In *Advances in Neural Information Processing Systems*, pages 307–315, 2015.

[27] Junpei Komiyama, Junya Honda, and Hiroshi Nakagawa. Copeland dueling bandit problem: Regret lower bound, optimal algorithm, and computationally efficient algorithm. *arXiv preprint arXiv:1605.01677*, 2016.