Multinomial Logit Bandit with Low Switching Cost

Kefan Dong [1] **Yingkai Li** [2] **Qin Zhang** [3] **Yuan Zhou** [4]

Abstract

We study multinomial logit bandit with limited adaptivity, where the algorithms change their exploration actions as infrequently as possible when achieving almost optimal minimax regret. We propose two measures of adaptivity: the assortment switching cost and the more fine-grained item switching cost. We present an anytime algorithm (AT-DUCB) with $O(N \log T)$ assortment switches, almost matching the lower bound $\Omega(\frac{N \log T}{\log \log T})$. In the fixed-horizon setting, our algorithm FH-DUCB incurs $O(N \log \log T)$ assortment switches, matching the asymptotic lower bound. We also present the ESUCB algorithm with item switching cost $O(N \log^2 T)$.

1. Introduction

The dynamic assortment selection problem with the multinomial logic (MNL) choice model, also called MNL-bandit, is a fundamental problem in online learning and operations research. In this problem we have N distinct items, each of which is associated with a known reward r_i and an *unknown* preference parameter v_i. In the MNL choice model, given a subset $S \subseteq [N] \overset{\text{def}}{=} \{1, 2, 3, \ldots, N\}$, the probability that a user chooses $i \in S$ is given by

$$p_i(S) = \begin{cases} \dfrac{v_i}{v_0 + \sum_{j \in S} v_j} & \text{if } i \in S \cup \{0\} \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

where "0" stands for the case that the user does not choose any item, and v_0 is the associated preference parameter. As a convention (see, e.g. Agrawal et al., 2019), we assume that no-purchase is the most frequent choice, which is very natural in retailing. W.l.o.g., we assume $v_0 = 1$, and $v_i \le 1$ for all $i \in [N]$. The *expected reward* of the set S under the preference vector $\boldsymbol{v} = \{v_0, v_1, \ldots, v_N\}$ is defined to be

$$R(S, \boldsymbol{v}) = \sum_{i \in S} r_i p_i(S) = \sum_{i \in S} \frac{r_i v_i}{1 + \sum_{j \in S} v_j}. \quad (2)$$

For any online policy that selects a subset $S_t \subseteq [N]$ ($|S_t| \le K$, where K is a predefined capacity parameter) at each time step t, observes the user's choice a_t to gradually learn the preference parameters $\{v_i\}$, and runs for a horizon of T time steps, we define the *regret* of the policy to be

$$\text{Reg}_T \overset{\text{def}}{=} \sum_{t=1}^{T} \left(R(S^\star, \boldsymbol{v}) - R(S_t, \boldsymbol{v}) \right), \quad (3)$$

where $S^\star = \arg\max_{S \subseteq [N], |S| \le K} R(S, \boldsymbol{v})$ is the optimal assortment in hindsight. The goal is to find a policy to minimize the expected regret $\mathbb{E}[\text{Reg}_T]$ for all MNL-bandit instances.

To motivate the definition of the MNL-bandit problem, let us consider a fast fashion retailer such as Zara or Mango. Each of its product corresponds to an item in $[N]$, and by selling the i-th item the retailer takes a profit of r_i. At each specific time in each of its shops, the retailer can only present a certain number of items (say, at most K) on the shelf due to the space constraints. As a consequence, customers who visit the store can only pick items from the presented assortment (or, just buy nothing which corresponds to item 0), following a choice model. There has been a number of choice models being proposed in the literature (see, e.g., (Train, 2009; Luce, 2012) for overviews), and the MNL model is arguably the most popular one. The retailer certainly wants to maximize its profit by identifying the best assortment $S^\star$ to present. However, it does not know in advance customers' preferences to items in $[N]$ (i.e., the preference vector $\boldsymbol{v}$), to get which it has to learn from customers' actual choices. More precisely, the retailer needs to develop a policy to choose at each time step t an assortment $S_t \subseteq [N]$ ($|S_t| \le K$) based on the previous presented assortments $S_1, \ldots, S_{t-1}$ and customers' choices in the past $(t-1)$ time steps. The retailer's expected reward in a time horizon T can be expressed by $\sum_{t=1}^{T} R(S_t, \boldsymbol{v})$, which is typically reformulated as the regret compared with the best policy in the form of (3).

Author names are listed in alphabetical order. [1] Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China. [2] Department of Computer Science, Northwestern University, Evanston, Illinois, USA. [3] Computer Science Department, Indiana University, Bloomington, Indiana, USA. [4] Department of ISE, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA. Correspondence to: Yuan Zhou <yuanz@illinois.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

The MNL-bandit problem has attracted quite some attention in the past decade (Rusmevichientong et al., 2010; Sauré & Zeevi, 2013; Agrawal et al., 2016; 2017; Chen & Wang, 2018). However, all these works do not consider an important practical issue for regret minimization: in reality it is often impossible to *frequently* change the assortment display. For example, in retail stores it may not be possible to change the display in the middle of the day, not mentioning doing it after each purchase. We thus hope to minimize the number of assortment switches in the selling time horizon *without* increasing the regret by much. Another advantage of achieving a small number of assortment switches is that such algorithms are easier to parallelize, which enables us to learn users' preferences much faster. This feature is particularly useful in applications such as online advertising where it is easy to show the same assortment (i.e., a set of ads) in a large amount of end users' displays simultaneously.

We are interested in two kinds of switching costs under a time horizon T. The first is the *assortment switching cost*, defined as

$$\Psi_T^{(\mathrm{asst})} \overset{\mathrm{def}}{=} \sum_{t=1}^{T} \mathbb{I}[S_t \neq S_{t+1}].$$

The second is the *item switching cost*, defined as

$$\Psi_T^{(\mathrm{item})} \overset{\mathrm{def}}{=} \sum_{t=1}^{T} |S_t \oplus S_{t+1}|\,,$$

where binary operator $\oplus$ computes the symmetric difference of the two sets. In comparison, the item switching cost is more fine-grained and put less penalty if two neighboring assortments are "almost the same". As a straightforward observation, we always have that

$$\Psi_T^{(\mathrm{asst})} \le \Psi_T^{(\mathrm{item})} \le \min\{2K, N\} \cdot \Psi_T^{(\mathrm{asst})}. \qquad (4)$$

Our results. In this paper we obtain the following results for MNL-bandit with low switching cost. By default all log's are of base 2.

We first introduce an algorithm, AT-DUCB, that achieves almost optimal regret (up to a logarithmic factor) and incurs an assortment switching cost of $O(N \log T)$; this algorithm is *anytime*, i.e., it does *not* need to know the time horizon T in advance. We then show that the AT-DUCB algorithm achieves almost optimal assortment switching cost. In particular, we prove that every anytime algorithm that achieves almost optimal regret must incur an assortment switching cost of at least $\Omega(N \log T / \log\log(NT))$. These results are presented in Section 2.

When the time horizon is known beforehand, we obtain an algorithm, FH-DUCB, that achieves almost optimal regret (up to a logarithmic factor) and incurs an assortment switching cost of $O(N \log\log T)$. We also prove the optimality of this switching cost by establishing a matching lower bound. See Section 3.

For item switches, while the trivial application of (4) leads to $O(N^2 \log T)$ and $O(N^2 \log\log T)$ item switching cost bounds for AT-DUCB and FH-DUCB respectively, in Section 4, we design a new algorithm, ESUCB, to achieve an item switching cost of $O(N \log^2 T)$. In Appendix F, we show that a more careful modification to the algorithm further improves the item switching cost to $O(N \log T)$.

We make two interesting observations from the results above: (1) there is a *separation* between the assortment switching complexities when knowing the time horizon T and when not; in other words, the time horizon T is useful for achieving a smaller assortment switching cost; (2) the item switching cost is only at most a logarithmic factor higher than the assortment switching cost.

Technical contributions. We combine the epoch-based offering algorithm for MNL-bandits (Agrawal et al., 2019) and a natural delayed update policy in the design of AT-DUCB. Although a similar delayed update rule has been recently analyzed for multi-armed bandits and Q-learning (Bai et al., 2019), and such a result does not seem surprising, we present it in the paper as a warm-up to help the readers get familiar with a few algorithmic techniques commonly used for the MNL-bandit problem.

Our first main technical contribution comes from the design of FH-DUCB algorithm, where we invent a novel delayed update policy that uses the horizon information to improve the switching cost from $O(N \log T)$ to $O(N \log\log T)$. We note that for the ordinary multi-armed bandit problem, recent works (Gao et al., 2019) and (Simchi-Levi & Xu, 2019) managed to show a similar $O(N \log\log T)$ switching cost with known horizon. However, their update rules do not have to utilize the learned parameters for the arms, and a straightforward conversion of such update rules to the MNL-bandit problem does not produce the desired guarantees. In contrast, our update rule, formally described in (6), carefully exploits the structure of the MNL-bandits and uses the information of the partially learned preference parameters (more specifically, $\hat{v}_{i,\tau_i}$ in (6)) to adaptively decide when to switch to a different assortment.

Our second main technical contribution is the ESUCB algorithm for the low item switching cost. The technical challenge here stems from the fact that the low item switching cost is a much stronger requirement than the low assortment switching cost, and simple lazy updates with the doubling trick and the straightforward analysis will show that the item switching cost is at most N times the assortment switching cost (see (4)), leading to a total item switching cost of $O(N^2 \log T)$. To reducing the extra factor N, we propose the idea of decoupling the learning for the optimal revenue and the assortment, so that the offering of the assortment is decided via optimizing a new objective function based on the (usually) fixed revenue estimate. Since the revenue estimates are fixed, the offered assortments enjoy improved stability, and the item switching cost can be upper bounded by careful analysis.

We remark that the item switching cost is a particularly interesting goal that arises in online learning problems when the actions are sets of elements, which is very different from traditional MAB and linear bandits. Thanks to our novel technical ingredients, we are able to bring the item switching cost down to almost the same order as the assortment switching cost. We hope our results will inspire future study of the switching costs in both settings for other online learning problems with set actions.

Related work. MNL-bandit was first studied in (Rusmevichientong et al., 2010) and (Sauré & Zeevi, 2013), where the authors took the "explore-then-commit" approach, and proposed algorithms with regret $O(N^2 \log^2 T)$ and $O(N \log T)$ respectively under the assumption that the gap between the best and second-to-the-best assortments is known. (Agrawal et al., 2016) removed this assumption using a UCB-type algorithm, which achieves a regret of $O(\sqrt{NT} \log T)$. An almost tight regret lower bound of $\Omega(\sqrt{NT})$ was later given by (Chen & Wang, 2018). (Agrawal et al., 2017) proposed an algorithm using Thompson Sampling, which achieves comparable regret bound to the UCB-type algorithms while demonstrates a better numerical performance.

Learning with low policy switches (also called learning in the *batched model* or *limited adaptivity*) has recently been studied in reinforcement learning for several other problems, including stochastic multi-armed bandits (Perchet et al., 2015; Jun et al., 2016; Agarwal et al., 2017; Gao et al., 2019; Esfandiari et al., 2019; Simchi-Levi & Xu, 2019), Q-learning (Bai et al., 2019), and online-learning (Cesa-Bianchi et al., 2013). This research direction is motivated by the fact that in many practical settings, the change of learning policy is very costly. For example, in clinical trials, every treatment policy switch would trigger a separate approval process. In crowdsourcing, it takes time for the crowd to answer questions, and thus a small number of rounds of interactions with the crowd is desirable. The performance of the learning would be much better if the data is processed in batches and during each batch the learning policy is fixed.

2. Warm-up: An anytime algorithm with $O(N \log T)$ assortment switches

As a warm-up, we begin with a simple anytime algorithm using at most $O(N \log T)$ assortment switches. Our algorithm combines the epoch-based offering framework introduce in (Agrawal et al., 2016) and a deferred update policy. We will first briefly explain the epoch-based offering procedure, and then present and analyze our algorithm.

The epoch-based offering. In the epoch-based offering framework, whenever we are to offer an assortment S, instead of offering it for only one time period, we keep offering S until a no-purchase decision (item 0) is observed, and refer to all the consecutive time periods involved in this procedure as an *epoch*. The detailed offering procedure is described in Algorithm 1, where t is the global counter for the time period, and $\{\Delta_i\}$ records the number of purchases made for each item i in the epoch.

Algorithm 1: EXPLORATION(S)

```
Initialize: $\Delta_i \leftarrow 0$ for all $i \in [N]$;
while TRUE do
    $t \leftarrow t + 1$;
    Offer assortment $S$, and observe purchase decision $a_t$;
    If $a_t = 0$ then return $\{\Delta_i\}$;
    $\Delta_{a_t} \leftarrow \Delta_{a_t} + 1$;
```

The following key observation for EXPLORATION(S) states that $\{\Delta_i\}$ forms an unbiased estimate for the utility parameters of all items in S.

Observation 1. *Let $\{\Delta_i\}$ be returned by* EXPLORATION(S). *For each $i \in S$, Δ_i is an independent geometric random variable with mean v_i. Moreover, one can verify that $\mathbb{E}[\Delta_i] = v_i$ and*

$$\Pr[\Delta_i = k] = \left(\frac{v_i}{1 + v_i}\right)^k \left(\frac{1}{1 + v_i}\right), \forall k \in \mathbb{N}.$$

At any time of the algorithm when an epoch has ended, for each item $i \in [N]$, we let $\bar{v}_i = n_i / T_i$ where T_i is the number of the past epochs in which i is included in the offered assortment, and n_i is the total number of purchases for item i during all past epochs. By Observation 1, we know that $\bar{v}_i$ is also an unbiased estimate of v_i. In (Agrawal et al., 2016), the following upper confidence bound (UCB) is constructed for each $i \in [N]$,

$$\hat{v}_i = \bar{v}_i + \sqrt{\frac{48 \bar{v}_i \ln(\sqrt{N}\ell + 1)}{T_i}} + \frac{48 \ln(\sqrt{N}\ell + 1)}{T_i}. \quad (5)$$

We will compute the assortment for the next epoch based on the vector of UCB values $\hat{\boldsymbol{v}} = (\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_n)$.

We describe our algorithm in Algorithm 2, which can be seen as an adaptation of the one in (Agrawal et al., 2016). The main difference from (Agrawal et al., 2016) is that the UCB values (and hence the assortment) is updated only when T_i reaches an integer power of 2 for any item $i \in [N]$.

Algorithm 2: Anytime Deferred Update UCB (AT-DUCB)

```
1 Initialize: $\hat{v}_i \leftarrow 1, T_i \leftarrow 0$ for all $i \in [N]$, $t \leftarrow 0$;
2 for $\ell \leftarrow 1, 2, 3, \dots,$ do
3     Compute $S_\ell = \arg\max_{S \subseteq [N]:|S| \le K} R(S, \hat{\boldsymbol{v}})$;
4     $\{\Delta_i\} \leftarrow$ EXPLORATION$(S)$;
5     for $i \in S$ do
6         $n_i \leftarrow n_i + \Delta_i$ and $T_i \leftarrow T_i + 1$;
7         if $T_i = 2^k$ for some $k \in \mathbb{Z}$ then
8             $\bar{v}_i \leftarrow n_i/T_i$; $\hat{v}_i \leftarrow \min\left\{\hat{v}_i, \bar{v}_i + \sqrt{\frac{48\bar{v}_i \ln(\sqrt{N}\ell+1)}{T_i}} + \frac{48\ln(\sqrt{N}\ell+1)}{T_i}\right\}$;
```

This deferred update strategy is implemented in Line 7. Also note that instead of directly evaluating (5), the update in Line 8 makes sure that $\hat{v}_i$ is non-increasing as the algorithm proceeds. We comment that the optimization task in Line 3 can be done efficiently, as studied in, for example, (Rusmevichientong et al., 2010).

Theorem 2. *For any time horizon T, the expect regret incurred by Algorithm 2 is*

$$\mathbb{E}\left[\mathrm{Reg}_T\right] \lesssim \sqrt{NT \log T},$$

and the expected number of assortment switches $\mathbb{E}[\Psi_T^{(\mathrm{asst})}]$ is $O(N \log T)$. [1]

The proof of the regret upper bound in Theorem 2 is similar to that of (Agrawal et al., 2016), except for a more careful analysis about the deferred update rule. For completeness, we prove this part in Appendix A.

Proof of the assortment switch upper bound in Theorem 2. Let $\mathcal{D}_i^{(\ell)}$ be the event that Line 8 is executed in Algorithm 2 for item i at the ℓ-th epoch. Recall that the assortment S_ℓ is computed by $S_\ell = \arg\max_{S \subseteq [N], |S| \le K} R(S, \hat{\boldsymbol{v}})$, and $\hat{\boldsymbol{v}}$ is updated after epoch ℓ only when $\mathcal{D}_i^{(\ell)}$ happens for some $i \in [N]$. Let L be the total number of epochs at or before time T; we thus have $\sum_{\ell=1}^{L} \mathbb{I}[\mathcal{D}_i^{\ell}] \le \log T$. We then have that

$$\mathbb{E}[\Psi_T^{(\mathrm{asst})}] = \mathbb{E} \sum_{t=1}^{T-1} \mathbb{I}[S_t \neq S_{t+1}]$$
$$\le \sum_{\ell=1}^{L} \sum_{i=1}^{N} \mathbb{I}[\mathcal{D}_i^{(\ell)}] = \sum_{i=1}^{N} \sum_{\ell=1}^{L} \mathbb{I}[\mathcal{D}_i^{(\ell)}] \lesssim N \log T.$$

□

The lower bound. We complement our algorithmic result with the following almost matching lower bound. The theorem states that the number of assortment switches has to be $\Omega(N \log T / \log\log(NT))$, if the algorithm is anytime and incurs only $\sqrt{NT} \times \mathrm{poly}\log(NT)$ regret. The proof of Theorem 3 can be found in Appendix E.1.

Theorem 3. *There exist universal constants $d_0, d_1 > 0$ such that the following holds. For any constant $C \ge 1$, if an anytime algorithm $\mathcal{A}$ achieves expected regret at most $d_0\sqrt{NT}(\ln(NT))^C$ for all T and all instances with N items, then for any $N \ge 2$, $T_0 \ge N$ and T_0 greater than a sufficiently large constant that only depends on C, there exists an instance with N items and a time horizon $T \in [T_0, T_0^2]$, such that the expected number of assortment switches before time T is at least $d_1 N \log T/(C \log\log(NT))$.*

3. Achieving $O(N \log\log T)$ assortment switch with known horizons

When the time horizon is known to the algorithm, we can exploit this advantage via more carefully designed update policy to achieve only $O(N \log\log T)$ assortment switches. For the convenience of presentation, we first introduce a few notations.

Algorithm 3: UPDATE(i)

```
$\tau_i \leftarrow \tau_i + 1$; $T_i^{(\tau_i)} \leftarrow T_i^{(\tau_i - 1)} + |\mathcal{T}(i, \tau_i - 1)|$;
$n_i^{(\tau_i)} \leftarrow n_i^{(\tau_i - 1)} + n_{i,\tau_i - 1}$; $\bar{v}_{i,\tau_i} \leftarrow n_i^{(\tau_i)}/T_i^{(\tau_i)}$;
$\hat{v}_{i,\tau_i} \leftarrow \min\left\{\hat{v}_{i,\tau_i - 1}, \bar{v}_{i,\tau_i} + \sqrt{\frac{48\bar{v}_{i,\tau_i} \ln(\sqrt{N}T^2+1)}{T_i^{(\tau_i)}}} + \frac{48\ln(\sqrt{N}T^2+1)}{T_i^{(\tau_i)}}\right\}$;
```

For each item $i \in [N]$, we divide the time periods into consecutive *stages* where the boundaries between any two neighboring stages are marked by the UCB updates for item i. Note that the division for the stages may be different for different items. For any $\tau \in \{1, 2, 3, \dots\}$, let $\mathcal{T}(i, \tau)$ be the set of epochs to offer item i, in stage τ for the item. Let $T_i^{(\tau)} = \sum_{\tau'=1}^{\tau-1} |\mathcal{T}(i, \tau')|$ be the total number of epochs to offer item i, *before* stage τ for the item, and let $n_i^{(\tau)}$ be the total number of purchases for item i in the epochs counted by $T_i^{(\tau)}$. We can therefore define $\bar{v}_{i,\tau} \overset{\mathrm{def}}{=} n_i^{(\tau)}/T_i^{(\tau)}$ as an unbiased estimate of v_i based on the observations before stage τ. Similarly to (5), we can define $\hat{v}_{i,\tau}$ as a UCB for v_i. The UPDATE(i) procedure (formally described in Algorithm 3) is invoked whenever the main algorithm decides to conclude the current stage for item i and update the UCB for v_i together with the quantities defined above, where τ_i is the counter for the number of stages for item i, and $n_{i,\tau}$ is the number of purchases observed in stage τ for

[1] For two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n = O(b_n)$ or $a_n \lesssim b_n$ if there exists a *universal* constant $C < \infty$ such that $\limsup_{n\to\infty} |a_n|/|b_n| \le C$. Similarly, we write $a_n = \Omega(b_n)$ or $a_n \gtrsim b_n$ if there exists a *universal* constant $c > 0$ such that $\liminf_{n\to\infty} |a_n|/|b_n| \ge c$.

item i.

The key to the design of our main algorithm for the fixed time horizon setting is a new trigger for updating the UCB values. Let $\tau_0 = \lceil \log\log(T/N) + 1 \rceil$, for each item $i \in [N]$, we will conclude the current stage τ_i and invoke $\textsc{Update}(i)$ whenever the following condition $\mathcal{P}(i, \tau_i)$ is satisfied. Note that $\mathcal{P}(i, \tau_i)$ is adaptive to the estimated parameters $\hat{v}_{i,\tau_i}$ to customize the number of epochs between assortment switches for each item. More specifically, the smaller $\hat{v}_{i,\tau_i}$ is, the less regret may be incurred by offering item i, and therefore the longer we can offer item i without switching and incurring too large regret, and this is reflected in the design of $\mathcal{P}$.

$$\mathcal{P}(i, \tau_i) \overset{\text{def}}{=} \begin{cases} |\mathcal{T}(i, \tau_i)| \geq 1 + \sqrt{\frac{T \cdot T_i^{(\tau_i)}}{N}} & \text{if } \tau_i < \tau_0 \\ |\mathcal{T}(i, \tau_i)| \geq 1 + \sqrt{\frac{T \cdot T_i^{(\tau_i)}}{N \cdot \hat{v}_{i,\tau_i}}} & \\ \qquad \text{and } \hat{v}_{i,\tau_0} > 1/\sqrt{NT} & \text{if } \tau_i \geq \tau_0 \end{cases}. \tag{6}$$

For each epoch ℓ, we use $\tau_i(\ell)$ to denote the stage (in terms of item i) where epoch ℓ belongs to. We present the details of our main algorithm in Algorithm 4. The algorithm is terminated whenever the time step t reaches the horizon T.

Theorem 4. *For any given time horizon $T \geq N^4$, we have the following upper bound for the expected regret:*

$$\mathbb{E}\left[\mathrm{Reg}_T\right] \lesssim \sqrt{NT \ln(\sqrt{N}T^2 + 1)} \cdot \log\log T,$$

and the following upper bound for the expected number of assortment switches:

$$\mathbb{E}\left[\Psi_T^{(\mathrm{asst})}\right] \lesssim N \log\log T.$$

To prove Theorem 4, we first define the desired events. Let

$$\mathcal{E}_{i,\tau}^{(1)} \overset{\text{def}}{=} \Big\{ \hat{v}_{i,\tau} \geq v_i \text{ and } \hat{v}_{i,\tau} \leq v_i + \sqrt{\frac{144 v_i \ln(\sqrt{N}T^2+1)}{T_i^{(\tau)}} + \frac{144\ln(\sqrt{N}T^2+1)}{T_i^{(\tau)}}} \Big\},$$

and

$$\mathcal{E}^{(1)} \overset{\text{def}}{=} \cap_{i,\tau} \mathcal{E}_{i,\tau}^{(1)}.$$

We also let

$$\mathcal{E}_{i,\tau}^{(2)} \overset{\text{def}}{=} \Big\{ n_{i,\tau} \geq \frac{1}{2} v_i |\mathcal{T}(i,\tau)|, \quad \text{if } v_i \geq \frac{1}{2}\sqrt{\frac{1}{NT}} \text{ and } |\mathcal{T}(i,\tau)| \geq \frac{T}{4N \cdot v_i} \Big\},$$

and

$$\mathcal{E}^{(2)} \overset{\text{def}}{=} \cap_{i,\tau} \mathcal{E}_{i,\tau}^{(2)}.$$

Finally, let $\mathcal{E} = \mathcal{E}^{(1)} \cap \mathcal{E}^{(2)}$. In Appendix B.1, we prove the following lemma.

Algorithm 4: Deferred Update UCB for Fixed Time Horizon (FH-DUCB)

```
Input : The time horizon $T$.
Initialize: $\tau_i \leftarrow 1, \hat{v}_{i,\tau_i} \leftarrow 1, n_{i,\tau_i} \leftarrow 0, \mathcal{T}(i,\tau_i) \leftarrow \emptyset, T_i^{(1)} \leftarrow 0, n_i^{(1)} \leftarrow 0$ for all $i \in [N]$;
$t \leftarrow 0$, $S_0 \leftarrow [N]$;
for $\ell \leftarrow 1, 2, 3, \ldots,$ do
    $S_\ell \leftarrow S_{\ell-1}$;
    if $\exists i : \mathcal{P}(i,\tau_i)$ holds then
        $\textsc{Update}(i)$ for all $i$ such that $\mathcal{P}(i,\tau_i)$ holds;
        Compute $S_\ell \leftarrow \arg\max_{S \subseteq [N]:|S| \leq K} R(S, \hat{\boldsymbol{v}}_\ell)$
          where $\hat{\boldsymbol{v}}_\ell = (\hat{v}_{i,\tau_i(\ell)})_{i \in [N]}$;
    $\{\Delta_i\} \leftarrow \textsc{Exploration}(S_\ell)$;
    for $i \in S$ do
        $n_{i,\tau_i} \leftarrow n_{i,\tau_i} + \Delta_i$; Add $\ell$ to $\mathcal{T}(i,\tau_i)$;
```

Lemma 5. *If $T \geq N^4$ and T is greater than a large enough universal constant, then $\Pr[\mathcal{E}] \geq 1 - \frac{14}{T}$.*

Bounds for the stage lengths. When $\mathcal{E}$ happens, we can infer the following useful lower bound for the lengths of the stages after τ_0. The lemma is proved in Appendix B.2.

Lemma 6. *Assume that $T \geq N^4$ and T is greater than a sufficiently large universal constant. Conditioned on $\mathcal{E}^{(1)}$, for each $i \in [N]$, if τ_0 is not the last stage for item i, we have that $v_i \geq \frac{1}{2}\sqrt{\frac{1}{NT}}$. Additionally, if $\hat{v}_{i,\tau_0} > 1/\sqrt{NT}$, then for all $\tau > \tau_0$ such that τ is not the last stage for i, we have that $|\mathcal{T}(i,\tau)| \geq (T/(2Nv_i))^{1-2^{-\tau+\tau_0+1}}$.*

Upper bounding the number of assortment switches. Suppose that there are L epochs before the algorithm terminates. We only need to upper bound $\mathbb{E}\sum_{i=1}^N \tau_i(L)$ which upper bounds the number of assortment switches $\mathbb{E}[\Psi_T^{(\mathrm{asst})}]$. For each $i \in [N]$, if $\tau_i(L) \geq \tau_0$ and $\hat{v}_{i,\tau_0} \leq 1/\sqrt{NT}$, we easily deduce that $\tau_i(L) \leq \tau_0 + 1$ because of the condition $\mathcal{P}(i,\tau_0)$. Otherwise, assuming that $\hat{v}_{i,\tau_0} > 1/\sqrt{NT}$, by Lemma 6, conditioned on $\mathcal{E}^{(1)}$, we have that $v_i \geq \frac{1}{2}\sqrt{\frac{1}{NT}}$ and $|\mathcal{T}(i,\tau)| \geq \frac{T}{4Nv_i}$ for all $\tau \in [\tau_0 + \log\log\frac{T}{2Nv_i} + 1, \tau_i(L) - 1]$. Because of $\mathcal{E}^{(2)}$, we have $n_{i,\tau} \geq \frac{v_i}{2} \cdot |\mathcal{T}(i,\tau)| \geq \frac{T}{8N}$ for all $\tau \in [\tau_0 + \log\log\frac{T}{2Nv_i} + 1, \tau_i(L) - 1]$. Therefore, we know that there are no more than $8N$ pairs of (i,τ) satisfying $\tau \in [\tau_0 + \log\log\frac{T}{2Nv_i} + 1, \tau_i(L) - 1]$. In total, conditioned on $\mathcal{E}$, we have that

$$\mathbb{E}\sum_{i=1}^N \tau_i(L)$$

$$\begin{aligned}
&\lesssim N\tau_0 + \sum_{i=1}^{N} \mathbb{I}\left[\hat{v}_{i,\tau_0} > 1/\sqrt{NT}\right] \log\log\frac{T}{2Nv_i} \\
&\quad + \mathbb{E}\sum_{i=1}^{N} \max\{\tau_i(L) - \tau_0 - \log\log\frac{T}{2Nv_i}, 0\} \\
&\lesssim N\log\log T + \sum_{i=1}^{N} \log\log\frac{T^{3/2}}{N^{1/2}} \lesssim N\log\log T, \qquad (7)
\end{aligned}$$

where the second inequality is because of Lemma 6. Finally, since the contribution to the expected number of assortment switches when $\mathcal{E}$ fails is at most $\Pr[\overline{\mathcal{E}}] \cdot T \le O(1)$ (because of Lemma 5), we prove the upper bound for the number of assortment switches in Theorem 4.

Upper bounding the expected regret. Let $E^{(\ell)}$ be the length of epoch ℓ, i.e., the number of time steps taken in epoch ℓ. Note that $E^{(\ell)}$ is a geometric random variable with mean value $(1 + \sum_{i \in S_\ell} v_i)$. Also recall that there are L epochs in total. Letting S^* be the optimal assortment, conditioned on event $\mathcal{E}^{(1)}$, we have that

$$\begin{aligned}
\mathbb{E}\left[\mathrm{Reg}_T\right] &= \mathbb{E}\sum_{\ell=1}^{L} E^{(\ell)}(R(S^\star, \boldsymbol{v}) - R(S_\ell, \boldsymbol{v})) \\
&= \mathbb{E}\sum_{\ell=1}^{L}\left(1 + \sum_{i \in S_\ell} v_i\right)(R(S^\star, \boldsymbol{v}) - R(S_\ell, \boldsymbol{v})) \\
&\le \mathbb{E}\sum_{\ell=1}^{L}\sum_{i \in S_\ell}(\hat{v}_{i,\tau_i(\ell)} - v_i) \\
&= \mathbb{E}\sum_{i=1}^{N}\sum_{\ell : i \in S_\ell}(\hat{v}_{i,\tau_i(\ell)} - v_i) \\
&= \mathbb{E}\sum_{i=1}^{N}\sum_{\tau=1}^{\tau_i(L)}\sum_{\ell \in \mathcal{T}(i,\tau)}(\hat{v}_{i,\tau} - v_i), \qquad (8)
\end{aligned}$$

where the inequality is due to Lemma 17. In the next lemma, we upper bound the contribution from each item i and stage τ to the upper bound in (8). The lemma is proved in Appendix B.3.

Lemma 7. *Conditioned on event $\mathcal{E}^{(1)}$, for any item i and any stage $\tau \le \tau_i(L)$, we have that*

$$\sum_{\ell \in \mathcal{T}(i,\tau)} (\hat{v}_{i,\tau} - v_i) \lesssim \sqrt{T\ln(\sqrt{N}T^2 + 1)/N}.$$

Combining Lemma 5, Lemma 7, inequalities (7) and (8), we have that

$$\begin{aligned}
\mathbb{E}\left[\mathrm{Reg}_T\right] &\le T \cdot \Pr[\overline{\mathcal{E}^{(1)}}] + \mathbb{E}\left[\mathrm{Reg}_T \;\middle|\; \mathcal{E}^{(1)}\right] \\
&\lesssim 1 + \mathbb{E}\sum_{i=1}^{N}\tau_i(L) \times \sqrt{\frac{T\ln(\sqrt{N}T^2 + 1)}{N}} \\
&\lesssim \sqrt{NT\ln(\sqrt{N}T^2 + 1)} \cdot \log\log T,
\end{aligned}$$

proving the expected regret upper bound in Theorem 4.

The lower bound. We prove the following matching lower bound in Appendix E.2.

Theorem 8. *For any constant $C \ge 0$ and time horizon T, if an algorithm $\mathcal{A}$ achieves expected regret $\mathbb{E}[\mathrm{Reg}_T]$ at most $\frac{1}{7525} \cdot \sqrt{NT}(\ln(NT))^C$ for all N-item instances, then there exists an N-item instance such that the expected number of assortment switches is*

$$\mathbb{E}[\Psi_T^{(\mathrm{asst})}] = \Omega(N\log\log T).$$

4. Optimizing the number of item switches

In this section, we study how to minimize the item switch cost while still achieving $\tilde{O}(\sqrt{NT})$ regret.

Algorithm 5: The Exponential Stride UCB algorithm (ESUCB) for MNL-Bandit

```
Initialize: $\hat{\theta} \leftarrow 1, \epsilon_1 \leftarrow 1/3, c_1 \leftarrow 44840$;
for $\tau \leftarrow 1, 2, 3, \ldots$ do
    $t_{\max} \leftarrow c_1 N \ln^3(NT/\delta)/\epsilon_\tau^2$;
    if CHECK$(\hat{\theta} - 3\epsilon_\tau, \hat{\theta} - \epsilon_\tau, t_{\max})$ then $\hat{\theta} \leftarrow \hat{\theta} - \epsilon_\tau$;
    $\epsilon_{\tau+1} \leftarrow \frac{2}{3}\epsilon_\tau$;
```

We now propose a new algorithm, Exponential Stride UCB (ESUCB), to achieve an item switching cost that is linear with N and poly-logarithmic with T. The specific guarantee of the ESUCB algorithm is presented in Theorem 10, the main theorem of this section. The key idea of the algorithm is to decouple the learning of the optimal expected revenue and the optimal assortment, which is made possible by the following lemma.

Lemma 9. *Define $G(\theta) \overset{def}{=} R(S_\theta, \boldsymbol{v})$, where $S_\theta \overset{def}{=} \arg\max_{S \subseteq [N] : |S| \le K}\left(\sum_{i \in S} v_i(r_i - \theta)\right)$. There exists a unique $\theta^\star$ such that*

$$G(\theta^\star) = \theta^\star = \max_{|S| \le K} R(S, \boldsymbol{v}).$$

Moreover,

(1) for any $\theta < \theta^\star$, we have that $G(\theta) > \theta$, and

(2) for any $\theta > \theta^\star$, we have that $G(\theta) < \theta$.

The proof of Lemma 9 is deferred to Appendix D.1. Motivated by the lemma, we present our ESUCB algorithm in Algorithm 5. The algorithm learns the optimal revenue $\theta^\star$ in the main loop, using a sequence of exponentially decreasing learning step size ϵ_τ. For each estimate $\hat{\theta}$, the CHECK procedure (Algorithm 6) learns the assortment $S_{\hat{\theta}}$ via the UCB method with deferred updates. (More precisely speaking, the algorithm learns $S_{\hat{\theta}-\epsilon_\tau}$ and $S_{\hat{\theta}-3\epsilon_\tau}$, and at Line 4, chooses one of them based on the UCB estimation $\hat{\rho}$ for the expected revenue of $S_{\hat{\theta}-\epsilon_\tau}$.) In the CHECK procedure, the variable t keeps the count of time steps and is updated in EXPLORATION. We also make the following notes: 1) The ESUCB algorithm needs the horizon T as input, and uses a confidence parameter δ, which is usually set as $1/T$. The whole algorithm terminates whenever the horizon T is reached. 2) At the optimization steps (Lines 6 and 9 of Algorithm 6), we have to adopt a deterministic tie breaking rule, e.g., we let the $\arg\max$ operator to return the S such that $\sum_{i \in S} 2^i$ is minimized among multiple maximizers.

Theorem 10. *Setting $\delta = 1/T$, we have the following upper bound for the expected regret of ESUCB:*

$$\mathbb{E}\left[\mathrm{Reg}_T\right] \lesssim \sqrt{NT} \cdot \log^{1.5}(NT),$$

and the item switching cost for ESUCB is

$$\mathbb{E}\left[\Psi_T^{(\mathrm{item})}\right] \lesssim N \log^2 T.$$

To prove Theorem 10, we upper bound the item switching cost and the expected regret separately.

Upper bounding the item switch cost. Since the estimate of $\theta^\star$ is fixed in CHECK, the outcome of $\arg\max_{S:|S| \le K} \sum_{i \in S} \hat{v}_i(r_i - \theta)$ (corresponding to Lines 6 and 9 of Algorithm 6) becomes more stable compared to that of $\arg\max_{S:|S| \le K} R(S, \hat{\boldsymbol{v}})$ in previous algorithms. Exploiting this advantage, we upper bound the number of item switches incurred by each call of CHECK as follows. The lemma is proved in Appendix D.2.

Lemma 11. *The item switch cost incurred by any invocation* CHECK$(\theta_l, \theta_r, t_{\max})$ *is $O(N \log T)$.*

Since the τ loop in Algorithm 5 iterates for only $O(\log T)$ times, Lemma 11 easily implies an $O(N \log^2 T)$ item switching cost upper bound for ESUCB. We also note that this bound can be improved to $O(N \log T)$ via a slight modification to the algorithm which is elaborated in Appendix F.

Upper bounding the expected regret. We first provide the following guarantees for CHECK.

Lemma 12 (Main Lemma for CHECK)**.** *For any invocation* CHECK$(\theta_l, \theta_r, t_{\max})$*, with probability at least $(1 - \delta/T)$, the following statements hold.*

(a) If CHECK *returns* true*, then $G(\theta_r) < \theta_r$.*

(b) If CHECK *returns* false*, then*

$$\theta^\star \ge \theta_r - \frac{2}{t_{\max}} \left(c_2 \sqrt{N t_{\max} \ln^3 \frac{NT}{\delta}} + c_3 N \ln^3 \frac{NT}{\delta} \right).$$

Algorithm 6: CHECK$(\theta_l, \theta_r, t_{\max})$

```
1  Initialize: $\hat{v}_i \leftarrow 1, T_i \leftarrow 0, n_i \leftarrow 0$ for all $i \in [N]$,
     $c_2 \leftarrow 688$, $c_3 \leftarrow 21732$;
2  $\rho \leftarrow 0, \hat{\rho} \leftarrow 1, b \leftarrow$ false, $t \leftarrow 0$;
3  for $\ell \leftarrow 1, 2, 3, \ldots$ do
4      if $\hat{\rho} < \theta_r$ then
5          $b \leftarrow$ true;
6          $S_\ell \leftarrow \arg\max_{S \subseteq [N], |S| \le K} \left(\sum_{i \in S} \hat{v}_i (r_i - \theta_l)\right)$;
7          $\{\Delta_i\} \leftarrow$ EXPLORATION$(S_\ell)$;
8      else
9          $S_\ell \leftarrow \arg\max_{S \subseteq [N], |S| \le K} \left(\sum_{i \in S} \hat{v}_i (r_i - \theta_r)\right)$;
10         $\{\Delta_i\} \leftarrow$ EXPLORATION$(S_\ell)$;
11         $\rho \leftarrow \rho + \sum_{i \in S_\ell} \Delta_i \cdot r_i$; $\hat{\rho} \leftarrow \frac{1}{t}\big(\rho +$
             $c_2 \sqrt{N t_{\max} \ln^3(NT/\delta)} + c_3 N \ln^3(NT/\delta)\big)$;
12     if $t \ge t_{\max}$ then return $b$;
13     for $i \in S_\ell$ do
14         $n_i \leftarrow n_i + \Delta_i, T_i \leftarrow T_i + 1$;
15         if $T_i = 2^k$ for some $k \in \mathbb{Z}$ then
16             $\bar{v}_i \leftarrow n_i / T_i$; $\hat{v}_i \leftarrow \min\Big\{\hat{v}_i, \bar{v}_i +$
                 $\sqrt{\frac{196 \bar{v}_i \log(NT/\delta+1)}{T_i}} + \frac{292 \log(NT/\delta+1)}{T_i}\Big\}$;
```

(c) Let $r_{\text{CHECK}}^{(t)}$ be the reward at time step t in this invocation. If $\theta_l \le \theta^\star$, then we have that

$$t_{\max}\theta_l - \mathbb{E}\left[\sum_{t=1}^{t_{\max}} r_{\text{CHECK}}^{(t)}\right] \lesssim \sqrt{N t_{\max} \ln^3(NT/\delta)} + N \ln^3(NT/\delta).$$

Proof of Lemma 12 is built upon Lemma 9 and deferred to Appendix D.3.

Let $\mathcal{Q}_\tau$ be the event that the statements $(a) - (c)$ hold for the invocation of CHECK at iteration τ of Algorithm 5, and let $\mathcal{Q}$ be the event that $\mathcal{Q}_\tau$ holds every all τ. By Lemma 12 and a union bound, we immediately have that $\Pr[\mathcal{Q}] \ge 1 - \delta$. The next lemma, built upon Lemma 9 and Lemma 12, shows that $\hat{\theta}$ in Algorithm 5 is always an upper confidence bound for the true parameter $\theta^\star$, and converges to $\theta^\star$ with a decent rate.

Lemma 13. *Let $\hat{\theta}^{(\tau)}$ be the value of $\hat{\theta}$ at the beginning of iteration τ of Algorithm 5. Conditioned on event $\mathcal{Q}$, for any iteration $\tau = 1, 2, 3, \ldots$, we have that $\hat{\theta}^{(\tau)} - 3\epsilon_\tau \le \theta^\star \le \hat{\theta}^{(\tau)}$.*

Proof. Recall that for every $\tau = 1, 2, 3, \ldots$, we need to prove

$$\hat{\theta}^{(\tau)} - 3\epsilon_\tau \le \theta^\star \le \hat{\theta}^{(\tau)}. \tag{9}$$

We prove this by induction. For iteration $\tau = 1$, (9) trivially holds since $0 \le r_i \le 1$ and therefore $0 \le \theta^\star \le 1$.

Now suppose (9) holds for iteration τ, we will establish (9) for iteration $(\tau + 1)$. Consider the invocation of $\textsc{Check}(\theta_l, \theta_r, t_{\max})$ at iteration τ, where $\theta_l = \hat{\theta}^{(\tau)} - 3\epsilon_\tau$ and $\theta_r = \hat{\theta}^{(\tau)} - \epsilon_\tau$. We discuss the following two cases.

Case 1. When the $\textsc{Check}$ procedure returns true, by Lemma 12 we have that $G(\theta_r) < \theta_r$. By Lemma 9, we have that $\theta_r > \theta^\star$. Therefore, by Line 4 and the induction hypothesis we have that $\hat{\theta}^{(\tau+1)} = \hat{\theta}^{(\tau)} - \epsilon_\tau = \theta_r > \theta^\star$, and $\hat{\theta}^{(\tau+1)} - 3\epsilon_{\tau+1} = \theta_r - 2\epsilon_\tau = \hat{\theta}^{(\tau)} - 3\epsilon_\tau \le \theta^\star$, proving (9).

Case 2. When the $\textsc{Check}$ procedure returns false, by Lemma 12, we have that

$$\theta^\star \ge \theta_r - \frac{1}{t_{\max}} \left((c_2 + 8)\sqrt{N t_{\max} \ln^3 \frac{NT}{\delta}} + c_3 N \ln^3 \frac{NT}{\delta} \right).$$

Recall that at Line 3 we set $t_{\max} = c_1 N \ln^3(NT/\delta)/\epsilon_\tau^2$. For large enough c_1, this implies that

$$\theta^\star \ge \theta_r - \epsilon_\tau = \hat{\theta}^{(\tau)} - 2\epsilon_\tau = \hat{\theta}^{(\tau+1)} - 3\epsilon_{\tau+1}.$$

By Line 4 and the induction hypothesis we have that $\hat{\theta}^{(\tau+1)} = \hat{\theta}^{(\tau)} \ge \theta^\star$, finishing the proof of (9). □

Finally we upper bound the expected regret of Algorithm 5.

Lemma 14. *With probability at least $1 - \delta$, the expected regret incurred by Algorithm 5 is $O(\sqrt{NT}\log^{1.5}(NT/\delta))$. Therefore, if we set $\delta = 1/T$, we have that*

$$\mathbb{E}[\mathrm{Reg}_T] \lesssim \sqrt{NT}\log^{1.5}(NT).$$

Proof. Throughout the proof we condition on the event $\mathcal{Q}$, which happens probability at least $(1 - \delta)$. We first prove that at iteration τ of Algorithm 5, the expected regret for this iteration is bounded by $\tilde{O}(N/\epsilon_\tau)$. Consider the invocation $\textsc{Check}(\theta_l, \theta_r, t_{\max})$ at Line 4. Recall that we define $t_{\max} = c_1 N \ln^3(NT/\delta)/\epsilon_\tau^2$. Combining with statement (c) of Lemma 12 and Lemma 13, the expected regret of this invocation is bounded by (where the $O(N)$ term is due to the last epoch that might run over time $t_{\max}$),

$$\begin{aligned}
&\mathbb{E}\left[\theta^\star \cdot t_{\max} - \sum_{t=1}^{t_{\max}} r^{(t)}_{\textsc{Check}}\right] + O(N) \\
&\lesssim t_{\max}(\theta^\star - \theta_l) + \mathbb{E}\left[\theta_l \cdot t_{\max} - \sum_{t=1}^{t_{\max}} r^{(t)}_{\textsc{Check}}\right] + O(N) \\
&\lesssim t_{\max}(\theta^\star - \theta_l) + N \ln^3(NT/\delta)/\epsilon_\tau. \qquad (10)
\end{aligned}$$

By Lemma 13, we have that $\theta^\star - \theta_l \lesssim \epsilon_\tau$. Therefore, (10) is upper bounded by $O(N \ln^3(NT/\delta)/\epsilon_\tau)$.

Since $\textsc{Check}(\theta_l, \theta_r, t_{\max})$ runs for at least $t_{\max}$ time steps, the second to the last iteration $(\tau_{\max} - 1)$ satisfies that $c_1 N \ln^3(NT/\delta)/\epsilon^2_{\tau_{\max}-1} \le T$, which means that

$$\epsilon_{\tau_{\max}} \gtrsim \sqrt{N \log^3(NT/\delta)/T}.$$

Since ϵ_τ is an exponential sequence, the overall expected regret is bounded by the order of

$$\sum_{\tau=1}^{\tau_{\max}} N \log^3(NT/\delta)/\epsilon_\tau \lesssim \sqrt{NT \log^3(NT/\delta)}.$$

□

Refined and non-trivial item switching cost upper bound for the AT-DUCB algorithm. Since an assortment switch may incur at most $2K$ item switches, Theorem 2 trivially implies that Algorithm 2 (AT-DUCB) incurs at most $O(KN \log T)$ item switches, which is upper bounded by $O(N^2 \log T)$ since $K = O(N)$.

In Appendix C, we present a refined analysis showing that the item switching cost of AT-DUCB is at most $O(N^{1.5} \log T)$. While it is not clear to us whether the dependence on N delivered by this analysis is optimal, we also discuss the relationship between the analysis and an extensively studied (but not yet fully resolved) geometry problem, namely the maximum number of planar K-sets. We hope that further study of this relationship might lead to improvement of both upper and lower bounds of the item switching cost of AT-DUCB. Please refer to Appendix C for more details.

5. Conclusion

In this paper, we present algorithms for MNL-bandits that achieve both almost optimal regret and assortment switching cost, in both anytime and fixed-horizon settings. We also design the ESUCB algorithm that achieves the almost optimal regret and item switching cost $O(N \log^2 T)$. For future directions, it is interesting to study whether it is possible to achieve an item switching cost of $O(N \log T)$ in the anytime setting and $O(N \log\log T)$ in the fixed-horizon setting. Also, as mentioned in Section 4 (and Appendix C), given the simplicity of our AT-DUCB algorithm, it is worthwhile to further refine the bounds for its item switching cost.

Acknowledgement

Part of the work done while Kefan Dong was a visiting student at UIUC. Kefan Dong and Yuan Zhou were supported in part by a Ye Grant and a JPMorgan Chase AI Research Faculty Research Award. Qin Zhang was supported in part by NSF IIS-1633215, CCF-1844234 and CCF-2006591.

References

Agarwal, A., Agarwal, S., Assadi, S., and Khanna, S. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In *COLT*, pp. 39–75, 2017.

Agrawal, S., Avadhanula, V., Goyal, V., and Zeevi, A. A near-optimal exploration-exploitation approach for assortment selection. In *EC*, pp. 599–600, 2016.

Agrawal, S., Avadhanula, V., Goyal, V., and Zeevi, A. Thompson sampling for the MNL-bandit. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, pp. 76–78, 2017.

Agrawal, S., Avadhanula, V., Goyal, V., and Zeevi, A. MNL-bandit: A dynamic learning approach to assortment selection. *Operations Research*, 67(5):1453–1485, 2019.

Bai, Y., Xie, T., Jiang, N., and Wang, Y.-X. Provably efficient q-learning with low switching cost. In *NeurIPS*, 2019.

Cesa-Bianchi, N., Dekel, O., and Shamir, O. Online learning with switching costs and other adaptive adversaries. In *NIPS*, pp. 1160–1168, 2013.

Chen, X. and Wang, Y. A note on a tight lower bound for capacitated MNL-bandit assortment selection models. *Oper. Res. Lett.*, 46(5):534–537, 2018.

Dey, T. K. Improved bounds for planar k-sets and related problems. *Discrete & Computational Geometry*, 19(3): 373–382, 1998.

Esfandiari, H., Karbasi, A., Mehrabian, A., and Mirrokni, V. S. Batched multi-armed bandits with optimal regret. *CoRR*, abs/1910.04959, 2019.

Gao, Z., Han, Y., Ren, Z., and Zhou, Z. Batched multi-armed bandits problem. In *NeurIPS*, 2019.

Jin, Y., Li, Y., Wang, Y., and Zhou, Y. On asymptotically tight tail bounds for sums of geometric and exponential random variables. *arXiv preprint arXiv:1902.02852*, 2019.

Jun, K., Jamieson, K. G., Nowak, R. D., and Zhu, X. Top arm identification in multi-armed bandits with batch arm pulls. In *AISTATS*, pp. 139–148, 2016.

Luce, R. D. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

Perchet, V., Rigollet, P., Chassang, S., and Snowberg, E. Batched bandit problems. In *COLT*, pp. 1456, 2015.

Pinsker, M. S. *Information and information stability of random variables and processes*. Holden-Day, 1964.

Rusmevichientong, P., Shen, Z. M., and Shmoys, D. B. Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations Research*, 58(6):1666–1680, 2010.

Sauré, D. and Zeevi, A. Optimal dynamic assortment planning with demand learning. *Manufacturing & Service Operations Management*, 15(3):387–404, 2013.

Simchi-Levi, D. and Xu, Y. Phase transitions and cyclic phenomena in bandits with switching constraints. In *NeurIPS*, 2019.

Tóth, G. Point sets with many k-sets. *Discrete & Computational Geometry*, 26(2):187–194, 2001.

Train, K. E. *Discrete Choice Methods with Simulation*. Cambridge university press, 2009.