Decentralized Online Integer Programming Problems with a Coupling Cardinality Constraint

Ezgi Karabulut, Shabbir Ahmed, George Nemhauser

Abstract

We consider a problem involving a set of agents who need to coordinate their actions to optimize the sum of their objectives while satisfying a common resource constraint. The objective functions of the agents are unknown to them a priori and are revealed in an online manner. The resulting problem is an online optimization problem to optimally allocate the resource among the agents prior to observing the item values. For any deterministic online algorithm for this problem, it has been shown that there exists a lower bound of $\Omega(T)$ on regret. When the agents' integer programs satisfy a discrete concavity condition, we propose a randomized online algorithm that is decentralized and guarantees an upper bound of $O(\sqrt{T})$ on the expected regret.

1 Introduction

The easy accessibility of data, its dynamic characteristics and the need to protect it, together with developments in multi-processor computing has led to the emergence of decentralized and online optimization models and algorithms. We consider an online optimization problem involving a set of collaborative agents who need to coordinate their actions to optimize the sum of their individual objectives while sharing a common resource. The objective functions of the agents are unknown to them a priori and are revealed in an online manner over time. The only coupling constraint for the agents is a common resource constraint. Given a resource allocation, the agents' problems become separable, and each agent's action is determined by solving a discrete optimization problem. Due to privacy issues, agents want to share limited information while solving this resource allocation problem in a decentralized way. The goal is to optimally allocate the resource among the agents prior to observing the objective functions.

Such problems arise, for example, in digital platforms. One application is ad allocations on search pages, where the agents are the companies wanting to purchase ad slots, the common resource is the number of ad slots on the page, and the value of the ad slots are changing over time. Another example is allocating IP addresses to local authorities, where the demand, i.e. the value, of the resource changes over time.

2 Problem Definition

In the most general form, our problem can be formulated as:

$$z^t = \max \left\{ \sum_{i=1}^{m} z_i^t(K_i) : \sum_{i=1}^{m} K_i = K \right\} \tag{1}$$

The set $i = 1, \ldots, m$ corresponds to agents. K is the total amount of resource to be shared among m agents, and each agent is assigned K_i units of resource. The global objective is the sum of individual agents' objective functions, $z_i^t(K_i)$. As seen in (1), once the resource is allocated such that each agent is assigned K_i units, their problems become separable.

Our distributed decision making process approaches problem (1) from this perspective, and only tackles the problem of resource allocation without having detailed information on the agents' individual problems. Each agent's individual problem is allowed to have a different structure, with different sets of local constraints, however we assume they are all discrete optimization problems. Throughout this paper, we will focus on problems that satisfy the following discrete concavity conditions:

$$z_i^t(K_i+1) - z_i^t(K_i) \geq z_i^t(K_i+2) - z_i^t(K_i+1) \text{ for all } K_i \text{ integer} \tag{2}$$

$$z_i^t(K_i) = z_i^t(K_i+\epsilon) \text{ for any } K_i \text{ integer, and } \epsilon \in [0,1). \tag{3}$$

In other words, we assume that for each agent the marginal value of each additional unit resource is non-increasing. Examples of problems that satisfy (2)-(3) are given in [9], that consist of a linear objective function, a coupling cardinality constraint, and a set of local constraints for each agent including matching, matroid, and intersection of two matroids, and some transportation problems. It is shown in [9] that any $z_i^t(\cdot)$ that satisfies (2)-(3) can be expressed as a cardinality problem (4) with newly defined objective coefficients.

$$z_i^t(K_i) = \max\left\{(c_i^t)^T x_i : \mathbf{1}^T x_i \leq K_i \ , \ x_i \in \{0,1\}^{N_i}\right\}. \tag{4}$$

In this simplified problem formulation, each agent i has their individual discrete optimization problem (4), where x_i is an integer decision variable vector of size N_i, and c_i is the vector of objective function coefficients. The data in (4) is available only to agent i. In order to determine the resource allocation, each agent i is expected to share a subset of the information available to them. Furthermore, at every iteration the objective function coefficients, i.e. the values of the items, change, and the online problem is allocating the resource to agents prior to observing the item values. Once the resource allocation is fixed, the objective function coefficients are revealed, and the objective values are realized. Our goal is to determine a decentralized online optimization algorithm with provably good performance, that takes $\left\{z_i^1, \ldots, z_i^{t-1}\right\}_{i=1}^m$ as input, and outputs a resource allocation $\left\{K_i^t\right\}_{i=1}^m$. This research is a continuation of the work [9], where the offline version of this decentralized resource allocation problem in [9] is now extended to an online setting.

Bubeck [5] and Hazan [7] provide two extensive pieces of work that cover a wide range of topics in online convex optimization. There exist various efficient online convex optimization algorithms in the literature with good error bounds, such as online gradient descent and its generalization online mirror descent. These convex optimization algorithms assume a smooth transition over the decision variables throughout the iterations. However, the decision variables in our problem (1), i.e. the resource allocations, are discrete. Therefore we cannot make use of the online convex optimization algorithms directly to solve our problem.

The most elementary version of our problem is described in [10]. The algorithm Koolen et al. propose works on discrete online optimization problems where the decision variables are binary, i.e. $y \in \{0,1\}^m$, and the objective functions are linear, i.e. $z_i^t(y_i) = l_i^t y_i$. Their problem is more general in the sense that it admits any subset of the unit hypercube to be the set of feasible decisions.

Audibert et al. [1] consider the same problem setting, and generalize the algorithm in [10] to online stochastic mirror descent. They further provide analysis for the semi-bandit and bandit feedback.

Liu and Zhao [12] solve the multi-armed bandit problem with multiple agents in a distributed setting. This problem is defined as follows: There are $j = 1, \ldots, N$ arms and each arm j returns a reward θ_j^t at each iteration t. Each agent $i = 1, \ldots, m$ picks an arm to play at every iteration t, and receives the respective reward. When multiple agents select the same arm, the reward θ_j^t is shared among them. The main difference between this problem and ours is that we assume each agent's decision set is disjoint.

Braun and Pokutta [4] consider the problem where the decisions $x^t \in A \subseteq \mathbf{R}^n$ from a finite set yield a linear loss $(L^t)^T x^t$. They assume the linear loss functions L^t are output by adaptive adversaries, and provide an efficient algorithm against them.

Lattimore et al. [11] study a similar problem in a continuous setting. Their goal is to determine resource allocations at each time period as a continuous variable. Our problem differs from theirs both in terms of the variable type, i.e. discrete or continuous, the decentralization, and the observed data. Their parameters don't change at each iteration; however they cannot observe them directly, and their task is to estimate the unknown parameter.

Our work is along the lines of [10] and [1]. Our contribution in this paper is to generalize the binary problem to integer programming problems, and the linear objective function to step functions. More importantly, we focus on the decentralization aspect of the problem and provide a decentralized protocol with $\Theta(m)$ messages shared at every iteration.

In the next section, we introduce a regret function objective for our online problem and show that deterministic algorithms for minimizing regret do not perform well. Then we show that random algorithms can give better performance. Our main result in Section 4 is a decentralized randomized algorithm with good performance. We provide experimental results that illustrate the performance of our algorithm in Section 5. Throughout the paper, we assume individual problems of agents are in form (4), and carry out the analyses on this elementary problem. In Section 6, we present error bounds for approximation functions for our problem. We close with concluding remarks.

3 Deterministic and Randomized Algorithms

Consider an online optimization problem that is solved for T iterations. A generally accepted measure for the performance of an online optimization algorithm $\mathcal{A}$ is the regret function defined as

$$R_{\mathcal{A}}(T) = \sup_{z_i^t \in \mathcal{F}} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*) - \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^t) \right\} \tag{5}$$

where in our case $\mathcal{F}$ is the set of objective functions z_i^t that can be expressed as (4), and $K^* = \{K_i^*\}_{i=1}^m$ is the best fixed decision, i.e.

$$K^* = \arg\max \left\{ \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i) \ : \ \sum_{i=1}^{m} K_i = K, \quad K_i \in \mathbf{Z}_+ \ \forall i = 1, \dots, m \right\}. \tag{6}$$

In other words, regret measures the maximum difference between the value of the best static decision, $\sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*)$, and the value of the algorithm output, $\sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^t)$ over all possible objective functions in order to consider the worst-case scenario. Notice that the comparison is made with the best static decision, not the true optimal decision that is allowed to have different resource allocation at each iteration. The reason behind that is usually explained as fairness. Namely, the optimal solution with full information revealed prior to decision making is too advantageous compared to the decisions made prior to observing the data. Because K^* is not the optimal resource allocation independently for all t, the regret $R_{\mathcal{A}}(T)$ is not necessarily a nonnegative value.

An algorithm is accepted to have a good performance if the regret is bounded from above by a sublinear function of T, i.e. $R_{\mathcal{A}}(T)$ is $o(T)$ [7]. There are many online convex optimization algorithms in the literature that satisfy this regret bound. For instance the online gradient descent algorithm, and even its generalization the online mirror descent algorithm have upper bounds of $O(\sqrt{T})$ on regret in online convex optimization problems. If the problem satisfies some additional conditions, such as being α-strongly concave, we can get an upper bound of $O(\log T)$ on regret for the online gradient descent algorithm [7]. It has been shown that for our problem, the desired bound of $o(T)$ on $R_{\mathcal{A}}(T)$ cannot be obtained by a deterministic online algorithm.

Theorem 1. *For every deterministic online algorithm $\mathcal{A}$, there exists a sequence of objective functions $\left\{z_i^1, \ldots, z_i^T\right\}_{i=1}^m$ in the family $\mathcal{F}$, that guarantees $R_{\mathcal{A}}(T) \geq \Omega(T)$.*

Proof. See Theorem 4.1 in [13].

□

Theorem 1 implies that for problem (4), the desired bound $R_{\mathcal{A}}(T) \leq o(T)$ cannot be obtained by a deterministic online algorithm.

Given the objective functions $\left\{z_i^1, \ldots, z_i^{t-1}\right\}_{i=1}^m$ as input, at iteration t a randomized algorithm $\mathcal{A}$ outputs resource allocation $K^t \in \left\{\bar{K}^1, \ldots, \bar{K}^r\right\}$ with probabilities $\lambda_1^t, \ldots, \lambda_r^t$ respectively. Let this randomization be such that the expectation $\mathbf{E}\left[K^t\right]$ equals the vector ω^t, i.e. the probabilities $\lambda_1^t, \ldots, \lambda_r^t$ satisfy $\sum_{j=1}^r \lambda_j^t \bar{K}^j = \omega^t$. The randomized algorithm updates the expectation vector ω^t at every iteration.

The performance of a randomized online optimization algorithm is determined by the expected regret function defined as

$$R_{\mathcal{A}}^E(T) = \sup_{z_i^t \in \mathcal{F}} \left\{ \sum_{t=1}^T \sum_{i=1}^m z_i^t(K_i^*) - \sum_{t=1}^T \sum_{i=1}^m \mathbf{E}\left[z_i^t(K_i^t)\right] \right\} \tag{7}$$

where K^* is the best static decision described in (6).

The structure of the problem in (4) is such that $z_i^t(K_i)$ equals the sum of the highest K_i elements in the vector c_i^t for $K_i \in \mathbf{Z}_+$, and $z_i^t(K_i) = z_i^t(K_i + \epsilon)$ for $K_i \in \mathbf{Z}_+$ and $\epsilon \in [0, 1)$. It is a step-function as illustrated in Figure 1. Consider the upper envelope of z_i^t, denoted by $\tilde{z}_i^t$, in other words the linear relaxation of the problem (4). By the structure of our problem, $\tilde{z}_i^t(K_i) = z_i^t(K_i)$ for all $K_i \in \mathbf{Z}_+$.

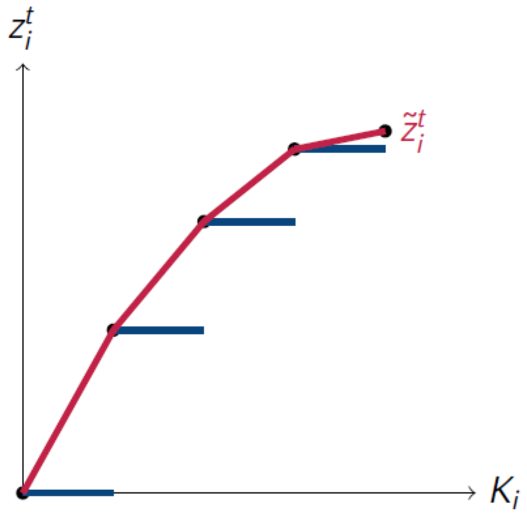


Figure 1: Functions in $\mathcal{F}$ and their upper envelopes

We construct the following online convex optimization problem over the continuous concave functions $\tilde{z}_i^t$ to be maximized

$$\tilde{R}_{\mathcal{A}}(T) = \sup_{\tilde{z}_i^t \in \tilde{\mathcal{F}}} \left\{ \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(\tilde{K}_i^*) - \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(\omega_i^t) \right\} \tag{8}$$

where the feasible decisions are $\omega^t \in \left\{\omega \in \mathbf{R}_+^m \ , \ \sum_{i=1}^m \omega_i = K\right\}$, $\tilde{\mathcal{F}}$ is the family of functions that can be described as the upper envelope of functions in $\mathcal{F}$, and $\left\{\tilde{K}_i^*\right\}_{i=1}^m$ is the best fixed decision in the continuous space, i.e.

$$\tilde{K}^* = \arg\max \left\{ \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(K_i) \ : \ \sum_{i=1}^m K_i = K, \quad K_i \geq 0 \ \forall i = 1, \ldots, m \right\}. \tag{9}$$

Theorem 2. *For a function z_i^t, and its upper envelope $\tilde{z}_i^t$, if $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$ holds, then for any online convex optimization algorithm $\mathcal{A}$, $R_{\mathcal{A}}^E(T) \leq \tilde{R}_{\mathcal{A}}(T)$.*

Proof. For any fixed function z_i^t and its upper envelope $\tilde{z}_i^t$, the following statements are true:

- $z_i^t(K_i^*) \leq \tilde{z}_i^t(K_i^*)$ ($\tilde{z}_i^t$ is the upper envelope of z_i^t)
- $\sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(K_i^*) \leq \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(\tilde{K}_i^*)$ ($\tilde{K}^*$ is the maximizer and K^* is a feasible solution of (9))
- $\tilde{z}_i^t(\omega_i^t) = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$ (the definition of $\mathbf{E}\left[K_i^t\right]$)
- $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$ (our assumption)

Hence we get

$$\sum_{t=1}^T \sum_{i=1}^m z_i^t(K_i^*) - \sum_{t=1}^T \sum_{i=1}^m \mathbf{E}\left[z_i^t(K_i^t)\right] \leq \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(\tilde{K}_i^*) - \sum_{t=1}^T \sum_{i=1}^m \tilde{z}_i^t(\omega_i^t). \tag{10}$$

□

The importance of Theorem 2 is that as long as the algorithm satisfies $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$, we can utilize any online convex optimization algorithm on the linear relaxations $\left\{\tilde{z}_i^1, \ldots, \tilde{z}_i^T\right\}_{i=1}^m$ over the expectation vectors ω^t, and the upper bound on the regret of the convex optimization algorithm holds as an upper bound on the expected regret $R_{\mathcal{A}}^E(T)$.

4 A Decentralized Randomized Online Algorithm

4.1 Framework

The result stated in Theorem 2 forms the basis for the algorithm that we propose. The remaining challanges are:

(a) to incorporate the decentralization into the online convex optimization algorithms,

(b) and to guarantee $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$.

The main framework of our algorithm is the *Online Stochastic Mirror Descent* in [1] using the update formulas in [10]. We describe this framework in Figure 2.

At every iteration, the expectation vector ω^t is updated in two steps. The intermediary point $\hat{\omega}^{t+1}$ is computed in Line 3 by optimizing the tradeoff between the relative entropy with respect to the previous iterate ω^t, $\sum_{i=1}^m \left(\omega_i \ln \frac{\omega_i}{\omega_i^t} + \omega_i^t - \omega_i\right)$, and a linear approximation of the objective function, $\sum_{i=1}^m \left(\omega_i \nabla \tilde{z}_i^t(\omega_i^t)\right)$. This is an unconstrained optimization problem and has a closed form solution, $\hat{\omega}_i^{t+1} = \omega_i^t \, e^{\eta \nabla \tilde{z}_i^t(\omega_i^t)}$. ω^t is a feasible solution, i.e. $\sum_{i=1}^m \omega_i^t = K$ and $\omega^t \geq 0$. With this update formula, $\hat{\omega}^{t+1}$ is not feasible, i.e. $\sum_{i=1}^m \hat{\omega}_i^{t+1} > K$, unless $\nabla \tilde{z}_i^t(\omega_i^t) = 0$ for all $i = 1, \ldots, m$. In Line 4, $\hat{\omega}^{t+1}$ is then projected back to the feasible solution polytope by minimizing the relative entropy with respect to $\hat{\omega}^{t+1}$. This problem also has a closed form solution, $\omega_i^{t+1} = \left(\frac{\hat{\omega}_i^{t+1}}{\sum_{j=1}^m \hat{\omega}_j^{t+1}}\right) K$. When $\eta = \sqrt{\frac{2 \ln m}{T}}$, the expected regret of this algorithm is upper bounded by $O(K\sqrt{T \ln m})$.

At iteration t:

1. Pick K^t randomly such that $\mathbf{E}\left[K_i^t\right] = \omega^t$ and $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$.
2. Observe the new objective function $\left\{z_i^t(\cdot)\right\}_{i=1}^m$
3. $\hat{\omega}^{t+1} = \arg\min_{\omega \in \mathbf{R}_+^m} \left\{\sum_{i=1}^m \left(\omega_i \ln \frac{\omega_i}{\omega_i^t} + \omega_i^t - \omega_i\right) - \eta \sum_{i=1}^m \left(w_i \nabla \tilde{z}_i^t(\omega_i^t)\right)\right\}$
   $\Rightarrow\ \hat{\omega}_i^{t+1} = \omega_i^t\ e^{\eta \nabla \tilde{z}_i^t(\omega_i^t)}$
4. $\omega^{t+1} = \arg\min_{\omega} \left\{\sum_{i=1}^m \left(\omega_i \ln \frac{\omega_i}{\hat{\omega}_i^{t+1}} + \hat{\omega}_i^{t+1} - \omega_i\right) : \sum_{i=1}^m \omega_i = K, \omega \geq 0\right\}$
   $\Rightarrow\ \omega_i^{t+1} = \left(\frac{\hat{\omega}_i^{t+1}}{\sum_{j=1}^m \hat{\omega}_j^{t+1}}\right) K$

Figure 2: The centralized framework.

The structure of the relative entropy function is very convenient for decentralization. As stated in Line 3 of Figure 2, the updates on $\hat{\omega}^t$ are separable over the agents. The updates on ω^t, on the other hand, require the sum $\sum_{j=1}^m \hat{\omega}_j^{t+1}$. Sharing this sum, denoted by ***ShareSum*** subroutine, can be achieved using at most $O(m)$ messages by protocols such as *Secure Sum Protocol* described in [6]. What remains is to select the resource allocation vector K^t randomly in a decentralized way, while satisfying $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$. Our proposed randomized decentralized algorithm is explained in the following section.

4.2 Our Proposed Algorithm

Figure 3 describes the decentralized version of Figure 2 from player i's perspective:

Agent i's routine at iteration t:

1. $K_i^t =$ ***Pick***K_i^t***Randomly( )***
2. Observe the new objective function $z_i^t(\cdot)$
3. $\hat{\omega}_i^{t+1} = \omega_i^t\ e^{\eta \nabla \tilde{z}_i^t(\omega_i^t)}$
4. $\sum_{j=1}^m \hat{\omega}_j^{t+1} =$ ***ShareSum***$(\hat{\omega}_i^{t+1})$
5. $\omega_i^{t+1} = \left(\frac{\hat{\omega}_i^{t+1}}{\sum_{j=1}^m \hat{\omega}_j^{t+1}}\right) K$
6. $\omega^{t+1} =$ ***ShareOmega***(ω_i^{t+1})

Figure 3: The decentralized algorithm.

Each iteration starts with agents coordinating to determine the resource allocation prior to observing their objective functions (Line 1), as it is in the nature of online optimization. Once the decision is fixed, each agent observes their own objective function (Line 2). Using the new objectives, the agents need to update their expected resource allocation vector ω (Lines 3-5). During this update, they need to communicate to jointly calculate $\sum_i \hat{\omega}_i$ (Line 4), which is then used in their individual calculation of ω_i, i.e. their expected resource in the next iteration. Finally, the expected resource vector ω becomes public (Line 6), to be used in the resource allocation in the upcoming iteration.

The main challenge while decentralizing the algorithm in Figure 2 to get the algorithm in Figure 3 is coordination while randomly picking the resource allocation vector K^t. The subroutine that we propose for the random selection of K^t, described in Figure 4, is motivated by the proof of Caratheodory Theorem explained in [2].

Let $\mathcal{K}^{\omega^t}$ be the set of feasible resource allocations defined as

$$\mathcal{K}^{\omega^t} = \left\{ \bar{K}^t \; : \; \bar{K}^t \in \mathbf{Z}_+^m \; , \; \sum_{i=1}^{m} \bar{K}_i^t = K \; , \; \bar{K}_i^t \in \left\{ \lfloor \omega_i^t \rfloor, \lceil \omega_i^t \rceil \right\} \quad \forall i = 1, \ldots, m \right\}. \tag{11}$$

In other words, $\mathcal{K}^{\omega^t}$ is the set of vertices of the unit hypercube around ω^t the sum of whose coordinates equals K, i.e. the feasible resource allocations. Clearly, ω^t lies within the convex hull of the points in $\mathcal{K}^{\omega^t}$. By the Caratheodory Theorem, there exists a subset $\mathcal{K}^t$ of $\mathcal{K}^{\omega^t}$ such that $|\mathcal{K}^t| = m$ and $\omega^t \in conv(\mathcal{K}^t)$. The subroutine described in Figure 4 simultaneously constructs the set $\mathcal{K}^t$ and randomly selects one element in the set with the probability of each element being the respective weight in the convex combination.

1: Set $K^t = \omega^t$
2: $f_i = K_i^t - \lfloor K_i^t \rfloor$ and $\delta = \sum_{i=1}^m f_i$
3: **if** $\delta = 0$ **then**
4:    **return** K_i^t
5: **end if**
6: $I^+ =$ indices of the first δ fractional components of K^t ,
   $I^- =$ indices of the remaining fractional components of K^t
7: $K_i^t = \lceil K_i^t \rceil \;\; \forall i \in I^+$, $K_i^t = \lfloor K_i^t \rfloor \;\; \forall i \in I^-$
8: $i^+ = \arg\min_{i \in I^+} \{f_i\}$
   $i^- = \arg\min_{i \in I^-} \{1 - f_i\}$
9: $\theta = \min\{f_{i^+}, 1 - f_{i^-}\}$,
   $i^* = i^+$ if $\theta = f_{i^+}$, $i^* = i^-$ if $\theta = 1 - f_{i^-}$
10: **if** $i = i^*$ **then**
11:    w.p. θ: ***Share***("Terminate"), w.p. $1 - \theta$: ***Share***("Continue")
12: **else**
13:    *Status* = ***Receive( )***
14: **end if**
15: **if** Status = Terminate **then**
16:    **return** K_i^t
17: **else if** Status = Continue **then**
18:    $K_i^t = \frac{K_i^t - \theta \lceil K_i^t \rceil}{1-\theta} \;\; \forall i \in I^+$, $K_i^t = \frac{K_i^t - \theta \lfloor K_i^t \rfloor}{1-\theta} \;\; \forall i \in I^-$
19: **end if**
20: **go to 2**

Figure 4: The ***Pick***K_i^t***Randomly*** subroutine

We illustrate the steps of the ***Pick***K_i^t***Randomly*** subroutine with the following example: Let $K^t = f = [0.3, 0.8, 0.9]$, and therefore $\delta = 0.3 + 0.8 + 0.9 = 2$. The set of indices of the first δ fractional components is $I^+ = \{1, 2\}$ and the set of remaining fractional indices is $I^- = \{3\}$. We round up f_1 and f_2 and round down f_3 to get the integer vector $[1, 1, 0]$ and set $K^t = [1, 1, 0]$. The goal is find the vector y such that $[0.3, 0.8, 0.9] = \theta[1, 1, 0] + (1 - \theta)y$. The constraint $y_i \geq 0$ is restrictive for $i \in I^+$, i.e. $\theta \leq f_i$, and we get $i^+ = 1$ as the agent with the tightest bound. The constraint $y_i \leq 1$ is restrictive for $i \in I^-$, i.e. $\theta \leq 1 - f_i$, and we get $i^- = 3$ as the agent with the tightest bound. The minimizer $\theta = 0.1$ and the agent with the tightest bound i^* is agent 3. At this point, agent 3 randomly selects to either terminate with probability $\theta = 0.1$, or to continue with probability $1 - \theta = 0.9$. If the subroutine terminates, then it returns the integer point $K^t = [1, 1, 0]$. Otherwise, the subroutine is repeated for $K^t = y = [2/9, 7/9, 1]$. Notice that if agent 3 chooses to continue, in the remaining iterations K_3^t is fixed to 1, and therefore agent 3 will no longer be a part of the random selection process and $K_3^t = 1$.

Theorem 3. *The* **PickK_i^tRandomly** *subroutine described in Figure 4 returns a point K^t such that $\mathbf{E}\left[K^t\right] = \omega^t$.*

Proof. The underlying idea in Figure 4 is along the lines of the proof of the Caratheodory theorem. Given a non-extreme point $y^0 \in conv(\mathcal{K}^{\omega^t})$, we want to find integer points $\bar{K}^1, \ldots, \bar{K}^m \in \mathcal{K}^{\omega^t}$, and weights $\lambda_1, \ldots, \lambda_m \in [0, 1]$ that satisfy $y^0 = \sum_{j=1}^m \lambda_j \bar{K}^j$ and $\sum_{j=1}^m \lambda_j = 1$, i.e. $y^0 \in conv(\{\bar{K}^j\}_{j=1}^m)$.

Let us start with the induction $y^0 = \theta^1 \bar{K}^1 + (1 - \theta^1) y^1$, where $\bar{K}^1 \in \mathcal{K}^{\omega^t}$ and $y^1 \in conv(\mathcal{K}^{\omega^t})$. In other words, let us fix one of the integer points, $\bar{K}^1$, and find the possibly fractional point y^1 on the other end of the line segment passing through y^0 and $\bar{K}^1$. $\mathcal{K}^{\omega^t}$ is an $m - 1$ dimensional polytope, and since $y^0 \in conv(\mathcal{K}^{\omega^t})$, it can be expressed as a convex combination of m points by Caratheodory Theorem, i.e. $y^0 \in conv(\{\bar{K}^j\}_{j=1}^m)$. Our final goal is to describe y^0 using m points, by being able to describe y^1 using $m - 1$ points. θ^1 needs to be selected such that the components of y^1, $y_i^1 = \frac{y_i^0 - \theta^1 \bar{K}_i^1}{1 - \theta^1}$, are within feasible bounds, in our case $\left[\lfloor \omega_i^t \rfloor, \lceil \omega_i^t \rceil\right]$ since we restrict ourselves to the unit hypercube around ω. Notice that when θ^1 is set to the maximum value it can take while y^1 remains feasible, at least one component of y^1, $y_{i^*}^1$, is fixed to one of the bounds. Hence $y^1 \in conv(\mathcal{K}^{\omega^t}) \cap \{y_{i^*}^1 = \lfloor \omega_{i^*}^t \rfloor\}$ or $y^1 \in conv(\mathcal{K}^{\omega^t}) \cap \{y_{i^*}^1 = \lceil \omega_{i^*}^t \rceil\}$, both $m - 2$ dimensional polytopes, and by Caratheodory Theorem y^1 can be described using $m - 1$ points. Next, we'll show that Lines 7 and 8 compute this maximum feasible value for θ^r.

The algorithm starts with $y^0 = \omega^t$ and uses the recursion $y^{r-1} = \theta^r \bar{K}^r + (1 - \theta^r) y^r$, where $y^r \in conv(\{\bar{K}^j\}_{j=r+1}^m)$. At each iteration $\bar{K}^r$ is defined such that the first δ fractional indices of y^{r-1} are rounded up and the remaining are rounded down, where δ is described as in Line 2. Since $y^{r-1} \in \left[\lfloor \omega_i^t \rfloor, \lceil \omega_i^t \rceil\right]$, for all fractional components of y^{r-1}, $\lfloor \omega_i^t \rfloor = \lfloor y_i^{r-1} \rfloor$ and $\lceil \omega_i^t \rceil = \lceil y_i^{r-1} \rceil$. The feasibility condition $y_i^r = \frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1 - \theta^r} \leq \lceil y_i^{r-1} \rceil$ and $y_i^r = \frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1 - \theta^r} \geq \lfloor y_i^{r-1} \rfloor$ only apply to the fractional components of y^{r-1}, since ***PickK_i^tRandomly*** subroutine leaves all integer components untouched. For all $i \in I^+$, the feasibility conditions can be reduced to the following:

$$
\begin{aligned}
\frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1 - \theta^r} &\leq \lceil \omega_i^t \rceil && (12) \\
\frac{y_i^{r-1} - \theta^r \lceil y_i^{r-1} \rceil}{1 - \theta^r} &\leq \lceil y_i^{r-1} \rceil && (13) \\
y_i^{r-1} - \theta^r \lceil y_i^{r-1} \rceil &\leq \lceil y_i^{r-1} \rceil - \theta^r \lceil y_i^{r-1} \rceil && (14) \\
y_i^{r-1} &\leq \lceil y_i^{r-1} \rceil && (15)
\end{aligned}
$$

and

$$\frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1-\theta^r} \geq \lfloor \omega_i^t \rfloor \quad (16)$$
$$\frac{y_i^{r-1} - \theta^r \lceil y_i^{r-1} \rceil}{1-\theta^r} \geq \lfloor y_i^{r-1} \rfloor \quad (17)$$
$$y_i^{r-1} - \theta^r \lceil y_i^{r-1} \rceil \geq \lfloor y_i^{r-1} \rfloor - \theta^r \lfloor y_i^{r-1} \rfloor \quad (18)$$
$$y_i^{r-1} - \lfloor y_i^{r-1} \rfloor \geq \theta^r (\lceil y_i^{r-1} \rceil - \lfloor y_i^{r-1} \rfloor) \quad (19)$$
$$f_i \geq \theta^r \quad (20)$$

(12) to (13) and (16) to (17) are merely substitutions for $\lceil \omega_i^t \rceil = \lceil y_i^{r-1} \rceil$ and $\lfloor \omega_i^t \rfloor = \lfloor y_i^{r-1} \rfloor$ as discussed previously, and $\bar{K}_i^r = \lceil y_i^{r-1} \rceil$ since for $i \in I^+$ the integer value is obtained by rounding up. As seen in (15), for $i \in I^+$, the upper bound constraint is satisfied trivially, and the only necessary constraint on θ^r is (20). For all $i \in I^-$, the feasibility conditions can be reduced to the following:

$$\frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1-\theta^r} \geq \lfloor \omega_i^t \rfloor \quad (21)$$
$$\frac{y_i^{r-1} - \theta^r \lfloor y_i^{r-1} \rfloor}{1-\theta^r} \geq \lfloor y_i^{r-1} \rfloor \quad (22)$$
$$y_i^{r-1} - \theta^r \lfloor y_i^{r-1} \rfloor \geq \lfloor y_i^{r-1} \rfloor - \theta^r \lfloor y_i^{r-1} \rfloor \quad (23)$$
$$y_i^{r-1} \geq \lfloor y_i^{r-1} \rfloor \quad (24)$$

and

$$\frac{y_i^{r-1} - \theta^r \bar{K}_i^r}{1-\theta^r} \leq \lceil \omega_i^t \rceil \quad (25)$$
$$\frac{y_i^{r-1} - \theta^r \lfloor y_i^{r-1} \rfloor}{1-\theta^r} \leq \lceil y_i^{r-1} \rceil \quad (26)$$
$$y_i^{r-1} - \theta^r \lfloor y_i^{r-1} \rfloor \leq \lceil y_i^{r-1} \rceil - \theta^r \lceil y_i^{r-1} \rceil \quad (27)$$
$$\theta^r (\lceil y_i^{r-1} \rceil - \lfloor y_i^{r-1} \rfloor) \leq \lceil y_i^{r-1} \rceil - y_i^{r-1} \quad (28)$$
$$\theta^r \leq 1 - f_i \quad (29)$$

Similarly, (25) to (26) and (21) to (22) are merely substitutions for $\lceil \omega_i^t \rceil = \lceil y_i^{r-1} \rceil$ and $\lfloor \omega_i^t \rfloor = \lfloor y_i^{r-1} \rfloor$ as discussed previously, and $\bar{K}_i^r = \lfloor y_i^{r-1} \rfloor$ since for $i \in I^-$ the integer value is obtained by rounding down. As seen in (24), for $i \in I^-$, the lower bound constraint is satisfied trivially, and the only necessary constraint on θ^r is (29). Line 8 in Figure 4 finds i^+ that has the tightest bound (20), and i^- that has the tightest bound (29), and Line 9 finds the tightest of both, hence setting θ^r to the maximum value possible.

The algorithm uses the recursion $y^{r-1} = \theta^r \bar{K}^r + (1-\theta^r) y^r$. Plugging the values in this recursion, we get $\omega^t = y^0 = \sum_{j=1}^m \lambda_j \bar{K}^j$, where $\lambda_j = \theta^j \prod_{r=1}^{j-1} (1-\theta^r)$ for $j = 1, \ldots, m-1$ and $\lambda_m = \prod_{r=1}^{m-1} (1-\theta^r)$. The ***Pick$K_i^t$Randomly*** subroutine does not necessarily compute all candidates $\bar{K}^1, \ldots, \bar{K}^m$, it concurrently carries out the process of construction of those points and the random selection. For $r = 1, \ldots, m-1$ it selects a point $\bar{K}^r$ right after its construction with probability λ^r. Namely, with probability $\prod_{j=1}^{r-1} (1-\theta^j)$ the previous $r-1$ points have not been selected, i.e. Status = Continue, and after that with probability θ^r $\bar{K}^r$ is selected, i.e. Status = Terminate. Hence we get $\lambda_r = \theta^r \prod_{j=1}^{r-1} (1-\theta^j)$. If none of the first $m-1$ points are selected, i.e. with probability $\lambda_m = \prod_{r=1}^{m-1} (1-\theta^r)$, then $\bar{K}^m$ is selected. With these probabilities and coefficients, we get:

$$\mathbf{E}\left[K^t\right] = \sum_{j=1}^{m} \lambda_j \bar{K}^j = \omega^t \tag{30}$$

□

Theorem 4. *The* **Pick**K_i^t**Randomly** *subroutine described in Figure 4 applied to problems with* $z_i^t \in \mathcal{F}$ *guarantees* $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$.

Proof. Because of the structure of $z_i^t \in \mathcal{F}$ and the concave nature of $\tilde{z}_i^t \in \tilde{\mathcal{F}}$ described in Figure 1, $\mathbf{E}\left[z_i^t(K_i^t)\right] \leq \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$ trivially holds for any $K_i \in [0, K]$ by Jensen's Inequality. Within the unit hypercube around ω^t described in (11), the functions $\tilde{z}_i^t$ behave as linear functions rather than piece-wise linear functions. For any $z_i^t \in \mathcal{F}$ and it's upper envelope $\tilde{z}_i^t$, $z_i^t(K_i) = \tilde{z}_i^t(K_i)$ for all $K_i \in \mathbf{Z}_+$ as shown in Figure 1, therefore $\mathbf{E}\left[K_i^t\right] = \omega^t$ implies $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$ for $K_i^t \in \left\{\lfloor\omega_i^t\rfloor, \lceil\omega_i^t\rceil\right\}$. Theorem 3 states that ***Pick***K_i^t***Randomly*** satisfies $\mathbf{E}\left[K_i^t\right] = \omega^t$, thus ***Pick***$K_i^t$***Randomly*** with $z_i^t \in \mathcal{F}$ guarantees $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$.

□

Theorem 5. *The* **Pick**K_i^t**Randomly** *subroutine described in Figure 4 applied to problems with* z_i^t *that satisfies (2) and (3) guarantees* $\mathbf{E}\left[z_i^t(K_i^t)\right] = \tilde{z}_i^t\left(\mathbf{E}\left[K_i^t\right]\right)$.

Proof. It has been shown in [9] that any problem that satisfies (2) and (3) can be expressed as a problem in $\mathcal{F}$ by the following argument: Let z_i^t satisfy (2) and (3), construct the following problem:

$$\bar{z}_i^t(K_i^t) = \max\left\{(\bar{w}_i^t)^T y_i \ : \ \mathbf{1}^T y_i \leq K_i^t, \ y_i \in \{0,1\}^{N_i}\right\} \tag{31}$$

where $N_i = K$ for all players $i = 1 \ldots m$, and the weights $\bar{w}_{ij}^t = z_i^t(j) - z_i^t(j-1)$ for $j = 1 \ldots K$. In other words, the marginal changes in the objective function value caused by the j^{th} unit of resource, i.e. $z_i^t(j) - z_i^t(j-1)$, equals the value of the j^{th} item in (31). Clearly $\bar{z}_i^t \in \mathcal{F}$. (2) and (3) imply that $z_i^t(K_i^t) = \bar{z}_i^t(K_i^t)$ for all $K_i^t \geq 0$, as thoroughly explained in [9]. Therefore, the concave upper envelope of the two functions, $\tilde{z}_i^t(K_i^t)$ and $\tilde{\bar{z}}_i^t(K_i^t)$ are identical. Notice the arguments for the proof of Theorem 4 holds for $\bar{z}_i^t$, and only use the values of the functions $\bar{z}_i^t$ and $\widetilde{\bar{z}}_i^t$. Because they are interchangeable with z_i^t and $\tilde{z}_i^t$ respectively, the arguments provided for the proof of Theorem 4 hold for any z_i^t that satisfies (2) and (3).

□

Overall, the communication complexity of the decentralized algorithm in Figure 3 is as follows: In Line 1 the agents need to communicate during the ***Pick***K_i^t***Randomly*** subroutine, which terminates in at most $m-1$ iterations, and at each iteration either a "Terminate" or "Continue" message is sent from a single agent to all remaining agents. This type of communication is called "*Single node accumulation*", and the communication complexity is given in Table 1. In line 4, algorithms such as *Secure Sum Protocol* [6] can share the sum in at most $O(m)$. Finally, in order to coordinate the selection of K^t, the updates on ω^{t+1} need to be shared among all agents. Communication complexity of this protocol, *multinode broadcast*, is provided in Table 1. With a maximum of $m-1$ Single node accumulations for ***Pick***K_i^t***Randomly***, the number of messages required per iteration of the decentralized algorithm in Figure 3 is upperbounded by $O(m^2)$.

It is also worth noting that the exact value of ω^t is not necessary for Figure 3. Using the fractional vector $f^t = \omega^t - \lfloor\omega^t\rfloor$ during the computations and finally shifting the resulting binary vector by $\lfloor\omega^t\rfloor$ yields the same result. This approach doesn't reduce the communication complexity, however it is advantageous in terms of privacy of data.

Table 1: Solution times of optimal algorithms for the basic communication problems using a ring, a binary balanced tree, and a d-dimensional symmetric mesh with m processors [3]

	Ring	Tree	Mesh
Single node accumulation	$\Theta(m)$	$\Theta(\log m)$	$\Theta(m^{1/d})$
Multinode broadcast	$\Theta(m)$	$\Theta(m)$	$\Theta(m)$

This concludes the reduction of our problem to an online convex optimization problem, from which we get a sublinear regret. The proof of sublinear regret stated in the following theorem, along with the selection of η follows [10].

Theorem 6. *When* $\eta = \sqrt{\frac{2\ln m}{T}}$ *and* $z_i^t \in \mathcal{F}$ *such that* $c_{ij}^t \in [0,1]$*, the expected regret of the decentralized algorithm described in Figure 2 is upper bounded by* $O(K\sqrt{T\ln m})$.

Proof. Let K^* be the best static decision described in (6), and $\Delta(x||y) = \sum_{i=1}^m x_i \ln \frac{x_i}{y_i} + y_i - x_i$ be the relative entropy function. Recall that the upper envelope of the functions $z_i^t \in \mathcal{F}$, i.e. $\tilde{z}_i^t$, are piecewise linear and concave, and therefore can be expressed as the minimum of K linear functions. $\tilde{z}_i^t(K_i) = \min_{\{j=1,\dots K\}} \left\{\beta_{ij}^t + c_{ij}^t K_i\right\}$, where $\beta_{ij}^t = \sum_{k=1}^j (c_{ik}^t - c_{ij}^t)$. Let $\beta_i^{*t} + c_i^{*t} K_i^*$ denote the line that is the minimizer for $K_i = \omega_i^{t-1}$. Recall once again that $z_i^t(K_i) = \tilde{z}_i^t(K_i)$ for integer K_i values. Consider the chain of inequalities:

$$\Delta(K^*\|\omega^t) - \Delta(K^*\|\omega^{t-1}) + \eta\left(\sum_{i=1}^m \min_{\{j=1,\dots K\}} \{\beta_{ij}^t + c_{ij}^t K_i^*\}\right) \tag{32}$$

$$\leq \Delta(K^*\|\hat{\omega}^t) - \Delta(K^*\|\omega^{t-1}) + \eta\left(\sum_{i=1}^m \beta_i^{*t} + c_i^{*t} K_i^*\right) \tag{33}$$

$$= \sum_{i=1}^m \omega_i^{t-1}(e^{\eta c_i^{*t}} - 1) + \eta\beta_i^{*t} \tag{34}$$

$$\leq \sum_{i=1}^m \omega_i^{t-1} c_i^{*t}(e^{\eta} - 1) + \eta\beta_i^{*t} \tag{35}$$

$$\leq (e^{\eta} - 1)\left(\sum_{i=1}^m c_i^{*t}\omega_i^{t-1} + \beta_i^{*t}\right). \tag{36}$$

For iteration t, (32) denotes the difference of the relative entropies of the best static decision K^* with respect to ω^t and ω^{t-1}, plus η times the value of the best static decision at iteration t, i.e. $\sum_{i=1}^m z_i^t(K_i^*)$. Among the linear functions, $\beta_i^{*t} + c_i^{*t} K_i^*$ is an upperbound on the minimizer and we get (33). Expanding the Δ functions gives (34). Due to our assumption $c_i^{*t} \in [0,1]$, (35) is an upper bound on (34). We can further bound this term by (36). Notice that the term in the summation in (36) is equal to $\tilde{z}_i^t(\mathbf{E}[K_i^t])$, since the line $\beta_i^{*t} + c_i^{*t} K_i^*$ is defined to be the minimizer at $K_i = \omega_i^{t-1}$. By the property of our algorithm, this is equal to $\mathbf{E}[z_i^t(K_i^t)]$. Hence, for iteration t, we have

$$\Delta(K^*\|\omega^t) - \Delta(K^*\|\omega^{t-1}) + \eta\sum_{i=1}^m z_i^t(K_i^*) \leq (e^{\eta} - 1)\sum_{i=1}^m \mathbf{E}[z_i^t(K_i^t)]. \tag{37}$$

Summing (37) over $t = 1, \dots, T$ gives

$$\Delta(K^*\|\omega^T) - \Delta(K^*\|\omega^0) + \eta\sum_{t=1}^T\sum_{i=1}^m z_i^t(K_i^*) \leq (e^{\eta} - 1)\sum_{t=1}^T\sum_{i=1}^m \mathbf{E}[z_i^t(K_i^t)]. \tag{38}$$

By rearranging (38) and plugging in $\eta = \sqrt{2(\ln m)/T}$, we get the desired bound.

$$\begin{aligned} R_{\mathcal{A}}^{E}(T) &\leq \frac{\Delta(K^* \| \omega^0)}{e^{\eta}-1} + \left(1 - \frac{\eta}{e^{\eta}-1}\right) \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*) \\ &\leq K\sqrt{2T \ln m} \end{aligned}$$

□

5 Experimental Results

The $K\sqrt{2T \ln m}$ upper bound on the expected regret is theoretically shown to be correct, however a natural question to ask is its tightness on average. In order to answer this question, we conduct experiments using the settings explained in the following subsections. These experiments also provide additional insight about how the expected regret function behaves throughout time within the known upper bound. Furthermore, we provide the realized regret values alongside the expected regret values to observe discrepancies both in values and in function behaviors.

For every experiment, we report three measures:

1. ω^t vector: How the expectation vector, $\mathbf{E}\left[(K^t)\right]$ changes over time.
2. Expected Regret: Using the K^* definition in (6), for every time period $t = 1, \ldots, T$, we report the following as the expected regret:

$$\sum_{t'=1}^{t} \sum_{i=1}^{m} z_i^{t'}(K_i^*) - \sum_{t'=1}^{t} \sum_{i=1}^{m} \mathbf{E}\left[z_i^{t'}(K_i^{t'})\right].$$

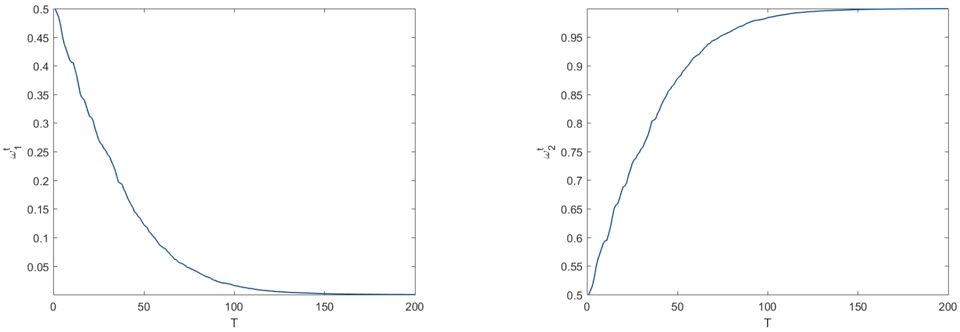
3. Realized Regret: Using the K^* definition in (6), for every time period $t = 1, \ldots, T$, we report the following as the realized regret:

$$\sum_{t'=1}^{t} \sum_{i=1}^{m} z_i^{t'}(K_i^*) - \sum_{t'=1}^{t} \sum_{i=1}^{m} z_i^{t'}(K_i^{t'}).$$

5.1 Experimental Setting 1

We consider a setting with two agents, $m = 2$, and a single unit of resource $K = 1$. The objective functions are generated such that agent 1 always gets the invaluable item, $c_1^t \sim U(0, 0.5)$, and agent 2 gets the valuable item, $c_2^t \sim U(0.5, 1)$. We observe this online problem for $T = 200$ iterations. Clearly, the best static decision K^* is $(0, 1)$, allocating the resource to agent 2.

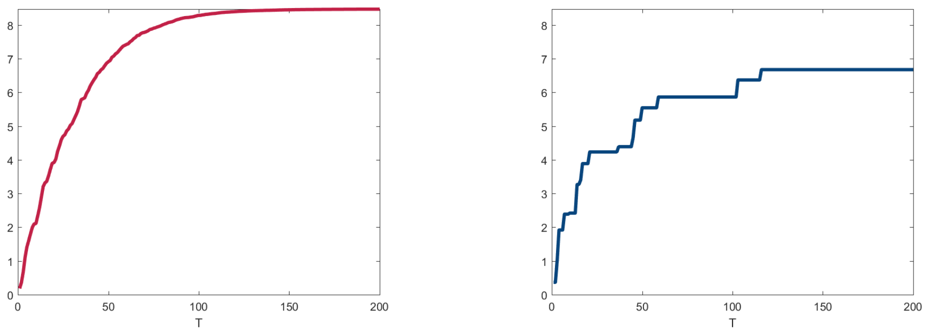
As shown in Figure 5, the expectation vector ω^t steadily converges to K^*. In addition to this convergence, we also observe a $\sqrt{T}$ shape in the expected regret over time, in Figure 6a. Notice that the theoretical upper bound on the expected regret $K\sqrt{2T \ln m}$ is 11.77, whereas the expected regret converges to 8.67. As ω^t converges to K^*, the probability of selecting the best static decision as the resource allocation, i.e. $K^t = K^*$, converges to 1. We observe this in Figure 6b, where for long periods of time the expected regret stays constant when $K^t = K^*$, but sometimes when the algorithm chooses the other resource allocation due to randomness a jump in the realized regret value occurs.



(a) Agent 1

(b) Agent 2

Figure 5: ω^t versus time



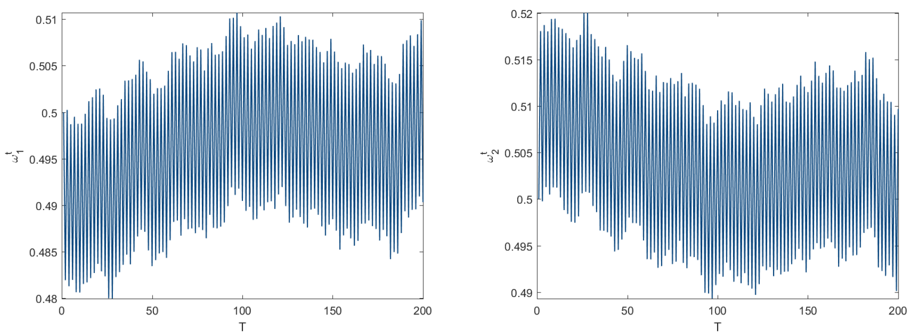
(a) Expected Regret

(b) Realized Regret

Figure 6: Expected and realized regret versus time

5.2 Experimental Setting 2

We again consider a setting with two agents, $m = 2$, and a single unit of resource $K = 1$. We construct the second experimental setting such that the two static decisions $(1, 0)$ and $(0, 1)$ do not have a significant dominance over each other. Namely, for t odd, agent 1 gets the valuable item, $c_1^t \sim U(0.9, 1)$, and agent 2 gets the invaluable item, $c_2^t \sim U(0, 0.1)$, and for t even, agent 2 gets the valuable item, $c_2^t \sim U(0.9, 1)$, and agent 1 gets the invaluable item, $c_1^t \sim U(0, 0.1)$. We observe this online problem for $T = 200$ iterations.



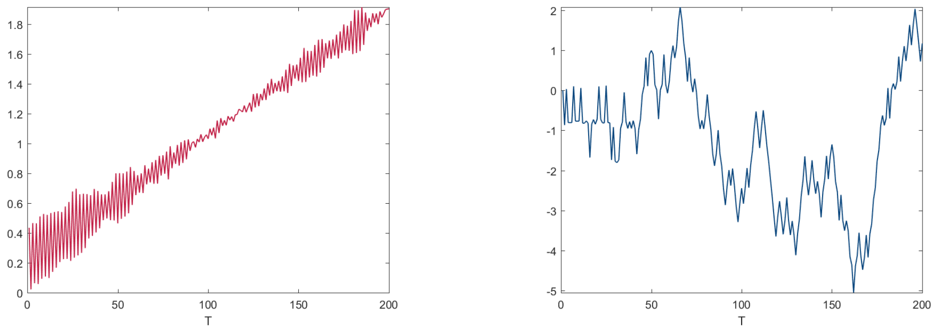
(a) ω^t (Agent 1)

(b) ω^t (Agent 2)

Figure 7: ω^t versus time

As shown in Figure 7, the expectation vector ω^t oscillates around $(0.5, 0.5)$. This oscillation causes a linear trend in the expected regret, which can be observed in Figure 8a.

A natural question to ask is whether this linear increase over time will eventually exceed the $O(\sqrt{T})$ bound as T increases. The answer is that it will not, by the following argument: The slope of the expected regret, which is a function of the length of the oscillation interval around $(0.5, 0.5)$ depends on the step-length parameter η. Recall that the definition $\eta = \sqrt{\frac{2 \ln m}{T}}$ includes the termination time for the online algorithm. Hence, if we mean to carry on for a longer period of time $T' > T$, η will be a smaller value, hence the length of the oscillation interval and the slope of the linear trend in expected regret will decrease accordingly, keeping the final value of the expected regret below the now $O(\sqrt{T'})$ threshold. Notice that the upper bound on the expected regret is the same as the experimental setting 1, since m, K, and T values are constant; however for this setting it is not a tight bound.



(a) Expected Regret (b) Realized Regret

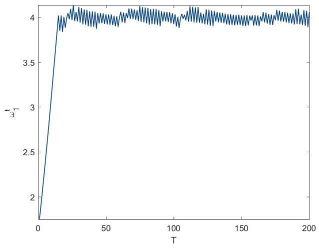
Figure 8: Expected and realized regret versus time

We observe an interesting phenomenon in Figure 8b when we look at the realized regret. Recall that the agent with the valuable item alternates at every iteration. So of the two possible candidates for the best static decision, each of them receives the valuable item at exactly $T/2$ iterations. The expectation vector ω^t, however, assigns almost equal probabilities to each agent. This means that in reality, due to this randomness, there may be periods of time when the random resource allocation indeed matches the assignment of the valuable item, hence being more successful than the best static decision. This appears in Figure 8b as regions when the regret value decreases, and sometimes even falls below 0.

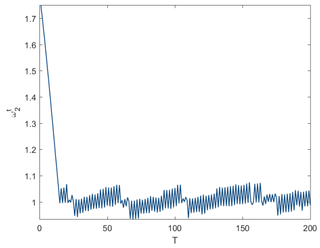
5.3 Experimental Setting 3

So far we have observed a setting where best static decision dominates other resource allocations, in which case the expectation vector ω^t converges to the best static decision (experimental setting 1), and a setting where the two decisions are not significantly different, in which case the expectation vector ω^t oscillates around the center (experimental setting 2). We construct the third setting such that there exists a best static decision that dominates other resource allocations, but which is not an extreme point of the feasible resource allocations polytope. We consider a setting with four agents, $m = 4$, and seven units of resource $K = 7$. The value of the items are such that at every iteration, agents 1, 2, 3, and 4 receive 4, 1, 1, and 1 valuable items respectively, i.e. $c_{ij}^t \sim U(0.9, 1)$. For the remaining items we have $c_{ij}^t \sim U(0, 0.1)$. The number of valuable items equals the total amount of resource, hence we have $K^* = (4, 1, 1, 1)$. We observe this online problem for $T = 200$ iterations.

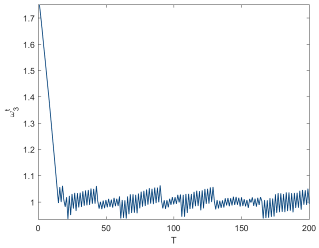
Figure 9 shows that once ω^t reaches a certain neighborhood of K^*, it starts oscillating around it, and the length of that neighborhood depends on T. More specifically, the algorithm reaches $\omega^t = (4-\epsilon_1, 1+\epsilon_2, 1+\epsilon_3, 1+\epsilon_4)$, and the objective coefficients force ω_1^t to increase and the remaining to decrease. Hence at iteration $t+1$ we get $\omega^{t+1} = (4+\epsilon‘_1, 1-\epsilon‘_2, 1-\epsilon‘_3, 1-\epsilon‘_4)$. Now the objective coefficients force ω_1^{t+1} to decrease and the remaining to increase, pushing



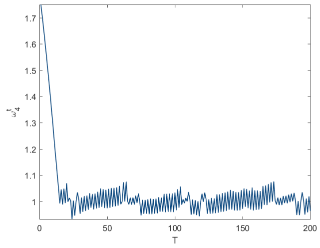
(a) ω^t (Agent 1)



(b) ω^t (Agent 2)



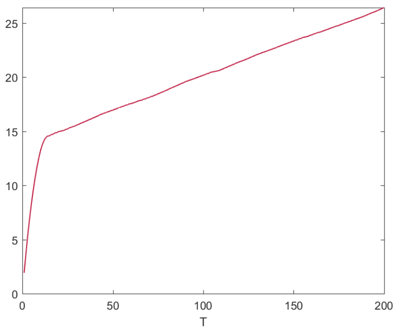
(c) ω^t (Agent 3)



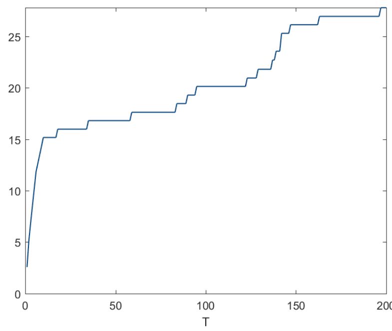
(d) ω^t (Agent 4)

Figure 9: ω^t versus time

ω_1^{t+2} slightly below 4, and the remaining slightly above 1. Because it is impossible to get $\omega^t = (4, 1, 1, 1)$, this oscillation continues.



(a) Expected Regret



(b) Realized Regret

Figure 10: Expected and realized regret versus time

We again observe the effect of the oscillation as a linear trend in the expected regret in Figure 10a. The previous argument about the linear trend exceeding the $\sqrt{T}$ upper bound still holds. With this parameter set, the value of the upper bound $K\sqrt{2T \ln m}$ is 116.56, which is a very loose bound for this parameter set. In the realized regret shown in Figure 10b, we observe a similar trend to the realized regret in experimental setting 1. Namely, ω^t gets arbitrarily close to K^*, meaning with high probability the algorithm will output $K^t = K^*$, in which case the realized regret value stays constant, and occasionally when K^t deviates from K^* due to randomness, we observe a sharp increase in the realized regret value.

6 Generalization of the Problem Structure

In the cases where the objective functions do not have the discrete concave structure, i.e. do not satisfy conditions (2)-(3), or are difficult to compute exactly, the agents may choose to use approximation functions instead. Assume we have a series of approximation functions

$\bar{z}_i^t$ that satisfy (2) and (3), i.e. $\bar{z}_i^t \in \mathcal{F}$, that can be bounded from above and below by a multiplicative scalar of the true objective function z_i^t as:

$$\rho_i^L \bar{z}_i^t(K_i) \leq z_i^t(K_i) \leq \rho_i^U \bar{z}_i^t(K_i). \tag{39}$$

For analyzing the approximation functions, we will use α-regret which is a notion of performance used in settings involving approximation functions [8]. In this line of thought, we define expected α-regret in the following way:

$$R_{\mathcal{A}}^{\alpha E}(T) = \sup_{z_i^t \in \mathcal{F}} \left\{ \alpha \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*) - \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[z_i^t(K_i^t)\right] \right\}. \tag{40}$$

For any online convex optimization algorithm, the upper bound on expected regret with respect to $\bar{z}_i^t$ that holds by Theorem 2 is a valid upper bound on expected α-regret with respect to z_i^t, where $\alpha = \frac{\rho_{min}^L}{\rho_{max}^U}$ defined by $\rho_{min}^L = \min_i \rho_i^L$ and $\rho_{max}^U = \max_i \rho_i^U$,. This holds by the following argument:

$$\begin{aligned}
o(T) &\geq \sum_{t=1}^{T} \sum_{i=1}^{m} \bar{z}_i^t(\bar{K}_i^*) - \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[\bar{z}_i^t(K_i^t)\right] && (41)\\
&\geq \sum_{t=1}^{T} \sum_{i=1}^{m} \bar{z}_i^t(K_i^*) - \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[\bar{z}_i^t(K_i^t)\right] && (42)\\
&\geq \sum_{t=1}^{T} \sum_{i=1}^{m} \frac{z_i^t(K_i^*)}{\rho_i^U} - \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[\frac{z_i^t(K_i^t)}{\rho_i^L}\right] && (43)\\
&\geq \frac{1}{\rho_{max}^U} \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*) - \frac{1}{\rho_{min}^L} \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[z_i^t(K_i^t)\right] && (44)\\
&= \frac{1}{\rho_{min}^L} \left(\frac{\rho_{min}^L}{\rho_{max}^U} \sum_{t=1}^{T} \sum_{i=1}^{m} z_i^t(K_i^*) - \sum_{t=1}^{T} \sum_{i=1}^{m} \mathbf{E}\left[z_i^t(K_i^t)\right] \right) && (45)
\end{aligned}$$

where K^* is the best static decision defined in (6), and $\bar{K}^*$ is defined as

$$\bar{K}^* = \arg\max \left\{ \sum_{t=1}^{T} \sum_{i=1}^{m} \bar{z}_i^t(K_i) \ : \ \sum_{i=1}^{m} K_i = K, \quad K_i \geq 0 \ \forall i = 1, \ldots, m \right\}. \tag{46}$$

(41) states that since $\bar{z}_i^t \in \mathcal{F}$, the expected regret of the selected online convex optimization algorithm is upper bounded by a sublinear function of T, by Theorem 2. We replace the maximizer $\bar{K}^*$ with K^* to get (42). Using the definition in (39), we lower bound this term by (43). By replacing the ρ_i^U with ρ_{max}^U and ρ_i^L with ρ_{min}^L we get (44) and rearranging the terms gives us the desired expected α-regret where $\alpha = \frac{\rho_{min}^L}{\rho_{max}^U}$.

This result can be utilized in the following way: assume that the true objective function value for our problem at every iteration is difficult to compute, or does not satisfy (2)-(3), however there exists a concave approximation algorithm to solve our problem. In such cases, by using an online convex optimization algorithm on the approximation function, we can get an upper bound of $o(T)$ on the expected α-regret. Examples of such concave approximations functions with their associated ρ_i^L and ρ_i^U coefficients are discussed in [9], and include the greedy heuristic and linear relaxation for some problems.

7 Conclusions

Online optimization problems involving multiple agents with the natural tendency of data privacy is a trending area. In this paper, we focused on the problem of resource allocation in a decentralized online setting. The results on the performance of deterministic online algorithms shifted the direction of our research to randomized algorithms, where by using linear relaxations we reduced our problem to an online convex optimization problem. With this observation, we derived a decentralized randomized algorithm that has an expected regret bound of $O(\sqrt{T})$. This results of this algorithm can be extended to any problem that has a discrete concave structure. Our results indicate that for any distributed integer programming problem where utility of each additional unit resource for the individual agents are non-increasing, we can obtain the state-of-the-art regret bounds achieved by online convex optimization algorithms. For problems that don't have the discrete concave structure, further experimental analysis can be carried out to analyze the deviation in realized regret from sublinear bounds.

8 Acknowledgements

This research was supported by ONR Grant N00014-15-1-2078 Distributed Integer Programming to the Georgia Institute of Technology.

References

[1] J.-Y. Audibert, S. Bubeck, and G. Lugosi. Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1):31–45, 2013.

[2] A. Ben-Tal and A. Nemirovski. Lectures on modern convex optimization. *Lecture Notes*, pages 1–590, 2013.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and distributed computation: Numerical methods*. Athena Scientific, 1997.

[4] G. Braun and S. Pokutta. An efficient high-probability algorithm for linear bandits. *arXiv preprint arXiv:1610.02072*, 2016.

[5] S. Bubeck. Introduction to online optimization. *Lecture Notes*, pages 1–86, 2011.

[6] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*, 4(2):28–34, 2002.

[7] E. Hazan. *Introduction to online convex optimization*. Foundations and Trends(r) in Optimization Series. Now Publishers, 2016.

[8] S. Kakade, A. T. Kalai, and K. Ligett. Approximation algorithms going online. *Technical Report CMU-CS-07-102*, 2007.

[9] E. Karabulut, S. Ahmed, and G. Nemhauser. Decentralized algorithms for distributed integer programming problems with a coupling knapsack constraint. *Optimization Online*, 2018.

[10] W. M. Koolen, M. K. Warmuth, J. Kivinen, et al. Hedging structured concepts. *COLT*, pages 93–105, 2010.

[11] T. Lattimore, K. Crammer, and C. Szepesvári. Optimal resource allocation with semi-bandit feedback. *arXiv preprint arXiv:1406.3840*, 2014.

[12] K. Liu and Q. Zhao. Distributed learning in multi-armed bandit with multiple players. *IEEE Transactions on Signal Processing*, 58(11):5667–5681, 2010.

[13] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic game theory*. Cambridge university press, 2007.