Optimizing budget allocation in creator marketing campaigns

Aleksi Päkkilä

School of Science

Bachelor's thesis
Helsinki 20.06.2024

Supervisor

Prof. Kai Virtanen

Advisor

BSc. Leo Lännenmäki

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the bachelor's thesis

Author Aleksi Päkkilä

Title Optimizing Budget Allocation in Creator Marketing Campaigns

Degree programme Bachelor's Programme in Science and Technology

Major Mathematics and Systems Sciences **Code of major** SCI3029

Teacher in charge Prof. Kai Virtanen

Advisor BSc. Leo Lännenmäki

Date 20.06.2024 **Number of pages** 24 **Language** English

Abstract

Budget allocation problem in the Youtube creator marketing context is a sequential resource allocation problem with a budget limitation in an uncertain environment. It can be formulated as a batched and budgeted multi-armed bandit problem. The solution is BB-MAB-TS algorithm for dynamic budget allocation. Thompson Sampling is suggested as the underlying heuristic for sampling the budget allocation weights that are needed to direct the budget allocation optimization system. The result of this thesis is an active learning system that is directed by the BB-MAB-TS algorithm. Multiple future research topics arise from this thesis on the areas of optimizing the convergence time of the optimization process, applying the BB-MAB-TS algorithm to other contexts, and statistical analysis of the performance of the process.

Keywords multi-armed bandit, Thompson Sampling, active learning, machine learning, creator marketing, budget allocation, price optimization

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Tekniikan kandidaatintyön tiivistelmä

Tekijä Aleksi Päkkilä

Työn nimi Optimizing Budget Allocation in Creator Marketing Campaigns

Koulutusohjelma Bachelor's Programme in Science and Technology

Pääaine Mathematics and Systems Sciences **Pääaineen koodi** SCI3029

Vastuuopettaja ja ohjaaja Prof. Kai Virtanen

Päivämäärä 20.06.2024 **Sivumäärä** 24 **Kieli** Englanti

Tiivistelmä

Youtube -vaikuttajamarkkinointikampanjan kontekstissa tapahtuva budjettiallokaatio voidaan kuvailla resurssiallokaatio-ongelmaksi epävarmassa ympäristössä. Budjettiallokaatio tulee optimoida ja se tehdään useiden päätöskierrosten yli itse budjetin rajoittaessa optimointiprosessin pituutta. Kyseinen ongelma muotoillaan batched & budgeted multi-armed bandit -ongelmaksi. Se ratkaistaan dynaamiseen budjettiallokaatioon soveltuvaa työn aikana kehitettyä BB-MAB-TS algoritmia hyödyntäen. Algoritmi valitsee budjettiallokaatiokertoimet perustuen Thompson Samplingia soveltavaan aktiiviseen koneoppimissysteemiin. Täten tämän kandidaatintyön tuloksena on aktiivinen koneoppimissysteemi, joka pohjautuu BB-MAB-TS algoritmiin. Työn seurauksena nousi useita tulevaisuuden tutkimusaiheita kuten oppimisprosessin konvergoitumisajan optimoiminen, BB-MAB-TS -algoritmin soveltaminen muihin resurssiallokaatio konteksteihin, ja oppimisprosessin suoriutumisen tilastollinen analysointi.

Avainsanat multi-armed bandit, Thompson Sampling, active learning, machine learning, creator marketing, budget allocation, price optimization

Foreword

I want to wholeheartedly thank my advisor Leo Lännenmäki for the opportunity to write this thesis during my summer job period and for all the support throughout the writing process of this thesis. Also, a special thanks goes to Niko Korvenlaita for sharing deep enthusiastic insights about Thompson Sampling and helping out with the mathematical formulation of the problem. Lastly, I want to thank my supervisor Professor Kai Virtanen for providing feedback on the clarity of the thesis and ensuring the scientific representation.

Helsinki, 20.06.2024

Aleksi Päkkilä

Contents

1 Introduction

This thesis introduces the BB-MAB-TS algorithm for dynamic budget allocation in the context of creator marketing campaigns. Thompson Sampling (Thompson, 1933) is used as the underlying mathematical heuristic behind the algorithm. It is used for creating an active learning system that optimizes the budget allocation. The term budget allocation refers to the action of allocating the finite campaign budget into multiple deals between the advertiser and different creators participating in the campaign. Each participant is offered a deal to participate in the campaign and the deals have monetary value. With appropriate modifications, the algorithm can also be used for finding optimal allocation for budget or other resources in other contexts.

Creator marketing campaigns are run through an online platform. It connects the advertiser and a group of Youtube creators who agree to take part in the campaign. The creators are offered a deal to do a short advertisement about the advertiser's product in their own Youtube video. The objective of the advertiser is to maximize the value gained from the marketing campaign. This value is measured as the amount of views that the advertisements get collectively. Therefore, the objective is to minimize the price per view. The lowest price per view will also minimize the deal price. Consecutively, the budget allocation of the entire campaign will be optimized as the deals are made with the lowest price point.

Hypothesis is that the acceptance rate of the deals as a function of price is a S-shaped curve. Concretely, this means that the higher price the creator is offered, the more likely they are to accept the deal. Therefore, finding the optimal budget allocation is achieved by finding the smallest price point for which the acceptance rate is high enough to use the entire given budget. During the active learning process, one needs to learn about the deal acceptance behavior of the creators and find the optimal price point with high enough acceptance rate.

The budget allocation optimization problem can be described as a sequential resource allocation problem in an uncertain environment. These types of optimization problems have previously been studied as multi-armed bandit (MAB) problems (Mahajan and Teneketzis, 2007). In this problem setting, an agent needs to maximize the value from the finite or infinite amount of resources by finding the optimal way to deploy them to different competing alternatives. The problem is related to the exploration-exploitation dilemma which is well-known in the context of reinforcement learning (Auer and Cesa-Bianchi, 2002).

The exploration-exploitation dilemma arises from the phenomenon that the known optimal alternative could be exploited to maximize the rewards gained from the resources. Still, by exploring the different alternatives, a more optimal alternative could be found and used for accumulating more rewards. Therefore, the agent needs to learn about the optimality of the different alternatives through a learning process (Lai and Robbins, 1985).

Previous research by Scott (2010), Auer and Cesa-Bianchi (2002), and Ding et al. (2013) has suggested different algorithms for solving the multi-armed bandit problem in similar contexts. These algorithms include Thompson sampling (Thompson, 1933), upper-confidence bound (UCB) (Auer and Cesa-Bianchi, 2002), and ϵ-greedy (Auer and Cesa-Bianchi, 2002) algorithms. Especially, the Bayesian Thompson Sampling method introduced by Scott (2010) has a wide applicability and has gained a lot of popularity in the machine learning community during the past years due to the need in online decision making and recommendation systems which require active learning. Companies such as Google, Amazon, Facebook, Salesforce, and Netflix utilize the algorithms in their products (Scott, 2015) for learning about the click-behavior of the users through the active learning systems ingrained within the products.

In this thesis, the budget allocation problem in creator marketing context is converted into a batched (Kalkanli and Ozgur, 2021) and budgeted (Ding et al., 2013) multi-armed bandit (BB-MAB) problem. Then, Thompson Sampling is suggested as a solution algorithm for solving the formulated BB-MAB problem and optimizing the budget allocation. The suggested BB-MAB-TS algorithm is used for creating an active learning process (Russo et al., 2018). During the process, the budget is allocated over multiple decision rounds which are called batches. Throughout the process, the optimal price point is learned while staying within the budget limitations. Therefore, the main result of this thesis is the BB-MAB-TS algorithm for adaptive and dynamic budget allocation. Furthermore, this thesis expands the use cases of Thompson Sampling by showing that it is applicable solution to budget allocation problems.

Section 2 gives an overview of MAB problems both in the classical, batched, and budgeted settings. Additionally, the Section 2 introduces some well-known decision policies used for solving the problem. In Section 3, the budget allocation problem is formalized as a BB-MAB problem in the context of creator marketing campaigns. The problem is then solved by applying Thompson Sampling in the problem context. Moreover, Section 3 describes the constraints for the active learning system and outlines the proposed algorithm as pseudo code. Finally, the Section 4 considers the future research topics arising from this thesis.

2 Introduction to the multi-armed bandit problem

Multi-armed bandit problems (MAB problems) are a class of sequential resource allocation problems where an agent faces a challenge to allocate some finite or infinite amount of resources between different competing alternatives. The goal of the agent is to maximize the received reward or realized value (Mahajan and Teneketzis, 2007). The idea of MAB was first introduced by Robbins (1952), and further formulated by Lai and Robbins (1985).

The name, multi-armed bandit, derives from an imaginary row of slot machines (see Picture 1) that have a total of $k \geq 2$ arms (Lai and Robbins, 1985). An agent faces a challenge to maximize the accrued rewards by utilizing an optimal decision policy for choosing which arm to pull at each time step. Every time an arm is pulled, the agent receives a random reward from an unknown reward distribution that is unique to each machine. The rewards are independent of the previous pulls. Thus, the agent needs to learn about the reward distributions of the different arms through exploration. In an optimal strategy, the agent needs to balance between exploiting the known optimal arm and explore to learn if there is more optimal arm.

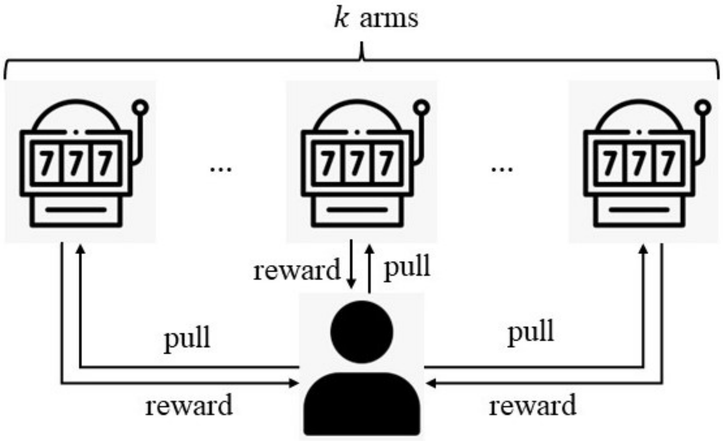

Figure 1: Illustrative picture of the multi-armed bandit problem set up.

Mahajan and Teneketzis (2007) describe the classical MAB with four key features: 1. only one machine is operated at a time, 2. machines that are not operated remain frozen, 3. machines are independent from each other, and 4. frozen machines do not give rewards.

The MAB problem outlines the fundamental conflict between exploiting the known optimal alternative and exploring for more optimal alternatives. This is also known as the exploration-exploitation dilemma extensively discussed in reinforcement learning (Auer and Cesa-Bianchi, 2002). In order to overcome the dilemma, a decision policy π must be designed to decide which arm to pull at each time step. The policy should enable finding the optimal arm and balance between exploring and exploiting different arms. Thus, the policy enables the agent to find the optimal arm and maximize the cumulative rewards.

The quality of the decision policy is assessed by its cumulative regret. Regret refers to the loss due to not pulling the optimal arm every time instant (Auer and Cesa-Bianchi, 2002). It is caused by either exploring other arms and not pulling the optimal arm or due to a non-optimal decision policy that never finds the optimal arm. The target is to minimize the cumulative regret accrued over the decision rounds. Lai and Robbins (1985) showed that the regret has to grow at least logarithmically when the number of pulls increases. Then, the decision policy would be asymptotically efficient. Thus, the optimal arm is pulled exponentially more often compared to the other arms (Auer and Cesa-Bianchi, 2002).

Multiple different problems in industry can be formulated as MAB problems. These include problems in clinical trials, sensor management, and online advertising (Xia et al., 2015; Mahajan and Teneketzis, 2007; Avadhanula et al., 2021). Thus, there are a lot of real life practical implementations for the solution algorithms of the MAB problem.

2.1 Batched multi-armed bandits

Batched bandits are a subcategory of multi-armed bandits where there are time restrictions that limit the number of decision rounds. Therefore, the agent must explore the optimal arm quickly at the beginning of the exploration phase. In this setting, the time horizon is divided into T decision rounds and at the end of each round the rewards from multiple pulls of arms are observed (Kalkanli and Ozgur, 2021). The multiple pulls during one decision rounds create a batch.

The data is accumulated more quickly, and the exploration process can be directed based on more data points within similar time horizon. Time restrictions can arise in contexts such as clinical trials, marketing, and surveys where the experiments are conducted in batches which creates limitations to the time steps between the experiments. Thus, the round constraints make some decision policies more beneficial to optimize the learning process.

The batches can be created either in a static or adaptive manner and they form a grid. In the static grid's case, the grid is fixed before starting to pull any arms and in the adaptive grid's case the consecutive batch is determined based on the results from the previous batch and utilizing some external randomness in the batch forming process (Gao et al., 2019). In the case of an adaptive grid, both the size of a batch and total number of batches can be varied.

Using an adaptive grid should lead to more efficient convergence of the learning process. By gathering more data and learning about the optimality of each arm, the agent can drop some arms from the exploration process leading to quicker convergence. The batched multi-armed bandit setting has previously been researched by, e.g., Kalkanli and Ozgur (2021) and Gao et al. (2019).

2.2 Multi-armed bandits with budget constraint and variable costs

Ding et al. (2013) were the first ones to introduce a subcategory of MABs with budget constraints and variable costs (MAB-BV). Whenever an arm is pulled, the agent needs to pay a random cost and the costs are further constrained by a limited budget. The objectiveu is to maximize the total rewards given the limited budget. Ding et al. (2013) researched the problem in the context of real-time bidding in an advertisement exchange where the costs are affected by the click behavior of the users and bidding behavior of the other advertisers. These types of application contexts are by nature more complex and have variable costs. Ingraining variable costs into the model makes it superior to using fixed costs although the model becomes more complex.

Ding et al. (2013) introduced two different upper confidence bound (UCB) algorithms as solutions for this specific type of MAB-BV problem in the context of online advertisement exchange. For the first UCB algorithm, the learning process is separated into clear exploration and exploitation phases. The budget constraint is only imposed on the exploration phase. The objective is to efficiently find the best performing arm during the exploration phase with the limited budget before exploiting it throughout the exploitation phase while using the rest of the budget. For the second algorithm, the costs are constrained by the limited budget regardless of the process being in the exploration or exploitation phase.

2.3 Decision policies as solutions for the MAB problem

The challenge for maximizing the rewards and minimizing the regret is to determine an optimal decision policy. The goal is to find the decision policy that can be implemented to the context and will converge to the optimal arm as fast as possible while minimizing the regret. There are multiple existing and well researched decision policies that can be implemented in different contexts. Three different widely used strategies will be introduced in the following subsections.

2.3.1 ϵ-greedy strategy

A well-known and widely used decision policy is the ϵ-greedy decision policy (Auer and Cesa-Bianchi, 2002). It is a simple decision policy and can be generalized for many sequential decision problems (Kuleshov and Precup, 2014). The policy allocates pulls for the arm with the highest average reward with the probability $1 - \epsilon$ and explores randomly chosen arm with the probability ϵ. Nevertheless, the regret of this decision policy will grow linearly instead of logarithmically due to the constant probability of exploration.

The decision policy can be made more efficient and achieves logarithmically growing regret if the ϵ is allowed to decrease to zero with the rate $\frac{1}{n}$, where n is the current time step (Auer and Cesa-Bianchi, 2002). With this modification, the decision policy will prioritize exploration at the beginning of the learning process while decreasing the amount of exploration over time. As a result, the process will converge towards the optimal arm while minimizing the regret by decreasing the amount of exploration at the end of the process.

With the value $\epsilon = 1$, the allocation strategy becomes an equal allocation strategy where the probability of pulling each arm is equal. According to Scott (2010), this is a poor allocation strategy because the algorithm continues to explore even when the optimal solution has come apparent. Therefore, the equal allocation strategy will never converge towards the optimal arm if there are not any context specific modifications.

2.3.2 Upper confidence bound

Kuleshov and Precup (2014) describe the upper confidence bound (UCB) decision policy as an "elegant implementation of the idea of optimism in the face of uncertainty". Auer and Cesa-Bianchi (2002) showed that the proposed decision policy, UCB1, achieves logarithmic cumulative regret meaning the regret does not increase linearly or exponentially. According to the UCB1 policy, every arm is played once at the beginning of the process. Thereafter, the arm that maximizes $\mu_k + \sqrt{\frac{2ln(n)}{n_k}}$ is played, where μ_k is the average reward from arm k, n_k is the number arm k has been played and n is the overall number of plays done so far.

2.3.3 Thompson Sampling

During the recent years, Thompson Sampling has become rapidly more used heuristic for creating a decision policy for different contexts. It has applications in various areas, e.g., website optimization, internet advertising, and revenue management (Russo et al., 2018). The idea behind the heuristic was initially suggested by Thompson in 1933 as a solution in the context of clinical trials.

The idea of Thompson sampling, also referred as random probability matching, is to allocate pulls for different arms according to the current belief of the optimality of each arm. The optimality of arms is quantified by the posterior distribution of the reward function which is then used as the pull weights for the different arms. The weights are defined with the equation

$$\pi_{at} = P(a \text{ is optimal} \mid y_t), \tag{1}$$

where a refers to the index of the arm, and y_t is the data gathered up to time t.

Scott (2010) describes broad applicability to different contexts and easy implementation as the advantages of Thompson Sampling. When compared to the efficiency of the ϵ-greedy strategy on general level, Thompson sampling algorithm gathers sufficient amount of information about a sub-optimal arm and after determining that the arm is sub-optimal allocates the pulls for other arms. Thus, Thompson sampling balances exploration and exploitation in a natural way, and it is compatible with batch updates.

2.3.4 Choosing the most applicable decision policy

As the cumulative regret needs to be minimized over the learning process, the chosen decision policy should explore the different arms efficiently while being applicable to the specific problem context. Additionally, this should reflect on the computational efficiency of the entire process. We can conclude from the previous sections that ϵ-greedy decision policy is a relatively naive approach though easy to apply to the context. UCB would likely be less efficient compared to Thompson Sampling due to exploring the different arms with the same amount of pulls. Thompson Sampling balances the exploration and exploitation more effectively compared to ϵ-greedy and UCB decision policies. Additionally, it is also applicable with batch updating which makes it great candidate as an optimal decision policy for the creator marketing context.

3 Formulation of the BB-MAB-TS budget allocation algorithm

To formulate the BB-MAB-TS budget allocation algorithm, the budget allocation problem in the creator marketing context is at first converted into a BB-MAB problem (Section 3.1). Then, to determine the budget allocation weights π_{kt}, at first the contextual reward function is formulated (Section 3.2) and Thompson Sampling is introduced as the sampling method for sampling the budget allocation weights (Section 3.3). Lastly, Poisson distribution based posterior distribution updating (Section 3.4), constraints for the active learning system (Section 3.5), and the BB-MAB-TS budget allocation algorithm as pseudocode (Section 3.6) are introduced.

3.1 Converting the budget allocation problem into the BB-MAB problem

In the problem context under consideration, an agent is playing a slot machine with $k \in \{1, ..., K\} \in \mathbb{Z}_+$ arms over $t \in \{1, ..., T\} \in \mathbb{Z}_+$ decision rounds. Due to the batched setting, the agent makes a decision which arms to pull and how many times one arm will be pulled during one decision round. The decisions are made based on a decision policy π. The consecutive pulls of an individual arm during a decision round are indexed with $j \in \{1, ..., J\} \in \mathbb{Z}_+$. Therefore, k, t, and j create the basis for the active learning system. Furthermore, the total number of pulls is limited by a finite budget $B \in \mathbb{R}$, and each pull of an arm results with a reward and cost. The total costs are limited by the budget B.

As the process is limited by the budget B, the agent needs to allocate the budget over the decision rounds t between the different arms k. Therefore, it needs to make a decision on how to allocate the budget B into T parts on batch level, i.e., $\sum_{t=1}^{T} b_t = B$. Within one batch, the agent allocates b_t part of the budget on an arm level, i.e., $\sum_{k=1}^{K} b_{kt} = b_t$. The arm level budget b_{kt} is used for pulling the arm as long as there is budget left. An individual pull is allocated with b_{ktj} part of the budget, i.e., $\sum_{j=1}^{J} b_{ktj} = b_{kt}$. Therefore, the allocation process is constrained by

$$\sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{j=1}^{J} b_{ktj} = B. \tag{2}$$

During each decision round t, the agent samples arms from the population $k \in C$ of all the available arms based on the decision policy $\vec{\pi} = [\pi_{1t}, ..., \pi_{Kt}]^T$. The decision policy determines the weights on which the different arms will be sampled and how many times each arm can be pulled with the budget allocated for the specific arm during the decision round. The weights π_{kt} are sampled from the posterior distribution of the reward function. The sampling method is based on a heuristic called Thompson Sampling. Therefore, the budget allocation will be determined based on the current understanding of the optimality of each arm as described in Section 2.3.3.

After sampling and pulling the arms, the agent receives a reward $f_k(b_{kt})$ from an unknown reward distribution for each pull of an arm. Over the decision rounds T, the agent must learn about the optimality of the different arms with the objective to maximize the cumulative rewards and minimizing the cumulative regret. The agent achieves the objective by updating the decision policy π after observing the rewards after each decision round. Consecutively, the decision policy $\vec{\pi}$ is updated based on the rewards f_{kt} observed after the pulls of arms.

To conclude, the objective is to maximize the sum of all rewards accumulated over the decision rounds T. This is achieved by finding the optimal arm over the decision rounds through the active learning process. The objective function of the process is defined as

$$\max \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{j=1}^{J} f_{kt}(b_{ktj}). \tag{3}$$

The performance of a decision policy is measured through the regret r which is the difference between the expected reward of the optimal arm and the observed reward. The index of the optimal arm is denoted by $k = *$. Thus, the regret for one pull is

$$r_{ktj} = f_*(b_{ktj}) - f_{kt}(b_{ktj})). \tag{4}$$

The objective is to minimize the cumulative regret R of all the pulls done over the rounds T

$$R = \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{j=1}^{J} (f_*(b_{ktj}) - f_k(b_{ktj})). \tag{5}$$

The budget allocation is optimized on three different levels: 1) on a system level as B is allocated between the decision rounds t into b_t parts, 2) on a decision round level as b_t is allocated between the arms k sampled during the decision round t, and 3) on an arm level as every time an arm is pulled it is defined how much budget is allocated for that specific pull.

It can be concluded that the budget allocation problem in the creator marketing context can be formulated as a batched and budgeted multi-armed bandit (BB-MAB) problem. To further adapt the BB-MAB problem into the creator marketing campaign context, the arms k denote to price points that the deals are sent with, and the individual pulls of arms denote individual deals that are sent to the creators.

3.2 Formulation the reward function

There are multiple factors affecting whether or not an individual creator accepts the deal they are offered. One of the main factors is the price of the sent deal. Other factors are the busyness of the creator, appeal of the brand of the advertiser, and other random factors caused by the everyday life of the creator. The variance in the acceptance of the deals is referred as deal acceptance behavior. Due to the varying deal acceptance behavior of the different creators, there is uncertainty related to the acceptance rate α. Therefore, the mean acceptance rate $\mu_{\alpha_{kt}}$ for each price ρ_k needs to be learned during the exploration part of the learning process. The optimal price point is hypothetically the lowest price ρ_k for which the acceptance rate α_{kt} would be high enough to use the rest of the available budget B_t.

The reward function f_{kt} is used for describing the rewards from the observations of the sent deals and estimating their optimality. The main variables for deriving the reward function are: the set of creators, $c \in C$; the average views the creators' videos are expected to have with some price ρ_k, $\mu_{v_{ktj}} \in \mathbb{R}$; the price per view of the sent deal, $\rho_k \in \mathbb{R}$; the number of accepted deals with a specific price during one decision round, $a_{kt} \in \mathbb{Z}$; and the total number of sent deals with a specific price during one decision round, $d_{kt} \in \mathbb{Z}$.

The acceptance data a_{kt} of all the deals for each price ρ_k is gathered during each decision round for all the sent deals. Based on the number of accepted deals and total number of sent deals, the acceptance rate for one price ρ_k during a decision round t is calculated as $\alpha_{kt} = \frac{a_{kt}}{d_{kt}}$. The average deal value ν_{kt} for some price ρ_k during a decision round t is $\mu_{\nu_{kt}} = \frac{\nu_{kt}}{a_{kt}} = \rho_k \mu_{v_{kt}}$. Thus, the value of the deals sent with some price ρ_k is $\nu_{kt} = d_{kt}\mu_{\nu_{kt}}$ and the value of accepted offers is $\nu_{a_{kt}} = \alpha_{kt}\mu_{\nu_{kt}}$.

The overall goal of the optimization is to maximize the number of expected views v for one price group. This can be calculated as following

$$E[v] = a_{kt}\mu_{\nu_{kt}} = \alpha_{\rho_{kt}} d_{kt}\mu_{v_{kt}} = \alpha_{kt}\nu_{kt}\frac{1}{\mu_{\nu_{kt}}}\mu_{v_{kt}}.$$

It is known that $\nu_{kt} = b_{kt}$ which can be considered as a given constant. The ratio between the expected view count and value of the deals $\frac{\mu_{v_{kt}}}{\mu_{\nu_{kt}}}$ can be described as $\frac{1}{\rho_k}$ which is the inverse of the cost per view, i.e., the inverse of the the price. Therefore, the reward function simplifies to the form of

$$f_k(\alpha_{kt}, \rho_k) = \frac{\alpha_{kt}}{\rho_k}. \tag{6}$$

3.3 Determining the budget allocation weight matrix with Thompson Sampling

The underlying sampling method for determining the budget allocation weights is based on a heuristic called Thompson Sampling. The suggested decision policy is based on the idea from paper of Scott published in 2010. In the paper, he describes the Bayesian solution for the MAB problem. Scott solves the MAB problem by using Thompson Sampling as the heuristic for creating the decision policy for allocating pulls for different arms.

The budget allocation weights are sampled from the posterior distribution of the reward function using Thompson Sampling during each decision round. The weights $\pi_{kt} = P(k \text{ is optimal} \mid y_t)$ reflect the optimality of each price given the observed historical data y_t. Therefore, the weights are used to determine how much each price ρ_k is to be allocated with the budget based on the current belief of the optimality of the price. The weights also determine how many deals will be sent with each price which is limited by the allocated budget.

Assume a sequence of rewards observed up to time t. The rewards are saved in a matrix $M = [\vec{y}_1, ..., \vec{y}_T]$ as they are observed over the decision rounds t. During one decision round, the rewards are saved as a vector $\vec{y_t} = [f_{1t}, ..., f_{Kt}]^T$. Therefore, the Matrix M is the form of

$$M = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1T} \\ f_{21} & f_{22} & \cdots & f_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ f_{K1} & f_{K2} & \cdots & f_{KT} \end{bmatrix}. \tag{7}$$

When a deal is sent with the price ρ_k at time t, the agent receives reward from the reward distribution f_{kt} that is generated independently of the historical and other simultaneously sent deals. As the optimality of each price ρ_k is not known, the agent needs to do exploration during the active learning phase to find the optimal price.

The high level computational steps of the BB-MAB-TS budget allocation algorithm are outlined in the following list and in the Picture 2.

1. Sampling budget allocation weights π_{kt} from the posterior distribution of the reward function,
2. Allocating budget b_t between prices ρ_k according to the weights π_{kt},
3. Sending deals according to the allocation and observing rewards,
4. Updating the posterior distribution of the reward function f_{kt} according to the observed data,
5. Repeating the steps 1-4 until all of the budget B is used.

The algorithmic steps are also outlined as a flowchart in the Picture 2 below.

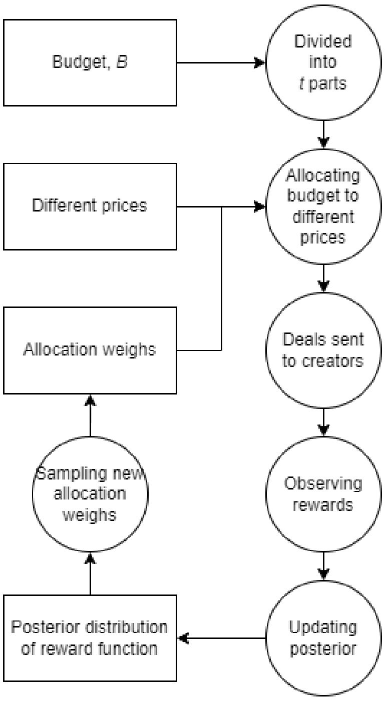

Figure 2: Flowchart of the algorithmic steps of the BB-MAB-TS dynamic budget allocation algorithm.

The following allocation vector $\vec{b}$ determines the budget allocation between the decision rounds

$$\vec{b} = \begin{bmatrix} b_1 & b_2 & \dots & b_T \end{bmatrix}. \tag{8}$$

The arm level allocation defined in Matrix 9 is achieved through weighing the budget determined by the vector $\vec{b}$ through utilizing the decision policy π that determines the weights. Therefore, $\vec{\pi}_t = [\pi_{1t}, ..., \pi_{Kt}]^T$ and the budget allocation weight matrix is $\Pi = [\pi_1, \pi_2, ..., \pi_T]$. Thus, the budget allocation matrix is in the form of

$$A = (\vec{b}\Pi^T)^T = \begin{bmatrix} b_1\pi_{11} & b_2\pi_{12} & ... & b_T\pi_{1T} \\ b_1\pi_{21} & b_2\pi_{22} & ... & b_T\pi_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ b_1\pi_{K1} & b_2\pi_{K2} & ... & b_T\pi_{KT} \end{bmatrix}. \tag{9}$$

The budget allocation matrix is used to determine how many deals are sent with every price. Conversely, referring back to the MAB problem, the allocation matrix denotes how much budget will be allocated to each pull. It also determines the magnitude of the reward received from the pull as the reward is directly proportional to the size of the budget allocated for the arm.

3.4 Posterior updating

Thompson Sampling is based on sampling the allocation weights from the posterior distribution of the reward function which reflects the current belief of the optimality of each price. As there is uncertainty related to the acceptance rate of the deals, a sent deal results as an accepted deal on average $\mu_{\alpha_{kt}}$. Thus, as the posterior distribution is updated based on the observed deal acceptance data, the ability of different prices to delivered value is learned about.

To model the posterior distribution, a starting prior distribution is needed. As shown by Agrawal and Goyal (2013), the prior can be any distribution, such as uniform, binomial or Bernoulli distribution. Nevertheless, Poisson distribution is better for modeling the deal acceptance as it describes the probability of getting some number of events happening given λ that represents the expected number of events. Additionally, the deal acceptance behavior during each decision round is independent of the previous rounds which Poisson distribution is well suited for. Therefore, assume that deal acceptances follow Poisson distribution

$$a_{kt} \sim Pois(\lambda_{kt}) \tag{10}$$

Also, assume that λ_{kt} is not a fixed number but it is distributed according to Gamma conjugate prior with parameters α_{kt} and β_{kt}

$$\lambda_{kt} \sim Gamma(\alpha_{kt}, \beta_{kt}). \tag{11}$$

As deal acceptances are observed over the decision rounds, the posterior distribution is updated based on the observed data. At the beginning of each decision round, the budget allocation weights π_{kt} are sampled from the posterior distribution.

Historical deal sending and acceptance data can be directly used to model α_{kt} and β_{kt} for the gamma conjugate. To model them, the entire historical data can be used up to the current decision round. Therefore, the parameters are

$$\alpha_{kt} = \sum_{n=1}^{t} a_{kn} \frac{1}{\rho_k},$$
$$\beta_{kt} = \sum_{n=1}^{t} d_{kn} \frac{1}{\rho_k}.$$

3.5 Constraints for the active learning system

The aim of the BB-MAB-TS algorithm is to find the smallest price for which the acceptance rate is high enough to use the rest of the budget the available after the decision round. Therefore, the learning process is constrained by the equation 12

$$\sum_{t=t+1}^{T} \mu_{\alpha_{kt}} \rho_k \mathbb{E}[v_C] - B_{t+1} \geq 0, \tag{12}$$

where $\mu_{\alpha_{kt}}$ is the mean acceptance rate for one price ρ_k during some decision round t, $\mathbb{E}[v_C]$ is the combined amount of views for the videos in the available creator pool $c \in C$, and B_{t+1} is the available budget after the decision round t. The first term in the Equation 12 quantifies how much budget the price ρ_k would be able to burn with the current belief of the mean acceptance rate for that specific price if all the creators $c \in C$ would be sent a deal with that price.

The sooner the optimal price point is found during the exploration process, the sooner the total campaign performance is maximized which is measured with the sum of all the views acquired with the given fixed budget B

$$\text{max. } V = \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{j=1}^{J} b_{ktj} \rho_k a_{ktj} \mathbb{E}[v_{ktj}], \tag{13}$$

where b is the allocated budget, ρ_k price, a_{ktj} the acceptance of deals, and $\mathbb{E}[v_{ktj}]$ the estimated views for the advertisement done with the sampled creator with the given index ktj.

Therefore, it can be concluded that the main constraints are the available budget B, the creator set C, the amount of decision rounds K, and the accessible price range K.

3.6 BB-MAB-TS algorithm as a solution for the budget allocation problem

This section outlines the algorithmic logic of the BB-MAB-TS dynamic budget allocation algorithm. The high level computational steps are:
1. Sampling budget allocation weights π_{kt} from the posterior distribution of the reward function,

2. Allocating budget b_t between prices ρ_k according to the weights π_{kt},
3. Sending deals according to the allocation & observing rewards,
4. Updating the posterior distribution of the reward function f_{kt} according to the observed data,
5. Repeating the steps 1-4 until all of the budget B is used.

The logic of the BB-MAB-TS algorithm is given as pseudocode in Algorithm 1:

Algorithm 1: BB-MAB-TS algorithm

```
Init:
$k \leftarrow 1$ ;                                        // price indices
$t \leftarrow 1$ ;                                        // time horizon indices
$j \leftarrow 1$ ;                                        // deal indices
$B \leftarrow B$ ;                                        // budget
$C \leftarrow C$ ;                                        // creator set
$\mathbb{E}[\nu] \leftarrow [\mathbb{E}[\nu_1], \ldots, \mathbb{E}[\nu_T]]$ ;      // estimated views
$\rho \leftarrow [\rho_1, \rho_2, ..., \rho_K]^T$ ;                  // prices
$\Pi \leftarrow [\pi_1, \pi_2, ..., \pi_T]$ ;                    // allocation weights
$a \leftarrow [a_1, a_2, ..., a_K]^T$ ;                      // acceptance data
$\alpha \leftarrow [\alpha_1, \alpha_2, ..., \alpha_T]$ ;                  // $\alpha$ for Gamma conjugate
$\beta \leftarrow [\beta_1, \beta_2, ..., \beta_T]$ ;                   // $\beta$ for Gamma conjugate
while $t \leq T$ do
    $b_t \leftarrow \frac{B_t}{T-t+1}$ ;        // Allocates budget $b_t$ for the decision round
    for $k \leq K$ do
        Sample $\pi_{kt}$ for each price ;
        $b_{kt} = b_t \pi_{kt}$ ;                  // weigh the budget for each price
        while $j \leq J$ do
            $J = \frac{b_{kt}}{\rho_k \mathbb{E}(v_{kt})}$ ;          // quantifies how many deals are sent
            Select $J$ creators from the creator pool $C$ Send each creator a deal
              with the value $b_{ktj} = \rho_k \mathbb{E}(v_{kt})$ ;
            Observe the acceptance data $a_{ktj}$ for all the sent deals ;
            $f_{kt} = \frac{\alpha_{kt}}{\rho_k}$ ;                          // computes rewards
            Update the posterior distribution of the reward function ;
        end
    end
end
```

4 Discussion

In the current form, the BB-MAB-TS algorithm is focused on optimizing the budget allocation between the different prices within one decision round but is not fine-tuned to optimize the allocation between different decision rounds or on a deal level. Therefore, multiple future research topics arising from this thesis. These topics are optimizing the convergence time to the optimal price, computational efficiency of the BB-MAB-TS algorithm, optimizing budget allocation between the decision rounds and on a deal level, applying the BB-MAB-TS algorithm to other resource and budget allocation problems, and statistical analysis of the performance of the deals. These aforementioned topics can be researched through, e.g., simulation, practical implementations, and developing the mathematical optimization model forward.

4.1 Optimizing convergence time to the optimal price and budget allocation on all levels

Minimizing the convergence time towards the optimal price could be achieved by fine-tuning the exploration phase of the learning process. One approach to achieve this is to allocate the budget to some limited number of prices during a decision round, e.g., sending deals with only five different prices during each decision round at the beginning of the exploration phase. These prices would be called active prices and the amount of active prices could be decreased over time as the confidence level about the optimality of the most optimal prices would increase. Thus, the budget would not be wasted on the non-optimal prices, and the process converges more quickly to the optimal price.

The risk is that the initially chosen price range would be far from the global optimal price which would as a consequence result in longer exploration phase and larger cumulative regret. To diminish this risk, historical data from previous campaigns should be used to determine the initial price range if it is available.

Alternatively, to optimize the budget allocation between the decision rounds, the budget b_t allocated to each batch could be determined according to some exponential function. This way less budget would be used at the beginning of the exploration phase. While the learning process advances and the confidence level about the optimality of the different prices increases, the budget usage increases. This should also decrease the cumulative regret. The exponential nature of allocation could be achieved, e.g., by introducing some tuning parameter γ that Scott (2010) also suggested in his paper.

Additionally, the budget allocation process could be clearly separated into exploratory and exploitation phase. This could be achieved by clearly allocating, e.g., 20% of the total budget for exploration and 80% for exploitation. Moreover, different prices could be attached with a confidence level indicating the minimum confidence level of the belief that some price is optimal. Prices with low confidence level would not be allocated budget and this budget could be allocated to the most optimal prices. Therefore, the more optimal prices would be explored increasingly more which would lead to increase of the confidence level of their optimality.

Currently, the BB-MAB-TS algorithm does not determine how the creators are chosen to one batch after the budget has been allocated to the different prices. In practice, the estimated view counts for the videos of the creators varies for which reason the value of the sent deals also varies. Therefore, some kind of mechanism that enables choosing creators within the budget limit should be built into the algorithm. The mechanism should select creators who are sent the deals within the budget limitation. Additionally, creators with smaller estimated view count could be preferred at the beginning of the learning process to gather more deal acceptance data and understanding of the acceptance behavior of the creators.

Testing the convergence and performance of the BB-MAB-TS algorithm could be achieved through simulations conducted, e.g., in simulation environments created with R or Python. Hypothetically, the most challenging part of the simulation is creating the pseudo data for simulating the uncertainty connected to the acceptance rate of the deals. The acceptance rate data could be created by disturbing sine function data with some noise that models the uncertainty to the data.

4.2 Applying the BB-MAB-TS algorithm to different resource and budget allocation problems

The BB-MAB-TS algorithm could be applied to different contexts where one needs to allocate some finite amount of resources between competing alternatives while getting binary reward data from the allocation process. To apply the algorithm, a contextual reward function needs to be formulated and the data saving and handling should be done with the context in mind. Additionally, to model the posterior distribution of the rewards function, one needs to decide a distribution that is applicable for that specific context.

The kind of contexts that the BB-MAB-TS algorithm could be applied to could be, e.g., other types of online marketplaces or platforms, conversion testing, and determining prices for consumer goods. Other online marketplaces where either budget needs to be dynamically allocated or a price determined, usually gather similar data and the dynamics are similar. For determining the prices of consumer goods, one would need to create a setting where we would test out different prices with a large number of customers that are offered the product with different prices. The dynamic for consumer good price optimization would be similar to the online one but the data should be gathered manually by an observer.

4.3 Statistical analysis of the performance of the deals

Through gathering data from the acceptance behavior of the creators and the performance of their video, the selection process of which creators are sent deals could be optimized. Thus, based on the historical data, the ability to deliver value measured by the ratio between deal price and realized value for creator could be learned. In the future campaigns, creators that have historically performed better, could be preferred as campaign participants.

Additionally, the variance of performance of the advertisements of some creator between different types of products or advertisers could be learned by analyzing the conversions of the advertisements and the total view amounts. Thus, the deals could be sent to such creators that have been historically performing well with the types of customer and type of advertisement than in previous campaigns.

Lastly, the acceptance behavior of the individuals creators could be learned. The different aspects to learn would be the probability of accepting some specific type of deal and the price point that they are most probable to accept a deal while optimizing the value per price ratio. Therefore, an individual price optimization model for each creator could be created if the system would learn about the acceptance behavior of the creators deeply enough.

5 Conclusion

This thesis introduces the BB-MAB-TS algorithm as a solution to the dynamic budget allocation problem in the context of creator marketing campaigns. The system allocates the budget over multiple decision rounds while simultaneously learning which price point is optimal for sending the campaign deals. The budget allocation problem is at first converted into a BB-MAB problem. Then, the contextual Thompson Sampling based sampling method is introduced for determining the budget allocation weights.

The first contribution of the thesis is that the given contextual problem can be converted into BB-MAB problem. Achieving this requires introducing the batched structure and budget constraints for the exploration-exploitation process. Secondly, introducing Thompson Sampling as the underlying heuristic for allocating the budget.

Lastly, multiple different research topics arose from this thesis. These topics aim at further developing the BB-MAB-TS algorithm to optimize the budget allocation more holistically both between batches and on a deal level. These modifications would also result at optimizing the convergence time towards the optimal price. Additionally, the BB-MAB-TS could be applied to other resource allocation problems with the required modifications.

References

Shipra Agrawal and Navin Goyal. Further Optimal Regret Bounds for Thompson Sampling. page 9, 2013.

Peter Auer and Nicolo Cesa-Bianchi. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235–256, 2002.

Vashist Avadhanula, Riccardo Colini Baldeschi, Stefano Leonardi, Karthik Abinav Sankararaman, and Okke Schrijvers. Stochastic bandits for multi-platform budget optimization in online advertising. In *Proceedings of the Web Conference 2021*, pages 2805–2817, Ljubljana Slovenia, April 2021. ACM. ISBN 978-1-4503-8312-7. doi: 10.1145/3442381.3450074. URL https://dl.acm.org/doi/10.1145/3442381.3450074.

Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-Armed Bandit with Budget Constraint and Variable Costs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 27(1):232–238, June 2013. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v27i1.8637. URL https://ojs.aaai.org/index.php/AAAI/article/view/8637.

Zijun Gao, Yanjun Han, Zhimei Ren, and Zhengqing Zhou. Batched Multi-armed Bandits Problem, October 2019. URL http://arxiv.org/abs/1904.01763. arXiv:1904.01763 [cs, math, stat].

Cem Kalkanli and Ayfer Ozgur. Asymptotic Performance of Thompson Sampling in the Batched Multi-Armed Bandits. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 539–544, July 2021. doi: 10.1109/ISIT45174.2021.9517843. URL http://arxiv.org/abs/2110.00158. arXiv:2110.00158 [cs].

Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems, February 2014. URL http://arxiv.org/abs/1402.6028. arXiv:1402.6028 [cs].

T. L. Lai and Herbert Robbins. Asymtotically Efficient Adaptive Allocation Rules. *Advances in Applied Mathematics*, 6:4–22, 1985. ISSN 0196-8858. doi: 10.1016/0196-8858(85)90002-8.

Aditya Mahajan and Demosthenis Teneketzis. Multi-Armed Bandit Problems. pages 121–151. October 2007. ISBN 978-0-387-27892-6. doi: 10.1007/978-0-387-49819-5_6.

Herbert Robbins. Some Aspects of the Sequential Design of Experiments. *American Mathematical Society*, 58(5):527 – 535, 1952.

Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A Tutorial on Thompson Sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, July 2018. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000070. URL https://www.nowpublishers.com/article/Details/MAL-070. Publisher: Now Publishers, Inc.

Steven L. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010. ISSN 1526-4025. doi: 10.1002/asmb.874. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asmb.874.

Steven L. Scott. Multi-armed bandit experiments in the online service economy. *Applied Stochastic Models in Business and Industry*, 31 (1):37–45, 2015. ISSN 1526-4025. doi: 10.1002/asmb.2104. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.2104. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asmb.2104.

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. ISSN 0006-3444. doi: 10.2307/2332286. URL https://www.jstor.org/stable/2332286. Publisher: [Oxford University Press, Biometrika Trust].

Yingce Xia, Haifang Li, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Thompson Sampling for Budgeted Multi-armed Bandits. page 7, 2015.