An Optimal Algorithm for Online Non-Convex Learning

LIN YANG, The Chinese University of Hong Kong, Hong Kong
LEI DENG, The Chinese University of Hong Kong, Hong Kong
MOHAMMAD H. HAJIESMAILI, Johns Hopkins University, MD, USA
CHENG TAN, The Chinese University of Hong Kong, Hong Kong
WING SHING WONG, The Chinese University of Hong Kong, Hong Kong

In many online learning paradigms, convexity plays a central role in the derivation and analysis of online learning algorithms. The results, however, fail to be extended to the non-convex settings, while non-convexity is necessitated by a large number of recent applications. The Online Non-Convex Learning (ONCL) problem generalizes the classic Online Convex Optimization (OCO) framework by relaxing the convexity assumption on the cost function (to a Lipschitz continuous function) and the decision set. The state-of-the-art result for the ONCL demonstrates that the classic online exponential weighting algorithm attains a sublinear regret of $O(\sqrt{T \log T})$. The regret lower bound for the OCO, however, is $\Omega(\sqrt{T})$, and to the best of our knowledge, there is no result in the context of the ONCL problem achieving the same bound. This paper proposes the Online Recursive Weighting (ORW) algorithm with regret of $O(\sqrt{T})$, matching the tight regret lower bound for the OCO problem, and fills the regret gap between the state-of-the-art results in the online convex and non-convex optimization problems.

CCS Concepts: • **Theory of computation → Online learning algorithms**; **Regret bounds**; • **Computing methodologies → Machine learning algorithms**;

Additional Key Words and Phrases: Online non-convex learning, online convex optimization, Lipschitz expert, regret, online recursive weighting

ACM Reference Format:
Lin Yang, Lei Deng, Mohammad H. Hajiesmaili, Cheng Tan, and Wing Shing Wong. 2018. An Optimal Algorithm for Online Non-Convex Learning. *Proc. ACM Meas. Anal. Comput. Syst.* 2, 2, Article 25 (June 2018), 25 pages.
https://doi.org/10.1145/3224420

1 INTRODUCTION

1.1 Background and Motivation

The Online Convex Optimization (OCO) framework has widely influenced the online learning community since the seminal work by Zinkevich [51]. The OCO is modeled as a repeated game composed of T iterations. At iteration t, the player chooses a point $\boldsymbol{x}_t$ from a bounded convex decision set $\mathcal{K} \subset \mathbb{R}^n$; after the choice is committed, a bounded convex cost function $f_t : \mathcal{K} \mapsto \mathbb{R}$ is revealed to the player. The goal of the player is to minimize the *regret*, which is defined as the

Authors' addresses: Lin Yang, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, 999077, yl015@ie.cuhk.edu.hk; Lei Deng, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, 999077, ldeng@ie.cuhk.edu.hk; Mohammad H. Hajiesmaili, Johns Hopkins University, 3400 N Charles St. Baltimore, MD, USA, hajiesmaili@jhu.edu; Cheng Tan, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, 999077, tancheng1987love@163.com; Wing Shing Wong, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, 999077, wswong@ie.cuhk.edu.hk.

2476-1249/2018/6-ART25 $15.00
https://doi.org/10.1145/3224420

difference between the online cumulative cost and the cumulative cost using an optimal offline choice in hindsight. This model can be applied to many real-world problems, such as online routing [7, 45], spam email filtering [17, 42], online metric learning [24], ad selection and content ranking in search engines [9, 35, 47], etc.

A promising result related to the OCO model is that the regret of the state-of-the-art efficient algorithms [18] is sublinear, implying that the average cost difference per iteration converges to zero as the number of iterations goes to infinity. It is well known that the lower bound of the regret for the OCO problem is $O(\sqrt{T})$ [18] and researchers have proposed a large number of online algorithms whose regret attains this lower bound, including the Online Gradient Decent method [51], the Stochastic Gradient Decent method [20, 39, 40, 44], the Online Newton Step, and many regularization-related methods [16, 26] (see the recent survey paper [18], and the references therein).

One of the most natural extensions for the OCO problem is to relax the convexity assumption on the decision set $\mathcal{K}$ and the cost function f_t. This extension brings out the Online Non-Convex Learning (the ONCL) problem, which is necessitated by tons of state-of-the-art applications. For example, in the portfolio selection problem [10, 25], the decision maker (e.g., the trader) chooses a distribution of her wealth allocation over n assets $\boldsymbol{x}_t$, at each round. By the end of each round, the adversary chooses the market returns of the assets with positive values. In some specific settings [5, 32, 46], the online portfolio selection problem is a non-convex one due to the non-convex diversification constraints and non-convex transaction costs, and thus the traditional OCO framework fails in modeling such case. In addition, there are extensive machine learning research focusing on non-convex loss functions in large margin classifiers [12, 37, 49]. In [12, 37], non-convex online Support Vector Machine (SVM) models has been studied which adopts a non-convex loss function, called Ramp Loss, to suppress the influence of outliers. In [49], a special non-convex penalty, called the smoothly clipped absolute deviation penalty, is imposed on the hinge loss function in the SVM. Such a new SVM is applied to identify important genes for cancer classification [49].

1.2 Related Results

The ONCL problem is not a new problem and there are plenty prior works on it. Among them, [12] and [15] propose respective heuristic online training algorithms, but neither of them are rigorously shown to satisfy any regret bound. In [19], Hazan and Kale tackle the ONCL problem with submodular cost functions, and propose an online algorithm that attains the regret of $O(\sqrt{T})$. In [50], an online bandit learning problem with non-convex losses is investigated. The cost function is again a special non-convex function, defined as the composition of a non-increasing scalar function with a linear function of small variation. An online algorithm is developed with $\tilde{O}(\text{poly}(d)T^{2/3})$ regret, where $\text{poly}(d)$ stands for a polynomial with respect to the dimension of the decision set d. The most related works to ours are [31] and [36], where by applying the exponential weighting method [6, 14] the regret of $O(\sqrt{T \log T})$ is attained.

In addition, the ONCL problem has been broadly investigated under a similar problem, called the Lipschitz Expert problem [27, 29], which generalizes the traditional Expert problem [14] to metric spaces. For such a problem, a regret of $O(\sqrt{T \log T})$ can be achieved [29]. However, to the best of our knowledge, for the general ONCL problem or the Lipschitz Expert problem, no online algorithm can achieve the regret of $O(\sqrt{T})$ as the well-known lower bound for the OCO problem.

1.3 Our Contributions and Adopted Techniques

This paper tackles the ONCL problem, with non-convex L-Lipschitz cost functions and non-convex decision set. We propose a novel online algorithm, called the Online Recursive Weighting (the

ORW) algorithm, and prove that the regret of the ORW is upper bounded by $O(\sqrt{T})$. The obtained regret bound matches the well-known lower bound of the regret for the OCO [18], hence, the ORW algorithm is asymptotically optimal for the general ONCL problem.

The general idea of the ORW algorithm is to divide the decision set into multiple subsets according to a *grid layered* structure. Any subset at the upper layer is divided into multiple smaller subsets at the lower layers. Our algorithm recursively selects the subset from the topmost layer to the bottommost layer until a decision point is identified. At each layer, the ORW algorithm leverages the classic Exponential Weighting or Hedge algorithm [6, 14] to select a subset at the lower layer. As the core technical contribution, the ORW properly partitions the subsets and sets the subset-selecting probabilities, thereby, it asymptotically achieves $O(\sqrt{T})$ regret.

Then, the ORW algorithm is extended to an adaptive version (the AORW algorithm), which increases the granularity parameter gradually as time goes on. The AORW guarantees $O(\sqrt{T})$ regret, reduces the computational complexity of the ORW, and works properly when the duration of time horizon is unknown to the online player.

1.4 Basic Notations and Organizations

In the rest of this paper, we use calligraphy font to denote sets, e.g., $\mathcal{K}$. In equations, we use Fraktur font to denote algorithms, e.g., $\mathfrak{R}$. We use bold math font to denote vectors whose entries are represented by the corresponding normal math font, e.g., $\boldsymbol{i} = (i_1, i_2, \cdots, i_n)$. We further gives the following definition for L-Lipschitz functions.

Definition 1. *A function $f : \mathcal{K} \to \mathbb{R}$ (where $\mathcal{K} \subset \mathbb{R}^n$) is L-Lipschitz if*

$$|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{K}. \tag{1}$$

The rest of the paper is organized as follows. Section 2 introduces details on the problem formulation as well as the notation used in this paper. In Section 3, we introduce the proposed ORW algorithm and its regret is analyzed in Section 4. In Sections 5, we extend our algorithm to an adaptive version called AORW. Section 6 reviews the related literature and potential extensions. Finally, Section 7 concludes the paper.

2 PROBLEM SETTING OF ONLINE NON-CONVEX LEARNING

We describe our online non-convex learning problem as a structured repeated game. At each iteration t, the player chooses a decision $\boldsymbol{x}_t \in \mathcal{K}$, where $\mathcal{K} \subseteq \mathbb{R}^n$ is a bounded decision set whose diameter is D, i.e., $\sup_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{K}} ||\boldsymbol{x} - \boldsymbol{y}||_2 = D$. After the player commits to a decision point at iteration t, the adversary chooses a cost function $f_t(\boldsymbol{x}) \in \mathcal{F}$, where $\mathcal{F} : \mathcal{K} \mapsto \mathbb{R}^+$ is a set of cost functions which are assumed to be non-negative and L-Lipschitz (as defined in Equation (1)).

Note that in the original OCO model, the cost function and decision set are assumed to be convex. In this work, we take into account the general class of cost functions and decision sets which are not necessarily convex.

In our model, we consider the full feedback case, in which the cost function $f_t(\boldsymbol{x})$ is revealed to the player only after a choice is made at iteration t. At each iteration, the player needs to make online decisions without knowing the current and future cost functions. Fed with the historical cost functions $\boldsymbol{H}_t = (f_1, f_2, \ldots, f_{t-1})$, the decision of an online algorithm $\mathfrak{A}$ at iteration t is denoted as $\boldsymbol{x}_t = \mathfrak{A}(\boldsymbol{H}_t)$. The summary of main notations related to problem setting is given in Table 1.

Table 1. Summary of notations related to problem setting

Notation	Description
t	Index of iteration
T	The number of iterations, $T \geq 1$
$\mathcal{T}$	Set $\mathcal{T} = \{1, 2, \ldots, T\}$
n	The dimension of the decision space
$\boldsymbol{x}_t$	Chosen decision point at t
$\mathcal{K}$	The decision set available for the online player
D	Diameter of the decision set $\mathcal{K}$
$f_t(\boldsymbol{x})$	$f_t(\boldsymbol{x}) : \mathcal{K} \mapsto \mathbb{R}^+$. Cost function at iteration t, known for $t-1$, unknown for $\tau \geq t$
L	Lipschitz constant for the cost functions
$\mathcal{F}$	The set of cost functions which are available for adversary
$\boldsymbol{H}_t$	$\boldsymbol{H}_t = (f_1, f_2, \ldots, f_{t-1})$. The historical cost functions available for the online algorithm at iteration t
$\mathfrak{A}$	An online algorithm which maps from the historical cost functions $\boldsymbol{H}_t$ to a decision point $\boldsymbol{x}_t$ for all $t = 1, 2, \ldots$

We use the pseudo-regret[1] as a common performance metric to evaluate our proposed online algorithm for the ONCL problem, formally defined as

$$\text{regret}_T(\mathfrak{A}) \stackrel{\text{def}}{=} \sup_{f_1,\ldots,f_T \in \mathcal{F}} \left\{ \sum_{t=1}^{T} \mathbb{E}[f_t(\boldsymbol{x}_t)] - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} f_t(\boldsymbol{x}) \right\}, \tag{2}$$

which is the cumulative difference between the expected cost of the online algorithm and the cost of the best fixed offline decision up to T. The expectation in (2) is taken over the randomness of the decision at each iteration t if the online algorithm $\mathfrak{A}$ is randomized.

Our goal is to design an online algorithm for the ONCL problem and try to minimize the regret. Ideally, it is desired to have a sublinear regret with respect to T, i.e., $\text{regret}_T(\mathfrak{A}) = o(T)$. The sublinear regret implies that time-average performance of the online algorithm is as good as the best fixed strategy as time goes to infinity.

In Section 3, we propose an online algorithm for the ONCL problem, called the Online Recursive Weighting (ORW). In Section 4, we analyze the performance of the ORW and demonstrate that its regret is $O(\sqrt{T})$ asymptotically, which attains the lower bound of the regret for the basic OCO problem [18].

3 ONLINE RECURSIVE WEIGHTING ALGORITHM

The Online Recursive Weighting (ORW) algorithm is based on the idea of dividing the entire decision set into several *sampling subsets* so that each subset contains highly correlated elements. An essential tuning parameter for the algorithm is the granularity of the sampling subsets, which determines the number of final sample points (details in Sections. 3.1 and 3.2).

To define the granularity of the sampling subsets and the decision policy of the ORW algorithm, we construct a *layered grid structure* . The topmost layer consists of a single grid and hence a single subset including the entire decision set $\mathcal{K}$. The sampling subsets are the subsets at the bottommost layer. In Section 3.1, we explain the details of constructing the layered grid structure.

[1]We call it regret in short in the rest of this paper.

Table 2. Summary of key notations related to the ORW algorithm

Notation	Description
$\mathcal{D}$	Cover cube of the decision set $\mathcal{K}$
l	Index of layer
$\boldsymbol{i}_l$	Index for a sub-cube/subset at l-th layer, $\boldsymbol{i}_l = (i_{l,1}, i_{l,2}, \cdots, i_{l,n})$
$\mathcal{D}_l(\boldsymbol{i}_l)$	Layer l sub-cube indexed by $\boldsymbol{i}_l$
$\mathcal{K}_l(\boldsymbol{i}_l)$	Layer l subset indexed by $\boldsymbol{i}_l$
$\mathcal{I}_l$	Index set of nonempty subsets at layer l
$\boldsymbol{v}_s(\boldsymbol{i}_l)$	Index of layer s subset that contains $\mathcal{K}_l(\boldsymbol{i}_l)$
$\mathcal{E}_s(\mathcal{K}_l(\boldsymbol{i}_l))$	Index set of nonempty layer s subsets within the subset $\mathcal{K}_l(\boldsymbol{i}_l)$, $\boldsymbol{i}_l \in \mathcal{I}_l$
m	The index of the layer where the subsets are sampled
$\boldsymbol{r}_m(\boldsymbol{i}_m)$	The sample point for the subset $\mathcal{K}_m(\boldsymbol{i}_m)$
$c_{m,t}(\boldsymbol{i}_m)$	The cost on the sample point of the subset $\mathcal{K}_m(\boldsymbol{i}_m)$ at iteration t, i.e., $c_{m,t}(\boldsymbol{i}_m) \stackrel{\text{def}}{=} f_t(\boldsymbol{r}_m(\boldsymbol{i}_m))$
L_m	The maximum cost difference between two points in the same subset at layer m
$\boldsymbol{I}_{l,t}$	The index of the subset chosen at layer l at iteration t
$\bar{c}_{l,t}(\boldsymbol{i}_l)$	The *expected normalized cost* conditioning on that subset $\mathcal{K}_l(\boldsymbol{i}_l)$ is chosen at layer l at iteration t. For short, we also call it the expected cost of choosing $\mathcal{K}_l(\boldsymbol{i}_l)$ at layer l at iteration t
$\bar{C}_{l,t}(\boldsymbol{i}_l)$	The cumulative sum for $\bar{c}_{l,\tau}(\boldsymbol{i}_l)$ up to iteration t, i.e., $\bar{C}_{l,t}(\boldsymbol{i}_l) = \sum_{\tau=1}^{t} \bar{c}_{l,\tau}(\boldsymbol{i}_l)$. For short, we also call it the cumulative expected cost of choosing $\mathcal{K}_l(\boldsymbol{i}_l)$ at layer l up to iteration t
$\Re$	The notation used for the ORW algorithm in the equations (Algorithm 1)

In the next step (Section 3.2), a single element is randomly selected from each sampling subset called *sample point*, and by grouping all these points, we construct the set of sample points. At each iteration, the algorithm selects a *decision point* randomly from the set of sample points according to a tree decision structure. Beginning with the topmost layer, the ORW algorithm probabilistically selects a subset by running Hedge update[2], and subsequently, the ORW algorithm continues among the subsets of one layer below that lies inside the selected subset of the previous layer.

Designing the tree decision structure to select among sample points is the core technical contribution of the ORW algorithm and is proposed in Section 3.3. The summary of key notations related to the algorithm design is given in Table 2.

3.1 A Layered Grid Structure

Recall that the diameter of bounded decision set $\mathcal{K}$ is D, hence, it is possible to find a bounded cube of length D, denoted by $\mathcal{D}$, that can cover $\mathcal{K}$ entirely, i.e., $\mathcal{K} \subset \mathcal{D}$. At layer $l \in \mathbb{N}^+$, we partition $\mathcal{D}$ into smaller identical sub-cubes with edge length of size $D/2^l$. The decision set is n-dimensional, hence the total number of sub-cubes is equal to 2^{nl}. For simplicity, each sub-cube is indexed by a distinct n-dimensional vector $\boldsymbol{i}_l = (i_{l,1}, i_{l,2}, \ldots, i_{l,n})$, where $1 \le i_{l,j} \le 2^l, j = 1, 2, \ldots, n$, and the sub-cube indexed by $\boldsymbol{i}_l$ is denoted by $\mathcal{D}_l(\boldsymbol{i}_l)$. One can refer to Fig. 1 for an illustrative example with $n = 2$ and $l = 1$.

[2]Hedge algorithm maintains weights for each expert/decision point, which are updated according to the observed losses at each step. For details, readers can refer to [14].

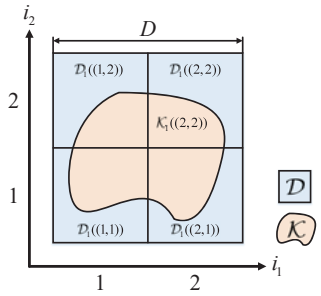

Fig. 1. Layer 1 grid construction for a general decision set $\mathcal{K}$.

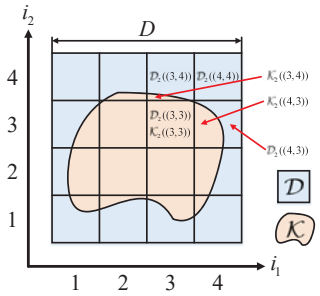

Fig. 2. Index set for overlapped sub-cubes when $n = 2$ and $l = 2$.

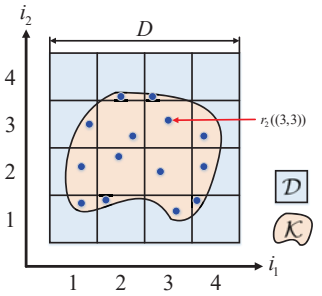

Fig. 3. Sampling for the case when $n = 2$ and $m = 2$.

The notation $\mathcal{K}_l(\boldsymbol{i}_l)$ is used to represent the intersection of the decision set $\mathcal{K}$ and corresponding sub-cube $\mathcal{D}_l(\boldsymbol{i}_l)$, i.e.,

$$\mathcal{K}_l(\boldsymbol{i}_l) \overset{\text{def}}{=} \mathcal{D}_l(\boldsymbol{i}_l) \cap \mathcal{K}.$$

By convention, we regard the decision set $\mathcal{K}$ as the only layer 0 subset, denoted by $\mathcal{K}_0(\mathbf{1}) = \mathcal{K}$. By the above partitioning structure, in total 2^{nl} subsets at layer l are constructed, whose union is the decision set $\mathcal{K}$. Note that there could be some empty subsets, and we denote the index set for nonempty layer l subsets by $\mathcal{I}_l$, i.e.,

$$\mathcal{I}_l \overset{\text{def}}{=} \{\boldsymbol{i}_l : \mathcal{K}_l(\boldsymbol{i}_l) \neq \emptyset\}.$$

Any subset indexed by $\boldsymbol{i}_l \in \mathcal{I}_l$, i.e., $\mathcal{K}_l(\boldsymbol{i}_l)$, consists of a group of neighboring lower-layer subsets. Specifically, for any $s > l$, we have

$$\mathcal{K}_l(\boldsymbol{i}_l) = \bigcup_{\substack{\boldsymbol{i}_s : \ i_{s,j} \geq 1 + (i_{l,j} - 1)2^{s-l}, \\ i_{s,j} \leq i_{l,j} 2^{s-l}, \\ j = 1, 2, \ldots, n.}} \mathcal{K}_s(\boldsymbol{i}_s).$$

To ease the presentation, we define the following notation:

▷ We use $\mathcal{E}_s\left(\mathcal{K}_l(\boldsymbol{i}_l)\right)$, $s \geq l$, to denote the index set of nonempty layer s subsets within $\mathcal{K}_l(\boldsymbol{i}_l)$, i.e.,

$$\mathcal{E}_s\left(\mathcal{K}_l(\boldsymbol{i}_l)\right) \overset{\text{def}}{=} \left\{\boldsymbol{i}_s \in \mathcal{I}_s : 1 + (i_{l,j} - 1)2^{s-l} \leq i_{s,j} \leq i_{l,j} 2^{s-l}, j = 1, 2, \ldots, n\right\}. \tag{3}$$

Indeed, we have $\left|\mathcal{E}_s\left(\mathcal{K}_l(\boldsymbol{i}_l)\right)\right| \leq 2^{(s-l)n}$.

▷ Assume $\boldsymbol{i}_l \in \mathcal{I}_l$ and $s \leq l$. We use $\boldsymbol{v}_s(\boldsymbol{i}_l)$ to denote an index for some layer s subset, and the elements of $\boldsymbol{v}_s(\boldsymbol{i}_l)$ satisfy

$$v_{s,j} = \left\lceil \frac{i_{l,j}}{2^{l-s}} \right\rceil, \ j = 1, 2, \ldots, n.$$

Due to the above definition of $\boldsymbol{v}_s(\boldsymbol{i}_l)$, $\boldsymbol{i}_l \in \mathcal{I}_l$, together with the grid construction method we adopt, it follows that $\mathcal{K}_l(\boldsymbol{i}_l) \subset \mathcal{K}_s(\boldsymbol{v}_s(\boldsymbol{i}_l))$.

Example 1. Fig. 1 illustrates a simple example of grid construction when $n = 2$ and $l = 1$. At layer 1, the cover cube $\mathcal{D}$ is partitioned into four smaller sub-cubes, each of whom intersects the decision set $\mathcal{K}$, forming four nonempty subsets, respectively.

$$\mathcal{I}_1 = \{(1,1), (1,2), (2,1), (2,2)\}.$$

In Fig. 2, the subsets at layer 1 are further divided and 14 fixed nonempty layer 2 subsets are constructed. Particularly, the subset $\mathcal{K}_1((2,2))$ contains three nonempty layer 2 subsets, i.e., $\mathcal{K}_2((3,3))$, $\mathcal{K}_2((3,4))$, and $\mathcal{K}_2((4,3))$. Thus, we have

$$\mathcal{E}_2\left(\mathcal{K}_1((2,2))\right) = \{(3,3),(3,4),(4,3)\}\,.$$

According to the definition of $\boldsymbol{v}_s(\boldsymbol{i}_l)$, we have

$$\boldsymbol{v}_1((3,3)) = \boldsymbol{v}_1((3,4)) = \boldsymbol{v}_1((4,3)) = (2,2),$$

and $\mathcal{K}_2((3,3)) \subset \mathcal{K}_1((2,2))$, $\mathcal{K}_2((3,4)) \subset \mathcal{K}_1((2,2))$, and $\mathcal{K}_2((4,3)) \subset \mathcal{K}_1((2,2))$.

3.2 Sampling

Assuming layer 0 contains the original decision set $\mathcal{K}$, we fix the number of layers to be $m+1$, where layer m contains at most 2^{nm} nonempty subsets. In the sampling procedure, a *sample point*, $\boldsymbol{r}_m(\boldsymbol{i}_m)$, is selected randomly as a representative, from each nonempty subset $\mathcal{K}_m(\boldsymbol{i}_m)$ at layer m. Fig. 3 depicts a simple example for $n=2$ and $m=2$, where the sample points are colored in blue. Due to the bijective correspondence between the sample points and the subsets at layer m, we interchangeably use choosing a sample point from a subset and choosing the corresponding layer m subset.

In Section 3.3, we define a recursive, probabilistic algorithm to select subsets as we go down the layered structure. Correspondingly, this defines a probabilistic approach to choose a *decision point* from the sample points as the final decision of the ORW algorithm.

Once the decision point is selected at each iteration, the cost of the decision, $f_t(\boldsymbol{r}_m(\boldsymbol{i}_m))$, along with the function f_t will be revealed.

For notation convenience, we denote the cost of the sample point from the layer m subset, $\mathcal{K}_m(\boldsymbol{i}_m)$, $\boldsymbol{i}_m \in \mathcal{I}_m$, at iteration t by $c_{m,t}(\boldsymbol{i}_m)$, i.e.,

$$c_{m,t}(\boldsymbol{i}_m) \stackrel{\text{def}}{=} f_t(\boldsymbol{r}_m(\boldsymbol{i}_m)).$$

Note that by this sample point construction step, the proposed algorithm reduces the original problem in principle to an Expert problem with $|\mathcal{I}_m|$ experts. As compared to the classic setting of the Expert problem in [8], the difference is that the cost function in our problem is a non-convex L-Lipschitz continuous one. Finally, the following lemma characterizes the cost difference between two sample points.

Lemma 3.1. *Assume $\boldsymbol{p},\boldsymbol{q} \in \mathcal{I}_m$, we have*

$$|c_{m,t}(\boldsymbol{p}) - c_{m,t}(\boldsymbol{q})| \le 2L_m||\boldsymbol{p}-\boldsymbol{q}||_1,$$

where $L_m = \sqrt{n}DL/(2^m)$ and $||\boldsymbol{p}-\boldsymbol{q}||_1 = \sum_{j=1}^{n}|p_j - q_j|$.

Proof. The maximum distance between two points within a subset at layer m is $\sqrt{n}D/(2^m)$. Considering the Lipschitz continuous condition, the maximum cost difference within the same subset at layer m is $L_m = \sqrt{n}DL/(2^m)$. We require that, any two subsets of the same layer, $\boldsymbol{p}$ and $\boldsymbol{q}$, are said to be neighboring subsets if their coordinates satisfy $p_j - q_j \le 1$ for $j = 1,2,\ldots,n$. Then, the maximum cost difference of two points in the union of any two neighboring layer m subsets is $2\sqrt{n}DL/(2^m)$. Thus, the maximum distance for any two points in the union of two layer m subsets, $\boldsymbol{p}$ and $\boldsymbol{q}$, is $2\sqrt{n}DL/(2^m)||\boldsymbol{p}-\boldsymbol{q}||_1$. This yields the result in Lemma 3.1. □

3.3 Recursive Choosing Policy

In this subsection, we propose a novel recursive decision structure to determine the index point for the bottom-layer subset over the index set $\mathcal{I}_m$.

At iteration t, $t = 1, 2, \ldots, T$, the final choice of the ORW algorithm is obtained by recursively choosing a nonempty subset from the topmost layer ($l = 0$) to the bottommost layer ($l = m$). In this way, the action of the ORW algorithm at each iteration t, consists of a sequence of indexes of subsets, i.e., $(\boldsymbol{I}_{0,t}, \boldsymbol{I}_{1,t}, \boldsymbol{I}_{2,t}, \ldots, \boldsymbol{I}_{m,t})$. By default, $\boldsymbol{I}_{0,t}$ is set to be $\mathbf{1}$, referring to the only decision set $\mathcal{K}$. Additionally, $\boldsymbol{I}_{l,t}$ should satisfy that $\boldsymbol{I}_{l,t} \in \mathcal{E}_l(\mathcal{K}_{l-1}(\boldsymbol{I}_{l-1,t}))$, i.e., $\mathcal{K}_l(\boldsymbol{I}_{l,t}) \subset \mathcal{K}_{l-1}(\boldsymbol{I}_{l-1,t})$ for $l = 1, 2, \ldots, m$.

At each layer, a stochastic policy is adopted to select a lower-layer subset. Suppose that, at iteration t and at layer l, $0 \le l < m$, the ORW algorithm has chosen subset $\boldsymbol{i}_l \in \mathcal{I}_l$, i.e., $\boldsymbol{I}_{l,t} = \boldsymbol{i}_l$. Then, the algorithm proceeds to choose a subset at the next layer (layer $(l+1)$) in $\mathcal{K}_l(\boldsymbol{i}_l)$ randomly according to a conditional probability.

In the ORW algorithm, and for $\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))$, the conditional probability, $\Pr^t_{l,l+1} = \Pr\left[\boldsymbol{I}_{l+1,t} = \boldsymbol{i}_{l+1} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right]$, is based on the cumulative expected cost of the *previous* iterations.

For iteration τ, $0 \le \tau \le t-1$, the *expected normalized cost* of subset $\mathcal{K}_l(\boldsymbol{i}_l)$, denoted by $\bar{c}_{l+1,\tau}(\boldsymbol{i}_{l+1})$, is defined as follows

$$\begin{aligned}\bar{c}_{l,\tau}(\boldsymbol{i}_l) &\overset{\text{def}}{=} \mathbb{E}\left[\frac{c_{m,\tau}(\boldsymbol{I}_{m,\tau}) - \min\limits_{\boldsymbol{i}_m \in Q} c_{m,\tau}(\boldsymbol{i}_m)}{2^{m-l+1}L_m} | \boldsymbol{I}_{l,\tau} = \boldsymbol{i}_l\right] \\ &= \sum_{\boldsymbol{j}_m \in \mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{c_{m,\tau}(\boldsymbol{j}_m) - \min\limits_{\boldsymbol{i}_m \in Q} c_{m,\tau}(\boldsymbol{i}_m)}{2^{m-l+1}L_m} \cdot \Pr\left[\boldsymbol{I}_{m,\tau} = \boldsymbol{j}_m | \boldsymbol{I}_{l,\tau} = \boldsymbol{i}_l\right],\end{aligned} \tag{4}$$

where $Q = \mathcal{E}_m(\mathcal{K}_{l-1}(\boldsymbol{v}_{l-1}(\boldsymbol{i}_l)))$. Note that $\bar{c}_{l+1,\tau}(\boldsymbol{i}_{l+1})$ can be computed at the end of iteration τ, once the cost function is revealed. In Equation (4), the conditional selection probability for subset $\boldsymbol{j}_m \in \mathcal{I}_m$, i.e., $\Pr\left[\boldsymbol{I}_{m,\tau} = \boldsymbol{j}_m | \boldsymbol{I}_{l,\tau} = \boldsymbol{i}_l\right]$ can be obtained by multiplying the selection probabilities of the corresponding subsets at each layer, i.e.,

$$\Pr\left[\boldsymbol{I}_{m,\tau} = \boldsymbol{j}_m | \boldsymbol{I}_{l,\tau} = \boldsymbol{i}_l\right] = \prod_{k=l}^{m-1} \Pr\left[\boldsymbol{I}_{k+1,\tau} = \boldsymbol{v}_{k+1}(\mathcal{K}_m(\boldsymbol{j}_m)) | \boldsymbol{I}_{k,\tau} = \boldsymbol{v}_k(\mathcal{K}_m(\boldsymbol{j}_m))\right]. \tag{5}$$

For a subset $\mathcal{K}_l(\boldsymbol{i}_l)$, the *cumulative expected cost*, denoted by $\bar{C}_{l,t}(\boldsymbol{i}_l)$, is defined as the sum of the *expected normalized cost* from iteration 1 to iteration t, i.e.,

$$\bar{C}_{l,t}(\boldsymbol{i}_l) \overset{\text{def}}{=} \sum_{\tau=1}^{t} \bar{c}_{l,\tau}(\boldsymbol{i}_l). \tag{6}$$

By convention, let the initial value of the cumulative expected cost be zero, i.e., $\bar{C}_{l,0}(\boldsymbol{i}_l) = 0$ for all $\boldsymbol{i}_l \in \mathcal{I}_l$ and for any layer $l = 1, 2, \cdots, m$.

At iteration t, the selection probability, $\Pr\left[\boldsymbol{I}_{l+1,t} = \boldsymbol{i}_{l+1} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right]$, is proportional to the exponent of the expected cumulative cost of choosing subset $\boldsymbol{i}_{l+1}$ at layer $(l+1)$ up to iteration $t-1$, i.e.,

$$\Pr\left[\boldsymbol{I}_{l+1,t} = \boldsymbol{i}_{l+1} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right] = \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)}{\sum\limits_{\boldsymbol{i}'_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}, \tag{7}$$

where η_t is a positive and decreasing parameter which implies that the changing on $\bar{C}(.)$ has decreasing influence on the choosing probability and thus the decision of the online algorithm gets

Algorithm 1 The Online Recursive Weighting (ORW) Algorithm

Input: index set $\mathcal{I}_m$, T, $\{\eta_t = \sqrt{\frac{n}{t}}\}$
Output: $\boldsymbol{r}_m(\boldsymbol{I}_{m,1}), \boldsymbol{r}_m(\boldsymbol{I}_{m,2}), \ldots$

```
//↑ Output sample points of layer m subsets
//↓ Initialize the cumulative expected cost for all subsets at all layers
```
Set $\bar{C}_{l,0}(\boldsymbol{i}_l) = 0, \forall \boldsymbol{i}_l \in \mathcal{I}_l, l = 1, 2, \ldots, m$
for $t = 1, 2, \ldots, T$ **do**
  $\boldsymbol{I}_{0,t} = \boldsymbol{1}$
  `//↓ Recursively select subsets at all layers`
  for $l = 0, 1, \ldots, m-1$ **do**
    Randomly select an index $\boldsymbol{I}_{l+1,t} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{I}_{l,t}))$ according to the probability distribution specified in Equation (7)
  end for
  Choose the subset indexed by $\boldsymbol{I}_{m,t}$ at iteration t
  Choose the sample point for subset $\mathcal{K}_m\left(\boldsymbol{I}_{m,t}\right)$, i.e., $\boldsymbol{r}_m(\boldsymbol{I}_{m,t})$, as the final decision
  `//↓ Get the normalized expected cost for all subsets at all layers based on the revealed cost function` $f_t(\boldsymbol{x})$ `at iteration` t
  for $l = 1, 2, \ldots, m$ **do**
    for $\boldsymbol{i}_l \in \mathcal{I}_l$ **do**
      for $\boldsymbol{j}_m \in \mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))$ **do**
        Calculate $\Pr\left[\boldsymbol{I}_{m,t} = \boldsymbol{j}_m | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right]$ according to Equation (5)
      end for
      Calculate $\bar{c}_{l,t}(\boldsymbol{i}_l)$ according to Equation (4)
      Update $\bar{C}_{l,t}(\boldsymbol{i}_l) = \bar{C}_{l,t-1}(\boldsymbol{i}_l) + \bar{c}_{l,t}(\boldsymbol{i}_l)$
    end for
  end for
end for

more unalterable as time goes on. Note that the denominator in Equation (7) is a normalizer such that right hand side of Equation (7) is a probability mass function.

Note that the ORW algorithm updates the cumulative expected cost at the end of each iteration. Once the conditional probabilities are defined and the cost function at iteration t is revealed, we then calculate the expected normalized cost and further update the cumulative expected cost for each subset, which is

$$\bar{C}_{l,t}(\boldsymbol{i}_l) = \bar{C}_{l,t-1}(\boldsymbol{i}_l) + \bar{c}_{l,t}(\boldsymbol{i}_l). \tag{8}$$

Then, the updated cumulative expected cost will be used to calculate the choosing probability at the next iteration as shown in Equation (7). The summary of the proposed ORW algorithm is listed as Algorithm 1.

Remark 1. (Technical differences with the traditional weighting methods) For the expert learning problem, the intuitive idea to attain a sublinear regret is to allocate more preference to the expert of smaller cumulative cost in a stochastic manner [34], which can be realized by the Exponential Weighting algorithm (or Hedge algorithm). The Hedge algorithm observes the costs on each point, and updates the weight of a point based on its own cost only. Such a point-by-point weighting method fails in utilizing the cost correlation among neighboring decision nodes and attains only sub-optimal performance within continuum decision space which could consist of infinitely many decision points. Than the Hedge algorithm, the ORW algorithm makes better use of the correlation of neighboring points by grouping highly correlated decision points as a high-level decision. As

expressed in Equation (4), a common evaluation is conducted for each decision group, and a particular amount of common preference will be allocated to the group based on its cumulative performance. Furthermore, by constructing a tree decision structure, the ORW algorithm chooses among at most 2^n "experts" or decision points at each layer, avoiding the performance degradation due to the large number of decision points that the Hedge algorithm might have been faced with.

4 REGRET ANALYSIS FOR THE ONLINE RECURSIVE WEIGHTING ALGORITHM

Since the ONCL problem is a more general model than the OCO, the lower bound for the OCO is still valid for the non-convex case. In the OCO framework, a well-known lower bound of the regret is $\Omega(\sqrt{T})$, and the reader can refer to [18] for the sketch of the proof. In this section, we analyze the regret of the ORW algorithm (represented by $\Re$ in equations) and demonstrate it matches the above lower bound. The main technical results are summarized in the following theorem.

THEOREM 4.1. *With $\eta_t = \sqrt{\frac{n}{t}}$, the* ORW *algorithm guarantees that*

$$\text{regret}_T(\Re) \le 2(\ln 2 + 1)nDL\sqrt{T} + 2\ln 2 \cdot nDL + \frac{\sqrt{n}}{2^m}DLT.$$

If m is further set to $\lceil \log_2 \sqrt{nT} \rceil$, it follows that

$$\text{regret}_T(\Re) < (4n + 1)DL\sqrt{T} + 2nDL.$$

Remark 2. (Dimensional dependency) Theorem 4.1 implies that the ORW algorithm attains a regret of $O(n\sqrt{T})$, with a mild polynomial dependency on the dimension. Note that the regret of the Hedge algorithm on a continuum [31] has dimension dependency and is shown to be $O(\sqrt{nT\log T}$ (see Corollary 2 in [31]), which has a milder dependency on the dimension as compared to our algorithm. This milder dependency is due to an additional assumption on the cost function in [31] to be always bounded by a constant no matter how large the dimension is. In our setting, we do not have such assumption on cost functions, and the regret is $O(n\sqrt{T})$.

Remark 3. [31] shows that when the decision set is uniformly fat, the Hedge algorithm can attain a regret of $O(\sqrt{T\log T})$. An additional advantage of our ORW algorithm is that the only assumption on decision set is that it is bounded.

To carry out the analysis, we split the regret analysis of the ORW algorithm into two parts:

▷ The first part is the regret due to the "imperfect choice" among sample points, i.e.,

$$\text{regret}_{\text{ImC},T}(\Re) \stackrel{\text{def}}{=} \sup_{f_1,\ldots,f_T\in\mathcal{F}} \left\{ \sum_{t\in\mathcal{T}} \mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})\right] - \min_{\boldsymbol{i}_m\in\mathcal{I}_m} \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m) \right\},$$

where the first term is the cumulative cost incurred by the online algorithm (whose choice at iteration t is denoted by $\boldsymbol{I}_{m,t}$), and the second term is the minimum cumulative cost among sampled points.

▷ The second part of the regret is introduced by "imperfect discretization", which is expressed as

$$\text{regret}_{\text{ImD},T}(\Re) \stackrel{\text{def}}{=} \sup_{f_1,\ldots,f_T\in\mathcal{F}} \left\{ \min_{\boldsymbol{i}_m\in\mathcal{I}_m} \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m) - \min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}) \right\},$$

where the second term is the minimum cumulative cost over the decision set $\mathcal{K}$. Since the supreme of sum is less than or equal to the sum of supreme, we have the following lemma.

LEMMA 4.2. $\text{regret}_T(\Re) \le \text{regret}_{\text{ImC},T}(\Re) + \text{regret}_{\text{ImD},T}(\Re)$.

In the following lemma, we derive a bound for the first-part regret of the ORW algorithm, which is related to the subproblem of choosing a point over the set $\mathcal{I}_m$. The regret analysis for the imperfect discretization is given in Lemma 4.4.

Lemma 4.3. *With $\eta_t = \sqrt{\frac{n}{t}}$, the* ORW *algorithm guarantees that*

$$\text{regret}_{\text{ImC},T}(\Re) \leq 2(\ln 2 + 1)nDL\sqrt{T} + 2\ln 2 \cdot nDL.$$

Proof. The ORW algorithm has a recursive decision-making structure to determine the final decision. Correspondingly, the regret due to "imperfect choice" can be further splited into multiple pieces which are introduced at each layer.

Suppose a nonempty subset $\mathcal{K}_l(\boldsymbol{i}_l)$ is chosen at layer $l \in \{0, 1, 2, \ldots, m-1\}$. In the next step, the ORW algorithm will further choose a subset whose index lies in $\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))$. Among the subsets of $\mathcal{K}_l(\boldsymbol{i}_l)$, there exists a local optimal subset (for example, $\boldsymbol{i}''_{l+1}$), whose performance is known in hindsight and potentially a regret loss due to imperfect choice at layer l will be incurred by the online algorithm. Equation (9) bounds such cost difference (or regret loss) between $\sum_{t\in\mathcal{T}} \mathbb{E}[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l,t} = \boldsymbol{i}_l]$ and $\sum_{t\in\mathcal{T}} \mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l+1,t} = \boldsymbol{i}''_{l+1}\right]$ at the l-th layer, where $\boldsymbol{i}''_{l+1}$ is any element in $\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))$.

In Equation (9), equality $(E1)$ is obtained simply by the law of total probability. Equality $(E2)$ is due to the definition of $\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})$ in (4) . Equality $(E3)$ is based on the conditional probability in (7). Inequality $(E4)$ is due to the following inequality,

$$\begin{aligned}
&\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)\cdot \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)} \\
\leq &-\frac{1}{\eta_t}\ln \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)\cdot \exp(-\eta_t \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1}))}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)} + \frac{\eta_t}{2}. \qquad (10)
\end{aligned}$$

The proof of Equation (10) is given in Appendix A.1. Equality $(E5)$ is due to the fact that $\bar{C}_{l+1,t}(\boldsymbol{i}_{l+1}) = \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1}) + \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})$. Equality $(E6)$ simplifies the expression for the log-sum-exp function by defining

$$\Phi_t(\alpha) \stackrel{\text{def}}{=} -\frac{1}{\alpha}\ln \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\alpha \bar{C}_{l+1,t}(\boldsymbol{i}_{l+1})\right),$$

where $\alpha > 0$. Equality $(E7)$ rearranges the first set of terms and Inequality $(E8)$ uses the following bound result

$$\sum_{t=1}^{T-1}[\Phi_t(\eta_t) - \Phi_t(\eta_{t+1})] \leq \sum_{t=1}^{T-1} n\left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) \leq \ln 2 \cdot \sqrt{nT}, \qquad (11)$$

which is proved in Appendix A.2. Inequality $(E9)$ entails the following two results:

$$-\Phi_0(\eta_1) = \frac{1}{\eta_1}\ln|\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))| = \frac{1}{\sqrt{n}}\ln|\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))| \leq \frac{1}{\sqrt{n}}\ln 2^n \leq \ln 2\cdot\sqrt{n},$$

and

$$\sum_{t=1}^{T}\frac{\eta_t}{2} < \sum_{t=1}^{T}\frac{\sqrt{n}}{\left(\sqrt{t-1}+\sqrt{t}\right)} = \sum_{t=1}^{T}\sqrt{n}\cdot\left(\sqrt{t}-\sqrt{t-1}\right) = \sqrt{nT}. \qquad (12)$$

$$
\begin{aligned}
&\sum_{t\in\mathcal{T}} \mathbb{E}[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l] \\
\overset{(E1)}{=}& \sum_{t\in\mathcal{T}} \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l]\cdot\mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}\right] \\
\overset{(E2)}{=}& \sum_{t\in\mathcal{T}} \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l]\cdot\left[\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\cdot 2^{m-l}L_m + \min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)\right] \\
=& \sum_{t\in\mathcal{T}}\left\{\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l]\cdot\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\cdot 2^{m-l}L_m + \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l]\cdot\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)\right\} \\
=& \sum_{t\in\mathcal{T}}\left\{\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[\boldsymbol{I}_{l+1,t}=\boldsymbol{i}_{l+1}|\boldsymbol{I}_{l,t}=\boldsymbol{i}_l]\cdot\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\cdot 2^{m-l}L_m + \min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)\right\} \\
\overset{(E3)}{=}& \sum_{t\in\mathcal{T}}\underbrace{\left[\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}\cdot\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\right]}_{\text{See (10), which is proved in Appendix A.1}}\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E4)}{\leq}& \sum_{t\in\mathcal{T}}\left\{\underbrace{\left[-\frac{1}{\eta_t}\ln\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)\cdot\exp\left(-\eta_t\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}+\frac{\eta_t}{2}\right]}_{\text{See (10), which is proved in Appendix A.1}}\cdot 2^{m-l}L_m + \min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)\right\} \\
\overset{(E5)}{=}& \sum_{t\in\mathcal{T}}\left[-\frac{1}{\eta_t}\ln\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}_{l+1,t}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}+\frac{\eta_t}{2}\right]\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E6)}{=}& \sum_{t\in\mathcal{T}}\left[\Phi_t(\eta_t)-\Phi_{t-1}(\eta_t)+\frac{\eta_t}{2}\right]\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E7)}{=}& \left\{\Phi_T(\eta_T)+\underbrace{\sum_{t=1}^{T-1}(\Phi_t(\eta_t)-\Phi_t(\eta_{t+1}))}_{\text{See (11) which is proved in Appendix A.2}}-\Phi_0(\eta_1)+\sum_{t=1}^{T}\frac{\eta_t}{2}\right\}\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E8)}{\leq}& \left\{\Phi_T(\eta_T)+\underbrace{\ln 2\cdot\sqrt{nT}}_{\text{See (11) which is proved in Appendix A.2}}-\Phi_0(\eta_1)+\sum_{t=1}^{T}\frac{\eta_t}{2}\right\}\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E9)}{\leq}& \underbrace{\left\{\Phi_T(\eta_T)+\ln 2\cdot\sqrt{nT}+\ln 2\cdot\sqrt{n}+\sqrt{nT}\right\}}_{\text{See (13) which is proved in Appendix A.3}}\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E10)}{\leq}& \underbrace{\left\{\sum_{t\in\mathcal{T}}\mathbb{E}\left[\frac{c_{m,t}(\boldsymbol{I}_{m,t})-\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)}{2^{m-l}L_m}|\boldsymbol{I}_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+(\ln 2+1)\sqrt{nT}+\ln 2\cdot\sqrt{n}\right\}}_{\text{See (13) which is proved in Appendix A.3}}\cdot 2^{m-l}L_m + \sum_{t\in\mathcal{T}}\min_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m) \\
\overset{(E11)}{=}& \sum_{t\in\mathcal{T}}\mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+\left((\ln 2+1)\sqrt{nT}+\ln 2\cdot\sqrt{n}\right)\cdot 2^{m-l}L_m \\
\overset{(E12)}{=}& \sum_{t\in\mathcal{T}}\mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+\underbrace{\left((\ln 2+1)nDL\sqrt{T}+\ln 2\cdot nDL\right)}_{\text{Denoted as }\phi}\cdot\frac{1}{2^l},\qquad\forall\boldsymbol{i}''_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))
\end{aligned}
\tag{9}
$$

Inequality ($E10$) is due to the following inequality

$$\Phi_T(\eta_T) \leq \sum_{t\in\mathcal{T}} \mathbb{E}\left[\frac{c_{m,t}(\boldsymbol{I}_{m,t}) - \min\limits_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_l(\boldsymbol{i}_l))} c_{m,t}(\boldsymbol{i}_m)}{2^{m-l}L_m} | \boldsymbol{I}_{l+1,t} = \boldsymbol{i}''_{l+1}\right], \tag{13}$$

which is proved in Appendix A.3. Equality ($E11$) combines the first term and the last term. Equality ($E12$) follows from the fact that $L_m = \sqrt{n}DL/2^m$.

For any cost functions $(f_1, f_2, \cdots, f_T)$, let $\boldsymbol{i}^*_m$ be the index of the sample point with the smallest cumulative cost, i.e.,

$$\boldsymbol{i}^*_m \overset{\text{def}}{=} \arg\min_{\boldsymbol{i}_m\in\mathcal{I}_m} \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m).$$

By repeatedly applying the result in Equation (9), we have

$$\begin{aligned}
&\sum_{t\in\mathcal{T}} \mathbb{E}[c_{m,t}(\boldsymbol{I}_{m,t})] - \min_{\boldsymbol{i}_m\in\mathcal{I}_m} \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m)\\
=&\sum_{t\in\mathcal{T}} \mathbb{E}[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{0,t} = \boldsymbol{1}] - \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}^*_m)\\
\leq&\sum_{t\in\mathcal{T}} \mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{1,t} = \boldsymbol{v}_1(\boldsymbol{i}^*_m)\right] + \phi\cdot\frac{1}{2^0} - \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}^*_m)\\
\leq&\sum_{t\in\mathcal{T}} \mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{2,t} = \boldsymbol{v}_2(\boldsymbol{i}^*_m)\right] + \phi\cdot\frac{1}{2^1} + \phi\cdot\frac{1}{2^0} - \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}^*_m)\\
&\vdots\\
\leq&\sum_{t\in\mathcal{T}} \mathbb{E}\left[c_{m,t}(\boldsymbol{I}_{m,t})|\boldsymbol{I}_{m,t} = \boldsymbol{v}_m(\boldsymbol{i}^*_m)\right] + \phi\cdot\left(\frac{1}{2^{m-1}} + \cdots\frac{1}{2^0}\right) - \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}^*_m)\\
=&\phi\cdot\left(\frac{1}{2^{m-1}} + \cdots\frac{1}{2^0}\right) = 2\phi\cdot\left[1 - \frac{1}{2^m}\right] \leq 2\phi.
\end{aligned} \tag{14}$$

Since (14) holds for any cost functions $(f_1, f_2, \cdots, f_T)$, we have

$$\text{regret}_{\text{lmC},T}(\mathfrak{R}) \leq 2\phi = 2(\ln 2 + 1)nDL\sqrt{T} + 2\ln 2\cdot nDL.$$

This completes the proof. □

Remark 4. At each layer, the ORW algorithm chooses among at most 2^n subsets. Moreover, due to the Lipschitz condition, the expected normalized cost difference among subsets to be chosen at each layer decreases exponentially as the algorithm goes down. These two facts result in the exponentially decreasing of the regret loss due to "imperfect choice" at each layer. This is also implied by Equation (9) in the proof of Lemma 4.3. Thus , the total regret loss of "imperfect choice" , $\text{regret}_{\text{lmC},T}$ is always upper bounded by $O(\sqrt{T})$ in timescale, no matter how many layers there are.

Moreover, the regret loss due to imperfect discretization can be reduced with larger m. Now, we state the the following lemma.

Lemma 4.4. *The* ORW *algorithm guarantees that*

$$\text{regret}_{\text{lmD},T}(\mathfrak{R}) \leq \frac{\sqrt{n}}{2^m}DLT.$$

Proof. For any cost functions $(f_1, f_2, \ldots, f_T)$, let

$$\boldsymbol{x}^* \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}).$$

Suppose that layer m subset containing $\boldsymbol{x}^*$ is indexed by $\boldsymbol{i}_m^*$. The sample point for subset $\mathcal{K}_m(\boldsymbol{i}_m^*)$ is $\boldsymbol{r}_m(\boldsymbol{i}_m^*) \in \mathcal{K}$. Clearly, $\boldsymbol{r}_m(\boldsymbol{i}_m^*) = (r_1, r_2, \ldots, r_n)$ and $\boldsymbol{x}^* = (x_1, x_2, \ldots, x_n)$ are in the same sub-cube whose edge length is $\frac{D}{2^m}$. Thus, we have

$$\begin{aligned}||\boldsymbol{r}_m(\boldsymbol{i}_m^*) - \boldsymbol{x}^*||_2 =& \sqrt{(r_1 - x_1)^2 + (r_2 - x_2)^2 + \cdots (r_n - x_n)^2} \\ \leq& \sqrt{\left(\frac{D}{2^m}\right)^2 + \left(\frac{D}{2^m}\right)^2 + \cdots + \left(\frac{D}{2^m}\right)^2} = \frac{D\sqrt{n}}{2^m}.\end{aligned}$$

Then, we have

$$\begin{aligned}&\min_{\boldsymbol{i}_m\in\mathcal{I}_m} \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m) - \min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}) \\ \leq& \sum_{t\in\mathcal{T}} c_{m,t}(\boldsymbol{i}_m^*) - \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}^*) \\ =& \sum_{t\in\mathcal{T}} f_t(\boldsymbol{r}_m(\boldsymbol{i}_m^*)) - f_t(\boldsymbol{x}^*) \leq TL||\boldsymbol{r}_m(\boldsymbol{i}_m^*) - \boldsymbol{x}^*||_2 \leq \frac{\sqrt{n}}{2^m} DLT,\end{aligned} \tag{15}$$

where the first inequality is due to the Lipschitz condition. Since (15) holds for any cost functions $(f_1, f_2, \cdots, f_T)$, we conclude that $\text{regret}_{\text{lmD},T}(\mathfrak{R}) \leq \frac{\sqrt{n}}{2^m} DLT$. This completes the proof. □

Putting together the results in lemmas 4.2, 4.3, and 4.4, the results in Theorem 4.1 is proved.

5 ADAPTIVE SAMPLING

The ORW algorithm described in Section 3, specifies a granularity parameter, m, and samples the costs at the bottommost layer (layer m). With m larger than $\log_2 \sqrt{nT}$, the ORW algorithm guarantees the regret to be $O(\sqrt{T})$. In many cases, however, the duration interval is long and unknown to the online player, and setting a large m to guarantee $O(\sqrt{T})$ regret might be costly to compute. In order to handle cases with unknown T and reduce this complexity, in this section, we devise an adaptive version of the ORW algorithm (called the AORW algorithm), which increases the granularity parameter gradually. The AORW algorithm is denoted by $\mathfrak{R}^{\text{adpt}}$ in equations.

5.1 Online Recursive Weighting Algorithm with Adaptive Sampling

The AORW algorithm adopts an increasing granularity parameter m_t. We denote the time interval that satisfies $m_t \geq l$ as $\mathcal{T}_l$ or $[t_l, T]$. With increasing m_t, the index and the index set for the sampling subsets are updated to the new notation, $\boldsymbol{i}_{m_t}$ and $\mathcal{I}_{m_t}$, respectively. $\boldsymbol{r}_{m_t}(\boldsymbol{i}_{m_t})$ is used to denote the sample point for the subset of index $\boldsymbol{i}_{m_t}$, and the cost on the sampled point of the subset $\boldsymbol{i}_{m_t} \in \mathcal{I}_{m_t}$ at iteration t is denoted as

$$c'_{m_t,t}(\boldsymbol{i}_{m_t}) \stackrel{\text{def}}{=} f_t(\boldsymbol{r}_{m_t}(\boldsymbol{i}_{m_t})).$$

The AORW algorithm maintains the *cumulative expected cost*, $\bar{C}'_{l,t}(\boldsymbol{i}_l)$, for each subset at layers $\{1, 2, \ldots, m_t\}$. The cumulative expected cost is initialized to be zero, i.e., $\bar{C}'_{l,0}(\boldsymbol{i}_l) = 0$. During run time, the cumulative expected cost is used to determine the choosing probability of subsets at layers $\{1, 2, \ldots, m_t\}$. The selection probability for the subset $\mathcal{K}_{l+1}(\boldsymbol{i}_{l+1})$ conditioning on that $\mathcal{K}_l(\boldsymbol{i}_l)$ has

$$\begin{aligned}\bar{c}'_{l,t}(\boldsymbol{i}_l) &\overset{\text{def}}{=} \mathbb{E}\left[\frac{c_{m_t,t}(\boldsymbol{I}_{m_t,t}) - \min\limits_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_{l-1}(\boldsymbol{v}_{l-1}(\boldsymbol{i}_l)))} c_{m_t,t}(\boldsymbol{i}_{m_t})}{2^{m_t-l+1}L_{m_t}} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right] \\ &= \sum_{\boldsymbol{j}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{c_{m_t,t}(\boldsymbol{j}_{m_t}) - \min\limits_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_{l-1}(\boldsymbol{v}_{l-1}(\boldsymbol{i}_l)))} c_{m_t,t}(\boldsymbol{i}_{m_t})}{2^{m_t-l+1}L_{m_t}} \cdot \Pr\left[\boldsymbol{I}_{m_t,t} = \boldsymbol{j}_{m_t} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right].\end{aligned} \tag{17}$$

been selected is

$$\Pr\left[\boldsymbol{I}_{l+1,t} = \boldsymbol{i}_{l+1} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right] = \frac{\exp\left(-\eta_t \bar{C}'_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)}{\sum\limits_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}'_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}. \tag{16}$$

Since there is a subset at each layer chosen in a recursive manner from the top layer to the bottom layer, the action sequence of the AORW algorithm at iteration t can be correspondingly denoted by a vector, $(\boldsymbol{I}_{0,t}, \boldsymbol{I}_{1,t}, \boldsymbol{I}_{0,t}, \ldots, \boldsymbol{I}_{m_t,t})$, where $\boldsymbol{I}_{0,t}$ is set to be $\mathbf{1}$ by default.

After the costs on the sampled points of the bottom-layer subsets are revealed, the AORW algorithm calculates the *expected normalized cost* for each subset at layers $\{1, 2, \ldots, m_t\}$, which is defined in Equation (17).

Then, the calculated *expected normalized cost* is used to update the cumulative expected cost, i.e.,

$$\bar{C}'_{l,t}(\boldsymbol{i}_l) = \bar{C}'_{l,t-1}(\boldsymbol{i}_l) + \bar{c}'_{l,t}(\boldsymbol{i}_l). \tag{18}$$

For the case when $m_{t+1} = m_t + 1$, the AORW algorithm will add the cumulative expected cost for those subsets at the new layer to maintenance, with previous values over $[1, t_{m_{t+1}} - 1]$ set to be 0, i.e.,

$$\bar{c}'_{m_{t+1},\tau}(\boldsymbol{i}_{m_{t+1}}) = 0, \text{ for } \tau = 0, 1, 2, \ldots, t_{m_{t+1}} - 1.$$

The summary of the AORW algorithm is listed in Algorithm 2.

5.2 Regret Analysis

By adaptive sampling, the algorithm parameter does not depend on the length of time interval. Meanwhile, an interesting result of the regret analysis is that adaptive sampling does not degrade the regret bound. We state the main result in the following theorem.

Theorem 5.1. *With $\eta_t = \sqrt{\frac{n}{t}}$ and m_t being set to $\lceil \log_2 \sqrt{nt} \rceil$, the* AORW *algorithm achieves the following regret bound*

$$\text{regret}_T(\mathfrak{R}^{\text{adpt}}) \leq \left(6n + 2\sqrt{n}\right) DL\sqrt{T}.$$

Let $\boldsymbol{i}^*_{m_t}$, $t \in \mathcal{T}$ indicate the bottom subset that contains the optimal decision point $\boldsymbol{x}^*$ at iteration t, i.e.,

$$\boldsymbol{i}^*_{m_t} : \boldsymbol{x}^* \in \mathcal{K}_{m_t}(\boldsymbol{i}^*_{m_t}),\ t \in \mathcal{T}.$$

Similar to the regret analysis in Section 4, we define the regret due to the imperfect choice among sample points as

$$\text{regret}_{\text{lmC},T}(\mathfrak{R}^{\text{adpt}}) \overset{\text{def}}{=} \sup_{f_1,\ldots,f_T\in\mathcal{F}} \left\{\sum_{t\in\mathcal{T}} \mathbb{E}\left[c'_{m_t,t}(\boldsymbol{I}_{m_t,t})\right] - \sum_{t\in\mathcal{T}} c'_{m_t,t}(\boldsymbol{i}^*_{m_t})\right\},$$

and that due to the imperfect sampling as

$$\text{regret}_{\text{ImD},T}(\mathfrak{R}^{\text{adpt}}) \stackrel{\text{def}}{=} \sup_{f_1,\ldots,f_T\in\mathcal{F}} \left\{ \sum_{t\in\mathcal{T}} c'_{m_t,t}(\boldsymbol{i}^*_{m_t}) - \min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}) \right\}.$$

Similar to Lemma 3.1, we have

$$\text{regret}_T(\mathfrak{R}^{\text{adpt}}) \le \text{regret}_{\text{ImC},T}(\mathfrak{R}^{\text{adpt}}) + \text{regret}_{\text{ImD},T}(\mathfrak{R}^{\text{adpt}}).$$

In the following two lemmas, we bound the regret loss due to imperfect choice and sampling, respectively.

Lemma 5.2. *With $\eta_t = \sqrt{\frac{n}{t}}$ and the $m_t = \lceil \log_2 \sqrt{t} \rceil$, the* AORW *algorithm guarantees that*

$$\text{regret}_{\text{ImC},T}(\mathfrak{R}^{adpt}) \le 6DLn\sqrt{T}.$$

The proof of Lemma 5.2 is analogous to that of Lemma 4.3. The details are given in Appendix A.4.

Lemma 5.3. *With $m_t = \lceil \log_2 \sqrt{t} \rceil$, the* AORW *algorithm guarantees that*

$$\text{regret}_{\text{ImD},T}(\mathfrak{R}^{adpt}) \le 2DL\sqrt{nT}.$$

Proof. For any cost functions $(f_1, f_2, \cdots, f_T)$, let

$$\boldsymbol{x}^* \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}).$$

Suppose that the bottom subset containing point $\boldsymbol{x}^*$ is indexed by $\boldsymbol{i}^*_{m_t}$, whose sample point is $\boldsymbol{r}_{m_t}(\boldsymbol{i}^*_{m_t}) \in \mathcal{K}$. Clearly, $\boldsymbol{r}_{m_t}(\boldsymbol{i}^*_{m_t}) = (r_{m_t,1}, r_{m_t,2}, \cdots, r_{m_t,n})$ and $\boldsymbol{x}^* = (x_1, x_2, \cdots, x_n)$ are in the same sub-cube whose edge length is $\frac{D}{2^{m_t}}$. Thus, we have

$$\begin{aligned}
||\boldsymbol{r}_{m_t}(\boldsymbol{i}^*_{m_t}) - \boldsymbol{x}^*||_2 =& \sqrt{(r_{m_t,1} - x_1)^2 + (r_{m_t,2} - x_2)^2 + \cdots (r_{m_t,n} - x_n)^2} \\
\le& \sqrt{\left(\frac{D}{2^{m_t}}\right)^2 + \left(\frac{D}{2^{m_t}}\right)^2 + \cdots + \left(\frac{D}{2^{m_t}}\right)^2} = \frac{D\sqrt{n}}{2^{m_t}} \\
\le& D\sqrt{\frac{n}{t}}.
\end{aligned}$$

Then we have

$$\begin{aligned}
\sum_{t\in\mathcal{T}} c'_{m_t,t}(\boldsymbol{i}^*_{m_t}) - \min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}) &= \sum_{t\in\mathcal{T}} c'_{m_t,t}(\boldsymbol{i}^*_{m_t}) - \sum_{t\in\mathcal{T}} f_t(\boldsymbol{x}^*) \\
&= \sum_{t\in\mathcal{T}} f_t(\boldsymbol{r}_m(\boldsymbol{i}^*_m)) - f_t(\boldsymbol{x}^*) \\
&\stackrel{(E1)}{\le} L \sum_{t\in\mathcal{T}} ||\boldsymbol{r}_{m_t}(\boldsymbol{i}^*_{m_t}) - \boldsymbol{x}^*||_2 \le 2DL\sqrt{nT},
\end{aligned} \tag{19}$$

where inequality $(E1)$ is due to the Lipschitz condition. Since (19) holds for any cost functions $(f_1, f_2, \cdots, f_T)$, we conclude that $\text{regret}_{\text{ImD},T}(\mathfrak{R}) \le 2DL\sqrt{nT}$. This completes the proof. □

The results in lemmas 5.2 and 5.3 immediately prove the results in Theorem 5.1.

Algorithm 2 Online Recursive Weighting with Adaptive Sampling

Input: $\{\eta_t = \sqrt{\frac{n}{t}}\}$, $\{m_t = \lceil \log_2 \sqrt{t} \rceil\}$
Output: $\boldsymbol{r}_{m_1}(\boldsymbol{I}_{m_1,1}), \boldsymbol{r}_{m_2}(\boldsymbol{I}_{m_2,2}), \ldots$

```
//↑ Output sample points of layer m subsets
//↓ Initialize the normalized expected cost for all subsets at all layers
```
Set $\bar{C}_{l,0}(\boldsymbol{i}_l) = 0, \forall \boldsymbol{i}_l \in \mathcal{I}_l, l = 1, 2, \ldots, m_1$
for $t = 1, 2, \ldots, T$ **do**
  $\boldsymbol{I}_{0,t} = \boldsymbol{1}$
  `//↓ Recursively select the subsets at all layers`
  for $l = 0, 1, \ldots, m_t - 1$ **do**
    Select an index $\boldsymbol{I}_{l+1,t} \in \mathcal{E}_{l+1}(\boldsymbol{I}_{l,t})$ according to the probability distribution calculated in Equation (16)
  end for
  Choose the subset indexed by $\boldsymbol{I}_{m_t,t}$ at iteration t
  Choose the sample point for subset $\mathcal{K}_{m_t}(\boldsymbol{I}_{m_t,t})$, i.e., $\boldsymbol{r}_{m_t}(\boldsymbol{I}_{m_t,t})$, as the final decision
  `//↓ Get the normalized expected cost for all subsets at all layers based on the revealed cost function` $f_t(\boldsymbol{x})$ `at iteration` t
  for $l = 1, 2, \ldots, m_t$ **do**
    for $\boldsymbol{i}_l \in \mathcal{I}_l$ **do**
      for $\boldsymbol{j}_{m_t} \in \mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))$ **do**
        Calculate $\Pr\left[\boldsymbol{I}_{m,t} = \boldsymbol{j}_{m_t} | \boldsymbol{I}_{l,t} = \boldsymbol{i}_l\right]$ according to Equation (5)
      end for
      Calculate $\bar{c}_{l,t}(\boldsymbol{i}_l)$ according to Equation (17)
      Update $\bar{C}'_{l,t}(\boldsymbol{i}_l) = \bar{C}'_{l,t-1}(\boldsymbol{i}_l) + \bar{c}'_{l,t}(\boldsymbol{i}_l)$
    end for
  end for
  `//↓ Initialize the cumulative expected cost for all subsets at the new layer`
  if $m_{t+1} > m_t$ **then**
    for $\boldsymbol{i}_{m_{t+1}} \in \mathcal{I}_{m_{t+1}}$ **do**
      Set $\bar{C}'_{m_{t+1},t}(\boldsymbol{i}_{m_{t+1}}) = 0$
    end for
  end if
end for

6 EXTENSIONS AND STATE-OF-THE-ART RESULTS

6.1 Online Optimization in a Decentralized Environment

Recently, Hosseini [23] and Lee [33] generalized the classic OCO problem to a decentralized optimization framework within a network of agents. In their work, consensus-based gradient-descent algorithms were proposed for distributed online optimization. In their setting, each agent aims to drive its individual average regret, which is the average over time of the regret function evaluated at this agent's estimation for the choice that the whole network should make, to zero. An interesting message of their work is that the $O(\sqrt{T})$ regret can still be attained by leveraging the communication among agents. In addition, [43] addresses decentralized online optimization in non-stationary environments using mirror decent, and in [4], distributed online optimization is studied for strongly convex objective functions over time-varying networks.

In parallel, a promising future work for our approach is to implement the proposed weighting method to solve the distributed ONCL problem. The extension is natural, while techniques to be adopted might be rather different. That is due to the fact that the ORW algorithm has to maintain an estimation for each subset and it might be very costly to exchange such information. Thus, in the opinion of the authors, the main challenge to implement such a weighting method within a decentralized environment is to alleviate the communication overhead among agents.

6.2 Partial Information Feedback

We emphasize that our online non-convex learning problem is based on *full information feedback*. That is, the whole cost function will be revealed at the end of each iteration. It is an interesting and important future direction to consider *partial information feedback* where only the cost value of the player's choice is revealed. The *partial information feedback* extension is motivated by many real-world systems in which the observer is not co-located with the controller and the feedback information is noisy, partial, or incomplete due to limited communication bandwidth. Some examples are online routing in data networks [7], power control in cellular networks [48], and the ad placement problem on a web page [47].

The bandit information feedback setting has been investigated in a huge number of works in the OCO framework such as [1, 2, 11, 13, 21, 22, 41], which is called the Bandit Convex Optimization (BCO) problem. Among those works, [1, 11, 13, 41] reported to attain a sublinear regret for the BCO problem, respectively, but none of them have attained the lower bound of the regret. For the special case of strongly convex and smooth losses, [2] obtained a regret of $\tilde{O}(\sqrt{T})$ in the unconstrained case, and [21] obtained the same rate even in the constrained case. Recently, a new algorithm was reported by Hazan *et al.* to attain a regret of $(\ln T)^{2d}\sqrt{T}$ [22]. This is the first algorithm to attain a $\tilde{O}(\sqrt{T})$ regret for the OCO model with bandit feedback. Different from the above works, [3, 28–30, 38] studied the model where the (expected) payoff function satisfies a Lipschitz condition with respect to the metric. [3, 30, 38] investigate this problem in a few specific metric spaces such as a one-dimensional real interval, while in [28, 29], the action set is from a general metric space. Their model is more general and the results in [29] show that there is an algorithm whose regret on any instance satisfies $R(T) = \tilde{O}(T^{\frac{d+1}{d+2}})$, where d is the dimension of the action set.

Despite of the above results, the optimal online algorithm and tight regret bound for the online non-convex optimization problem with bandit feedback are still open, calling for more investigation from the community.

7 CONCLUSION

In this paper, we investigated the online non-convex learning problem, which removes the convexity assumption of the cost function of the classic online convex optimization problem. This generalization is necessitated by a huge amount of important applications which incur non-convex penalties, such as in portfolio selection and SVM training problems. Without the convexity assumption, it is far more challenging to design an efficient online algorithm with sublinear regret. The classic exponential weighting online algorithm is shown to attain a sublinear regret of $O(\sqrt{T \log T})$ and the convex optimization methods even fail to converge to the optimum. Our analysis prove that by properly partitioning subsets and using the online recursive weighting method, the regret can be reduced to match the known lower bound for OCO, i.e., $O(\sqrt{T})$.

ACKNOWLEDGEMENT

The authors would like to thank Professor Giulia Fanti for her kind shepherding and the anonymous reviewers of SIGMETRICS 2018 for their insightful comments and suggestions to improve the quality of the paper.

REFERENCES

[1] J. Abernethy, E. Hazan, and A. Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *COLT*, pages 263–274, 2008.
[2] A. Agarwal, O. Dekel, and L. Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT*, pages 28–40. Citeseer, 2010.
[3] R. Agrawal. The continuum-armed bandit problem. *SIAM journal on control and optimization*, 33(6):1926–1951, 1995.
[4] M. Akbari, B. Gharesifard, and T. Linder. Distributed online convex optimization on time-varying directed graphs. *IEEE Transactions on Control of Network Systems*, 2015.
[5] D. Ardia, K. Boudt, P. Carl, K. M. Mullen, and B. Peterson. Differential evolution (deoptim) for non-convex portfolio optimization. 2010.
[6] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.
[7] B. Awerbuch and R. Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.
[8] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM (JACM)*, 44(3):427–485, 1997.
[9] R. Combes, S. Magureanu, A. Proutiere, and C. Laroche. Learning to rank: Regret lower bounds and efficient algorithms. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):231–244, 2015.
[10] T. Cover. Universal portfolios. *Mathematical finance*, 1(1):1–29, 1991.
[11] O. Dekel, R. Eldan, and T. Koren. Bandit smooth convex optimization: Improving the bias-variance tradeoff. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2926–2934, 2015.
[12] S. Ertekin, L. Bottou, and C. Giles. Non-convex online support vector machines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):368–381, 2011.
[13] A. Flaxman, A. Kalai, and H. McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 385–394, 2005.
[14] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.
[15] G. Gasso, L. Pappaioannou, M. Spivak, and L. Bottou. Batch and online learning algorithms for nonconvex neyman-pearson classification. *ACM Transactions on Intelligent Systems and Technology*, 2(3):28, 2011.
[16] A. J. Grove, N. Littlestone, and D. Schuurmans. General convergence results for linear discriminant updates. *Machine Learning*, 43(3):173–210, 2001.
[17] T. Guzella and W. Caminhas. A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7):10206–10222, 2009.
[18] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3–4):157–325, 2016.
[19] E. Hazan and S. Kale. Online submodular minimization. *Journal of Machine Learning Research*, 13(Oct):2903–2922, 2012.
[20] E. Hazan and S. Kale. Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization. *Journal of Machine Learning Research*, 15(1):2489–2512, 2014.
[21] E. Hazan and K. Levy. Bandit convex optimization: Towards tight bounds. In *Advances in Neural Information Processing Systems (NIPS)*, pages 784–792, 2014.
[22] E. Hazan and Y. Li. An optimal algorithm for bandit convex optimization. *arXiv preprint arXiv:1603.04350*.
[23] S. Hosseini, A. Chapman, and M. Mesbahi. Online distributed convex optimization on dynamic networks. *IEEE Transactions on Automatic Control*, 61(11):3545–3550, 2016.
[24] P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in neural information processing systems*, pages 761–768, 2009.
[25] A. Kalai and S. Vempala. Efficient algorithms for universal portfolios. *Journal of Machine Learning Research*, 3(Nov):423–440, 2002.
[26] J. Kivinen and M. K. Warmuth. Relative loss bounds for multidimensional regression problems. In *Advances in neural information processing systems (NIPS)*, pages 287–293, 1998.

[27] R. Kleinberg and A. Slivkins. Sharp dichotomies for regret minimization in metric spaces. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 827–846. Society for Industrial and Applied Mathematics, 2010.
[28] R. Kleinberg, A. Slivkins, and E. Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690. ACM, 2008.
[29] R. Kleinberg, A. Slivkins, and E. Upfal. Bandits and experts in metric spaces. *arXiv preprint arXiv:1312.1277*, 2013.
[30] R. D. Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *Advances in Neural Information Processing Systems*, pages 697–704, 2005.
[31] W. Krichene, M. Balandat, C. Tomlin, and A. Bayen. The hedge algorithm on a continuum. In *the 32nd International Conference on Machine Learning (ICML-15)*, pages 824–832, 2015.
[32] P. Krokhmal, J. Palmquist, and S. Uryasev. Portfolio optimization with conditional value-at-risk objective and constraints. *Journal of risk*, 4:43–68, 2002.
[33] S. Lee, A. Nedich, and M. Raginsky. Stochastic dual averaging for decentralized online optimization on time-varying communication graphs. *IEEE Transactions on Automatic Control*, 2017.
[34] N. Littlestone and M. Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.
[35] S. Magureanu, A. Proutiere, M. Isaksson, and B. Zhang. Online learning of optimally diverse rankings. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):32, 2017.
[36] O.-A. Maillard and R. Munos. Online learning in adversarial lipschitz environments. *Machine Learning and Knowledge Discovery in Databases*, pages 305–320, 2010.
[37] L. Mason, P. L. Bartlett, and J. Baxter. Improved generalization through explicit optimization of margins. *Machine Learning*, 38(3):243–255, 2000.
[38] S. Pandey, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for taxonomies: A model-based approach. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 216–227. SIAM, 2007.
[39] A. Rakhlin, O. Shamir, and K. Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 449–456, 2012.
[40] H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.
[41] A. Saha and A. Tewari. Improved regret guarantees for online smooth convex optimization with bandit feedback. In *AISTATS*, pages 636–642, 2011.
[42] D. Sculley and G. Wachman. Relaxed online svms for spam filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 415–422, 2007.
[43] S. Shahrampour and A. Jadbabaie. Distributed online optimization in dynamic environments using mirror descent. *IEEE Transactions on Automatic Control*, 2017.
[44] O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.
[45] M. S. Talebi, Z. Zou, R. Combes, A. Proutiere, and M. Johansson. Stochastic online shortest path routing: The value of feedback. *IEEE Transactions on Automatic Control*, 2017.
[46] S. Uryasev. Conditional value-at-risk: Optimization algorithms and applications. In *Computational Intelligence for Financial Engineering, 2000.(CIFEr) Proceedings of the IEEE/IAFE/INFORMS 2000 Conference on*, pages 49–57. IEEE, 2000.
[47] F. Wauthier, M. Jordan, and N. Jojic. Efficient ranking from pairwise comparisons. In *International Conference on Machine Learning (ICML)*, pages 109–117, 2013.
[48] W. Wong and C. Sung. Robust convergence of low-data rate-distributed controllers. *IEEE transactions on automatic control*, 49(1):82–87, 2004.
[49] H. H. Zhang, J. Ahn, X. Lin, and C. Park. Gene selection using support vector machines with non-convex penalty. *bioinformatics*, 22(1):88–95, 2005.
[50] L. Zhang, T. Yang, R. Jin, and Z. Zhou. Online bandit learning for a special class of non-convex losses. In *AAAI*, pages 3158–3164, 2015.
[51] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 928–936, 2003.

A PROOFS

A.1 Proof of Inequality (10)

The proof relies on the following Hoeffding's Lemma.

Lemma A.1. *(Hoeffding's Lemma) Let Z be any real-valued random variable with expected value $\mathbb{E}[Z] = 0$ and such that $a \le Z \le b$ almost surely. Then, for all $\lambda \in \mathbb{R}$,*

$$\mathbb{E}\left[\exp(\lambda Z)\right] \le \exp\left(\frac{\lambda^2(b-a)^2}{8}\right).$$

We thus define a random variable Y whose probability mass function is specified as

$$\Pr\left[Y = \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})\right] = \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}, \tag{20}$$

for $\forall \boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))$.

Since $\bar{c}_{l+1,t}(\boldsymbol{i}_{l+1}) \in [0,1], \forall \boldsymbol{i}_{l+1}$, we have that $0 \le Y \le 1$ for sure. However, random variable Y may not have zero mean. We thus denote another random variable $Z = Y - \mathbb{E}[Y]$. Clearly, $\mathbb{E}[Z] = 0$ and $-1 \le Z \le 1$ for sure. We further let $\lambda = -\eta_t$. Thus, by Hoeffding's Lemma, we have

$$\mathbb{E}[\exp(-\eta_t Z)] \le \exp\left(\frac{(-\eta_t)^2(1-(-1))^2}{8}\right) = \exp\left(\frac{\eta_t^2}{2}\right). \tag{21}$$

In addition, we have

$$\mathbb{E}[\exp(-\eta_t Z)] = \mathbb{E}[\exp(-\eta_t(Y - \mathbb{E}[Y]))] = \mathbb{E}[\exp(-\eta_t Y)] \cdot \exp[\eta_t \mathbb{E}[Y]]. \tag{22}$$

Combining (21) and (22), we get

$$\mathbb{E}[\exp(-\eta_t Y)] \cdot \exp\left(\eta_t \mathbb{E}[Y]\right) \le \exp\left(\frac{\eta_t^2}{2}\right). \tag{23}$$

Taking logarithm and rearranging the items in (23), we have

$$\mathbb{E}[Y] \le -\frac{1}{\eta_t} \ln \mathbb{E}[\exp(-\eta_t Y)] + \frac{\eta_t}{2}. \tag{24}$$

Now applying the probability mass function (20) into (24), we get that

$$\begin{aligned}
\mathbb{E}[Y] &= \sum_{\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right) \cdot \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1})}{\sum_{\boldsymbol{i}'_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)} \\
&\le -\frac{1}{\eta_t} \ln \mathbb{E}[\exp(-\eta_t Y)] + \frac{\eta_t}{2}. \\
&= -\frac{1}{\eta_t} \ln \sum_{\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}_{l+1})\right) \cdot \exp(-\eta_t \bar{c}_{l+1,t}(\boldsymbol{i}_{l+1}))}{\sum_{\boldsymbol{i}'_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_t \bar{C}_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)} + \frac{\eta_t}{2}.
\end{aligned}$$

A.2 Proof of Equality (11)

To prove (11), we only need to show that

$$\Phi_t(\eta_t) - \Phi_t(\eta_{t+1}) \le \ln 2 \cdot n \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right). \tag{25}$$

The second inequality of (11) simply follows from $\eta_t = \sqrt{\frac{n}{t}}$. We next prove (25).

We call a vector $\boldsymbol{z} = (z_1, z_2, \ldots, z_m) \in \mathbb{R}^m$ *a non-increasing non-negative vector* if $z_1 \geq z_2 \geq \cdots \geq z_m \geq 0$. We then define a function

$$\Psi(a, \boldsymbol{z}) \stackrel{\text{def}}{=} -\frac{1}{a} \ln \sum_{j=1}^{m} \exp\left(-a z_j\right), \quad a > 0,$$

where $\boldsymbol{z} = (z_1, z_2, \cdots, z_m)$ is a non-increasing non-negative vector in $\mathbb{R}^m$. We further define a function $\hbar(\bullet)$ mapping from a non-increasing non-negative vector to another non-increasing non-negative vector, where $\hbar(\boldsymbol{z})$ changes all largest elements of $\boldsymbol{z}$ into the second-largest element of $\boldsymbol{z}$. Namely, if the first k elements $\boldsymbol{x}$ are the largest, i.e., $z_1 = z_2 = \cdots = z_k > z_{k+1} \geq \cdots \geq z_m$, then

$$\hbar(\boldsymbol{z}) = (\underbrace{z_{k+1}, z_{k+1}, \ldots, z_{k+1}}_{\text{In total } k}, z_{k+1}, z_{k+2}, \ldots, z_m).$$

Clearly, if we apply function $\hbar(\bullet)$ to a non-increasing non-negative vector $\boldsymbol{x}$ for m times, we get a constant vector $\boldsymbol{z}_{\min} = (z_m, z_m, \cdots, z_m)$.

We have the following lemma.

Lemma A.2. *For $a > b > 0$, any non-increasing non-negative vector $\boldsymbol{z}$, we have*

$$\Psi(a, \boldsymbol{z}) - \Psi(b, \boldsymbol{z}) \leq \Psi(a, \hbar(\boldsymbol{z})) - \Psi(b, \hbar(\boldsymbol{z})). \tag{26}$$

Proof. (i) If $\boldsymbol{z}$ is a constant vector, i.e., $z_1 = z_2 = \cdots = z_m$, then $\hbar(\boldsymbol{z}) = \boldsymbol{z}$ and thus (26) holds as an equality.

(ii) If $\boldsymbol{z}$ is not a constant vector, we assume that the largest element of $\boldsymbol{z}$ is z and there are in total $k < m$ largest elements in $\boldsymbol{z}$, i.e., $z = z_1 = z_2 = \cdots = z_k > z_{k+1} \geq z_{k+2} \cdots \geq z_m$.

Then

$$\begin{aligned}
&\Psi(a, \boldsymbol{z}) - \Psi(b, \boldsymbol{z}) \\
&= -\frac{1}{a} \ln\left(k \exp\left(-az\right) + \sum_{j=k+1}^{m} \exp\left(-a z_j\right)\right) + \frac{1}{b} \ln\left(k \exp\left(-bz\right) + \sum_{j=k+1}^{m} \exp\left(-b z_j\right)\right) \stackrel{\text{def}}{=} \Upsilon(z),
\end{aligned} \tag{27}$$

where the last equality defines a function with respect to z satisfying $z > z_{k+1}$. Taking the derivative on $\Upsilon(z)$ with respect to z, we get

$$\frac{\partial \Upsilon(z)}{\partial z} = \frac{k \exp\left(-az\right)}{k \exp\left(-az\right) + \sum_{j=k+1}^{m} \exp\left(-a z_j\right)} - \frac{k \exp\left(-bz\right)}{k \exp\left(-bz\right) + \sum_{j=k+1}^{m} \exp\left(-b z_j\right)}.$$

Define

$$\Theta(\sigma) \stackrel{\text{def}}{=} \frac{k \exp\left(-\sigma z\right)}{k \exp\left(-\sigma z\right) + \sum_{j=k+1}^{m} \exp\left(-\sigma z_j\right)}.$$

Then we have that

$$\frac{\partial \Upsilon(z)}{\partial z} = \Theta(a) - \Theta(b). \tag{28}$$

Taking derivative on $\Theta(\sigma)$ with respect to σ, we get

$$\frac{\partial \Theta(\sigma)}{\partial \sigma} = \frac{k \exp\left(-\sigma z\right) \sum_{j=k+1}^{m} (z_j - z) \exp\left(-\sigma z_j\right)}{\left(k \exp\left(-\sigma z\right) + \sum_{j=k+1}^{m} \exp\left(-\sigma z_j\right)\right)^2}.$$

Since $z > z_j$, for $j = k+1, k+2, \ldots, m$, we have $\frac{\partial \Theta(a)}{\partial a} < 0$. Because $a > b$, so we have

$$\frac{\partial \Upsilon(z)}{\partial z} = \Theta(a) - \Theta(b) < 0, \ \forall z > z_{i+1}.$$

Thus, $\Upsilon(z) < \Upsilon(z_{i+1})$. By noting that $\Upsilon(z_{i+1}) = \Psi(a, \hbar(z)) - \Psi(b, \hbar(z))$, we thus have

$$\Psi(a, z) - \Psi(b, z) < \Psi(a, \hbar(z)) - \Psi(b, \hbar(z)).$$

Part (i) and part (ii) complete the proof. □

Based on Lemma A.2, we can immediately obtain the following result.

Lemma A.3. *For $a > b > 0$ and any non-increasing non-negative vector $z \in \mathbb{R}^m$, we have*

$$\Psi(a, z) - \Psi(b, z) \leq \left(\frac{1}{b} - \frac{1}{a}\right) \ln m. \tag{29}$$

Proof. From Lemma A.2, we have

$$\begin{aligned}
\Psi(a, z) - \Psi(b, z) \leq & \Psi(a, \hbar(z)) - \Psi(b, \hbar(z)) \\
\leq & \Psi(a, \hbar(\hbar(z))) - \Psi(b, \hbar(\hbar(z))) \\
\leq & \cdots \\
\leq & \Psi(a, z_{\min}) - \Psi(b, z_{\min}) \\
= & \left(\frac{1}{b} - \frac{1}{a}\right) \ln m,
\end{aligned}$$

where $z_{\min} = (z_m, z_m, \ldots, z_m)$. This completes the proof. □

We can sort all $\bar{C}_{l+1,t}(\boldsymbol{i}_{l+1})$'s where $\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))$ in a descending order and construct a non-increasing non-negative vector $z \in \mathbb{R}^{|\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))|}$. Then based on Lemma A.2, we have

$$\begin{aligned}
\Phi_t(\eta_t) - \Phi_t(\eta_{t+1}) = & \Psi(\eta_t, z) - \Psi(\eta_{t+1}, z) \\
\leq & \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) \ln |\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))| \\
\leq & \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) n \ln 2,
\end{aligned} \tag{30}$$

where the last inequality follows from $|\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))| \leq 2^n$.

This completes the proof for inequality (11).

A.3 Proof of Inequality (13)

First, we have

$$\exp\left(\eta_T \max_{\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\boldsymbol{i}_l)} [-\bar{C}_{l+1,T}(\boldsymbol{i}_{l+1})]\right) \leq \sum_{\boldsymbol{i}_{l+1} \in \mathcal{E}_{l+1}(\boldsymbol{i}_l)} \exp\left(-\eta_T \bar{C}_{l+1,T}(\boldsymbol{i}_{l+1})\right). \tag{31}$$

Taking logarithm and dividing by $-\frac{1}{\eta_T} < 0$ in both sides of (31), we can get that

$$\begin{aligned}
\Phi_T(\eta_T) =& -\frac{1}{\eta_T}\ln \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_T \bar{C}_{l+1,T}(\boldsymbol{i}_{l+1})\right)\\
\leq& \min_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \bar{C}_{l+1,T}(\boldsymbol{i}_{l+1})\\
\leq& \bar{C}_{l+1,T}(\boldsymbol{i}''_{l+1}) \quad (\forall \boldsymbol{i}''_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l)))\\
=& \sum_{t=1}^{T} \bar{c}_{l+1,t}(\boldsymbol{i}''_{l+1})\\
=& \sum_{t=1}^{T} \mathbb{E}\left[\frac{c_{m,t}(\boldsymbol{I}_{m,t}) - \min\limits_{\boldsymbol{i}_m\in\mathcal{E}_m(\mathcal{K}_{l-1}(\boldsymbol{v}_{l-1}(\boldsymbol{i}_l)))} c_{m,t}(\boldsymbol{i}_m)}{2^{m-l+1}L_m} | \boldsymbol{I}_{l+1,t} = \boldsymbol{i}''_{l+1}\right],
\end{aligned} \tag{32}$$

where the second last equality follows from the definition of $\bar{C}_{l,t}(\boldsymbol{i}_l)$ in (8) and the last equality follows from the definition of $\bar{c}_{l,t}(\boldsymbol{i}_l)$ in (4) and $\boldsymbol{v}_l(\boldsymbol{i}''_{l+1}) = \boldsymbol{i}_l$.

This completes the proof.

A.4 Proof of Lemma 5.2

Proof. Analogous to Equation (9), we have Equation (33) which characterizes the regret loss at each layer. In Equation (33), equality ($E1$) is according to the definition for the *normalized expected cost* shown in Equation (17). Equality ($E2$) is simply by the choosing probability for subsets at each layer. Inequality ($E3$) has been proved in Section (A.1). Equality ($E4$) follows from the update law for the *cumulative expected cost*. Inequality ($E5$) is based on the Equation (11) and (12), as well as the following inequality

$$\begin{aligned}
-\Phi_{t_{l+1}-1}(\eta_{t_{l+1}}) =& \frac{1}{\eta_{t_{l+1}}}\ln \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \exp\left(-\eta_{t_{l+1}} \bar{C}_{l+1,t_{l+1}-1}(\boldsymbol{i}_{l+1})\right)\\
=& \frac{1}{\eta_{t_{l+1}}}\ln|\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))| \leq \sqrt{nt_{l+1}}.
\end{aligned}$$

Inequality ($E6$) is proved in Section A.3.

With the results in Equation (33), we have Equation (34).

In Equation (34), the equalities follow from the fact that

$$\begin{aligned}
&\sum_{t\in\mathcal{T}_l} \mathbb{E}[c'_{m_t,t}(\boldsymbol{I}_{m_t,t})|\boldsymbol{I}_{l,t} = \boldsymbol{i}^*_l] - \sum_{t\in\mathcal{T}_l} c'_{m_t,t}(\boldsymbol{i}^*_{m_t})\\
=& \sum_{t\in\mathcal{T}_{l+1}} \mathbb{E}[c'_{m_t,t}(\boldsymbol{I}_{m_t,t})|\boldsymbol{I}_{l,t} = \boldsymbol{i}^*_l] - \sum_{t\in\mathcal{T}_{l+1}} c'_{m_t,t}(\boldsymbol{i}^*_{m_t}).
\end{aligned}$$

□

Received February 2018; revised May 2018; accepted June 2018

$$
\begin{aligned}
&\sum_{t\in\mathcal{T}_{l+1}} \mathbb{E}[c'_{m_t,t}(I_{m_t,t})|I_{l,t}=\boldsymbol{i}_l] \\
&= \sum_{t\in\mathcal{T}_{l+1}} \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[I_{l+1,t}=\boldsymbol{i}_{l+1}|I_{l,t}=\boldsymbol{i}_l]\cdot\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{l+1,t}=\boldsymbol{i}_{l+1}\right] \\
&\overset{(E1)}{=} \sum_{t\in\mathcal{T}_{l+1}} \sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[I_{l+1,t}=\boldsymbol{i}_{l+1}|I_{l,t}=\boldsymbol{i}_l]\cdot\left[\bar{c}'_{l+1,t}(\boldsymbol{i}_{l+1})\cdot 2^{m_t-l}L_{m_t}+\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t})\right] \\
&= \sum_{t\in\mathcal{T}_{l+1}} \left\{\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \Pr[I_{l+1,t}=\boldsymbol{i}_{l+1}|I_{l,t}=\boldsymbol{i}_l]\cdot\bar{c}'_{l+1,t}(\boldsymbol{i}_{l+1})\cdot 2^{m_t-l}L_{m_t}+\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t})\right\} \\
&\overset{(E2)}{=} \sum_{t\in\mathcal{T}_{l+1}} \left\{\left[\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}'_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}'_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}\cdot\bar{c}'_{l+1,t}(\boldsymbol{i}_{l+1})\right]\cdot 2^{m_t-l}L_{m_t}+\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t})\right\} \\
&\overset{(E3)}{\le} \sum_{t\in\mathcal{T}_{l+1}} \left\{\left[-\frac{1}{\eta_t}\ln\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}'_{l+1,t-1}(\boldsymbol{i}_{l+1})\right)\cdot\exp\left(-\eta_t\bar{c}'_{l+1,t}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}'_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}+\frac{\eta_t}{2}\right]\cdot 2^{m_t-l}L_{m_t}+\min_{\boldsymbol{i}_m\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t})\right\} \\
&\overset{(E4)}{=} \sum_{t\in\mathcal{T}_{l+1}} \left[-\frac{1}{\eta_t}\ln\sum_{\boldsymbol{i}_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))} \frac{\exp\left(-\eta_t\bar{C}'_{l+1,t}(\boldsymbol{i}_{l+1})\right)}{\sum_{\boldsymbol{i}'_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))}\exp\left(-\eta_t\bar{C}'_{l+1,t-1}(\boldsymbol{i}'_{l+1})\right)}+\frac{\eta_t}{2}\right]\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&= \sum_{t\in\mathcal{T}_{l+1}}\left[\Phi_t(\eta_t)-\Phi_{t-1}(\eta_t)+\frac{\eta_t}{2}\right]\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&= \left\{\Phi_T(\eta_T)+\sum_{t=t_{l+1}}^{T-1}(\Phi_t(\eta_t)-\Phi_t(\eta_{t+1}))-\Phi_{t_{l+1}-1}(\eta_{t_{l+1}})+\sum_{t=t_{l+1}}^{T}\frac{\eta_t}{2}\right\}\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&\overset{(E5)}{\le} \left\{\Phi_T(\eta_T)+\ln 2\cdot n\left(\frac{1}{\eta_T}-\frac{1}{\eta_{t_{l+1}}}\right)-\Phi_{t_{l+1}-1}(\eta_{t_{l+1}})+\sum_{t=t_{l+1}}^{T}\frac{\eta_t}{2}\right\}\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&\le \left\{\Phi_T(\eta_T)+\sqrt{nT}+\sqrt{nt_{l+1}}+\sqrt{nT}\right\}\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&\overset{(E6)}{\le} \left\{\sum_{t\in\mathcal{T}_{l+1}}\mathbb{E}\left[\frac{c'_{m_t,t}(I_{m_t,t})-\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t})}{2^{m_t-l}L_{m_t}}|I_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+3\sqrt{nT}\right\}\cdot 2^{m_t-l}L_{m_t}+\sum_{t\in\mathcal{T}_{l+1}}\min_{\boldsymbol{i}_{m_t}\in\mathcal{E}_{m_t}(\mathcal{K}_l(\boldsymbol{i}_l))} c'_{m_t,t}(\boldsymbol{i}_{m_t}) \\
&\le \sum_{t\in\mathcal{T}_{l+1}}\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+3\sqrt{nT}\cdot 2^{m_t-l}L_{m_t} \\
&= \sum_{t\in\mathcal{T}_{l+1}}\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{l+1,t}=\boldsymbol{i}''_{l+1}\right]+3DLn\sqrt{T}\cdot\frac{1}{2^l},\qquad\forall\boldsymbol{i}''_{l+1}\in\mathcal{E}_{l+1}(\mathcal{K}_l(\boldsymbol{i}_l))
\end{aligned}
\tag{33}
$$

$$
\begin{aligned}
&\sum_{t\in\mathcal{T}}\mathbb{E}[c'_{m_t,t}(I_{m_t,t})]-\sum_{t\in\mathcal{T}}c'_{m_t,t}(\boldsymbol{i}^*_{m_t})=\sum_{t\in\mathcal{T}}\mathbb{E}[c'_{m_t,t}(I_{m_t,t})|I_{0,t}=\boldsymbol{i}^*_0]-\sum_{t\in\mathcal{T}}c'_{m_t,t}(\boldsymbol{i}^*_{m_t}) \\
&\overset{(E1)}{=}\sum_{t\in\mathcal{T}_1}\mathbb{E}[c'_{m_t,t}(I_{m_t,t})|I_{0,t}=\boldsymbol{i}^*_0]-\sum_{t\in\mathcal{T}_1}c'_{m_t,t}(\boldsymbol{i}^*_{m_t})\le\sum_{t\in\mathcal{T}_1}\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{1,t}=\boldsymbol{i}^*_1\right]-\sum_{t\in\mathcal{T}_1}c'_{m_t,t}(\boldsymbol{i}^*_{m_t})+3DLn\sqrt{T}\cdot\frac{1}{2^0} \\
&\overset{(E2)}{=}\sum_{t\in\mathcal{T}_2}\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{1,t}=\boldsymbol{i}^*_1\right]-\sum_{t\in\mathcal{T}_2}c'_{m_t,t}(\boldsymbol{i}^*_{m_t})+3DLn\sqrt{T}\cdot\frac{1}{2^0} \\
&\le\sum_{t\in\mathcal{T}_2}\mathbb{E}\left[c'_{m_t,t}(I_{m_t,t})|I_{1,t}=\boldsymbol{i}^*_2\right]-\sum_{t\in\mathcal{T}_2}c'_{m_t,t}(\boldsymbol{i}^*_{m_t})+3DLn\sqrt{T}\cdot\left(\frac{1}{2^0}+\frac{1}{2^1}\right) \\
&\le\cdots\le 3DLn\sqrt{T}\cdot\left(\frac{1}{2^0}+\frac{1}{2^1}+\cdots++\frac{1}{2^m}\right)\le 6DLn\sqrt{T}.
\end{aligned}
\tag{34}
$$