UNIVERSITÀ COMMERCIALE ”LUIGI BOCCONI”

BSc in Economics, Management and Computer Science

Analysis of Bandits with Replenishable Knapsacks:

Algorithms and Applications

Tutor
Prof. Andrea Celli
Department of Computing Sciences

Candidate: *Luigi Liu*

Academic Year: 2024

Contents

Introduction

I study the Bandit with Replenishable Knapsack (BwRK) framework, proposed by Kumar and Kleinberg [16], which extends the well-known Bandit with Knapsack (BwK) framework. While the BwK framework assumes nonnegative consumption and ends when the budget of any resource becomes nonpositive, BwRK removes this limitation by allowing the budget to be replenished.

In the first chapter, I introduce the problem of bandit with replenishable knapsack by explaining, in a broad way, the multi-armed bandit and bandit with knapsack frameworks. In the same chapter, I discuss the motivations and challenges of studying the BwRK framework. In the second chapter, I outline the basic setup of the BwRK framework and the primal-dual template proposed by Martino Bernasconi et al. [5].

Moving to chapter three, I present several algorithms that can be used within the Primal-Dual Template. Moreover, I include their regret bounds, which are essential for deriving the overall regret bound of the primal-dual template. In the last chapter, I explore the primal-dual template in depth through two applications: dynamic pricing and online ad allocation. I experiment with how the primal-dual template functions and conclude with the results of their behavior. Finally, I propose a dynamic per-iteration budget variable, which, although lacking an analytic analysis, makes resource consumption more reasonable, improving the regret bound.

Chapter 1

Online Machine Learning

1.1 Online learning

Traditional machine learning models typically use a batch learning method, where a set of data is collected in advance and then used to train the model. However, this approach often has scalability limitations in large-scale applications, as it requires retraining the model whenever new training data becomes available.

Online machine learning, on the other hand, processes data as it comes in sequentially, continuously learning and refining the model to improve future predictions. Thus, online learning addresses the limitations of batch learning, providing more efficient and scalable algorithms that can handle machine learning tasks where data is not only of large volume but also arrives with high velocity.

Online learning is performed in a sequence of T consecutive rounds, where at each round t, the decision-maker predicts p_t. Subsequently, it receives feedback, which may be the prediction's correct outcome or reward signal. Based on the feedback, the decision-maker suffers from a loss, $l\left(p_t, \text{feedback}_t\right)$, and uses the same feedback to update itself and improve predictions for future data. The goal of the algorithm is to minimize the cumulative loss $\sum_{t=1}^{T} l\left(p_t, \text{feedback}_t\right)$. A general template is shown in algorithm 1.

Algorithm 1 Online Learning

1: **for** $t = 1, 2, \ldots, T$ **do**
2: `predict` p_t
3: `observe` feedback$_t$
4: `suffer loss` $l\,(p_t, \text{feedback}_t)$
5: `update itself using` feedback$_t$

1.2 Multi-armed bandits

The multi-armed bandit (MAB) problem is an important branch of online learning where a decision-maker makes choices over time under uncertainty. In each round $t \in \{1, \ldots, T\}$, the decision-maker chooses an arm a_t, i.e., an action, from a given set A and receives a reward r_t[1]. Action a_t depends only on the history $\{a_1, r_1, \ldots, a_{t-1}, r_{t-1}\}$.

The fundamental challenge of the bandit problem is that the decision-maker cannot observe the reward before choosing the action. Therefore, the decision-maker needs to gain knowledge by *exploring* different arms and maximize the reward by *exploiting* all available information. However, too much exploration would waste resources on inferior alternatives, while too much exploitation would cause the decision-maker to miss better opportunities. Hence, we have a trade-off between exploration and exploitation.

MAB problem features three types of feedback:

- bandit feedback: the decision-maker only observes the reward of the chosen arm.
- full feedback: the decision-maker observes the rewards of all arms in A.
- partial feedback: the decision-maker observes the rewards of some arms that could have been selected in addition to the one chosen.

How do we evaluate the effectiveness of an algorithm across different problem instances when the reward distributions are unknown and differ from one instance to another? A common method involves measuring the algorithm's total reward and comparing it to the best-arm benchmark `OPT`, the reward obtained from consistently choosing the optimal arm. This metric is called *regret*, and is defined as:

$$R(T) = \texttt{OPT} - \sum_{t=1}^{T} r_t(a_t) \tag{1.1}$$

[1]Reward and loss can be interchanged in context by multiplying by -1.

Hence, while the object of the algorithm is to maximize the total reward $\sum_{t=1}^{T} r_t(a_t)$, the approach of minimizing regret inherently seeks a balance between exploration and exploitation to achieve theoretical optimal performance over time.

1.3 Bandits with replenishable knapsacks

We begin with Bandits with Knapsacks (BwK) since Bandits with Replenishable Knapsacks (BwRK) extends from the BwK framework.

BwK is a general framework for multi-armed bandits problems with global constraints. Initially, d numbers of budgets were given to the algorithm. In each round $t \in \{1, \ldots, T\}$, a decision-maker chooses an arm a_t from a given set A and receives both the reward r_t and a set of d resources consumption $(c_1, \ldots, c_d)$ with $c_i \geq 0 \ \forall i \in \{1, \ldots, d\}$.The objective remains to maximize the cumulative reward function; however, the procedure terminates either after T rounds or when the cumulative consumption of any resource i exceeds its assigned budget B_i. A general template is shown in algorithm 2.

Algorithm 2 Bandits with Knapsacks (BwK)

Input: T, A, `budgets for` d `resources` $B_1, \ldots, B_d$.
for $t = 1, 2, \ldots, T$ **do**
    `choose an arm` $a_t \in A$
    `observe reward and consumption` $(r_t, c_{t,1}, \ldots, c_{t,d})$.
    `stop if` $\exists i \in \{1, \ldots, d\} : \sum_{j=1}^{t} c_{j,i} > B_i$

BwK problem is more complex than classical bandit problems for three reasons:

- While in bandits, one would usually try the different arms first, this approach is not viable in the case of BwK due to the potential risk of exhausting the available resources with such exploration.
- In bandits, the goal is to identify which arm yields the highest reward, whereas, in BwK, the same arm could consume excessive resources, so it is crucial to weigh the total reward against resource consumption over the entire time horizon.
- In bandits, the best arm is consistently chosen during the exploitation phase;

however, the choice of the arm in BwK depends on the current availability of resources, i.e., the algorithm might select a 'worse' arm as the budget is exhausted.

One of the main assumptions in the BwK model is that the resources consumption has to be nonnegative, i.e. $c_{t,i} \geq 0$ for all $t \in \{1, \ldots, T\}, i \in \{1, \ldots, d\}$. The BwRK framework removes this restriction, allowing resources to be replenished over time.

In this framework, the problem becomes even more complicated. Since resources can be replenished, decisions are no longer just about which action to take based on current rewards and resource consumption but also about when it might be best to take certain actions in anticipation of resource replenishment. Therefore, the strategy must consider the trade-off between the current resources and saving or spending resources in anticipation of future replenishments. Furthermore, if the replenishment rates or amounts are uncertain or vary over time, this uncertainty adds to the complexity of the problem.

Although the BwK framework presents several motivating applications, such as dynamic pricing with limited supply, online ad allocation, and repeated auctions, BwRK can handle scenarios closer to the real world. For instance, under dynamic pricing with limited supply, the supplier can close the store for a round and purchase new goods; similarly, under online ad allocation, the advertiser might increase their ad investments.

Chapter 2

Bandits with Replenishable Knapsacks (BwRK)

2.1 Basic set-up

The algorithm is performed in T rounds and has d resources with their respective budget $B_1, \ldots, B_d$. In each round $t \in \{1, \ldots, T\}$, the decision-maker selects an action a_t from an non-empty set of actions A. Following each action, it receives a reward function $r_t : A \to [0, 1]$ and a resource-consumption function $\boldsymbol{c}_t : A \to [-1, 1]^d$ of the d resources. In round t, resource i is depleted if $c_{t,i}(a_t) > 0$ and restored for a positive amount when $c_{t,i}(a_t) < 0$. The goal is

$$\text{max} \sum_{t=1}^{T} r_t(a_t) \text{ such that } \sum_{t=1}^{T} c_{t,i}(a_t) \leq B_i \; \forall i \in \{1, \ldots, d\} \tag{2.1}$$

In the BwRK framework, the process terminates when the time horizon, T, is reached because the algorithm can restore resources as they are depleted, enabling continuous decision-making up to the final round.

Let ρ_i be the resource i's per-iteration budget, which is defined as $\rho_i = B_i/T \geq 0$, and let $\boldsymbol{\rho} = [\rho_1 \; \cdots \; \rho_d]^T \in \mathbb{R}^m$. Moreover, given the reward and resource-consumption

functions, we define the Lagrangian function as

$$\mathcal{L}_{r,\boldsymbol{c}}(a, \boldsymbol{\lambda}) \coloneqq r(a) + \langle \boldsymbol{\lambda}, \boldsymbol{\rho} - \boldsymbol{c}(a) \rangle \ \forall a \in A, \boldsymbol{\lambda} \in \mathbb{R}^d \tag{2.2}$$

2.1.1 Adversarial setting

In an adversarial setting, the sequence of inputs $\boldsymbol{\omega} = \{(r_1, \boldsymbol{c}_1), \ldots, (r_T, \boldsymbol{c}_T)\}$ is determined by an oblivious adversary. The total reward from the best fixed unconstrained strategy of A is defined by

$$\texttt{OPT}_{\boldsymbol{\omega}} = \sup_{a \in A} \sum_{t=1}^{T} r_t(a) \tag{2.3}$$

Under the adversarial setting, the primal-dual template makes the following assumption:

Assumption. 1. *In an adversarial setting, there exists a void action $\varnothing \in A$ and a constant $\beta \geq 0$ such that $c_{t,i}(\varnothing) \leq -\beta$ for all resources $i \in \{1, \ldots, d\}$ and $t \in \{1, \ldots, T\}$.*

2.1.2 Stochastic setting

In a stochastic setting, each input pair $(r_t, \boldsymbol{c}_t)$ is drawn i.i.d. from a fixed yet unknown distribution $\mathcal{P}$ over a set of possible input pairs. Furthermore, we denote a linear program by $\texttt{OPT}^{\text{LP}}_{\bar{r},\bar{\boldsymbol{c}}}$, which selects the optimal action mixture that maximizes expected reward while adhering to resource consumption constraints:

$$\texttt{OPT}^{\text{LP}}_{\bar{r},\bar{\boldsymbol{c}}} = \sup_{\xi \in \Xi} \mathbb{E}_{a \in \xi}\left[\bar{r}(a)\right] \ \text{s.t.} \ \mathbb{E}_{a \in \xi}\left[\bar{\boldsymbol{c}}(a)\right] \leq \boldsymbol{\rho} \tag{2.4}$$

where $\bar{r}$ is the expected reward function, $\bar{\boldsymbol{c}}$ is the expected resource-consumption function, and Ξ is a set of action mixtures[1], i.e. the set of probability measures on the Borel sets of A. Hence, the upper bound of expected reward is $T \cdot \texttt{OPT}^{\text{LP}}_{\bar{r},\bar{\boldsymbol{c}}}$.

Under the stochastic setting, the primal-dual template makes the following assumption:

[1] We assume that all possible reward function r_t and resource-consumption function $\boldsymbol{c}_t$ are measurable with respect to every possible measure $\xi \in \Xi$.

Assumption. 2. *In a stochastic setting, there exists a void action $\varnothing \in A$ and a constant $\beta \geq 0$ such that $\mathbb{E}[c_i(\varnothing)] \leq -\beta$ holds for all resources $i \in \{1, \ldots, d\}$, where the expectation is with respect to the draw of the $\boldsymbol{c}$ from $\mathcal{P}$.*

2.2 Primal-dual template

The primal-dual template, proposed by Martino Bernasconi et al. [5], is a best-of-both-worlds algorithm that can tackle BwRK problems. The template ensures a regret bound of $\tilde{O}\left(T^{1/2}\right)$ in a stochastic setting. Furthermore, it guarantees a constant competitive ratio α, defined as the ratio of the reward from the optimal fixed policy to the (expected) reward of the algorithm, that recovers the standard guarantees of adversarial bandits when replenishment is not allowed and $B = \Omega(T)$. In the case of a positive constant of per-round replenishment, the competitive ratio is improved.

2.2.1 Algorithm

The framework relies on two regret minimizer algorithms for a set $\mathcal{W}$ implemented with the following functions: (i) the function `NEXTELEMENT()` selects and returns an element $w_t \in \mathcal{W}$, and (ii) the function `OBSERVEUTILITY(·)` receives some form of feedback and uses it to adjust its internal state. These regret minimization algorithms aim to minimize the cumulative regret while respecting budget constraints. Let $\mathcal{I} \subseteq \{1, \ldots, T\}$ be a subset of rounds, we define the regret function by

$$\text{Regret}_{\mathcal{I}} \coloneqq \sup_{w^* \in \mathcal{W}} \sum_{t \in \mathcal{I}} \left(u_t(w^*) - u_t(w_t)\right) \tag{2.5}$$

When $\mathcal{I} = \{1, \ldots, T\}$, we denote the regret by $\text{Regret}_T = \text{Regret}_{\{1,\ldots,T\}}$.

Assuming the two regret minimizers, primal regret minimizer and dual regret minimizer, are available with the following characteristics. The primal regret minimizer $\mathcal{A}^P$ is an algorithm of any type of feedback depending on the problem environment that outputs an action $a_t \in A$ in each round t. Sequentially, it receives the realized utility

function $u_t^P(a_t) = r_t(a_t) + \langle \boldsymbol{\lambda}_t, \boldsymbol{\rho} - \boldsymbol{c}_t(a_t) \rangle$ as feedback. The dual regret minimizer $\mathcal{A}^D$ is a full feedback function that outputs $\boldsymbol{\lambda}_t$, which is used by the primal regret minimizer A^P, and receives the utility function $u_t^D : \boldsymbol{\lambda} \mapsto \langle \boldsymbol{\lambda}, \boldsymbol{c}_t(a_t) - \boldsymbol{\rho} \rangle$ as input.

The general structure of the primal-dual template is shown in algorithm 3. In each round t, if the budget for any resource i, denoted as $B_{t,i}$, is less than 1, the algorithm chooses the void action $\varnothing$ and updates the budgets accordingly, ensuring they never fall below 0. Otherwise, the primal minimizer plays action a_t at time t by invoking `NEXTELEMENT()`. Consequently, the algorithm updates the budget according to the realized costs $\boldsymbol{c}_t$. Next, both regret minimizers update their internal state through `OBSERVEUTILITY(·)`, taking the primal utility $u_t^P(a_t)$ and the dual utility function u_t^D as feedback. The algorithm ends upon reaching the time horizon T.

Algorithm 3 Primal-Dual template

Input: T, budgets for d resources $B_1, \ldots, B_d$, regret minimizers $\mathcal{A}^P$ and $\mathcal{A}^D$
Initialization: $B_{1,i} \leftarrow B_i \; \forall i \in \{1, \ldots, d\}$; initialize $\mathcal{A}^P$ and $\mathcal{A}^D$
for $t = 1, 2, \ldots, T$ **do**
    if $\exists i \in \{1, \ldots, d\} : B_{t,i} < 1$ **then**
        Primal action $a_t \leftarrow \varnothing$
        Observe reward $r_t(a_t)$ and costs $\boldsymbol{c}_t(a_t)$
        Update available budgets $\boldsymbol{B}_{t+1} \leftarrow \boldsymbol{B}_t - \boldsymbol{c}_t(a_t)$
    else
        Dual action $\boldsymbol{\lambda}_t \leftarrow \mathcal{A}^D$.NEXTELEMENT()
        Primal action $a_t \leftarrow \mathcal{A}^P$.NEXTELEMENT()
        Observe reward $r_t(a_t)$ and costs $\boldsymbol{c}_t(a_t)$
        Update available budgets $\boldsymbol{B}_{t+1} \leftarrow \boldsymbol{B}_t - \boldsymbol{c}_t(a_t)$
        Primal update
            $u_t^P(a_t) \leftarrow r_t(a_t) + \langle \boldsymbol{\lambda}_t, \boldsymbol{\rho} - \boldsymbol{c}_t(a_t) \rangle$
            $\mathcal{A}^P$.OBSERVEUTILITY $\left(u_t^P(a_t)\right)$
        Dual update
            $u_t^D : \boldsymbol{\lambda} \mapsto \langle \boldsymbol{\lambda}, \boldsymbol{c}_t(a_t) - \boldsymbol{\rho} \rangle$
            $\mathcal{A}^D$.OBSERVEUTILITY $\left(u_t^D\right)$

Intuitively, the primal regret minimizer aims to maximize rewards but might select actions that, while yielding high rewards, also consume considerable resources, making them less optimal. Here, the dual regret minimizer intervenes, applying a large lambda to penalize excessive resource use. Conversely, when the budget is sufficient, the dual regret minimizer should apply a smaller lambda, allowing the primal regret minimizer to choose actions that yield higher rewards.

2.2.2 General guarantees of the primal-dual template

Let $\mathcal{T}_G \coloneqq \{t \in \{1,\dots,T\} : \forall i \in \{1,\dots,d\}, B_{t,i} \geq 1\}$ be a set of rounds wherein $\mathcal{A}^P$ and $\mathcal{A}^D$ were invoked and $\mathcal{T}_\varnothing \coloneqq \{t \in \{1,\dots,T\} : \forall i \in \{1,\dots,d\}, B_{t,i} < 1\}$ be a set of rounds wherein the void action is taken. Let $\tau \in \mathcal{T}_\varnothing$ be the last time the void action was chosen. We partition $\mathcal{T}_G$ into two disjoint sets: (i) $\mathcal{T}_{G,<\tau} \coloneqq \{1,\dots,\tau\} \setminus \mathcal{T}_\varnothing$, and (ii) $\mathcal{T}_{G,>\tau} \coloneqq \{1,\dots,T\} \setminus \{1,\dots,\tau\}$. We further denote these regrets $\mathcal{R}^P_{\mathcal{T}_G}$, $\mathcal{R}^D_{\mathcal{T}_G<\tau}$, and $\mathcal{R}^D_{\mathcal{T}_G>\tau}$ as seen in 2.5, accumulated by the primal regret minimizer and dual regret minimizer. Finally, let $\mathcal{D} \coloneqq \left\{\boldsymbol{\lambda} \in \mathbb{R}^d_+ : \|\boldsymbol{\lambda}\|_1 \leq 1/\nu\right\}$ where $\nu = \beta + \rho$ indicating the budget availability at each round and how fast the budget can be restored.

Let $\alpha \coloneqq \nu/(1+\beta)$. Martino Bernasconi et al. [5] show that, in the adversarial setting, the algorithm 3 chooses actions such that the following inequality holds.

$$\sum_{t=1}^{T} r_t(a_t) \geq \alpha \cdot \texttt{OPT}_{\boldsymbol{\omega}} - \left(\frac{2}{\nu} + \mathcal{R}^P_{\mathcal{T}_G} + \mathcal{R}^D_{\mathcal{T}_{G,<\tau}}(\mathcal{D}) + \mathcal{R}^D_{\mathcal{T}_{G,>\tau}}(\mathcal{D})\right) \tag{2.6}$$

While in the stochastic setting, the algorithm ensures

$$\sum_{t=1}^{T} r_t(a_t) \geq T \cdot \texttt{OPT}^{\texttt{LP}}_{\bar{r},\bar{c}} - \left(\frac{2}{\nu} + \frac{1}{\nu}\mathcal{E}_{T,\delta} + \mathcal{R}^P_{\mathcal{T}_G} + \mathcal{R}^D_{\mathcal{T}_{G,<\tau}}(\mathcal{D}) + \mathcal{R}^D_{\mathcal{T}_{G,>\tau}}(\mathcal{D})\right) \tag{2.7}$$

with probability at least $1-\delta$ for $\delta \in (0,1]$ and $\mathcal{E}_{T,\delta} = \sqrt{8T \log(4dT/\delta)}$.

In both settings, the guarantees depend on the choice of primal and dual regret minimizers. Since the bound depends on the dual algorithm's regret before and after τ, the dual regret minimizer must be weakly adaptive. Moreover, to achieve a meaningful bound, sublinear bounds in T are necessary for both primal and dual regret minimizers.

Chapter 3

Algorithms of the Primal-Dual Template

3.1 Algorithms

I present some algorithms used as regret minimizers such that $\mathcal{R}^D_{\mathcal{T}_{G,<\tau}}(\mathcal{D})$, $\mathcal{R}^D_{\mathcal{T}_{G,>\tau}}(\mathcal{D})$, and $\mathcal{R}^P_{\mathcal{T}_G}$ all grow sublinearly in T.

3.1.1 Hedge

The Hedge algorithm is a well-known full-information algorithm for online prediction with expert advice. There are N experts denoted by $1, \ldots, N$. At each round t, each expert recommends action, and the decision-maker chooses an expert, i_t, according to a distribution $\boldsymbol{p}_t$ over the experts, where $p_{t,i} \geq 0$ is the probability of choosing expert i, and $\sum_{i=1}^{N} p_{t,i} = 1$. That decision-maker incurs a loss l_{t,i_t} for choosing expert i_t.

Algorithm 4 shows the logic of the Hedge algorithm. The algorithm maintains a weight vector $\boldsymbol{w}_t = \begin{bmatrix} w_{t,1} & \cdots & w_{t,N} \end{bmatrix} \in \mathbb{R}^N$ where, initially, each weigh must be non-negative. Typically, they are set to be equal, e.g., $w_{1,i} = 1$ for all i. The algorithm uses the normalized distribution to make predictions, i.e., $\boldsymbol{p}_t = \boldsymbol{w}_t / \sum_{i=1}^{N} w_{t,i}$. After the loss $\boldsymbol{l}_t$ is observed, the algorithm updates the weight vector according to the rule $w_{t+1,i} = w_{t,i} \cdot e^{-\eta l_{t,i}}$ for $i \in \{1, \ldots, N\}$, where $\eta \in [0, \infty]$ is the learning rate.

Algorithm 4 Hedge Algorithm

Input: $\eta \geq 0$
Initialization: $w_{1,i} = 1 \; \forall i \in \{1, \ldots, N\}$
for $t = 1, 2, \ldots, T$ **do** do
    `Set distribution` $\boldsymbol{p}_t = \frac{\boldsymbol{w}_t}{\sum_{i=1}^{N} w_{t,i}}$
    `Pick expert` $i_t \sim \boldsymbol{p}_t$
    `Incur loss` l_{t,i_t}
    `Update weight` $w_{t+1,i} = w_{t,i} \cdot e^{-\eta l_{t,i}} \; \forall i \in \{1, \ldots, N\}$

The expected loss of the algorithm at round t is given by

$$\mathbb{E}\left[l_{t,i_t}\right] = \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i} \tag{3.1}$$

which represents the weighted average loss of the experts according to the distribution selected by the decision-maker.

Theorem. 1. *Let $\eta > 0$ and assume all losses to be nonnegative. For any $j \in \{1, \ldots, N\}$, The Hedge algorithm satisfies:*

$$\sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i} \leq \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i}^2 + \frac{\ln N}{\eta} + \sum_{t=1}^{T} l_{t,j}$$

Note that $\sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i}$ is the total expected loss of the algorithm and by choosing $i^* = \arg\min_{i \in \{1,\ldots,N\}} \sum_{t=1}^{T} l_{t,i}$, the minimum loss that can be attained is $\sum_{t=1}^{T} l_{t,i^*}$. Thus, the expected regret function can be expressed

$$\mathbb{E}\left[\text{Regret}_T\right] = \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i} - \sum_{t=1}^{T} l_{t,i^*}$$

Since theorem 1 holds for any expert, we conclude with the following theorem by choosing an appropriate learning rate η.

Theorem. 2. *There exists a learning rate, $\eta > 0$, such that the Hedge algorithm has an expected regret bound of $O(\sqrt{T \log N})$ for all $T \geq 1$.*

3.1.2 EXP3

The EXP3 algorithm stands for ”exponential-weight algorithm for exploration and exploitation”. It is similar to the Hedge algorithm but with an addition step: it introduces the random loss, $\hat{l}_{t,i} = \frac{l_{t,i_t}}{p_{t,i_t}} \mathbb{I}_{i_t=i}$, which is used for updating the weights. EXP3 is nearly optimal, meaning that its expected regret cannot be significantly improved in the worst-case scenario. The general structure of EXP3 is outlined in algorithm 5.

Algorithm 5 EXP3

Input: $\eta \geq 0$
Initialization: $w_{1,i} = 1 \; \forall i \in \{1, \ldots, N\}$
for $t = 1, 2, \ldots, T$ **do** do
    `Set distribution` $\boldsymbol{p}_t = \frac{\boldsymbol{w}_t}{\sum_{i=1}^{N} w_{t,i}}$
    `Pick expert` $i_t \sim \boldsymbol{p}_t$
    `Incur loss` l_{t,i_t}
    `Let` $\hat{l}_{t,i} = \frac{l_{t,i_t}}{p_{t,i_t}} \mathbb{I}_{i_t=i}$
    `Update weight` $w_{t+1,i} = w_{t,i} \cdot e^{-\eta \hat{l}_{t,i}} \; \forall i \in \{1, \ldots, N\}$

Theorem. 3. *EXP3 algorithm attains an expected regret bound of* $O(\sqrt{TN \log N})$ *for all* $T \geq 1$.

3.1.3 EXP3-IX

EXP3-IX is a variant of the EXP3 algorithm that uses the Implicit eXploration (IX) loss estimates defined as

$$\tilde{l}_{t,i} = \frac{l_{t,i}}{p_{t,i} + \gamma} \mathbb{I}_{i_t=i}$$

for all i and an appropriately chosen $\gamma > 0$. The IX loss function adjusts the probability distribution for arm selection based on loss estimates that implicitly incorporate the probability of choosing each arm to manage variance. Hence, the algorithm takes two parameters: η and γ. The algorithm results in a more efficient exploration and potentially better performance in environments with high loss variance. A general structure of EXP3-IX is presented as algorithm 6.

Using the IX loss, Gergely Neu [21] proved the following lemma and corollary

Algorithm 6 EXP3-IX

Input: $\eta \geq 0; \gamma \geq 0$
Initialization: $w_{1,i} = 1 \; \forall i \in \{1, \ldots, N\}$
for $t = 1, 2, \ldots, T$ **do** do
    `Set distribution` $\boldsymbol{p}_t = \frac{\boldsymbol{w}_t}{\sum_{i=1}^{N} w_{t,i}}$
    `Pick expert` $i_t \sim \boldsymbol{p}_t$
    `Incur loss` l_{t,i_t}
    `Let` $\tilde{l}_{t,i} = \frac{l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i}$
    `Update weight` $w_{t+1,i} = w_{t,i} \cdot e^{-\eta \tilde{l}_{t,i}}$

Lemma. 1. *Let $\gamma \geq 0$ and $\alpha_{t,i}$ be a random variable based on the observation history of the algorithm up to round $t-1$ such that $0 \leq \alpha_{t,i} \leq 2\gamma$ for all $t \in \{1, \ldots, T\}$ and $i \in \{1, \ldots, N\}$. Then, with probability at least $1-\delta$,*

$$\sum_{t=1}^{T}\sum_{i=1}^{N} \alpha_{t,i}\left(\tilde{l}_{t,i} - l_{t,i}\right) \leq \log\frac{1}{\delta}$$

Corollary. 1. *Let $\gamma \geq 0$. With probability at least $1-\delta$,*

$$\sum_{t=1}^{T}\left(\tilde{l}_{t,i} - l_{t,i}\right) \leq \frac{1}{2\gamma}\log\frac{N}{\delta}$$

simultaneously holds for all $i \in \{1, \ldots, N\}$.

Thus, we can derive the high-probability regret bound for EXP3-IX using the above lemma and corollary.

Theorem. 4. *For any $\delta \in (0, 1]$ and with $\eta = 2\gamma = \sqrt{\frac{2\log N}{NT}}$, EXP3-IX guarantees*

$$\textit{Regret}_T \leq 2\sqrt{2TN\log N} + \left(\sqrt{\frac{2NT}{\log N}} + 1\right)\log\left(\frac{2}{\delta}\right)$$

with probability at least $1-\delta$.

3.1.4 EXP3-SIX

EXP3-SIX extends the concepts from EXP3-IX by including a mechanism to adaptively shrink the learning rate and recursively update weights, allowing for better tracking and

adaption to changing optimal actions. This is particularly beneficial in environments where the optimal sequence of actions varies significantly over time. The algorithm targets sequences that switch actions at most k times. Thus, the regret function against the best sequence from this class $C(k) \subseteq 1, \ldots, N^T$ is defined as

$$\text{Regret}_T(j_1, \ldots, j_N) = \sum_{t=1}^{T} l_{t,i_t} - \min_{\{j_1,\ldots,j_N\} \in C(k)} \sum_{t=1}^{T} l_{t,j_t}$$

A general structure is presented in algorithm 7. The algorithm recursively updates weights $\boldsymbol{w}_t$ over the arms. Initially, EXP3-SIX sets $w_{1,i} = 1/N$ for all i. In subsequent rounds, the weights for each arm i are updated as follows:

$$w_{t+1,i} = (1-\alpha)\, w_{t,i} \cdot e^{-\eta \tilde{l}_{t,i}} + \frac{\alpha}{N} \sum_{j=1}^{N} w_{t,j} \cdot e^{-\eta \tilde{l}_{t,j}}$$

Algorithm 7 EXP3-SIX

Input: $\eta \geq 0; \gamma \geq 0; \alpha \in [0,1]$
Initialization: $w_{1,i} = 1/N \; \forall i \in \{1, \ldots, N\}$
for $t = 1, 2, \ldots, T$ **do** do
    `Set distribution` $\boldsymbol{p}_t = \frac{\boldsymbol{w}_t}{\sum_{i=1}^{N} w_{t,i}}$
    `Pick expert` $i_t \sim \boldsymbol{p}_t$
    `Incur loss` l_{t,i_t}
    `Let` $\tilde{l}_{t,i} = \frac{l_t(i)}{p_{t,i}+\gamma} \mathbb{I}_{i_t=i}$
    `Update weight` $w_{t+1,i} = (1-\alpha)\, w_{t,i} \cdot e^{-\eta \tilde{l}_{t,i}} + \frac{\alpha}{N} \sum_{j=1}^{N} w_{t,j} \cdot e^{-\eta \tilde{l}_{t,j}}$

Theorem. 5. *For any* $\delta \in (0,1]$ *and seting* $\eta = 2\gamma = \sqrt{\frac{2(k+1)\log N}{NT}}$ *and* $\alpha = \frac{k}{T-1}$. *Then, the regret of EXP3-SIX satisfies*

$$\textit{Regret}_T \leq 2\sqrt{2T\,(k+1)\,N \log\left(\frac{eNT}{k}\right)} + \left(\sqrt{\frac{2NT}{(k+1)\log N}} + 1\right) \log\left(\frac{2}{\delta}\right)$$

with probability at least $1-\delta$.

3.1.5 Fixed share algorithm

The fixed share algorithm, outlined in algorithm 8, is a variant of the Hedge algorithm. It incorporates a fixed exploration term alongside multiplicative updates to ensure that the

weight of any expert does not diminish excessively. This addition guarantees a regret bound that effectively tracks the best expert over any given interval.

Algorithm 8 Fixed Share Algorithm

Input: $\alpha < \frac{1}{2}$
Initialization: $\forall i \in 1, \ldots, N, w_{1,i} = 1/N$
for $t = 1, 2, \ldots, T$ **do**
    `Set distribution` $\boldsymbol{p}_t = \frac{\boldsymbol{w}_t}{\sum_{i=1}^{N} w_{t,i}}$
    `Pick expert` $i_t \sim \boldsymbol{p}_t$
    `Incur loss` l_{t,i_t}
    `Update weight` $\hat{w}_{t+1,i} = w_{t,i} \cdot e^{-\eta l_{t,i}}$
    `Share update` $w_{t+1,i} = (1-\alpha)\,\hat{w}_{t+1,i} + \frac{\alpha}{N}\hat{W}_{t+1}$ where $\hat{W}_{t+1} = \sum_{i=1}^{N} \hat{w}_{t+1,i}$

We denote $w'_t(i_1, \ldots, i_T)$ as the weight assigned at time t to the compound action $(i_1, \ldots, i_T)$ and it is defined as:

$$w'_t(i_1, \ldots, i_T) = w'_1(i_1, \ldots, i_T) \cdot e^{-\eta \sum_{s=1}^{t-1} l_{t,i_s}}$$

where the initial weights are recursively computed as follows

$$w'_1(i_1) = \frac{1}{N}$$

$$w'_1(i_1, \ldots, i_{t+1}) = w'_1(i_1, \ldots, i_t)\left(\frac{\alpha}{N} + (1-\alpha)\mathbb{I}_{i_{t+1}=i_t}\right)$$

Let $k = \sum_{t=2}^{T} \mathbb{I}_{i_{t-1}=i_t}$ be the number of switches occur in the sequence $\{i_1, \ldots, i_T\}$, it is not hard to see that

$$w'_1(i_1, \ldots, i_T) = \frac{1}{N}\left(\frac{\alpha}{N}\right)^k \left(1 - \alpha + \frac{\alpha}{N}\right)^{T-k-1}$$

Moreover, let the marginalized weights be defined as

$$w'_1(i_1, \ldots, i_t) = \sum_{i_{t+1}, \ldots, i_T} w'_1(i_1, \ldots, i_t, i_{t+1}, \ldots, i_T)$$

so the weight of any expert i at time t is

$$w'_t(i) = \sum_{i_1,\ldots,i_t,i_{t+2},\ldots,i_T} w'_1(i_1,\ldots,i_t,i,i_{t+2},\ldots,i_T)$$

Theorem. 6. *For all $t \in \{1,\ldots,T\}$ and $i \in \{1,\ldots,N\}$, the weight of the experts $w_{t,i}$ in fixed share algorithm is identical to $w'_t(i)$ for any $\alpha \in [0,1]$.*

Using this theorem, we show the following regret bound of the fixed share algorithm.

Theorem. 7. *For all $T \geq 1$, the Fixed Share algorithm satisfies*

$$\textit{Regret}_T(i_1,\ldots,i_T) \leq \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\alpha^k(1-\alpha)^{T-k-1}} + \frac{\eta}{8}T$$

for all action sequences $i_1,\ldots,i_n$, where $k = \sum_{t=2}^{T}\mathbb{I}_{i_{t-1}=i_t}$ is the number of switches occur in the sequence $\{i_1,\ldots,i_T\}$.

Moreover, by choosing appropriate parameters, we have the following theorem

Theorem. 8. *For all $T \geq 1$ and k such that $0 \leq k \leq T$, if $\alpha = \frac{k}{T-1}$ and $\eta = \sqrt{\frac{8}{T}\left((k+1)\ln N + (T-1)H\left(\frac{k}{T-1}\right)\right)}$, then the following bound holds*

$$\textit{Regret}_T(i_1,\ldots,i_T) \leq \sqrt{\frac{T}{2}\left((k+1)\ln N + (T-1)H\left(\frac{k}{T-1}\right)\right)}$$

for all action sequences $i_1,\ldots,i_n$ and $H(x) = -x\ln x - (1-x)\ln(1-x)$ for $x \in [0,1]$.

3.1.6 Online gradient descent

The goal of Online Gradient Descent (OGD) is to minimize the function of an online convex optimization problem. The algorithm extends from standard gradient descent for offline optimization.

The idea is to update the next point in the opposite direction of the subgradients of the current function at the current point in each round $t \in \{1,\ldots,T\}$. If this step results in a point outside the convex set $\mathcal{K}$, the algorithm projects this point to the closest point

in the $\mathcal{K}$ by applying the projection operation, $\Pi_{\mathcal{K}}$. Moreover, although the function can differ in each round, OGD attains a sublinear regret. A general structure of the OGD is shown in Algorithm 9.

Algorithm 9 Online Gradient Descent

1: **Input:** convex set $\mathcal{K}$, T, $\boldsymbol{x}_1 \in \mathcal{K}$, step size η
2: **for** $t = 1, 2, \ldots, T$ **do**
3: `Play` $\boldsymbol{x}_t$
4: `Obeserve function` $\boldsymbol{f}_t(\boldsymbol{x}_t)$
5: `Update` $\boldsymbol{y}_{t+1} = \boldsymbol{y}_t - \eta \nabla f_t(\boldsymbol{x}_t)$
6: `Project` $\boldsymbol{x}_{t+1} = \Pi_{\mathcal{K}} \left(\boldsymbol{y}_{t+1} \right)$

Denote G an upper bound on the norm of the subgradients of f over $\mathcal{K}$, i.e., $\forall \boldsymbol{x} \in \mathcal{K}, t \in \{1, \ldots, T\}, \|\nabla f_t(\boldsymbol{x})\| \leq G$.

Theorem. 9. *For any interval of rounds $\mathcal{I} = \{t_1, \ldots, t_2\} \subseteq \{1, \ldots, T\}$, Online Gradient Descent with step sizes η guarantees the following regret:*

$$\textit{Regret}_{\mathcal{I}} = \sum_{t \in \mathcal{I}} f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}^* \in \mathcal{K}} \sum_{t \in \mathcal{I}} f_t(\boldsymbol{x}^*) \leq \frac{\|\boldsymbol{x}_{t_1} - \boldsymbol{x}^*\|^2}{2\eta} + \frac{\eta G^2 T}{2} \tag{3.2}$$

3.2 Combining primal and dual regret minimizers

The primal regret algorithms must guarantee that their regret $\mathcal{R}^P_{\mathcal{T}_G}$ is sublinear in T to attain meaningful bound, and they do not have any constraints when the replenishment factor is known. However, in case of not knowing the replenishment factor, we need an additional assumption on the primal regret minimizer A^P.

Assumption. 3. *For any $\mathcal{I} = \{t_1, \ldots, t_2\} \subseteq \{1, \ldots, T\}$, the primal regret minimizer A^P facing adversarial losses with unknown range L is upper bounded by $L^2 \epsilon^P_{T,\delta}$ with probability at least $1 - \delta$, where $\epsilon^P_{T,\delta}$ is independent from the range of payoffs.*

While Hedge, EXP3, and EXP3-IX do not satisfy this assumption, EXP3-SIX does. Thus, we can use EXP3-SIX as a primal regret minimizer when the replenishment factor is unknown.

The dual regret algorithms must be full-feedback by construction and weakly adaptive. Furthermore, since the guarantees depend on $\mathcal{R}^D_{\mathcal{T}_{G,<\tau}}(D)$ and $\mathcal{R}^D_{\mathcal{T}_{G,>\tau}}(D)$, we must choose the dual regret algorithm such that these terms are sublinear in T so that the bound is meaningful.

In a scenario wherein the replenishment factor is known, we can use the fixed share algorithm as a dual regret minimizer and instantiate it to play on the set $\mathcal{D} \coloneqq \left\{\boldsymbol{\lambda} \in \mathbb{R}^d_+ : \|\boldsymbol{\lambda}\|_1 \leq 1/\nu\right\}$ where $\nu = \beta + \rho$. However, if the replenishment factor is unknown, we will use OGD as a dual regret minimizer.

Using the analysis of each algorithm and guarantees 2.6 and 2.7, Martino Bernasconi et al.[5] shows the following two theorem:

Theorem. 10. *Let $\alpha = \frac{\nu}{1+\beta}$. In the adversarial setting of any BwRK problem, there exists an algorithm satisfying the following bound on the regret:*

$$\alpha \cdot \texttt{OPT}_{\boldsymbol{\omega}} - \sum_{t=1}^{T} r_t(a_t) \leq \tilde{O}\left(\frac{d^4}{\nu^2}\sqrt{NT}\log\frac{1}{\delta}\right) \tag{3.3}$$

with probability at least $1-\delta$.

Theorem. 11. *In the stochastic setting of any BwRK problem, there exists an algorithm satisfying the following bound on the regret:*

$$T \cdot \texttt{OPT}^{\texttt{LP}}_{\bar{r},\bar{c}} - \sum_{t=1}^{T} r_t(a_t) \leq \tilde{O}\left(\frac{d^4}{\nu^2}\sqrt{NT}\log\frac{1}{\delta}\right) \tag{3.4}$$

with probability at least $1-2\delta$.

Chapter 4

Application and Performance Analysis of the Primal-Dual Template

4.1 Dynamic pricing

One critical application of Bandits with Replenishable Knapsacks is Dynamic Pricing. This strategy enables firms to modify prices in real time, considering factors such as inventory levels and budget limitations, which can be replenished. Dynamic pricing can be applied across various economic sectors, including e-commerce and retail, as well as airline revenue management. In the latter case, airlines adopt dynamic pricing techniques to maximize their profit while effectively managing their seat inventory and adding flights as necessary.

4.1.1 Problem formulation

Consider a firm that has to make a pricing decision for n product over a finite time horizon with T periods. At each period t, the firm makes a pricing decision $p_{t,i}$ for all $i = 1, \ldots, n$. The demand of the product, $d_{t,i}$, is a stochastic function of price defined by

$$d_{t,i} = D_i(p_{t,i}) + \varepsilon_i$$

with ε as an unknown random variable such that $\mathbb{E}[\varepsilon_i] = 0$, and $D_i(p_{t,i})$ being a continuous and monotone decreasing function representing the actual demand function.

Initially, we assume that the firm has no inventory, i.e., $B_i = 0$. Furthermore, in any period t, the firm can replenish the inventory of any products, but it cannot sell them in that round. Thus, the firm can restore all products in any round, and assumption 2 is satisfied. However, the firm is unaware of the replenishment factor, as it depends on the goods available from the suppliers.

Let $\boldsymbol{p}_t$ be the vector of pricing decisions made at round t, the reward function of the firm is

$$r_t\left(\boldsymbol{p}_t\right) = \sum_{i=1}^{n} p_{t,i} d_{t,i}$$

and the cost is the number of units of each product sold.

4.1.2 Simulated scenario analysis

This BwRK problem has a stochastic setting and an unknown replenishment factor. Thus, we can use EXP3-SIX as the primal regret algorithm as it satisfies assumption 3 and OGD as the dual regret algorithm. EXP3-SIX is instantiated as in theorem 5 with $k = 2$, whereas OGD is initiated on $\mathbb{R}_+$ and its starting point is $\lambda = 0$.

The EXP3-SIX algorithm operates within a framework of discrete action choices. However, in our scenario, the firm's decision-making involves a continuous price action space. A typical approach to handle this is by discretizing the continuous action space into a finite and discrete set of possible actions, i.e., $p_{t,i} \in P_i$ with P_i being a finite and discrete set. Therefore, we assume that the firm makes its pricing decisions from a set of predefined options. This assumption is generally practical because firms often conduct market research to narrow their pricing strategies to a few viable options. Then, for the sake of consistency, we scale the reward function and the cost function so that $r_t(a_t) \in [0, 1]$ and $c_t(a_t) \in [-1, 1]$ for all $t \in \{1, \ldots, T\}$.

Consider a company that offers three different products, and their true demand func-

tion, D_i, of each product are

$$D_1(p_{t,1}) = -\frac{7}{17}p_{t,1} + 10 \quad D_2(p_{t,2}) = -\frac{1}{100}e^{p_{t,2}} + 13 \quad D_3(p_{t,3}) = -\frac{1}{150}p_{t,1}^3 + 11$$

For each product, after conducting market research, the company has narrowed down its pricing strategy to three potential price points, which are

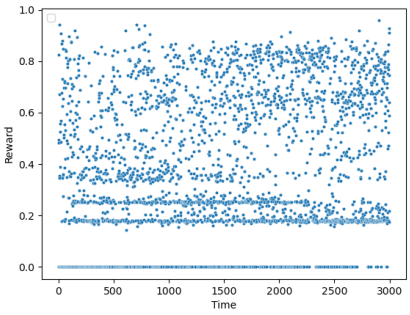
$$P_1 = \{12, 5, 18\} \quad P_2 = \{5.5, 3, 6.5\} \quad P_3 = \{7.5, 4, 9.5\}$$

Additionally, the firm can decide not to sell some products and restore them by an unknown replenishment factor. Notice that the firm can choose to restore all products in a round so that the assumption 2 is satisfied.

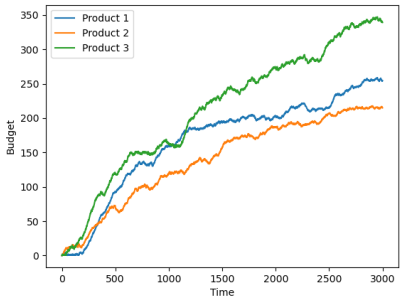
Let the unknown replenishment factor be 0.5, 0.6, and 0.7 for each product, and assume that the unknown random variable in the demand function is drawn from a Gaussian distribution with a mean of 0 and standard deviation of 0.3, 0.1, and 0.6, respectively. After running the setup for a total period of $T = 3 \cdot 10^4$, figure 4.1 shows the results of the primal-dual template.

As expected, when the budget is depleted, the dual minimizer assigns a higher λ, emphasizing the per-iteration budget constraint. This adjustment prompts the primal minimizer to take action to replenish the budget. Once the budget is restored, the dual minimizer reduces λ, allowing the primal minimizer to prioritize actions that maximize rewards and, if necessary, replenish only the needed products, as indicated by the dual minimizer.

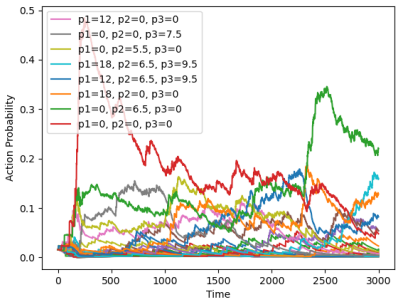
Figure 4.1c illustrates how the probability of each action changes over time. Initially, when the budgets are 0, the primal minimizer will likely choose actions that replenish all products. It then realizes that it does not have to replenish all products in the same round but can replenish only some, leading to an increasing probability of actions restoring only some products. Finally, when the dual minimizer deems the budgets adequate for all three products, the primal minimizer selects actions that balance reward maximization and resource consumption. In other words, the primal minimizer does not



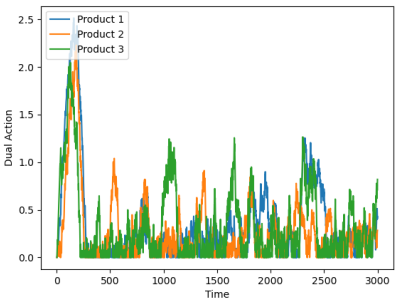
(a) Reward obtained



(b) Budget situation



(c) Probability of each action



(d) Dual minimizer action played

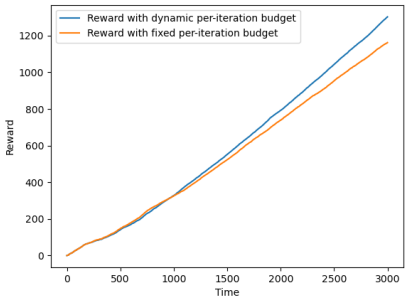
Figure 4.1: Results of performing primal-dual template on dynamic pricing.

choose prices that solely maximize the reward but also considers the consumption of each action. This behavior is also shown in future 4.1a, which shows an increase in the selection of pricing strategies that yield higher rewards.

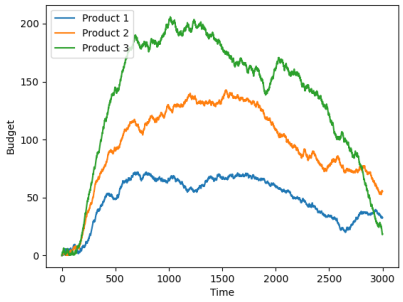
4.1.3 Dynamic pricing with dynamic per-iteration budget

From the previous setting, since the initial budget for all products is zero, their per-iteration budget, defined by $\rho_i = B_i/T$, is also 0. However, as depicted in figure 4.1b, a fixed per-iteration budget can mislead the dual minimizer into applying a higher λ even when the budget is sufficient for the remaining rounds because it assumes that the algorithm should consume zero resources each round. Therefore, consider a scenario where the per-iteration budget is updated as follows:

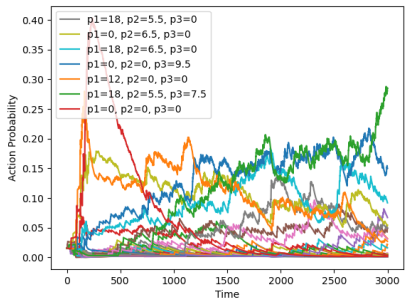
$$\rho_{t,i} = \frac{B_{t,i}}{T-t} \; \forall i \in \{1, \ldots, d\} \tag{4.1}$$



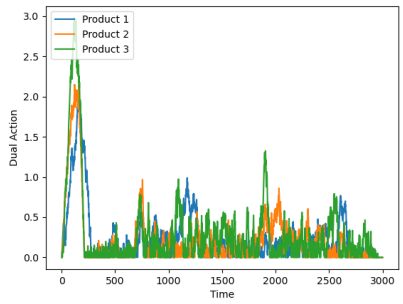
(a) Cumulative reward comparison



(b) Budget situation



(c) Probability of each action



(d) Dual minimizer action played

Figure 4.2: Results of performing primal-dual template on dynamic pricing with dynamic per-iteration budget.

where $B_{t,i}$ is the budget for product i at round t. Figures 4.2 show the result with the dynamic per-iteration budget.

The budget consumption is now more judicious than before. Initially, the algorithm replenishes resources as it would with a fixed per-iteration budget. Once the budgets are deemed sufficient by the dual minimizer, it reduces λ to prioritize reward maximization until all budgets are consumed. In the final stages, with an adequate budget to cover all rounds, the primal minimizer focuses solely on actions that maximize rewards. Hence, as shown in figure 4.2c, there is a dramatic increase in the replenishment action. Over time, as the probability of playing replenishment actions decreases, the probability of the action that yields the maximum reward increases.

Therefore, it is logical to observe a higher reward with a dynamic per-iteration budget than a fixed one. The updating rule for the per-iteration budget can significantly alter the strategies of both the primal and dual regret minimizers, making it a compelling area for further investigation. In practice, this approach is highly beneficial as it allows

the decision-maker to reduce or cease budget restoration when the budget is sufficient and redirect these funds to other projects. However, the paper does not include an analytical analysis of this updating rule, which might improve the regret bound stated in theorems 11 and 10.

4.2 Online ad allocation

Another simple yet fundamental application of BwRK is resource allocation. I will focus on online ad allocation, a prime example of resource allocation. Different channels attract different audience segments, and an advertiser must determine which channels draw larger audiences. This section shows how an advertiser can decide which channels to promote their product under budget constraints, which can be increased.

4.2.1 Problem formulation

Consider an advertiser who wants to promote his product over a finite time horizon consisting of T periods. In each period t, the advertiser selects a channel $a_t \in A$ to place their advertisement. At the end of each period, the advertisement receives the number of clicks, which is stochastic and depends solely on the selected channel. The reward $r_t : \mathbb{R} \to [0, 1]$ and the cost $c_t : \mathbb{R} \to [-1, 1]$ are then computed based entirely on the number of clicks $l_t : A \to \mathbb{R}$. The higher the number of clicks, the greater the reward and the cost, i.e., the cost of clicks.

Furthermore, the advertiser can decide to increase the budget at each round. This decision, however, is made without knowledge of the replenishment factor, which often varies based on the performance of other activities. For example, a firm might increase its budget if other activities yield favorable outcomes.

4.2.2 Simulated scenario analysis

Similar to the dynamic pricing problem, this is a BwRK problem with a stochastic setting and an unknown replenishment factor. Thus, we can use the same primal and dual regret minimizers and their parameters as in dynamic pricing.

I ran the problem in a total period of $T = 3 \cdot 10^4$ days, where the advertiser has five channels to choose from: Instagram, Facebook, Twitter, Snapchat, and TikTok. The advertiser starts with a budget that can at least cover the $T/15$ days.

We assume that each channel's number of clicks follows a Poisson distribution with different parameters and that the variable cost of click, v_t, is also different.

The reward function and cost function are defined by

$$r_t(l_t) = l_t \quad \text{and} \quad c_t(l_t) = l_t \cdot v_t + 0.1$$

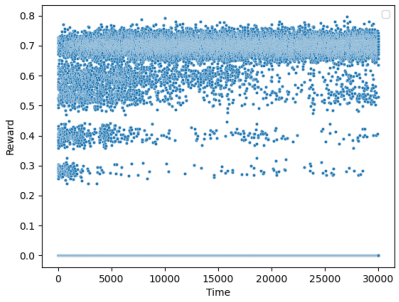
where $v_t \in [0.0005, 0.0007]$. Moreover, we scale the reward and cost functions so that $r_t(l_t) \in [0, 1]$ and $c_t(l_t) \in [-1, 1]$ for all $t \in \{1, \ldots, T\}$.

To illustrate the conclusion of the primal-dual template, I will assume that the unknown replenishment factor follows a normal distribution with a mean of 0.8 and a standard deviation of 0.1. Additionally, I will use the following number to complete the setup

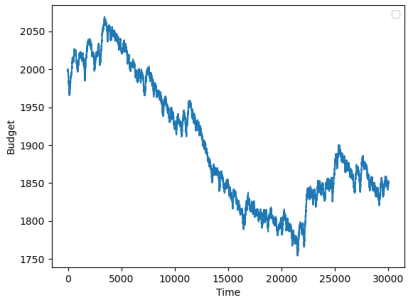
Table 4.1: Example of a Simple Table

Data	Instagram	Facebook	Twitter	Snapchat	TikTok
Poisson parameter	520	370	777	912	693
Variable cost per click	0.00053	0.00062	0.00068	0.0007	0.00057

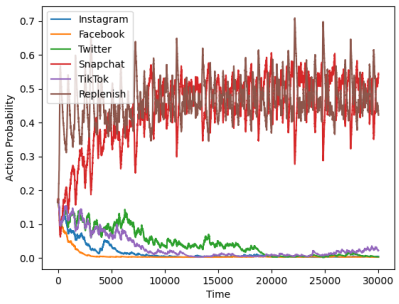
After performing the primal-dual template, figure 4.3 shows the results. The algorithm explores different channels in the initial phase to assess their rewards and costs. After gathering all the information, it alternates between the action that yields the highest reward and the replenishment action. Moreover, the oscillation in dual actions is attributed to the small value of the fixed per-iteration budget. Consequently, the primal algorithm assigns similar probabilities to both actions, depending on whether there was a replenishment or consumption in the previous round.



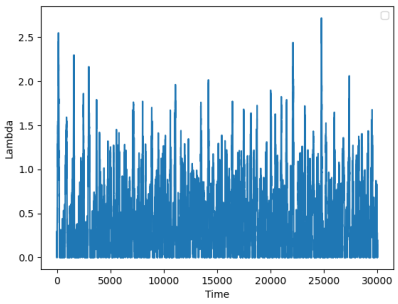
(a) Reward obtained



(b) Budget situation



(c) Probability of each action



(d) Dual minimizer action played

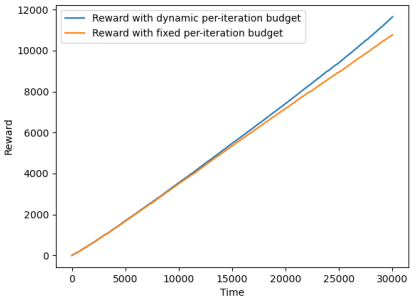
Figure 4.3: Results of performing primal-dual template on online ad allocation.

4.2.3 Online ad allocation with dynamic per-iteration budget

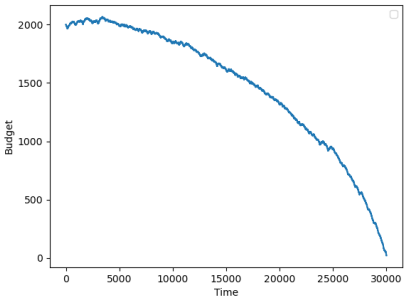
Like dynamic pricing, a fixed per-iteration budget causes the dual minimizer to base its actions on the initial rather than the current budget. Figure 4.3b illustrates how the standard primal-dual template consumes the budget illogically: it replenishes resources even when the budget is quite adequate and consumes resources so rapidly that it must skip several rounds to replenish them, and yet resources are not fully consumed by the final round. Figure 4.4 demonstrates the benefits of using the updated per-iteration budget rule defined in 4.1. As a result, we achieve more rational budget consumption, leading to higher cumulative revenue.

4.2.4 Online ad allocation with known replenishment factor

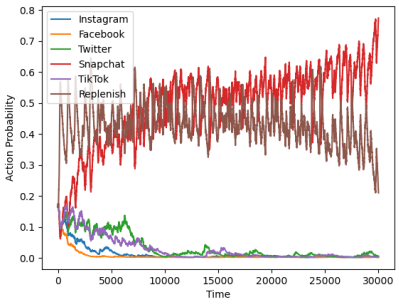
Consider the same setting of online ad allocation with a fixed per-iteration budget, but now the replenishment factor is known. This might happen when the advertiser has a clear plan for future investment and knows how much to boost the budget. In this



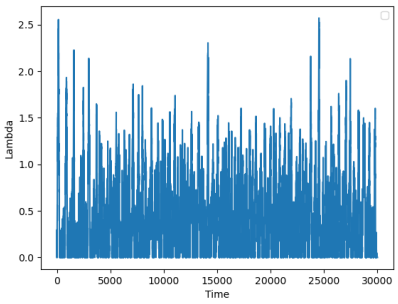
(a) Cumulative reward comparison



(b) Budget situation



(c) Probability of each action



(d) Dual minimizer action played

Figure 4.4: Results of performing primal-dual template on online ad allocation with dynamic per-iteration budget.

scenario, instead of using Online Gradient Descent (OGD) as the dual regret minimizer, the Fixed Share Algorithm becomes a viable option. Moreover, since assumption 3 is no longer required, we can employ other primal regret minimizers such as EXP3 and EXP3-IX.

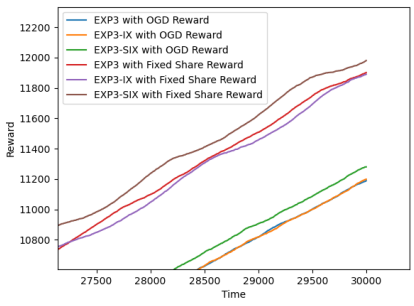


Figure 4.5: Reward comparison among different primal and dual regret minimizers.

Figure 4.5 shows that when advertisers have prior knowledge about the replenishment factor, they are better off using this information to make decisions. As expected, the fixed share algorithm outperforms OGD by returning higher rewards regardless of

the choice of primal regret minimizer.

4.3 Conclusion and discussion

In this chapter, I examine the performance of the primal-dual template across two simulated settings: dynamic pricing and online ad allocation. Both simulations demonstrate that the primal-dual template operates as expected. The primal regret minimizer adjusts its actions based on the actions of the dual regret minimizer, which rapidly modulates its responses—applying smaller or larger actions when budgets are assessed as sufficient or constrained, respectively. However, an issue arises with the standard primal-dual template when the initial budget is not intended to cover the entire period T. The problem lies in the per-iteration budget being calculated at the outset, which does not accurately reflect the actual consumption needed each round. As evidenced in both applications, introducing a dynamic per-iteration budget allows the dual regret minimizer to base its decisions on the current budget rather than the initial estimate, leading to more effective resource utilization and a reduction of the regret bound.

A potential improvement could involve developing a general update rule for the per-iteration budget to manage budget consumption more effectively. Figure 4.2b illustrates a strategy where the budget is initially replenished enough to cover upcoming periods. However, a more evenly distributed replenishment strategy might be preferable in practical scenarios throughout the timeline. One approach could be to incorporate how many rounds the current budget can sustain while factoring in the (expected) replenishment rate into the per-iteration budget update rule.

Appendix A

Theorem. 1. *Let $\eta > 0$ and assume all losses to be nonnegative. For any $j \in \{1, \ldots, N\}$, The Hedge algorithm satisfies:*

$$\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i} \cdot l_{t,i} \leq \frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i} \cdot l_{t,i}^2 + \frac{\ln N}{\eta} + \sum_{t=1}^{T} l_{t,j}$$

Proof. Assuming that the algorithm initializes the weights to be 1, it is not hard to see that the weight of each expert after the last round is

$$w_{T+1,i} = w_{1,i}\prod_{t=1}^{T} e^{-\eta l_{t,i}} = e^{-\eta\sum_{t=1}^{T} l_{t,i}}$$

Let $W_t = \sum_{i=1}^{N} w_{t,i}$ be the total weight of all expert at round t. We have

$$W_{T+1} > w_{T+1,j} = e^{-\eta\sum_{t=1}^{T} l_{t,j}} \tag{4.2}$$

Since the initial weight of each expert is 1, we have $W_1 = N$. Moreover, for all $x \geq 0$, we have $e^{-x} \leq 1 - x + \frac{x^2}{2}$. Thus, for each round t and each expert i, we get

$$\begin{aligned}
\frac{W_{t+1}}{W_t} &= \sum_{i=1}^{N} \frac{w_{t,i}}{W_t} \cdot e^{-\eta l_{t,i}} \\
&= \sum_{i=1}^{N} p_{t,i} \cdot e^{-\eta l_{t,i}} \\
&\leq \sum_{i=1}^{N} p_{t,i} \cdot \left(1 - \eta \cdot l_{t,i} + \frac{\eta^2}{2} \cdot l_{t,i}^2\right) \\
&= 1 - \eta \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i} + \frac{\eta^2}{2} \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i}^2
\end{aligned} \tag{4.3}$$

Taking a logarithm on both sides of 4.3 and use the fact about logarithm that $\ln(1+x) \leq x$ for all $x > -1$

$$\ln \frac{W_{t+1}}{W_t} \leq \ln \left(1 - \eta \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i} + \frac{\eta^2}{2} \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i}^2\right) \leq -\eta \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i} + \frac{\eta^2}{2} \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i}^2$$

Summing over t and inverting the signs on both sides, we obtain

$$\begin{aligned}
\sum_{t=1}^{T} \left[\eta \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i} - \frac{\eta^2}{2} \cdot \sum_{i=1}^{N} p_{t,i} l_{t,i}^2\right] &\leq -\sum_{t=1}^{T} \ln \frac{W_{t+1}}{W_t} \\
&= -\ln \prod_{t=1}^{T} \frac{W_{t+1}}{W_t} \\
&= -\ln \frac{W_{T+1}}{W_1} \\
&= \ln W_1 - \ln W_{T+1} \\
&\leq \ln N + \eta \cdot \sum_{t=1}^{T} l_{t,j} \qquad (4.4)
\end{aligned}$$

where we used 4.2 in the last step. Rearrange the inequality, we get:

$$\eta \sum_{i=1}^{N} p_{t,i} l_{t,i} \leq \frac{\eta^2}{2} \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} l_{t,i}^2 + \ln N + \eta \cdot \sum_{t=1}^{T} l_{t,j}$$

and the theorem follows by dividing η on both sides. □

Theorem. 2. *There exists a learning rate, $\eta > 0$, such that the Hedge algorithm has an expected regret bound of $O(\sqrt{T \log N})$ for all $T \geq 1$.*

Proof. Let $i^* = \arg\min_{i \in \{1,\ldots,N\}} \sum_{t=1}^{T} l_{t,i}$. Since theorem 1 holds for any expert $j \in \{1, \ldots, N\}$, we apply $j = i^*$

$$\sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i} \leq \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i}^2 + \frac{\ln N}{\eta} + \sum_{t=1}^{T} l_{t,i^*}$$

Due to the boundedness of the loss function, we have $\sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \cdot l_{t,i}^2 \leq T$ and

rearranging it, we get

$$\mathbb{E}\left[\text{Regret}_T\right] = \sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\cdot l_{t,i} - \sum_{t=1}^{T} l_{t,i^*} \leq \frac{\eta}{2}T + \frac{\ln N}{\eta}$$

Setting $\eta = \sqrt{\frac{2\ln N}{T}}$, we obtain

$$\mathbb{E}\left[\text{Regret}_T\right] \leq \frac{1}{2}\sqrt{\frac{2\ln N}{T}}T + \frac{\ln N}{\sqrt{\frac{2\ln N}{T}}} = \sqrt{2T\ln N}$$

□

Theorem. 3. *EXP3 algorithm attains an expected regret bound of* $O(\sqrt{TN\log N})$ *for all* $T \geq 1$.

Proof. Notice that the expect random loss of an expert, $\hat{l}_{t,i}$, is

$$\mathbb{E}\left[\hat{l}_{t,i}\right] = Pr\left[i_t = i\right]\cdot\frac{l_{t,i_t}}{p_{t,i}} = p_{t,i}\cdot\frac{l_{t,i_t}}{p_{t,i}} = l_{t,i_t}$$

Moreover, notice that

$$\begin{aligned}\mathbb{E}\left[\sum_{i=1}^{N} p_{t,i}\hat{l}_{t,i}^2\right] &= \sum_{i=1}^{N} Pr\left[i_t = i\right]\cdot p_{t,i}\hat{l}_{t,i_t}^2 \\ &= \sum_{i=1}^{N} p_{t,i}^2\hat{l}_{t,i_t}^2 = \sum_{i=1}^{N} l_{t,i_t}^2 \leq N \end{aligned} \tag{4.5}$$

and

$$l_{t,i_t} = p_{t,i_t}\hat{l}_{t,i_t} = \sum_{i=1}^{N} p_{t,i}\hat{l}_{t,i}$$

The Exp3 algorithm applies Hedge to the losses given by $\hat{l}_t$, which are all non-negative. Therefore, we can directly use theorem 1 and apply $j = i^* = \arg\min_{i\in\{1,\ldots,N\}}\sum_{t=1}^{T} l_{t,i}$

to get

$$\begin{aligned}
\mathbb{E}\left[\text{Regret}_T\right] &= \mathbb{E}\left[\sum_{t=1}^{T} l_{t,i_t} - \sum_{t=1}^{T} l_{t,i^*}\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{l}_{t,i} - \sum_{t=1}^{T} \hat{l}_{t,i^*}\right] \\
&\leq \mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i} l_{t,i}^2 + \frac{\ln N}{\eta}\right] \\
&\leq \frac{\eta}{2} TN + \frac{\ln N}{\eta}
\end{aligned}$$

The first step is because $\hat{l}$ is independent of i^*. Setting $\eta = \sqrt{\frac{2\ln N}{TN}}$, we obtain

$$\mathbb{E}\left[\text{Regret}_T\right] \leq \sqrt{2TN\log N}$$

□

Theorem. 4. *For any $\delta \in (0, 1]$ and with $\eta = 2\gamma = \sqrt{\frac{2\log N}{NT}}$, EXP3-IX guarantees*

$$\textit{Regret}_T \leq 2\sqrt{2TN\log N} + \left(\sqrt{\frac{2NT}{\log N}} + 1\right)\log\left(\frac{2}{\delta}\right)$$

with probability at least $1 - \delta$.

Proof. Let us fix an arbitrary $\delta' \in (0, 1)$. Assuming positive learning rates and non-negative losses, we can obtain the bound from theorem 1

$$\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i} \cdot \tilde{l}_{t,i} \leq \frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i} \cdot \tilde{l}_{t,i}^2 + \frac{\ln N}{\eta} + \sum_{t=1}^{T} \tilde{l}_{t,j}$$

Observe that

$$\begin{aligned}\sum_{i=1}^{N} p_{t,i}\tilde{l}_{t,i} &= \sum_{i=1}^{N} \frac{p_{t,i}l_{t,i} + \gamma l_{t,i} - \gamma l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i} \\ &= \sum_{i=1}^{N} \frac{(p_{t,i}+\gamma)\, l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i} - \sum_{i=1}^{N} \frac{\gamma l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i} \\ &= l_{t,i_t} - \gamma \sum_{i=1}^{N} \tilde{l}_{t,i} \end{aligned} \tag{4.6}$$

Moreover, note that

$$\sum_{i=1}^{N} p_{t,i}\tilde{l}_{t,i}^2 = \sum_{i=1}^{N} \frac{p_{t,i}l_{t,i}}{p_{t,i}+\gamma}\frac{l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i} \leq \sum_{i=1}^{N} \frac{l_{t,i}}{p_{t,i}+\gamma}\mathbb{I}_{i_t=i} = \sum_{i=1}^{N} \tilde{l}_{t,i}$$

The inequality is due to the boundedness of the losses, i.e. $l_{t,i} \in [0,1]$. Thus, we obtain

$$\begin{aligned}\sum_{t=1}^{T}(l_{t,i_t} - l_{t,j}) &= \sum_{t=1}^{T}\left(\sum_{i=1}^{N} p_{t,i}\tilde{l}_{t,i} + \gamma\sum_{i=1}^{N}\tilde{l}_{t,i} - \tilde{l}_{t,j} + \tilde{l}_{t,j} - l_{t,j}\right) \\ &= \sum_{t=1}^{T}\left(\sum_{i=1}^{N} p_{t,i}\tilde{l}_{t,i} - \tilde{l}_{t,j}\right) + \sum_{t=1}^{T}\gamma\sum_{i=1}^{N}\tilde{l}_{t,i} + \sum_{t=1}^{T}\left(\tilde{l}_{t,j} - l_{t,j}\right) \\ &\leq \sum_{t=1}^{T}\frac{\eta}{2}\sum_{i=1}^{N} p_{t,i}\tilde{l}_{t,i}^2 + \frac{\ln N}{\eta} + \sum_{t=1}^{T}\gamma\sum_{i=1}^{N}\tilde{l}_{t,i} + \sum_{t=1}^{T}\left(\tilde{l}_{t,j} - l_{t,j}\right) \\ &\leq \sum_{t=1}^{T}\frac{\eta}{2}\sum_{i=1}^{N}\tilde{l}_{t,i} + \frac{\ln N}{\eta} + \sum_{t=1}^{T}\gamma\sum_{i=1}^{N}\tilde{l}_{t,i} + \sum_{t=1}^{T}\left(\tilde{l}_{t,j} - l_{t,j}\right) \\ &= \sum_{t=1}^{T}\left(\frac{\eta}{2}+\gamma\right)\sum_{i=1}^{N}\tilde{l}_{t,i} + \frac{\ln N}{\eta} + \sum_{t=1}^{T}\left(\tilde{l}_{t,j} - l_{t,j}\right) \\ &\leq \sum_{t=1}^{T}\left(\frac{\eta}{2}+\gamma\right)\sum_{i=1}^{N} l_{t,i} + \log\left(\frac{1}{\delta'}\right) + \frac{\ln N}{\eta} + \frac{\log\frac{N}{\delta'}}{2\gamma}\end{aligned}$$

holds with probability at least $1-2\delta'$. Notice that we apply lemma 1 with $\alpha_{t,i} = \frac{\eta}{2}+\gamma$ and corollary 1 in the last line. Let j be the expert that minimize the total loss, i.e., $j = i^*$, and set $\delta' = \delta/2$. By using the boundedness of the losses, we get

$$\text{Regret}_T = \sum_{t=1}^{T}(l_{t,i_t} - l_{t,i^*}) \leq N\sum_{t=1}^{T}\left(\frac{\eta}{2}+\gamma\right) + \log\left(\frac{2}{\delta}\right) + \frac{\ln N}{\eta} + \frac{\log\frac{2N}{\delta}}{2\gamma}$$

Setting $\eta = 2\gamma = \sqrt{\frac{2\log N}{NT}}$, we get the final regret bound

$$\begin{aligned}\text{Regret}_T &\leq N\sum_{t=1}^{T}\sqrt{\frac{2\log N}{NT}} + \log\left(\frac{2}{\delta}\right) + \frac{\log N}{\sqrt{\frac{2\log N}{NT}}} + \frac{\log\frac{2N}{\delta}}{\sqrt{\frac{2\log N}{NT}}} \\ &= 2\sqrt{2TN\log N} + \left(\sqrt{\frac{2NT}{\log N}} + 1\right)\log\left(\frac{2}{\delta}\right)\end{aligned}$$

with probability at least $1-\delta$. □

Theorem. 5. *For any $\delta \in (0,1]$ and seting $\eta = 2\gamma = \sqrt{\frac{2(k+1)\log N}{NT}}$ and $\alpha = \frac{k}{T-1}$. Then, with probability at least $1-\delta$, the regret of EXP3-SIX satisfies*

$$\textit{Regret}_T \leq 2\sqrt{2T\left(k+1\right)N\log\left(\frac{eNT}{k}\right)} + \left(\sqrt{\frac{2NT}{(k+1)\log N}} + 1\right)\log\left(\frac{2}{\delta}\right)$$

Proof. Let us fix an arbitrary $\delta' \in (0,1)$. Assuming positive learning rates and non-negative losses, we can obtain from inequality 4.4 assuming that $W_1 \leq 1$

$$\begin{aligned}\sum_{t=1}^{T}\left[\eta\cdot\sum_{i=1}^{N}p_{t,i}\tilde{l}_{t,i} - \frac{\eta^2}{2}\cdot\sum_{i=1}^{N}p_{t,i}\tilde{l}_{t,i}^2\right] &\leq \ln W_1 - \ln W_{T+1} \\ &\leq -\ln W_{T+1}\end{aligned}$$

Moreover, since the update rule in EXP3-SIX is the same as the one on the Fixed Share algorithm, we can replace W_{T+1} as we did in the proof of theorem 7. Hence, we have

$$\begin{aligned}\sum_{t=1}^{T}\sum_{i=1}^{N}p_{t,i}\tilde{l}_{t,i} &\leq \frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}p_{t,i}\tilde{l}_{t,i}^2 - \frac{1}{\eta}\ln W_{T+1} \\ &\leq \frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}p_{t,i}\tilde{l}_{t,i}^2 + \sum_{t=1}^{T}\tilde{l}_{t,i_t} + \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\alpha^k\left(1-\alpha\right)^{T-k-1}} \\ &\leq \frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\tilde{l}_{t,i} + \sum_{t=1}^{T}\tilde{l}_{t,i_t} + \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\alpha^k\left(1-\alpha\right)^{T-k-1}}\end{aligned}$$

where the last step uses $p_{t,i}\cdot\tilde{l}_{t,i} \leq 1$. From corollary 1, we know that with probability at least $1-\delta$,

$$\sum_{t=1}^{T}\left(\tilde{l}_{t,j_t} - l_{t,j_t}\right) \leq \frac{\log\frac{|C(k)|}{\delta}}{2\gamma}$$

simultaneously holds for all $j_1, \dots j_T \in C(k)$. Further, notice that

$$|C(k)| = \sum_{r=1}^{k} \binom{T-1}{r} N\,(N-1)^j \leq N^{k+1} \sum_{r=1}^{k} \binom{T-1}{r} \leq N^{k+1} \left(\frac{eT}{k}\right)^k$$

Using the corollary 1, inequality above and equation 4.6, our regret becomes

$$\sum_{t=1}^{T} \left(l_{t,i_t} - l_{t,j_t^*}\right) \leq \frac{\bar{k}}{\eta} \ln N + \frac{1}{\eta} \ln \frac{1}{\alpha^k \left(1-\alpha\right)^{T-\bar{k}}} + \frac{\bar{k} \log N + k \log \frac{2eT}{k\delta'}}{2\gamma} + \left(\frac{\eta}{2} + \gamma\right) NT + \left(\frac{\eta}{2} + \gamma\right) \frac{\log\left(2/\delta\right)}{2\gamma}$$

holds with probability at least $1-\delta$ and $\bar{k} = k+1$. Lastly, setting $\alpha = \frac{k}{T-1}$ and using the inequality $\log x \geq \frac{x-1}{x}$ for all $x > 0$, we get

$$\begin{aligned}
\ln \frac{1}{\alpha^k \left(1-\alpha\right)^{T-\bar{k}}} &= -k \ln \alpha - (T-\bar{k}) \ln(1-\alpha) \\
&= -k \ln \left(\frac{k}{T-1}\right) - (T-\bar{k}) \ln \left(\frac{T-\bar{k}}{T-1}\right) \\
&= k \ln \left(\frac{T-1}{k}\right) - (T-1) \frac{T-\bar{k}}{T-1} \ln \left(\frac{T-\bar{k}}{T-1}\right) \\
&\leq k \ln \left(\frac{T-1}{k}\right) - (T-1) \left(\frac{T-\bar{k}}{T-1} - 1\right) \\
&= k \ln \left(\frac{T-1}{k}\right) - k \\
&= k \ln \left(\frac{eT}{k}\right)
\end{aligned}$$

The proof concludes by using the inequality above and plugging in η and γ. □

Theorem. 6. *For all $t \in \{1, \dots, T\}$ and $i \in \{1, \dots, N\}$, the weight of the experts $w_{t,i}$ in fixed share algorithm is identical to $w'_t(i)$ for any $\alpha \in [0, 1]$.*

Proof. We prove it by induction on t.

Base case: $t = 1$:

$$w_{1,i} = w'_1(i) = \frac{1}{N} \; \forall i$$

Assume that $w_{s,i} = w'_s(i)$ for all i and $s < t$, we get

$$\begin{aligned}
w'_t(i) &= \sum_{i_1,\ldots,i_t,i_{t+2},\ldots,i_T} w'_t\left(i_1,\ldots,i_t,i,i_{t+2},\ldots,i_T\right) \\
&= \sum_{i_1,\ldots,i_t} e^{-\eta\sum_{s=1}^{t-1} l_{s,i_s}} \cdot w'_1\left(i_1,\ldots,i_t,i\right) \\
&= \sum_{i_1,\ldots,i_t} e^{-\eta\sum_{s=1}^{t-1} l_{s,i_s}} \cdot w'_1\left(i_1,\ldots,i_t\right)\left(\frac{\alpha}{N}+(1-\alpha)\mathbb{I}_{i_t=i}\right) \\
&= \sum_{i_1,\ldots,i_t} e^{-\eta l_{t-1,i_{t-1}}} \cdot w'_{t-1}\left(i_1,\ldots,i_t\right)\left(\frac{\alpha}{N}+(1-\alpha)\mathbb{I}_{i_t=i}\right) \\
&= \sum_{i_t} e^{-\eta l_{t-1,i_{t-1}}} \cdot w'_{t-1}(i_t)\left(\frac{\alpha}{N}+(1-\alpha)\mathbb{I}_{i_t=i}\right) \\
&= \sum_{i_t} e^{-\eta l_{t-1,i_{t-1}}} \cdot w_{t-1,i_t}\left(\frac{\alpha}{N}+(1-\alpha)\mathbb{I}_{i_t=i}\right) \\
&= \sum_{i_t} \hat{w}_{t,i_t}\left(\frac{\alpha}{N}+(1-\alpha)\mathbb{I}_{i_t=i}\right) \\
&= w_{t,i}
\end{aligned}$$

□

Theorem. 7. *For all $T \geq 1$, the Fixed Share algorithm satisfies*

$$\textit{Regret}_T(i_1,\ldots,i_T) \leq \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\alpha^k(1-\alpha)^{T-k-1}} + \frac{\eta}{8}T$$

for all action sequences $i_1,\ldots,i_n$, where $k = \sum_{t=2}^{T}\mathbb{I}_{i_{t-1}=i_t}$ is the number of switches occur in the sequence $\{i_1,\ldots,i_T\}$.

Proof. By definition of w'_1, we have

$$w'_1(i_1,\ldots,i_T) = \frac{1}{N}\left(\frac{\delta}{N}\right)^k\left(1-\delta+\frac{\delta}{N}\right)^{T-k-1} \geq \frac{1}{N}\left(\frac{\delta}{N}\right)^k(1-\delta)^{T-k-1}$$

Moreover, I claim that

$$\sum_{t=1}^{N}\sum_{i=1}^{N} p_{t,i}l_{t,i} \leq \frac{1}{\eta}\ln\frac{1}{W_T} + \frac{\eta T}{8} \tag{4.7}$$

We first show that the inequality above holds. Notice that

$$\begin{aligned}\frac{W_t}{W_{t-1}} &= (1-\alpha)\sum_{i=1}^{N}\frac{w_{t-1,i}}{W_{t-1}}e^{-\eta l_{t-1,i}} + \alpha\sum_{i=1}^{N}\frac{w_{t-1,i}}{W_{t-1}}e^{-\eta l_{t-1,i}} \\ &= \sum_{i=1}^{N} p_{t-1,i}e^{-\eta l_{t-1,i}}\end{aligned}$$

Taking the logarithm on both sides and using Jersey's inequality, we obtain

$$\begin{aligned}\ln\frac{W_{t+1}}{W_t} &= \ln\sum_{i=1}^{N} p_{t,i}e^{-\eta l_{t,i}} \\ &\leq -\eta\sum_{i=1}^{N} p_{t,i}l_{t,i} + \frac{\eta^2}{8}\end{aligned}$$

Summing both sides over t and solving for $\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}l_{t,i}$, we get

$$\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}l_{t,i} \leq \frac{1}{\eta}\ln\frac{1}{W_{T+1}} + \frac{\eta T}{8}$$

$\ln W_1$ is removed becauese $W_1 \leq 1$. Using inequality 4.7, we have

$$\begin{aligned}\sum_{t=1}^{N}\sum_{i=1}^{N} p_{t,i}l_{t,i} &\leq \frac{1}{\eta}\ln\frac{1}{W_{T+1}} + \frac{\eta T}{8} \\ &\leq \frac{1}{\eta}\ln\frac{1}{w'_{T+1}(i_1,\ldots,i_T)} + \frac{\eta T}{8} \\ &= \frac{1}{\eta}\left(\ln\frac{1}{w'_1(i_1,\ldots,i_T)} + \eta\sum_{t=1}^{T} l_{t,i_t}\right) + \frac{\eta T}{8} \\ &\leq \sum_{t=1}^{T} l_{t,i_t} + \frac{1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{(\delta/N)^k(1-\delta)^{T-k-1}} + \frac{\eta T}{8} \\ &= \sum_{t=1}^{T} l_{t,i_t} + \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\delta^k(1-\delta)^{T-k-1}} + \frac{\eta T}{8}\end{aligned}$$

□

Theorem. 8. *For all $T \geq 1$ and k such that $0 \leq k \leq T$, if $\alpha = \frac{k}{T-1}$ and $\eta =$*

$\sqrt{\frac{8}{T}\left((k+1)\ln N + (T-1)H\left(\frac{k}{T-1}\right)\right)}$, *then the following bound holds*

$$\textit{Regret}_T(i_1,\ldots,i_T) \leq \sqrt{\frac{T}{2}\left((k+1)\ln N + (T-1)H\left(\frac{k}{T-1}\right)\right)}$$

for all action sequences $i_1,\ldots,i_n$ *and* $H(x) = -x\ln x - (1-x)\ln(1-x)$ *for* $x \in [0,1]$.

Proof. Starting with theorem 7, we have

$$\begin{aligned}
\text{Regret}_T(i_1,\ldots,i_T) &\leq \frac{k+1}{\eta}\ln N + \frac{1}{\eta}\ln\frac{1}{\alpha^k(1-\alpha)^{T-k-1}} + \frac{\eta}{8}T \\
&= \frac{k+1}{\eta}\ln N - \frac{k}{\eta}\ln\frac{k}{T-1} - \frac{T-k-1}{\eta}\ln\left(1-\frac{k}{T-1}\right) + \frac{\eta}{8}T \\
&= \frac{k+1}{\eta}\ln N + \frac{1}{\eta}(T-1)H\left(\frac{k}{T-1}\right) + \frac{\eta}{8}T
\end{aligned}$$

The proof concludes by replacing the choice of η into the regret function. □

Theorem. 9. *For any interval of rounds* $\mathcal{I} = \{t_1,\ldots,t_2\} \subseteq \{1,\ldots,T\}$*, Online Gradient Descent with step sizes* η *guarantees the following regret:*

$$\textit{Regret}_{\mathcal{I}} = \sum_{t\in\mathcal{I}} f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}^*\in\mathcal{K}}\sum_{t\in\mathcal{I}} f_t(\boldsymbol{x}^*) \leq \frac{\|\boldsymbol{x}_{t_1} - \boldsymbol{x}^*\|^2}{2\eta} + \frac{\eta G^2 T}{2} \tag{4.8}$$

Proof. Let us first consider the following inequality:

$$\begin{aligned}
\|\boldsymbol{x}_{t+1} - \boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^*\|^2 &= \|\boldsymbol{\Pi}_{\mathcal{K}}\left(x_t - \eta\nabla f_t(\boldsymbol{x}_t)\right) - \boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^*\|^2 \\
&\leq \|\left(\boldsymbol{x}_t - \boldsymbol{x}^*\right) - \eta\nabla f_t(\boldsymbol{x}_t)\|^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^*\|^2 \\
&= \eta^2\left\|\nabla f_t(\boldsymbol{x}_t)\right\|^2 - 2\eta\nabla f_t(\boldsymbol{x}_t)^\top\left(\boldsymbol{x}_t - \boldsymbol{x}^*\right)
\end{aligned}$$

If we sum both sides over time, the intermediate term of the left side is cancelled out,

leaving us with the following

$$\begin{aligned}\sum_{t\in\mathcal{I}} \|\boldsymbol{x}_{t+1}-\boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_t-\boldsymbol{x}^*\|^2 &= \|\boldsymbol{x}_{t_2+1}-\boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_{t_1}-\boldsymbol{x}^*\|^2 \\ &\leq \sum_{t\in\mathcal{I}} \left[\eta^2 \|\nabla f_t(\boldsymbol{x}_t)\|^2 - 2\eta \nabla f_t(\boldsymbol{x}_t)^\top (\boldsymbol{x}_t-\boldsymbol{x}^*)\right] \\ &\leq \eta^2 G^2 T - 2\eta \sum_{t\in\mathcal{I}} \left[\nabla f_t(\boldsymbol{x}_t)^\top (\boldsymbol{x}_t-\boldsymbol{x}^*)\right]\end{aligned}$$

Rearranging the terms, we get

$$\sum_{t\in\mathcal{I}} \nabla f_t(\boldsymbol{x}_t)^\top (\boldsymbol{x}_t-\boldsymbol{x}^*) \leq \frac{1}{2}\eta G^2 T - \frac{\|\boldsymbol{x}_{t_2+1}-\boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_{t_1}-\boldsymbol{x}^*\|^2}{2\eta}$$

Thus, we have

$$\begin{aligned}\text{Regret}_{\mathcal{I}} &= \sum_{t\in\mathcal{I}} f_t(\boldsymbol{x}_t) - f_t(\boldsymbol{x}^*) \\ &\leq \sum_{t\in\mathcal{I}} \nabla f(\boldsymbol{x})^\top (\boldsymbol{x}_t-\boldsymbol{x}^*) \\ &\leq \frac{1}{2}\eta G^2 T - \frac{\|\boldsymbol{x}_{t_2+1}-\boldsymbol{x}^*\|^2 - \|\boldsymbol{x}_{t_1}-\boldsymbol{x}^*\|^2}{2\eta} \\ &\leq \frac{1}{2}\eta G^2 T + \frac{\|\boldsymbol{x}_{t_1}-\boldsymbol{x}^*\|^2}{2\eta}\end{aligned}$$

where the first inequality is derived from the definition of subgradient. □

Bibliography

[1] Shipra Agrawal and Nikhil R. Devanur. *Bandits with concave rewards and convex knapsacks*. 2014. arXiv: 1402.5758 [cs.LG].

[2] Peter Auer et al. “The Nonstochastic Multiarmed Bandit Problem”. In: *SIAM Journal on Computing* 32.1 (2002), pp. 48–77.

[3] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. “Bandits with Knapsacks”. In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE, Oct. 2013.

[4] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. “Bandits with Knapsacks”. In: *J. ACM* 65.3 (Mar. 2018). ISSN: 0004-5411.

[5] Martino Bernasconi et al. *Bandits with Replenishable Knapsacks: the Best of both Worlds*. 2023. arXiv: 2306.08470 [cs.LG].

[6] Djallel Bouneffouf and Irina Rish. *A Survey on Practical Applications of Multi-Armed and Contextual Bandits*. 2019. arXiv: 1904.10040 [cs.LG].

[7] Sébastien Bubeck and Nicolò Cesa-Bianchi. *Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems*. 2012. arXiv: 1204.5721 [cs.LG].

[8] Matteo Castiglioni, Andrea Celli, and Christian Kroer. *Online Learning under Budget and ROI Constraints via Weak Adaptivity*. 2024. arXiv: 2302.01203 [cs.GT].

[9] Andrea Celli, Matteo Castiglioni, and Christian Kroer. *Best of Many Worlds Guarantees for Online Learning with Knapsacks*. 2023. arXiv: 2202.13710 [cs.LG].

[10] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[11] Nicolò Cesa-Bianchi et al. *Mirror Descent Meets Fixed Share (and feels no regret)*. 2012. arXiv: 1202.3323 [cs.LG].

[12] Boxiao Chen et al. “Dynamic Pricing and Inventory Control with Fixed Ordering Cost and Incomplete Demand Information”. In: *Management Science* 68 (Dec. 2021).

[13] Mark Herbster and Manfred K. Warmuth. “Tracking the Best Expert”. In: *Machine Learning* 32 (1995), pp. 151–178.

[14] Steven C.H. Hoi et al. “Online learning: A comprehensive survey”. In: *Neurocomputing* 459 (2021), pp. 249–289. ISSN: 0925-2312.

[15] Nicole Immorlica et al. *Adversarial Bandits with Knapsacks*. 2023. arXiv: 1811.11881 [cs.DS].

[16] Raunak Kumar and Robert Kleinberg. *Non-monotonic Resource Utilization in the Bandits with Knapsacks Problem*. 2022. arXiv: 2209.12013 [cs.LG].

[17] Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

[18] Xiaocheng Li, Chunlin Sun, and Yinyu Ye. *The Symmetry between Arms and Knapsacks: A Primal-Dual Approach for Bandits with Knapsacks*. 2021. arXiv: 2102.06385 [cs.LG].

[19] Shang Liu, Jiashuo Jiang, and Xiaocheng Li. *Non-stationary Bandits with Knapsacks*. 2022. arXiv: 2205.12427 [cs.LG].

[20] Will Ma. “Improvements and Generalizations of Stochastic Knapsack and Multi-Armed Bandit Approximation Algorithms: Extended Abstract”. In: *Proceedings of the 2014 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1154–1163.

[21] Gergely Neu. *Explore no more: Improved high-probability regret bounds for non-stochastic bandits*. 2015. arXiv: 1506.03271 [cs.LG].

[22] Anshuka Rangi et al. “Online Learning and Decision Making with Partial Information, a Feedback Perspective”. AAI28541527. PhD thesis. 2021.

[23] Karthik Abinav Sankararaman and Aleksandrs Slivkins. *Bandits with Knapsacks beyond the Worst-Case*. 2021. arXiv: 2002.00253 [cs.LG].

[24] Shai Shalev-Shwartz. *Online Learning and Online Convex Optimization*. 2012.

[25] Aleksandrs Slivkins. *Introduction to Multi-Armed Bandits*. 2022. arXiv: 1904.07272 [cs.LG].

[26] Niranjan Srinivas et al. “Information-Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting”. In: *IEEE Transactions on Information Theory* 58.5 (May 2012), pp. 3250–3265. ISSN: 1557-9654.